56F801X

Peripheral Reference Manual

56F8000
16-bit Digital Signal Controllers (DSC)

MC56F8000RM
Rev. 5
08/2007

freescale.com

This manual is one of a set of three documents. For complete product information, it is necessary to have all three documents. They are: DSP56800E Reference Manual, 56F801X Peripheral User Manual, and Device Technical Data Sheet.

Order this document by **MC56F8000RM**

Revision History:

See revision history in each chapter

TABLE OF CONTENTS

Chapter 7
General Purpose Input/Output (GPIO)

Chapter 8
Joint Test Action Group Port (JTAG)

Chapter 9
Power Supervisor (PS)

Chapter 10
Pulse Width Modulator (PWM)

Chapter 13
Quad Timer (TMR)

LIST OF FIGURES

LIST OF TABLES

Preface

About This Manual

Features of the 56F801X Series of 16-bit Digital Signal Controllers (DSC) are described in this preliminary manual release. Peripheral modules are documented here. This manual is intended to be used with the *DSP56800E Reference Manual* (DSP56800RM), describing the Central Processing Unit (CPU), programming models, and instruction set details. The *56801X Technical Data Sheet* provides electrical specifications as well as timing, pinout, packaging descriptions and chip specific details of architecture and memory map particulars.

Audience

Information in this manual is intended to assist design and software engineers integrate the 56F801X digital signal processors into a design and/or while developing application software.

Manual Organization

This manual is arranged in chapters described below:

- **Chapter 1, Overview**—provides a brief overview to the core, devices, and peripherals, describing the structure of this document, including lists of other documentation necessary to use these chips.
- **Chapter 2, Analog-to-Digital Converter (ADC)**—describes features, functions and registers of the analog-to-digital converter.
- **Chapter 3, Computer Operating Properly (COP)**—Computer Operating Properly (COP) module is used to help software recover from runaway code.
- **Chapter 4, Inter-Integrated Circuit Interface (I2C)**—describes a two-wire, bi-directional serial bus suitable for short distance communication among devices.
- **Chapter 5, On-Chip Clock Synthesis (OCCS)**—elaborates on the internal oscillator, relaxation oscillator, Phase Lock Loop (PLL), and timer distribution chain for the 56801X.
- **Chapter 6, Flash Memory (FM)**—describes the Program Flash and Boot Flash features and registers.

- **Chapter 7, General Purpose Input/Output (GPIO)**—describes how peripheral pins are multiplexed with GPIO functions.
- **Chapter 8, Joint Test Action Group Port (JTAG)**—explains the Joint Test Action Group (JTAG) testing methodology and its capabilities with Test Access Port (TAP) and Enhanced OnCE (fully explained in the DSP56800E Reference Manual).
- **Chapter 9, Power Supervisor (PS)**—details how the on-chip PS module monitors on-chip voltages.
- **Chapter 10, Pulse Width Modulator (PWM)**—describes the function, configuration, and registers of the PWM.
- **Chapter 11, Serial Communications Interface (SCI)**—communicates with devices such as other DSPs, microprocessors, or peripherals providing the primary data input path as codecs.
- **Chapter 12, Serial Peripheral Interface (SPI)**—is described with the capability to communicate with external devices, such as Liquid Crystal Displays (LCDs) and Microcontroller Units (MCUs).
- **Chapter 13, Quad Timer (TMR)**—outlines the internal Quad Timer devices available, including features and registers.
- **Chapter 14, Voltage Regulator (VREG)**—describes the on-chip mechanism used to regulate an external 3.3 V supply down to 2.5 V levels for use with internal logic.
- **Appendix A,Glossary**—provides definitions of terms, peripherals, acronyms and register names used in this manual.
- **Appendix B, Programmer's Sheets**—supplies concise, one-location registers and their preference tables intended to simplify programming of the 56F801X.

Suggested Reading

A list of DSP-related books is provided here as an aid to those who may be new to DSPs:

Advanced Topics in Signal Processing, Jae S. Lim and Alan V. Oppenheim (Prentice-Hall: 1988).

Applications of Digital Signal Processing, A. V. Oppenheim (Prentice-Hall: 1978).

Digital Processing of Signals: Theory and Practice, Maurice Bellanger (John Wiley and Sons: 1984).

Digital Signal Processing, Alan V. Oppenheim and Ronald W. Schafer (Prentice-Hall: 1975).

Digital Signal Processing: A System Design Approach, David J. DeFatta, Joseph G. Lucas, and William S. Hodgkiss (John Wiley and Sons: 1988).

Discrete-Time Signal Processing, A. V. Oppenheim and R.W. Schafer (Prentice-Hall: 1989).

Foundations of Digital Signal Processing and Data Analysis, J. A. Cadzow (Macmillan: 1987).

Handbook of Digital Signal Processing, D. F. Elliott (Academic Press: 1987).

Introduction to Digital Signal Processing, John G. Proakis and Dimitris G. Manolakis (Macmillan: 1988).

IP Bus Specifications, Semiconductor Reuse Standard, SRSIPB1, v 2.0, Draft 1.6.

Multirate Digital Signal Processing, R. E. Crochiere and L. R. Rabiner (Prentice-Hall: 1983).

Signal Processing Algorithms, S. Stearns and R. Davis (Prentice-Hall: 1988).

Signal Processing Handbook, C. H. Chen (Marcel Dekker: 1988).

Signal Processing: The Modern Approach, James V. Candy (McGraw-Hill: 1988).

Theory and Application of Digital Signal Processing, Lawrence R. Rabiner and Bernard Gold (Prentice-Hall: 1975).

Manual Conventions

Conventions used in this manual:

- Bits within registers are always listed from Most Significant Bit (MSB) to Least Significant Bit (LSB).
- Bits within a register are formatted AA[n:0] when more than one bit is involved in a description. For purposes of description, the bits are presented as if they are contiguous within a register. However, this is not always the case. Refer to the programming model diagrams or to the programmer's sheets to see the exact location of bits within a register.
- When a bit is described as *set*, its value is set to *one*. When a bit is described as *cleared*, its value is set to *zero*.
- The word *pin* is a generic term for any pin on the chip.
- Pins or signals asserted low, made active when pulled to ground, have an over-bar above their name. For example, the $\overline{\text{SS0}}$ pin is asserted low.
- Hex values are indicated with a dollar sign ($) preceding the hex value, as follows: $F1A0 is the X memory address for an Interrupt Priority Register (IPR).
- Code examples follow in a single spaced font.

```
      BFSET  #$0007,X:PCC ; Configure:                          line 1
                          ; MISO0, MOSI0, SCK0 for SPI master   line 2
                          ; ~SS0 as PC3 for GPIO                line 3
```

- Pins or signals listed in code examples asserted as low have a tilde in front of their names. In the previous example, line three refers to the $\overline{\text{SS0}}$ pin, shown as ~SS0.
- The word *reset* is used in three different contexts in this manual. They are described as:
  - — a reset pin is always written as $\overline{\text{RESET}}$, in uppercase, using the over bar
  - — the processor state occurs when the $\overline{\text{RESET}}$ pin is asserted. It is always written as Reset, with a capitalized first letter
  - — the word *reset* refers to the reset function. It is written in lowercase, without italics, used here only for differentiation. The word may require a capital letter as style dictates, such as in headings and captions
- The word *assert* means a high true (active high) signal is pulled high to V_{DD}, or a low true (active low) signal is pulled low to ground. The word *deassert* means a high true signal is pulled low to ground, or a low true signal is pulled high to V_{DD}.

Pin Conventions

Signal/Symbol	Logic State	Signal State	Voltage[1]
$\overline{\text{PIN}}$	True	Asserted	V_{IL}/V_{OL}
$\overline{\text{PIN}}$	False	Deasserted	V_{IH}/V_{OH}
PIN	True	Asserted	V_{IH}/V_{OH}
PIN	False	Deasserted	V_{IL}/V_{OL}

1. Values for V_{IL}, V_{OL}, V_{IH}, and V_{OH} are defined by individual product specifications.

Throughout the manual, registers and tables displaying a grayed area designate reserved bits.

▨ = Reserved or not implemented bit, write or read as zero for future compatibility

Chapter 1
Overview

Document Revision History for Chapter 1, Overview

Version History	Description of Change
Rev. 0	Initial Release
Rev. 1	Correction to **Section 1.2.6.6**, fourth bullet item changes to: Usable as a keypad interface Correction to **Section 1.2.6.8** makes feature list consistent with corresponding list in chapter 9 Addition to **Section 1.2.6.9** eighth bullet item adds TMR to the list by which PWM can be controlled

1.1 Introduction to 56800E Core

The 56800E core, combines digital processing power with the functionality of a microcontroller. Adding a flexible set of peripherals creates an extremely cost-effective system solution on a single chip. Because of its low cost, configuration flexibility, and compact program code, the 56800E is well-suited for many applications.

The 56800E core is based on a Harvard-style architecture consisting of three execution units operating in parallel allowing as many as six operations per instruction cycle. The MCU-style programming model and optimized instruction set allow straightforward generation of efficient, compact 16-bit control code. The instruction set is also highly efficient for C/C++ Compilers, enabling rapid development of optimized control applications.

1.1.1 56800E Core Enhancements

The 56800E core architecture is C source-code compatible with 56F801X family devices, extending the family architecture, thereby adding these features:

- Byte and long data types, supplementing the word data type of the 56F801X
- 24-bit data memory address space
- 21-bit program memory address space
- Two additional 24-bit address registers
- Two additional 36-bit accumulator registers
- Full-precision integer multiplication
- 32-bit logical and shifting operations
- Second read in dual read instruction can access off-chip memory
- Loop Count (LC) register extended to 16 bits
- Support for nested DO looping through additional loop address and count registers
- Loop address and hardware stack extended to 24 bits
- Three additional interrupt levels with a software interrupt for each level
- Enhanced On-Chip Emulation (EOnCE) with three debugging modes:
  - non-intrusive real-time debugging
  - minimally intrusive real-time debugging
  - Breakpoint and Step Modes (core is halted)

1.1.2 56800E Core

The 56800E core is composed of several independent functional units. The program controller, Address Generation Unit (AGU), and data Arithmetic Logic Unit (ALU) contain their own register sets and control logic, allowing them to operate independently and in parallel to increase throughput. There is also an independent bit manipulation unit to enable efficient bit field operations. Each functional unit interfaces with the other units, memory, and the memory-mapped peripherals over the core's internal address and data buses. A block diagram of the 56800E core architecture is illustrated in **Figure 1-1**.

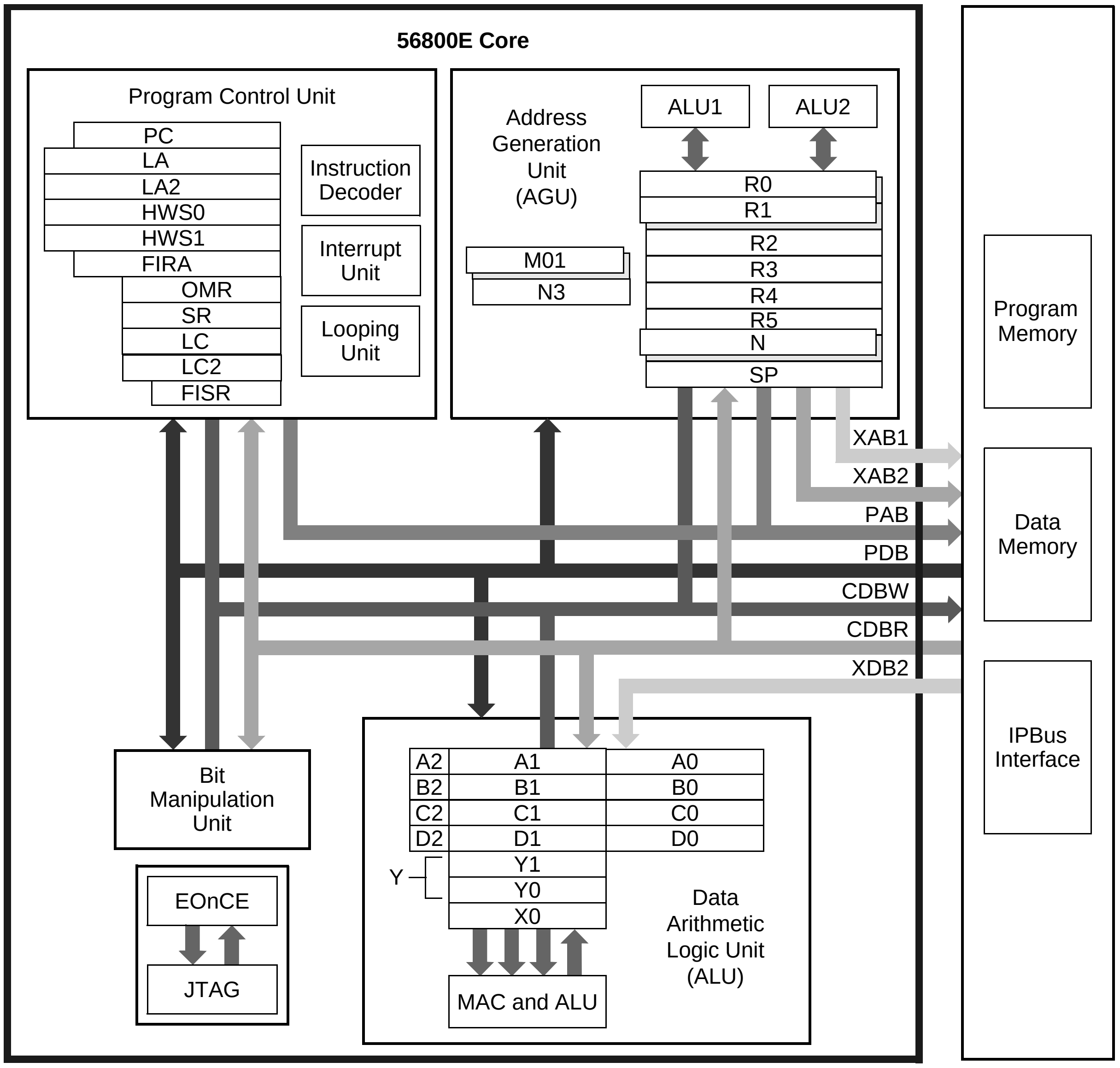


Figure 1-1. 56800E Core Block Diagram

Instruction execution is pipelined to take advantage of the parallel units, thereby significantly decreasing execution time for each instruction. For example, within a single execution cycle, it is possible for the:

- Data ALU to perform a multiplication operation
- AGU to generate up to two addresses
- Program controller to prefetch the next instruction

The major components of the 56800E core include:

- Address buses
- Data buses
- Data Arithmetic Logic Unit (ALU)
- Address Generation Unit (AGU)
- Program controller and hardware looping unit
- Bit manipulation unit
- Enhanced OnCE debugging module
- Clock generation
- Reset circuitry

1.1.2.1 Address Buses

The core contains three address buses:

1. Program Memory Address Bus (PAB) –21 bits
2. Primary Data Address Bus (XAB1) – 24 bits
3. Secondary Data Address Bus (XAB2) – 24 bits

The program address bus is used to access (16-bit) instruction words in program memory. The two data address buses allow for two simultaneous accesses to data (X) memory. The XAB1 bus can address byte, word, and long data types. The XAB2 bus is limited to (16-bit) word accesses.

All three buses address on-chip memory. They can also address off-chip memory on devices containing an external bus interface unit.

1.1.2.2 Data Buses

Data transfers inside the chip occur over these buses:

- Two unidirectional 32-bit buses:
  - Core Data Bus for Reads (CDBR)
  - Core Data Bus for Writes (CDBW)
- Two unidirectional 16-bit buses:
  - Secondary X Data Bus (XDB2)
  - Program Data Bus (PDB)
- IPBus interface

Data transfers between the data ALU and data memory use the CDBR and CDBW when a single memory read or write is performed. When two simultaneous memory reads are performed, the transfers use the CDBR and XDB2 buses. All other data transfers to core blocks occur using the CDBR and CDBW buses. Peripheral transfers occur through the IPBus interface. Instruction word fetches occur over the PDB.

This bus structure supports up to three simultaneous 16-bit transfers. Any one of the following can occur in a single clock cycle:

- One instruction fetch
- One read from data memory
- One write to data memory
- Two reads from data memory
- One instruction fetch and one read from data memory
- One instruction fetch and one write to data memory
- One instruction fetch and two reads from data memory

An instruction fetch will take place on every clock cycle, although it is possible for data memory accesses to be performed without an instruction fetch. Such accesses typically occur when a hardware loop is executed and the repeated instruction is only fetched on the first loop iteration.

1.1.2.3 Data Arithmetic Logic Unit (Data ALU)

The data Arithmetic Logic Unit (ALU) performs all of the arithmetic, logical, and shifting operations on data operands. The data ALU contains the following components:

- Three, 16-bit data registers (X0, Y0, and Y1)
- Four, 36-bit accumulator registers (A, B, C, and D)
- One Multiply-Accumulator (MAC) unit
- A single-bit accumulator shifter

- One arithmetic and logical multi-bit shifter
- One MAC output limiter
- One data limiter

The data ALU can perform multiplication, multiply-accumulation (with positive or negative accumulation), addition, subtraction, shifting, and logical operations in a single cycle. Division and normalization operations are provided by iteration instructions. Signed and unsigned multiple precision arithmetic is also supported. All operations are performed using two’s-complement fractional or integer arithmetic.

Data ALU source operands can be 8, 16, 32, or 36 bits in size and can be located in memory, in immediate instruction data, or in the data ALU registers. Arithmetic operations and shifts can have 16-, 32-, or 36-bit results. Logical operations are performed on 16- or 32-bit operands and yield results of the same size. The results of data ALU operations are stored either in one of the data ALU registers or directly in memory.

1.1.2.4 Address Generation Unit (AGU)

The Address Generation Unit (AGU) performs all of the calculations of effective addresses for data operands in memory. It contains two address ALUs, allowing up to two 24-bit data addresses to be generated every instruction cycle:

- One for the Primary Data Address Bus (XAB1)
- One for the Secondary Data Address Bus (XAB2)

The address ALU can perform both linear and modulus address arithmetic. The AGU operates independently of the other core units, minimizing address-calculation overhead.

The AGU can directly address 2^{24} (16M) words on the XAB1 and XAB2 buses. It can access 2^{21} (2M) words on the PAB. The XAB1 bus can address byte, word, and long data operands. The PAB and XAB2 buses can only address words in memory.

The AGU consists of the following registers and functional units:

- Seven 24-bit address registers (R0–R5 and N)
- Four 24-bit shadow registers (for R0, R1, N, and M01)
- A 24-bit dedicated Stack Pointer (SP) register
- Two offset registers (N and N3)
- A 16-bit modifier register (M01)
- A 24-bit adder unit
- A 24-bit modulus arithmetic unit

Each of the address registers, R0–R5, can contain either data or an address. All of these registers can provide an address for the XAB1 and PAB address buses; addresses on the XAB2 bus are provided by the R3 register. The N offset register can be used either as a general-purpose address register, or as an offset, or update value for the addressing modes supporting those values. The second 16-bit offset register, N3, is used only for offset or update values. The modifier register, M01, selects between linear and modulus address arithmetic.

1.1.2.5 Program Controller and Hardware Looping Unit

The program controller is responsible for instruction fetching and decoding, interrupt processing, hardware interlocking, and hardware looping. Actual instruction execution takes place in the other core units, such as in the data ALU, AGU, or bit manipulation unit.

The program controller contains the following:

- An instruction latch and decoder
- The hardware looping control unit
- Interrupt control logic
- A Program Counter (PC)
- Two special registers for Fast Interrupts:
  - Fast Interrupt Return Address (FIRA) register
  - Fast Interrupt Status Register (FISR)
- Seven user-accessible Status and Control registers
  - two-level deep Hardware Stack (HWS)
  - Loop Address (LA) register
  - Loop Address 2 (LA2) register
  - Loop Count (LC) register
  - Loop Count 2 (LC2) register
  - Status Register (SR)
  - Operating Mode Register (OMR)

The Operating Mode Register (OMR) is a programmable register controlling the operation of the 56800E core, including the memory map configuration. The initial operating mode is typically latched on reset from an external source; it can subsequently be altered under program control.

The Loop Address (LA) register and Loop Count (LC) register work in conjunction with the Hardware Stack (HWS) to support no-overhead hardware looping. The hardware stack is an internal Last-In-First-Out (LIFO) buffer consisting of two 24-bit words to store the address of the first instruction of a hardware DO loop. When executing the DO instruction, the address of the first instruction in the loop is pushed onto the HWS. When a loop finishes normally or an ENDDO instruction is encountered, the value is popped from the HWS. This process allows for one hardware DO loop to be nested inside another.

1.1.2.6 Bit Manipulation Unit

The bit manipulation unit performs bit field operations on data memory words, peripheral registers, and registers within the 56800E core. It is capable of testing, setting, clearing, or inverting individual or multiple bits within a 16-bit word. The bit manipulation unit can also test bytes for branch-on-bit field instructions.

1.1.2.7 Enhanced On-Chip Emulation (EOnCE) Module

The Enhanced On-Chip Emulation (EOnCE) module allows interaction in a debug environment with the 56800E core and its peripherals. Its capabilities include:

- Examining registers
- Accessing memory, or on-chip peripherals
- Setting breakpoints in memory
- Stepping or tracing instructions

The EOnCE module provides simple, inexpensive, and speed independent access to the 56800E core for sophisticated debugging and economical system development. The JTAG port allows access to the EOnCE module and through the 56800E device to its target system, retaining debug control without sacrificing other user accessible on-chip resources. This technique eliminates the costly cabling and access to processor pins required by traditional emulator systems. The EOnCE interface is fully described in the 56800E Reference manual.

1.1.3 System Bus Controller

The 56800E System Bus Controller (SBC) provides an interface between the 56800E core and other modules on the system bus. The SBC is composed of a set of buffers for the address and control signals originating at the core, and a separate set of multiplexers, routing data from each memory-mapped block back to the core.

The 56800E architecture includes two separate bus models:

1. System bus
2. IPBus

Internal memories, the external memory interface and the core are located on the system buses. All peripherals connect to the IPBus. Access to the IPBus by the 16-bit controller core is facilitated by the IPBus bridge. The System Bus Controller does not participate in IPBus transactions. Within this document, all descriptions of bus operations pertain only to the system bus.

For performance reasons, all system bus signals in the 56800E architecture have a single driver, as opposed to the more common three-state bus configurations. Read data from each memory-mapped device is multiplexed to avoid contention. Since the 16-bit controller core is the only system bus master, there is no need for multiplexers on the address, control or write data buses.

1.1.4 Operation Method

The 56800E system utilizes a pipelined memory architecture and separate program and data buses. Each memory cycle is completed in three or more system clock cycles. During the first of these cycles, the core presents an address, along with control signals, indicating the type of memory cycle being initiated. This clock cycle is referred to as the address phase of a memory cycle. The following cycle is an intermediate step not involving bus activity related to the memory cycle in progress. Finally, the data phase occurs. During this phase data is transferred to or from the master, depending upon the type of cycle initiated during the address phase.

Memory cycles can overlap in each clock cycle, a new address phase can begin while the data phase for a preceding memory cycle occurs. In certain cases, memory devices or the 16-bit controller core may require additional time to complete operations. When this occurs, clock edges to other modules are withheld by the clock generation circuitry.

1.2 Introduction to 56F801X Devices

1.2.1 Applications

The 56F801X family of products includes many peripherals useful for applications such as:

- BLDC motor control
- Dimming lamp ballast
- Switched-mode power supply
- Soft-switching PFC
- Appliance motor control
- DC-DC power supplies
- Smart sensors
- Instrumentation

1.2.2 Features

The 56F801X family of products provides a variety of memory options and peripherals, thereby enhancing performance, and reducing application cost while promoting an ease of product development. Features making these benefits possible include:

- Program Flash with built-in write protection and security
- Unified Program and Data RAM
- 12-bit Analog-to-Digital (ADC)
- Inter-Integrated Circuit Serial Bus Interface (I^2C)
- Serial Communication Interface (SCI)
- Serial Peripheral Interface (SPI)
- Quad Timer (TMR)
- Pulse Width Modulator (PWM)
- Computer Operating Properly (COP)/Watchdog
- JTAG/Enhanced On-Chip Emulation (EOnCE) for debugging
- GPIO lines
- LQFP package
- Competitive cost sensitive packaging

1.2.3 System Architecture and Peripheral Interface

The 56800E system architecture encompasses all on-chip components, including the core, on-chip memory, peripherals, and the buses necessary to connect them. **Figure 1-2** illustrates the overall system architecture for a device with an external bus.

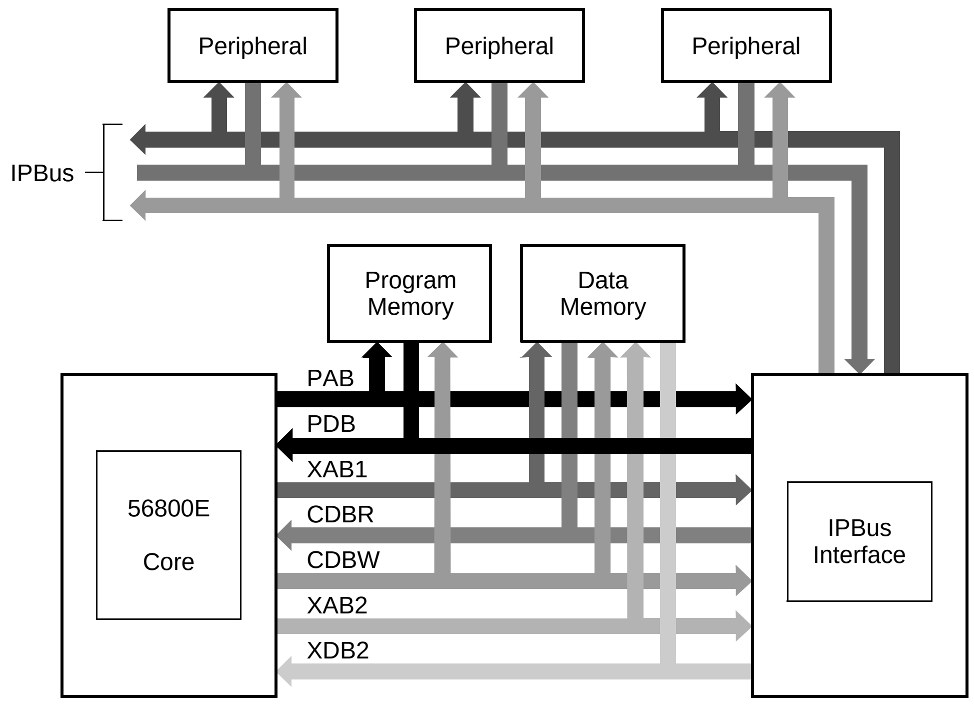


Figure 1-2. 56800E Chip Architecture with External Bus

The complete architecture includes the following components:

- 56800E core
- On-chip program memory
- On-chip data memory
- On-chip peripherals
- Motorola IPBus peripheral interface

Some 56800E devices might not implement an external bus interface. Regardless of the implementation, all peripherals communicate with the 56800E core via the IPBus interface. The program and data memory buses are not connected to peripherals.

1.2.4 IPBus Bridge (IPBB)

The IPBus Bridge (IPBB) provides a means for communication between the high speed core and the low-bandwidth devices on the IP peripheral bus. Among other functions, the bridge is responsible for maintaining an orderly and synchronized communication between devices on both sides potentially running at different clock frequencies.

The IPBus architecture supports a variety of on-chip peripherals:

- Analog-to-Digital Converters (ADC)
- Computer Operating Properly (COP) module
- Phase-Locked Loop (PLL) module
- Flash Memory (FM) module
- Programmable General-Purpose I/O (GPIO) modules
- Joint Test Action Group Port (JTAG) module
- Power Supervisor (PS) module
- Pulse Width Module (PWM) module
- Serial Communication Interface (SCI/LIN) modules
- Serial Peripheral Interface (SPI) modules
- 16-bit Quad Timer (TMR) modules
- Inter-Integrated Circuit Interface Bus (I^2C) module
- Voltage Regulator (VREG) module

Figure 1-3 denotes the position and interface of the IPBus Bridge with other main blocks within the chip.

Other connections in the figure not pertaining to the primary function of the bridge are omitted for clarity; nevertheless they will be discussed as appropriate. A brief description of the bridge's interface with various main components on both sides is also provided.

1.2.4.1 System Side Operation

On the system side the IPBus Bridge operates at 16-bit controller core frequency and fully supports pipelined communication with the core. The bridge acts as a slave device on this bus. The bridge is responsible for initiating IPBus transactions only per requests initiated by the 16-bit controller core.

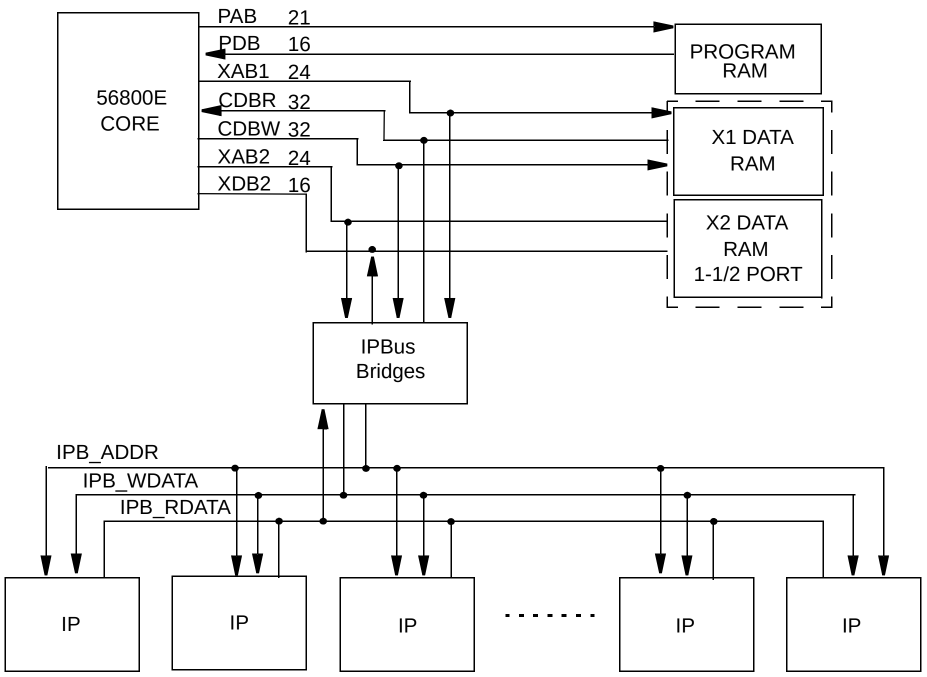


Figure 1-3. IPBus Bridge Interface With Other Main Components System Side Operation

1.2.4.2 Peripheral Side Operation

On the peripheral side, the IPBus Bridge accesses various devices through a standard non-pipelined IPBus interface. Separate bus lines are used for read and write transactions.

1.2.5 Peripheral Interrupts/Interrupt Controller Module

The peripherals on the 56F801X use the interrupt channels found on the 56800E core. Each peripheral has its own interrupt vector (often more than one interrupt vector for each peripheral), and can selectively be enabled or disabled via the Interrupt Priority Register (IPR) found in the Interrupt Controller (ITCN) module. Detailed information regarding the Interrupt Controller is located in the Data Sheet. Design includes these distinctive features:

- Programmable priority levels for each IRQ
- Two programmable Fast Interrupts
- Notification to the SIM module to restart clocks out of Wait and Stop modes

Note: Please see the device Data Sheet for detailed information about this module.

1.2.5.1 System Integration Module (SIM)

The SIM module is a system catchall for the glue logic tieing together the system-on-chip. It controls distribution of resets and clocks and provides a number of control features. The system integration module is responsible for the following functions:

- Reset sequencing
- Clock control & distribution
- STOP/WAIT control
- System status registers
- I/O Pad multiplexing
- Registers for software access to the JTAG ID of the chip
- Enforcing Flash security

Note: Please see the device Data Sheet for detailed information about this module.

1.2.6 56F801X Peripheral Features

1.2.6.1 Analog-to-Digital Converter (ADC)

Detailed information regarding the ADC peripheral is located in Chapter 2 of this manual.

- 12-bit resolution
- Maximum ADC clock frequency is 5.33MHz with 187.5ns period
- Sampling rate up to 1.78 million samples per second[1]
- Single conversion time of 8.5 ADC clock cycles (8.5 × 187.5ns = 1.595μs)
- Additional conversion time of 6 ADC clock cycles (6 × 187.5ns = 1.126μs)
- Eight conversions in 26.5 ADC clock cycles (26.5 × 187.5ns = 4.972μs) using simultaneous mode
- ADC conversions can be synchronized by both the PWM and TMR modules
- Simultaneous or sequential sampling
- Internal multiplexer to select two of eight inputs
- Ability to sequentially scan and store up to eight measurements
- Ability to simultaneously sample and hold two inputs
- Optional interrupts at the end of a scan, if an out-of-range limit is exceeded, or at zero crossing
- Optional sample correction by subtracting a pre-programmed offset value
- Signed or unsigned result
- Single-ended or differential inputs

1. Once in Loop mode, the time between each conversion is six ADC Clock cycles (1.125μs). Using simultaneous conversion two samples are captured in 1.126μs, providing an overall sample rate of 1,776,667 samples per second.

1.2.6.2 Computer Operating Properly (COP)

Detailed information regarding this peripheral is located in Chapter 3 of this manual.

- Programmable timeout period = (1024 × (CT + 1)) oscillator clock cycles, where CT can be from $0000 to $FFFF
- Programmable Wait and Stop modes
- COP timer is disabled while host controller is in Debug mode

1.2.6.3 Inter-Integrated Circuit Interface (I^2C)

Detailed information regarding the I^2C peripheral is located in Chapter 4 of this manual.

- Compatible with I^2C Bus Standard
- Multi-master operation
- Software programmable serial clock frequency
- Software selectable acknowledge bit
- Interrupt driver byte-by-byte to data transfer
- Arbitrator Lost interrupt with auto mode switching from master to slave
- Calling Address Identification interrupt
- Start and Stop signal generation/detection
- Repeated Start signal generation
- Acknowledge bit generation/detection
- Busy bus detection

1.2.6.4 On-Chip Clock Synthesis (OCCS)

Detailed information regarding this peripheral is located in Chapter 5 of this manual.

- Built-in 8MHz relaxation oscillator
- Three-bit postscaler can divide PLL output by 2, 4, 8, 16, or 32 prior to use as system clock
- Ability to power down the internal PLL
- Provides 2x master clock frequency to SIM module
- Provides 3x master clock frequency to PWM and Quad Timer
- Safety shutdown feature available in the event the PLL reference clock disappears

1.2.6.5 Flash Memory (FM)

Detailed information regarding this peripheral is located in Chapter 6 of this manual.

- 12k/16k bytes (chip dependent) of Program Flash Memory
- 32MHz single cycle operation for Program Flash access is at 150ºC (T_J) and 2.25V

- Automated program and erase operation
- Interrupts on command completion, command buffer empty and access error
- Fast page erase
- Single power supply program and erase
- Security feature prohibits flash access from external devices
- Protection feature prohibits accidental program/erase

1.2.6.6 General Purpose Input/Output (GPIO)

Detailed information regarding this peripheral is located in Chapter 7 of this manual.

- Individual control for each pin to be in either Normal (peripheral) or GPIO modes
- Individual direction control for each pin in GPIO mode
- Individual pull-up enable control for each pin in either Normal or GPIO modes
- Usable as a keypad interface
- Ability to monitor pad logic values even when GPIO are not enabled by using the RDATA register
- Interrupt Assert Capability
- I/O drive strength control

1.2.6.7 Joint Test Action Group Port (JTAG)

Detailed information regarding this peripheral is located in Chapter 8 of this manual.

- Perform boundary scan operations to test circuit board electrical continuity
- Bypass the TAP for a given circuit board test by replacing the Boundary Scan Register (BSR) with a single-bit register
- Sample system pins during operation and transparently shift-out the results in the BSR
- Preload output pins prior to invoking the EXTEST instruction
- Disable the output drive to pins during circuit board testing
- Provide a means of accessing the EOnCE module controller and circuits to control a target system
- Query the IDCODE from any TAP in the system
- Force test data onto the peripheral outputs while replacing its BSR with a single bit register
- Enable/disable pull-up devices on peripheral boundary scan pins

1.2.6.8 Power Supervisor (PS)

Detailed information regarding this peripheral is located in Chapter 9 of this manual.

- Power-On Reset (POR)
  - Holds device in reset until:
    - V_{DD} core voltage exceeds 1.8V
    - regulator voltages have risen above LVI thresholds (2.2V and 2.7V)
- Core Low Voltage Interrupt (LVI22)
  - generated when the 2.5V rail drops below 2.2V
- I/O Low Voltage Interrupt (LVI27)
  - generated when the 3.3V rail drops below 2.7V

1.2.6.9 Pulse Width Modulator (PWM)

Detailed information regarding this peripheral is located in Chapter 10 of this manual.

- Three complementary PWM signal pairs, or six independent PWM signals
- Operation at speeds up to 96MHz
- Features of complementary channel operation
  - deadtime insertion
  - separate top and bottom pulse width correction via current status inputs or software
  - separate top and bottom polarity control
- Edge-aligned or center-aligned PWM signals
- 15-bits of resolution
- Half-cycle reload capability
- Integral reload rates from one to 16
- PWM output can be controlled by:
  - software
  - ADC
  - GPIO
  - TMR
- Programmable fault protection
- Polarity control
- 5/8mA current source/sink capability on PWM pins
- Write protected registers

1.2.6.10 Serial Communications Interface (SCI)

Detailed information regarding this peripheral is located in Chapter 11 of this manual.

- Full-duplex or single wire operation
- Standard mark/space Non-Return-to-Zero (NRZ) format
- 13-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Separate receiver and transmitter 16-bit controller interrupt requests
- Programmable polarity for transmitter and receiver
- Two receiver wake up methods:
  - idle line
  - address mark
- Interrupt-driven operation with seven flags:
  - transmitter empty
  - transmitter idle
  - receiver full
  - receiver overrun
  - noise error
  - LIN synchronization error
  - framing error
  - parity error
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection
- LIN slave capability

1.2.6.11 Serial Peripheral Interface (SPI)

Detailed information regarding this peripheral is located in Chapter 12 of this manual.

- Full-duplex operation
- Master and Slave modes
- Double-buffered operation with separate transmit and receive registers
- Programmable length transmissions (2 to 16 bits)
- Programmable transmit and receive shift order (MSB first or last bit transmitted)

- Eight master mode frequencies (maximum = module clock ÷ 2)
- Maximum Slave mode frequency = module clock ÷ 2
- Clock ground for reduced Radio Frequency (RF) interference
- Serial clock with programmable polarity and phase
- Two separately enabled interrupts
  - SPI Receiver Full (SPRF)
  - SPI Transmitter Empty (SPTE)
- Mode fault error flag interrupt capability
- Wired OR mode functionality enabling connection to multiple SPIs

1.2.6.12 Quad Timer (TMR)

Detailed information regarding this peripheral is located in Chapter 13 of this manual.

- Four, 16-bit counters/timers
- Capable of counting up and down
- Counters will cascade
- Count modulus can be programmed
- Maximum count rate equals peripheral clock/2 for external clocks
- Maximum count rate up to 96MHz for internal clocks
- Will count once or repeatedly
- Counters can be preloaded
- Counters can share available input pins
- Separate prescaler for each counter
- Each counter has capture and compare capability
- Counters can run at 3x the peripheral clock rate
- Counters can run when the chip is in Stop mode

1.2.6.13 Voltage Regulator (VREG)

Detailed information regarding this peripheral is located in Chapter 14 of this manual.

- Provide a 2.5V ±10 percent accuracy
- Provide an average current of at least 250mA for the larger regulator
- Provide an average current of at least 1mA for the smaller regulators

1.3 Energy Information

- Fabricated in high-density CMOS with 5.0V tolerant, TTL-compatible digital inputs
- On-board 3.3V down to 2.5V voltage regulator for powering internal logic and memories
- On-chip regulators for digital and analog circuitry to lower cost and reduce noise
- Wait and Stop modes available
- ADC smart power management
- Each peripheral can be individually disabled to save power

Chapter 2
Analog-to-Digital Converter (ADC)

Document Revision History for Chapter 2, Analog-to-Digital Converter (ADC)

Version History	Description of Change
Rev. 0	Initial Release
Rev. 1	Grammatical/naming convention issues throughout chapter resolved
Rev. 2	Correction to Figure 2-28 values. PSTS2, PSTS1, and PSTS0 reset values should be 1 not 0. Same corrections made to the Programmers Sheets, Appendix B as well. Added Section 2.4.6.1, “Manual Power Down of Unused Converters,” on page 2-16, Correction to equation in **Section 2.4.2.2**, Additional grammatical issues throughout chapter resolved
Rev. 5	Corrected the entry for CTRL2 in **Figure 2-12**.

2.1 Introduction

The Analog-to-Digital Converter (ADC) for the 56F801X Series consists of two separate and complete ADCs, each with their own sample and hold circuits. The converters share a common voltage reference and common digital control module.

2.2 Features

The ADC's characteristics include:

- 12-bit resolution
- Maximum ADC clock frequency of 5.33MHz (1/6 of system clock), 187.5ns period
- Sampling rate up to 1.78 million samples per second[1]
- Single conversion time of 8.5 ADC clock cycles (8.5×187.5ns = 1.595µs)
- Additional conversion time of 6 ADC clock cycles (6×187.5ns = 1.126µs)
- Eight conversions in 26.5 ADC clocks (26.5×187.5ns = 4.972µs) using Simultaneous mode
- ADC conversions can be synchronized by both the PWM and TMR modules
- ADC can provide PWM output with hysteresis using high and low limits for samples 0-2.
- Ability to simultaneously sample and hold of two inputs
- Ability to sequentially scan and store of up to eight measurements
- Internally multiplex to select two of eight inputs
- Power savings modes allow automatic shutdown/startup of all or part of ADC
- Those inputs not selected tolerate injected/sourced current without affecting ADC performance, supporting operation in noisy industrial environments. See the *Device Data Sheet*'s *ADC Parameters* table for more information on current limits.
- Optional interrupts at the end of a scan, if an out-of-range limit is exceeded, (high or low) or at zero crossing
- Optional sample correction by subtracting a pre-programmed offset value
- Signed or unsigned result
- Single ended or differential inputs for all input pins with support for an arbitrary mix of input types

1. Once in Loop mode, the time between each conversion is six ADC Clock cycles (1.125 µs). Using Simultaneous conversion two samples are captured in 1.126 µs, providing an overall sample rate of 1,776,667 samples per second.

2.3 Block Diagram

The ADC function, shown in **Figure 2-1**, consists of two four-channel input select functions, interfacing with two independent Sample and Hold (S/H) circuits, which feed two 12-bit ADCs. The two converters store their results in a buffer, awaiting further processing.

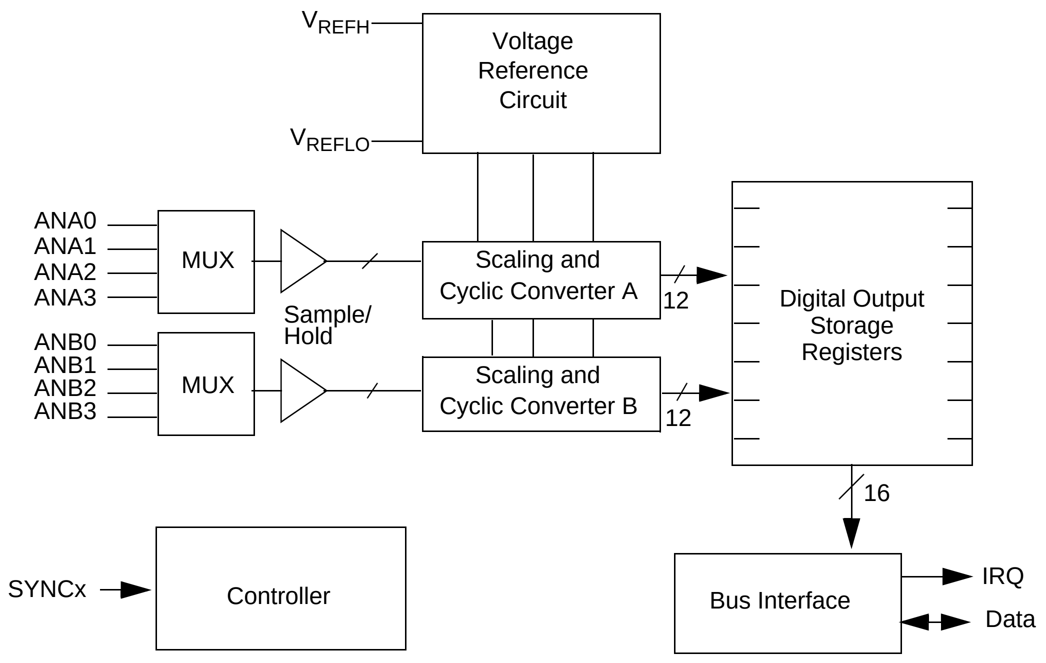

VREFH
VREFLO
Voltage Reference Circuit
ANA0
ANA1
ANA2
ANA3
MUX
ANB0
ANB1
ANB2
ANB3
MUX
Sample/ Hold
Scaling and Cyclic Converter A
Scaling and Cyclic Converter B
12
12
Digital Output Storage Registers
16
SYNCx
Controller
Bus Interface
IRQ
Data

Figure 2-1. Option 1: Dual ADC Block Diagram

2.4 Functional Description

The ADC's conversion process is either initiated by a sync signal from one of the on-chip timer channels (see chip specific documentation for the specific timer channel associated with the ADC) or by writing 1 to a START*n* bit.

Starting a single conversion actually begins a sequence of conversions, or a scan up to eight single ended or differential samples one at a time in sequential scan mode. The operation of the module in sequential scan mode is shown in **Figure 2-2**.

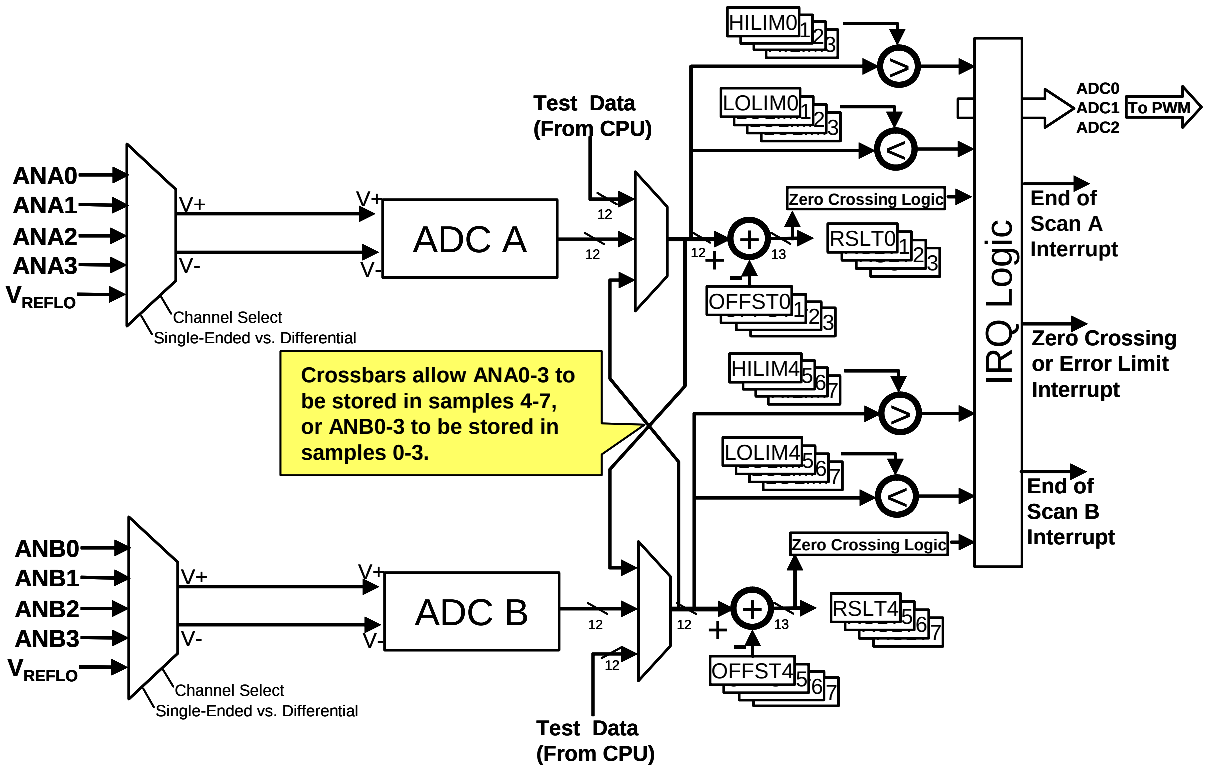


Figure 2-2. Sequential Mode Operation of the ADC

Scan sequence is determined by defining eight sample slots in CLST1/2 registers, processed in order SAMPLE0-7 during sequential scan or in order SAMPLE0-3 by Converter A and in order SAMPLE4-7 by Converter B in parallel Scan. SAMPLE slots may be disabled using the SDIS register.

The following pairs of analog inputs can be configured as a differential pair ANA0-1, ANA2-3, ANB0-1, and ANB2-3. When configured as a differential pair, a reference to either member of the differential pair by a sample slot results in a differential measurement using that differential pair.

Parallel scan can be simultaneous or non-simultaneous. During Simultaneous scan, the scans in the two converters are done simultaneously always resulting in simultaneous pairs of conversions, one by Converter A and one by Converter B. The two converters share the same start, stop, sync, end-of-scan interrupt enable control, and interrupt. Scanning in both converters is terminated when either converter encounters a disabled sample. In non-simultaneous scan, the parallel scans in the two converters are achieved independently. The two converters have their own start, stop, sync, end-of-scan interrupt enable controls, and end-of-scan interrupts. Scanning in either converter terminates only when that converter encounters a disabled sample in its part of SDIS register (DS0-DS3 for A, DS4-DS7 for B).

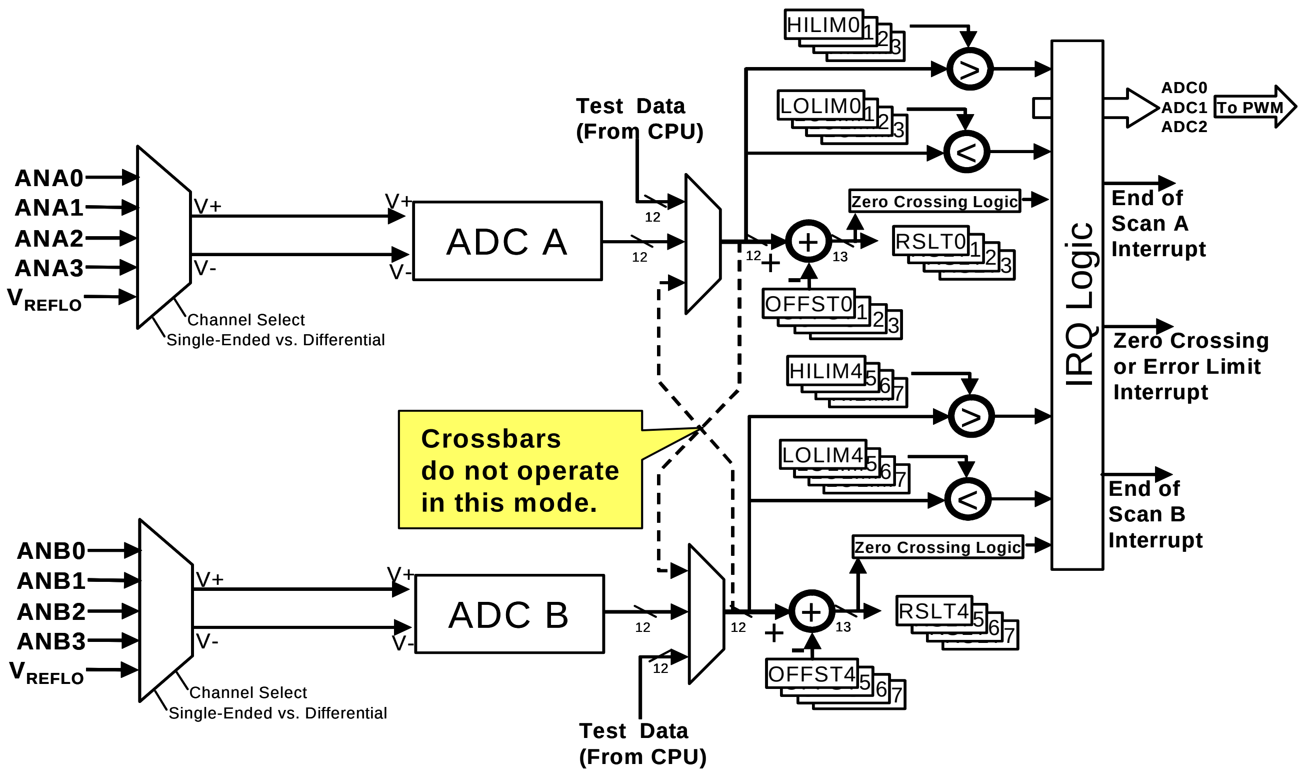


Figure 2-3. Parallel Mode Operation of the ADC

The ADC can be configured to perform a single scan and halt, perform a scan whenever triggered, or perform the scan sequence repeatedly until manually stopped. The single scan (Once mode) differs from the Triggered mode only in that SYNC input signals must be re-armed after each using a Once mode scan and subsequent SYNC inputs are ignored until the SYNC input is re-armed. This arming can occur anytime after the SYNC pulse occurs, even while the scan it initiated is still in process.

Optional interrupts can be generated at the end of a scan sequence. Interrupts are available simply to indicate the scan ended, that a sample was out of range, or at several different zero crossing conditions. *Out-of-Range* is determined by the High and Low Limit registers.

To understand the operation of the ADC it is important to understand the features and limitations of each of the functional parts.

2.4.1 Input MUX Function

The input MUX function is shown in **Figure 2-4**. The Channel Select and Single Ended vs. Differential switches are indirectly controlled based on settings within the LIST1, LIST2, SDIS registers and the CHNCFG field of the CTRL1 register.

1. MUXing for Sequential Mode, Single-Ended Conversions—During each conversion cycle (sample), any one input of the two four muxes can be directed to any RSLT*n* register.
2. MUXing for Sequential Mode, Differential Conversions—During any conversion cycle (sample), either member of a differential pair may be referenced as a SAMPLE, resulting in a differential measurement on that pair being stored in the corresponding RSLT*n* register.
3. MUXing for Parallel Mode, Single-Ended Conversions—During any conversion cycle (sample), any of ANA0-ANA3 can be directed to an RSLT0-3 Result register and any of ANB0-ANB3 can be directed to the RSLT4-7.
4. MUXing for Parallel Mode, Differential Conversions—During any conversion cycle (sample), either member of differential pair ANA0/1 or either member of differential pair ANA2/3 can be referenced as a SAMPLE, resulting in a differential measurement of that pair being stored in one of the RSLT0-3 registers. Likewise either member of differential pair ANB0/1 or either member of differential pair ANB2/3 can be referenced as a SAMPLE, resulting in a differential measurement of that pair being stored in one of the RSLT4-7 registers.

Details of switch operation is shown in **Table 2-1**. Internally, all measurements are performed differentially. During single ended measurements, V_{REFLO} is used as the negative (-) input voltage while the selected analog input is used as the positive (+) input.

Table 2-1. Analog MUX Controls for Each Conversion Mode

Conversion Mode	Channel Select Switches	Single Ended Differential Switches
Sequential, Single Ended	The two 1-of-4 select muxes can be set for the appropriate input line.	The lower switch selects Vreflo for the V- input of the A/D. The upper switch is always closed so that any of the four inputs can get to the V+ A/D input.
Sequential, Differential	The channel select switches are turned on in pairs, providing a dual 1-of-2 select function, such that either of the two differential channels can be routed to the A/D input.	The upper switch is open and the bottom switch selects the differential channel for the V- input of the A/D.
Parallel, Single Ended	The two 1-of-4 select muxes can be set for the appropriate input line.	The lower switch is selects Vreflo for the V- input of the A/D. The upper switch is always closed so that any of the four inputs can get to the V+ A/D input.
Parallel, Differential	The channel select switches are turned on in pairs, providing a dual 1-of-2 select function, such that either of the two differential channels can be routed to the A/D input.	The upper and lower switches are open and the middle switch is closed, providing the differential channel to the differential input of the A/D.

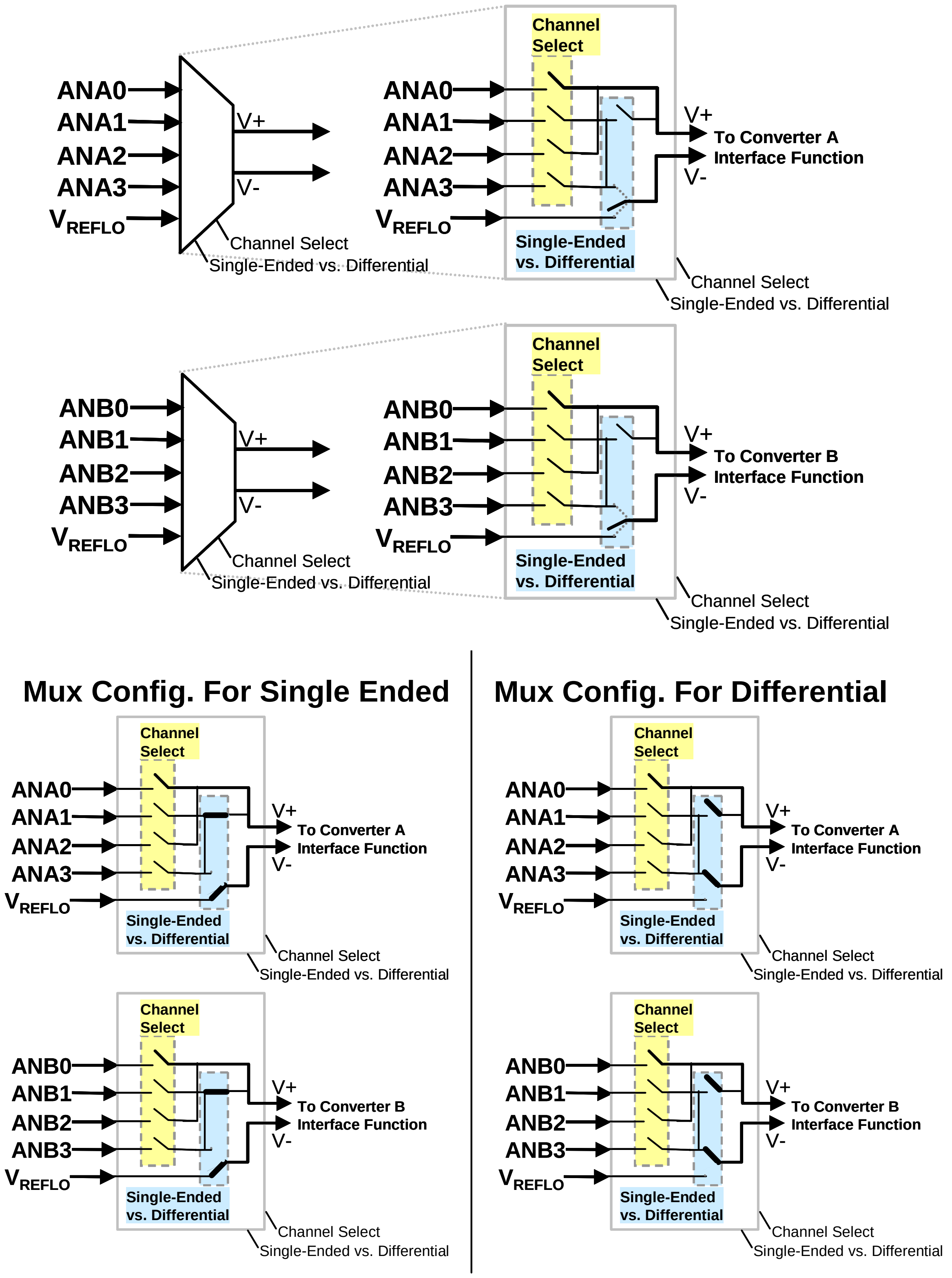

ANA0
ANA1
ANA2
ANA3
V_{REFLO}
V+
V-
Channel Select
Single-Ended vs. Differential
Channel Select
Single-Ended vs. Differential
To Converter A Interface Function
ANB0
ANB1
ANB2
ANB3
To Converter B Interface Function
Mux Config. For Single Ended
Mux Config. For Differential

Figure 2-4. Input Select Mux

2.4.2 ADC Sample Conversion

The ADC consists of a cyclic, algorithmic architecture using two recursive sub-ranging sections (RSD#1 and RSD#2), shown in **Figure 2-5.** Each sub-ranging section resolves a single bit for each conversion clock, resulting in an overall conversion rate of two bits per clock cycle. Each sub-ranging section is designed to run at a maximum clock speed of 5.33MHz. Thus a complete 12-bit conversion takes six ADC clocks (1.125ms), not including sample or post processing time.

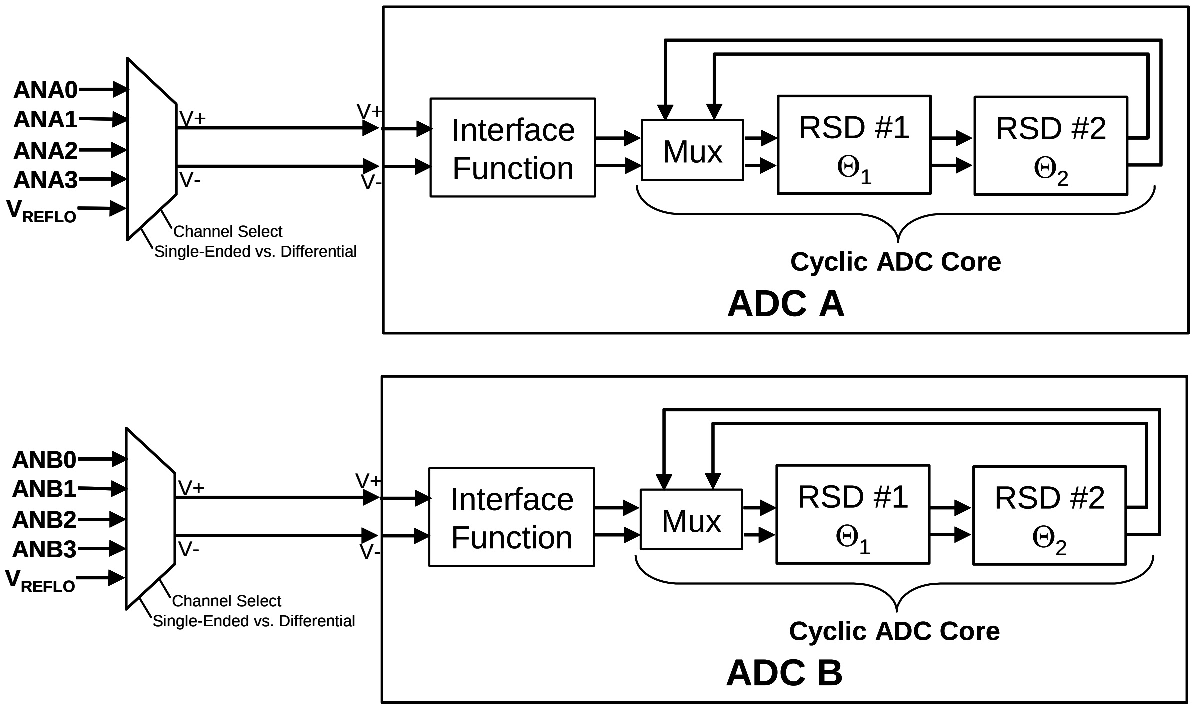


Figure 2-5. Cyclic ADC – Top Level Block Diagram

The ADC has two input modes. The input mode for a given sample is determined by the CHNCFG field of the CTRL1 register. The two input modes are:

1. Single-Ended Mode (CHNCFG bit=0)—In Single-Ended mode, input mux of the ADC selects one of the analog inputs and directs it to the plus terminal of the A/D core. The minus terminal of the A/D core is connected to the V_{REFLO} reference during this mode. The ADC measures the voltage of the selected analog input and compares it against the (V_{REFH} - V_{REFLO}) reference voltage range.
2. Differential Mode (CHNCFG bit = 1)—In Differential mode, the ADC measures the voltage difference between two analog inputs and compares that against the (V_{REFH} - V_{REFLO}) voltage range. The input is selected as an input pair: ANA0/1, ANA2/3, ANB0/1 or ANB2/3. In this mode, the plus terminal of the A/D core is connected to the even analog input while the minus terminal is connected to the odd analog input.

A mix and match combination of single-ended and differential configurations may exist. For example:

- ANA0 and ANA1 differential, ANA2 and ANA3 single-ended
- ANB0 and ANB1 differential, and ANB2 and ANB3 single-ended

2.4.2.1 Single-Ended Samples

The ADC module performs a ratio metric conversion. For single ended measurements, the digital result is proportional to the ratio of the analog input to the reference voltage in the following formula:

$$SingleEndedValue = round\left(\frac{V_{IN} - V_{REFLO}}{V_{REFH} - V_{REFLO}} \times 4095\right) \times 8$$

V_{IN} = Applied voltage at the input pin

V_{REFH} and V_{REFLO} = Voltage at the external reference pins on the device (typically $V_{REFH} = V_{SSA}$ and $V_{REFLO} = V_{DDA}$)

Note: The 12-bit result is rounded to the nearest LSB**.**

Note: The ADC is a 12-bit function with 4096 possible states. However, the 12 bits have been left shifted three bits on the 16-bit data bus so its magnitude, as read from the data bus, is now 32760**.**

2.4.2.2 Differential Samples

For differential measurements, the digital result is proportional to the ratio of the difference in the inputs to the difference in the reference voltages (V_{REFH} and V_{REFLO}). **Figure 2-6** shows typical configurations for differential inputs.

When converting differential measurements, the following formula is useful:

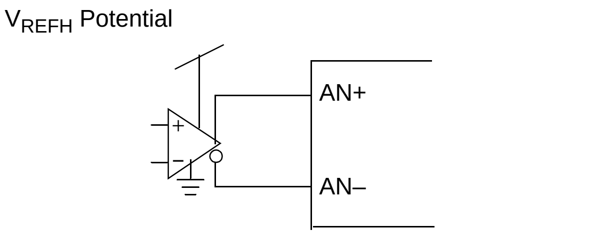
$$Differential\ Value = round\left\{\left(\frac{V_{IN+} - V_{IN-}}{V_{REFH} - V_{REFLO}} * 2048\right) + 2048\right\} * 8$$

V_{IN} = Applied voltage at the input pin

V_{REFH} and V_{REFLO} = Voltage at the external reference pins on the device (typically $V_{REFH} = V_{SSA}$ and $V_{REFLO} = V_{DDA}$)

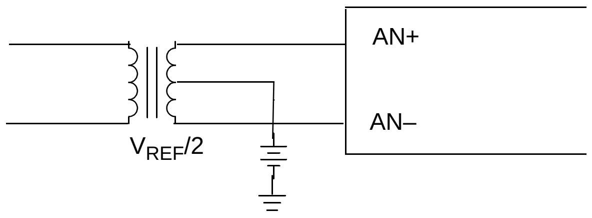
Note: The 12-bit result is rounded to the nearest LSB.

Note: The ADC is a 12-bit function with 4096 possible states. However, the 12 bits have been left shifted three bits on the 16-bit data bus so its magnitude, as read from the data bus, is now 32760.



Differential Buffer will center about mid point.

Note: Normally, V_{REFLO} is set to V_{SSA}=0V.



Center tap held at (V_{REFH} + V_{REFLO}) / 2

Figure 2-6. Typical Connections for Differential Measurements

2.4.3 ADC Data Processing

As shown in **Figure 2-7**, the raw result of the ADC conversion process is sent to an adder for offset correction. The adder subtracts the OFFST*n* register value from each sample and the result is then stored in the corresponding Result Register (RSLT*n*). Concurrent to this the raw ADC value is checked for limit violations and the RSLT*n* values are checked for zero-crossing. Appropriate interrupts are asserted, if enabled.

The sign of the result is calculated from the ADC unsigned result minus the respective offset register. If the offset register is programmed with a value of zero, the result register value is unsigned and equals the cyclic converter unsigned result. The range of the Result (RSLT) register is $0000–$7FF8, assuming the Offset (OFFST) register is set to zero.

The processor can write to the result registers whenever the ADC is in Stop mode or powered down. The data from this write operation is treated as if it came from the ADC analog core; so the limit checking, zero crossing, and the Offset registers' function as if in Normal mode. For example, if the ADC is stopped and the processor writes to RSLT5 register, the data written to the RSLT5 register is muxed to the ADC digital logic inputs, processed, and stored into RSLT5 as if the analog core had provided the data. This test data must be left justified by three bits (as shown in the RSLT register definition) and does not include the sign bit. The sign bit (SEXT) is calculated during subtraction of the corresponding OFFST*n* offset value.

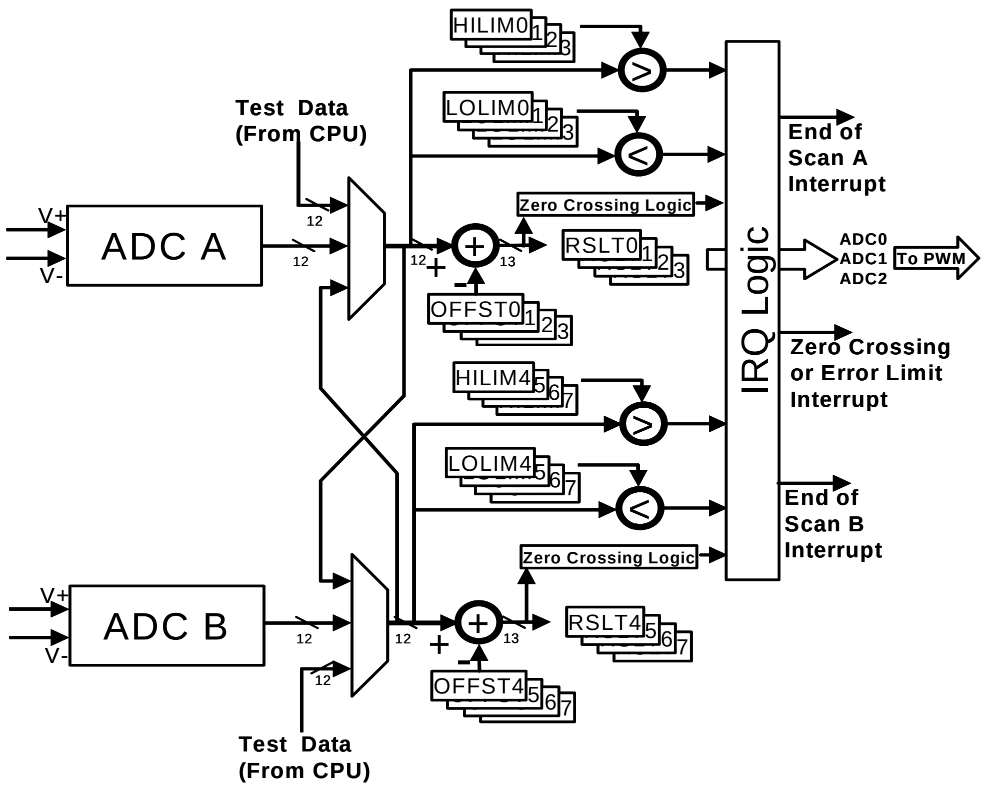


Figure 2-7. Result Register Data Manipulation

2.4.4 Sequential vs. Parallel Sampling

All scan modes make use of the eight SAMPLE slots in the CLST1 and CLST2 registers. These slots are used to define which single-ended input or differential input pair is measured at each step in a scan sequence. The SDIS register is used to disable unneeded slots.

Differential measurements are made on input pairs ANA0/1, ANA2/3, ANB0/1, and ANB2/3 using the CHNCFG field of the CTRL1 register. A single ended measurement will be made if a SAMPLE slot refers to an input not configured as a member of a differential pair by CHNCFG. A differential measurement will be made if a SAMPLE slot refers to either member of a differential pair. Refer to the CHNCFG field description in the CTRL1 register for details of differential and single ended measurement.

Scan modes are either Sequential or Parallel, as defined by the SMODE field of the CTRL1 register.   In sequential scans, up to eight SAMPLE slots are sampled one at a time in the order SAMPLE 0-7. Each SAMPLE slot may refer to any of the eight analog inputs (ANA0-3 and ANB0-3), thus the same input may be referenced by more than one SAMPLE slot. Scanning is initiated when the START0 bit is written as 1 or, if the SYNC0 bit is 1, when the SYNC0 input goes high. A scan ends when the first disabled sample slot is encountered in the SDIS register. Completion of the scan triggers the EOSI0 interrupt if the interrupt is enabled by the EOSIE0 bit. The START0 bit and SYNC0 input are ignored while a scan is in process. Scanning stops and cannot be initiated when the STOP0 bit is set.

Parallel scans differ in that Converter A collects up to four samples (SAMPLE 0-3) in parallel to Converter B collecting up to four samples (SAMPLE 4-7). SAMPLEs 0-3 may only reference inputs ANA0-3 and SAMPLEs 4-7 may only reference inputs ANB0-3. Within these constraints, any sample may reference any pin and the same input may be referenced by more than one sample slot.

By default (when SIMULT=1), Parallel scans of the converters are initiated together when the START0 bit is written as 1 or, if the SYNC0 bit is 1, when the SYNC0 input goes high. The scan in both converters terminates when either converter encounters a disabled sample slot in SDIS. Completion of a scan triggers the EOSI0 interrupt provided the EOSIE0 interrupt enable is set. Samples are always taken simultaneously in both the A and B converters. Setting the STOP0 bit stops and prevents the initiation of scanning in both converters.

Setting SIMULT=0 (non-simultaneous mode) causes parallel scanning to operate independently in the A and B converter. Each converter has its own set of START*n*, STOP*n*, SYNC*n*, and EOSIE*n* control bits, SYNC*n* input, EOSI*n* interrupt, and Conversion in Progress (CIP*n*) status indicators (*n* = 0 for converter A, *n* = 1 for converter B). Though still operating in parallel, the scans in the A and B converter start and stop independently according to their own controls and may be simultaneous, phase shifted, or asynchronous depending on when scans are initiated on the respective converters. The A and B converter may be of different length (still up to a maximum of four) and each converter’s scan completes when a disabled sample is encountered in that converters sample list only. STOP0 only stops the A converter and STOP1 only stops the B converter. Looping scan modes repeat independently, with the A converter capturing SAMPLE 0-3 and B converter capturing SAMPLE 4-7. In Loop modes, each converter independently restarts its scan after capturing its samples.

2.4.5 Scan Sequencing

Scan modes break down into three types based on how they repeat. These types are Once, Triggered, or Loop. Be certain to read **Section 2.4.4** to understand the operation of sequential and parallel scan modes before proceeding further.

During a Once mode scan a single sequential or parallel scan is executed. Once scan modes differ from triggered scan modes in that they must be re-armed after each use. While all scan modes ignore sync pulses occurring while a scan is in process, Once scan modes will continue to ignore sync pulses even after the scan completes until re-armed. Re-arming, however, can occur any time including during the scan by writing to a CTRL*n* register. If operating in a Sequential or Simultaneous Parallel mode write to the CTRL1 register. If operating in a non-simultaneous Parallel mode, re-arm Converter A by writing to the CTRL1 register and Converter B by writing to the CTRL2 register.

Triggered scan modes are identical to the corresponding Once scan modes except that re-arming of sync inputs is not necessary.

Loop scan modes automatically restart a scan as soon as the previous scan completes. In the Loop Sequential mode up to eight samples are captured in each loop and the next scan starts immediately after the completion of the previous scan. In Loop Parallel scan modes, both Converters restart together if SIMULT=1 and restart independently if SIMULT=0. All subsequent start and sync pulses will be ignored after the scan begins. Scanning can only be terminated by setting a STOP*n* bit. Use STOP0 in the CTRL1 register if operating in a Sequential or simultaneous Parallel mode. If operating in a non-simultaneous Parallel mode use STOP0 to stop Converter A and STOP1 in the CTRL2 register to stop Converter B.

2.4.6 Power Management

The simplest power management technique is to turn off ADCs that are not being used. This is described in **Section 2.4.6.1** below. The five supported ADC power modes are described in **Section 2.4.6.2**. These modes are presented in order from highest to lowest power utilization and lowest to highest conversion latency and/or startup delay. It is recommended that changes to the SIM and OCCS registers affecting power modes be made while the ADC is powered down (PD0=1, PD1=1 in PWR register). Please see the Clocks section (**Figure 2.4.7**) for details of the various clocks referenced below.

2.4.6.1 Manual Power Down of Unused Converters

If the Channel List Registers (see **Section 2.4.5**) do not use input pins ANB0-ANB3 then ADC B is never used. (See **Figure 2-2**) Similarly, if ANA0-ANA3 are not used then ADC A is never used. In such cases the unused ADC can be manually powered down using the PD0 or PD1 bit in the Power Control Register (see **Section 2.5.13**).

Since the default value of the PD0 and PD1 bits is "powered down", this technique can be used in most applications by simply not powering up the unused ADC.

2.4.6.2 Power Management Mode

1. NORMAL POWER MODE
   This mode operates when:
   a. At least one ADC converter is powered up (PD0 or PD1=0 in the PWR register);
   b. Both Auto Power-Down and Auto Standby modes are disabled (APD=0, ASB=0 in ADCPOWER);
   c. The ADC’s clock is enabled (ADC=1 in the SIM module’s SIM_PCE register).

In this mode the ADC uses the Conversion Clock as the ADC clock source both when active or idle. To minimize conversion latency it is recommended the conversion clock be configured to 5.33MHz. No startup delay (defined by PUDELAY in the PWR register) is imposed.

2. AUTO POWER-DOWN MODE
   This mode operates when:
   a. At least one ADC converter is powered up (PD0 or PD1=0 in the PWR register);
   b. Auto Power-Down mode enabled (APD=1 in the PWR register);
   c. The ADC’s clock is enabled (ADC=1 in the SIM module’s SIM_PCE register).

It is recommended the conversion clock be configured at or near 5.33MHz to minimize conversion latency when active. In this mode, the ADC uses the conversion clock when active and gates off the Conversion Clock and powers down the converters when idle. A startup delay of PUDELAY ADC clocks is executed at the start of all scans, allowing the ADC to stabilize when switching to normal current mode from a completely powered off condition. This mode uses less power than Normal and more power than Auto Standby. It requires more startup latency (than Auto Standby) when leaving the idle state to start a scan (higher PUDELAY value).

3. AUTO STANDBY MODE
   This mode operates when:
   a. At least one ADC converter is powered up (PD0 or PD1=0 in the PWR register);
   b. Auto Power-Down is disabled (APD=0 in the PWR register);
   c. Auto Standby is enabled (ASB=1 in the PWR register);

d. The ADC’s clock is enabled (ADC=1 in the SIM module’s SIM_PCE register);

e. The Relaxation Oscillator is powered up and operating at 8MHz. (ROPD=ROSB=0 in OCCS register OCTRL). Even when using an external clock source (PRECS=1 in ODDS’s OCTRL register) the Relaxation Oscillator is required to generate a 8MHz reference.

In Auto Standby Mode, the ADC uses the Conversion Clock when active and the100kHz Standby Clock when idle. The standby (low current) state automatically engages when the ADC is idle. It is recommended that the conversion clock be configured at or near 5.33MHz to minimize conversion latency. The ADC will execute a startup delay of PUDELAY ADC clocks at the start of all scans, allowing the ADC to switch to the Conversion Clock and to revert from Standby to Normal Current mode.

4. STANDBY MODE
   This mode operates when:

a. At least one ADC converter is powered up (PD0 or PD1=0 in the PWR register);

b. Auto Standby mode is disabled (ASB=0 in the PWR register);

c. The ADC’s clock is enabled (ADC=1 in the SIM module’s SIM_PCE register);

d. The PLL is bypassed (ZSRC=01 in the OCCS module’s CTRL register);

e. The MSTR_OSC clock is driven from the Relaxation Oscillator (PRECS=0 in OCCS’s CTRL register) in Standby mode (ROSB=1 and ROPD=0 in OCCS’s OCTRL register) (at 400kHz).

In this configuration the ADC clock operates at 100kHz and Standby Current mode is enabled without loss of conversion accuracy. While no startup delay (PUDELAY) is imposed for each scan, the latency of a scan is increased by the 100kHz frequency of the ADC clock. This mode is not available while using an external clock source because there is no way to determine the MSTR_OSC clock is operating at the correct frequency.

Auto Power-Down and Standby modes can be used together by setting APD=1 in the above configuration. This hybrid mode converts at an ADC clock rate of 100kHz using Standby Current mode when active and gates off the ADC clock and powers down the converters when idle. A startup delay of PUDELAY ADC clock cycles execute at the start of all scans while the ADC engages the conversion clock and the ADC powers up, stabilizing in the Standby Current mode. This provides the lowest possible power configuration for ADC operation.

5. POWER-DOWN MODE
   This mode operates when:

a. Both ADC converters are powered down (PD0=PD1=1 in the PWR register);

b. The ADC’s clock is disabled (ADC=0 in the SIM module’s SIM_PCE register).

In this configuration, the clock trees to the ADC and all of its analog components are shut down and the ADC uses no power.

2.4.6.3 Power Management Details

The ADC voltage reference and converters are powered down (PDn=1 in the PWR register) on reset. Individual converters can be manually powered down when not in use (PD0=1 or PD1=1) and the voltage reference can be automatically powered down when no converter is in use (PD2=1), or manually powered up when no converters are powered (PD2=0). When the ADC voltage reference is powered down, output reference voltages are set to Low (V_{SSA}) and the ADC data output (to the PWM module) is driven low.

A delay of PUDELAY ADC clock cycles is imposed whenever PD0 or PD1 are cleared to power-up a converter and whenever the ADC goes from an idle (neither converter has a scan in process) to an active state when not operating in Normal Power mode. The ADC is active when at least one converter has a scan in process. A device recommends the use of two PUDELAY values, a large value for full power-up and a smaller value for going from standby current levels to full power-up. The following paragraphs provide an explanation of how to use PUDELAY when starting the ADC up or changing modes.

When starting up in Normal mode clear the PD0 and or PD1 bits to power-up the required converters. Poll the status bits (PSTS*n* in the PWR register) until all required converters are powered up. Following polling, start scan operations.

When starting up using Auto Standby mode, first use the Normal mode startup procedure. Before starting scan operations, set PUDELAY to the smaller value, and then set ASB in the PWR register. Auto Standby mode will automatically reduce current levels until active and then impose a PUDELAY wait to allow current levels to rise from standby to normal levels.

When starting up using Auto Power-Down mode, first use the Normal mode startup procedure. Before starting scan operations, set PUDELAY to the large power-up value. Next, set APD in the PWR register. Finally, clear the PD0 and or PD1 bits for the required converters. Converters remain powered off until scanning goes active at which time the large PUDELAY executes as the ADC goes from powered down to fully powered at the start of the scan.

In Auto Power-Down mode, when the ADC goes from idle to active, a converter is only powered up if it is required for the scan, as determined by the CLST1, CLST2, and SDIS registers.

It is recommended to power-off both converters (PD0=PD1=1 in the PWR register) when re-configuring clocking or power controls to avoid generating bad samples, ensuring proper delays are applied when powering up or starting scans.

Attempts to start a scan during the PUDELAY time-out will be ignored until the appropriate PSTS*n* bits are cleared in the PWR register.

Any attempt to use a converter when powered down, or with the voltage reference disabled, results in invalid results. It is possible to read ADC result registers after converter power down to see results calculated before power-down. However, a new scan sequence must be started with a SYNC*n* pulse or a write to the START*n* bit before new results will be available.

2.4.6.4 ADC STOP Mode of Operation

Any conversion sequence in progress can be stopped by setting the relevant STOP*n* bit. Any further sync pulses, or writes to the START*n* bit, are ignored until the STOP*n* bit is cleared. Once in this stop mode, the results registers can be modified by writes from the processor. Any write to RSLT*n* registers in the ADC Stop mode is treated as if the analog core supplied the data, so limit checking, zero crossing, and associated interrupts can occur if enabled.

2.4.7 ADC Clock

2.4.7.1 General

The ADC has two external clock inputs used to drive two clock domains within the ADC module.

Table 2-2. ADC Clock Summary

Clock input	Source	Characteristics
Peripheral Clock (=System Clock)	OCCS by way of the SIM	Max rate is 32Mhz. When PLL is on and selected, it is PLL output divided by 6. When PLL is not selected, it is MSTR_OSC/2
ADC 8MHz Clock	Relaxation Oscillator	Provides 8Mhz for auto standby power saving mode.

2.4.7.2 Description of Clock Operation

The Peripheral Clock rate is determined by the OCCS module configuration. One of two sources (an external clock pin, or the Relaxation Oscillator) can be selected via PRECS in OCCS's CTRL register to generate the Peripheral Clock. The maximum rate of the Peripheral Clock to the ADC is therefore 32MHz. The ADC is clocked only when the ADC bit in the SIM_PCE register is set. The ADC bit must be enabled while the ADC is in use.

As shown in **Figure 2-8**, the Conversion Clock is the primary source for the ADC clock and is always selected as the ADC clock when conversions are in process. The clock source controls in the OCCS (PRECS, ROPD, ROSB), and DIV value in the CTRL2 register should be configured so the Conversion Clock frequency falls between 100kHz and 5.33MHz. Operating the ADC at out-of-spec clock frequencies will degrade conversion accuracy. Similarly, modifying the parameters affect clock rates or power modes while the regulators are powered up (PD0=0 or PD1=0) will also degrade conversion accuracy.

The Conversion Clock ADC uses for sampling is calculated using the IPBus Clock and the clock divisor bits within the CTRL2 register. Please see **Section 2.5.1** or **Section 2.5.2.** The ADC clock is active 100 percent of the time while in Loop modes, or if power management is set to Normal. It is also active during all ADC power-up for a period of time determined by the PUDELAY field in the Power (PWR) Register. After the power-up delay times out, the ADC clock continues until the completion of the ADC*n* scan when operating in Auto Standby or Auto Power-Down modes.

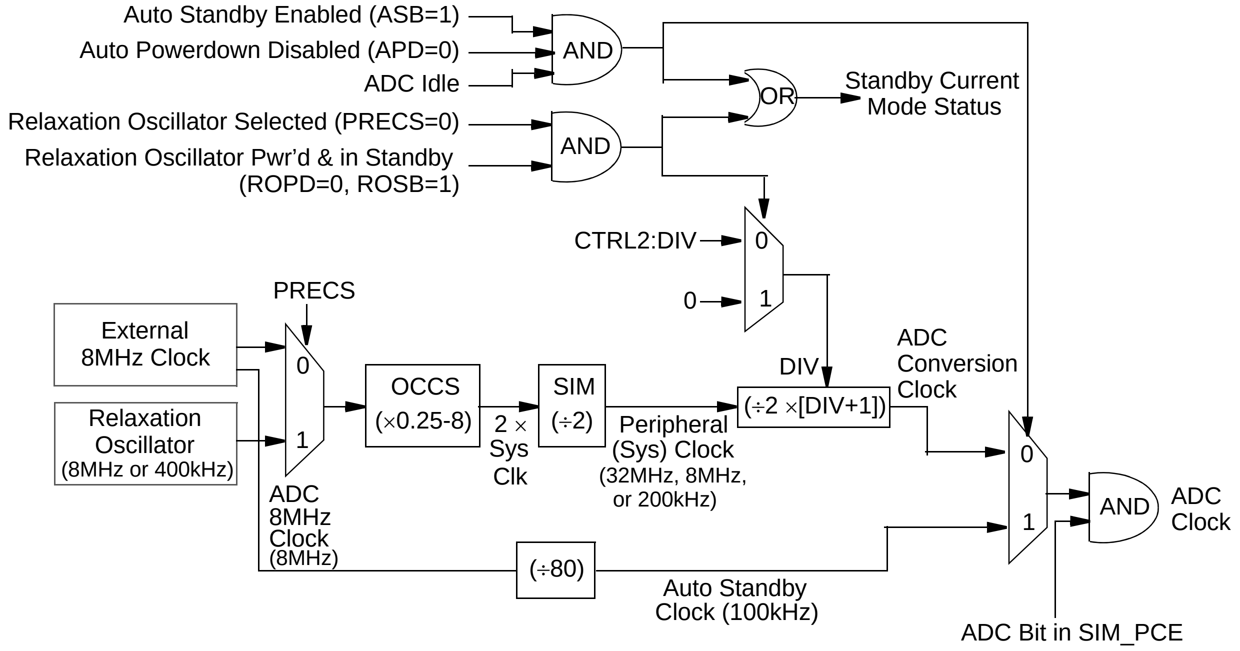

Auto Standby Enabled (ASB=1)
Auto Powerdown Disabled (APD=0)
ADC Idle
AND
OR
Standby Current Mode Status
Relaxation Oscillator Selected (PRECS=0)
Relaxation Oscillator Pwr'd & in Standby (ROPD=0, ROSB=1)
AND
CTRL2:DIV
0
0
1
PRECS
External 8MHz Clock
Relaxation Oscillator (8MHz or 400kHz)
0
1
OCCS (×0.25-8)
2 × Sys Clk
SIM (÷2)
Peripheral (Sys) Clock (32MHz, 8MHz, or 200kHz)
DIV
(÷2 ×[DIV+1])
ADC Conversion Clock
0
1
AND
ADC Clock
ADC 8MHz Clock (8MHz)
(÷80)
Auto Standby Clock (100kHz)
ADC Bit in SIM_PCE

Figure 2-8. ADC Clock Generation

The ADC 8MHz Clock feeds a 80:1 divider, generating the Auto Standby clock. The Auto Standby clock is selected as the ADC clock during the Auto Standby Power mode when both converters are idle. The Auto Standby Power mode requires the Relaxation Oscillator to be powered and Normal mode (ROPD ROSB=0 in OCCS's OCTRL register).

2.4.7.3 ADC Clock Resynchronization at Start of Scan

At the fastest ADC speed, each ADC clock period is 6 System Clock periods long. When asserting the start of a scan, either by writing to a START*n* bit or by a SYNC*n* signal, the ADC clock is re-synchronized to align it to the system clock. This allows the commanded scan to begin as soon as possible rather than wait up to five additional system clocks for the start of the next ADC clock period. This is shown in **Figure 2-9** for both Sequential and Simultaneous Parallel modes of operation. In these modes both ADC operate off of the same start signal.

In a Parallel scan mode when SIMULT=0 both ADCs operate using independent START*n* bits and SYNC*n* signals. As shown in **Figure 2-10**, the first scan started will be re-synchronized to the system clock but the second scan may wait up to five additional system clocks before starting. Also, please note that which converter is synchronized to the system clock depends on which convert first starts to use the ADC. The case shown has ADCA synchronized, but one could easily imagine the case where the ADCA start comes after instead of before the ADCB start. In this case ADCAs start would be delayed up to five additional system clock periods instead of ADCBs.

If there is a known timing relationship between ADCA and ADCB when operating in a non-simultaneous Parallel mode then the application can control which ADC starts first and gets the re-synchronized clock. The application can also control the delay to starting the second ADC scan so that its start signal aligns with the ADC clock and the start of the second ADC is not delayed.

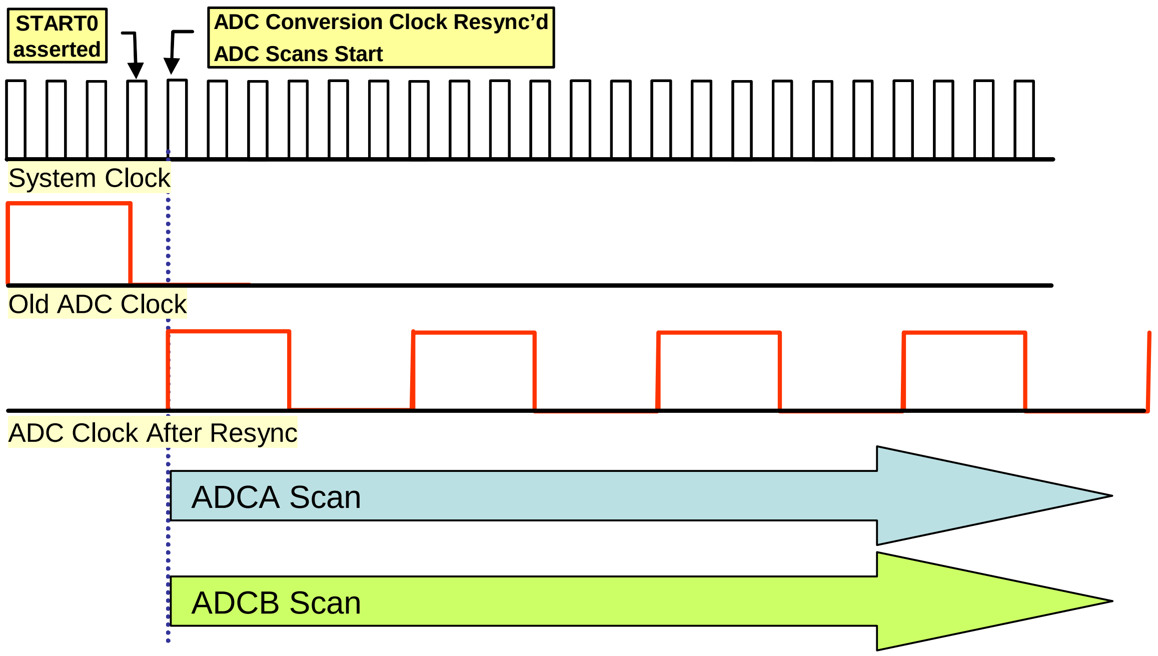


Figure 2-9. ADC Clock Resynchronization for Sequential and Simultaneous Parallel Modes

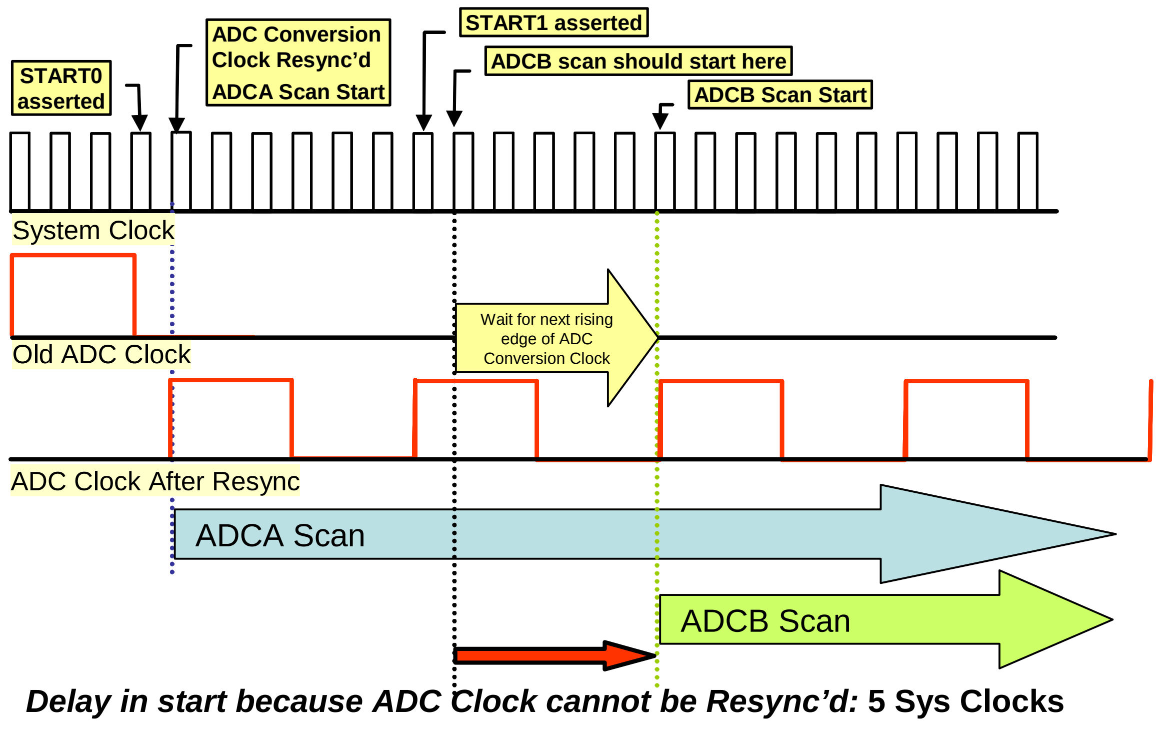

START0 asserted
ADC Conversion Clock Resync'd
ADCA Scan Start
START1 asserted
ADCB scan should start here
ADCB Scan Start
System Clock
Old ADC Clock
Wait for next rising edge of ADC Conversion Clock
ADC Clock After Resync
ADCA Scan
ADCB Scan
Delay in start because ADC Clock cannot be Resync'd: 5 Sys Clocks

Figure 2-10. ADC Clock Resynchronization for Non-Simultaneous Parallel Modes

2.4.8 Voltage Reference Pins V_{REFH} & V_{REFLO}

The voltage difference between V_{REFH} and V_{REFLO} provides the reference voltage all analog inputs are measured against. V_{REFH} is nominally set to V_{DDA}. V_{REFLO} is nominally set to V_{SSA}. An external reference voltage should be provided from a low noise filtered source capable of providing up to 1mA of reference current.

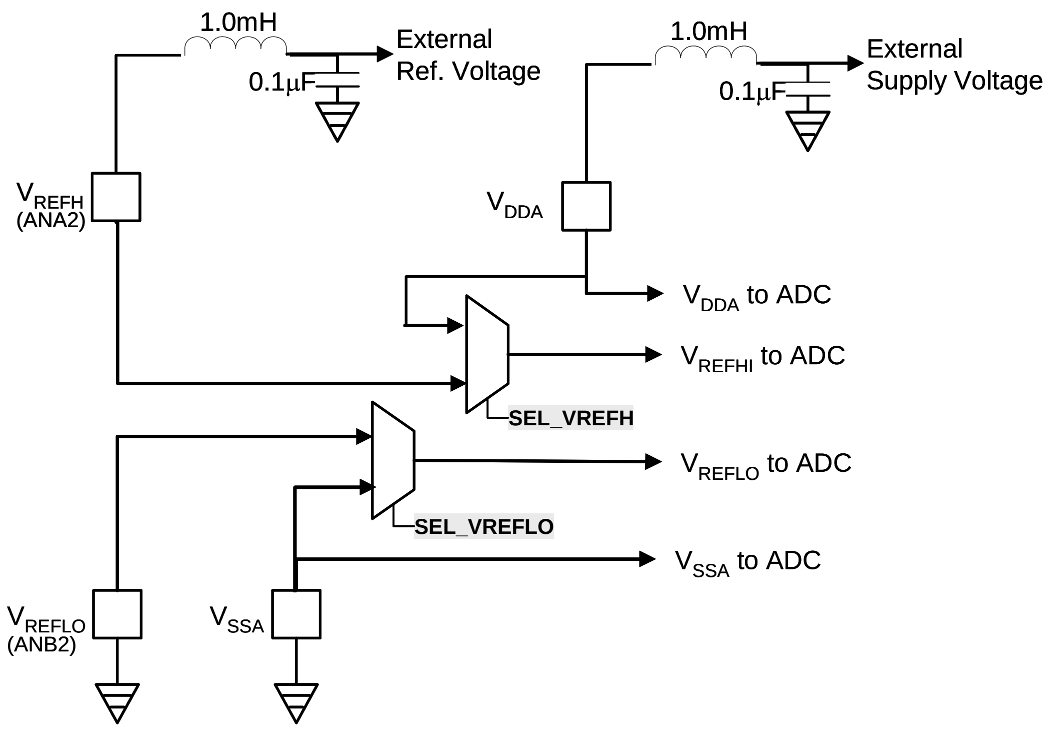


Figure 2-11. ADC Voltage Reference Circuit

When tying V_{REFH} to the same potential as V_{DDA} relative measurements are being made with respect to the amplitude of V_{DDA}. It is imperative special precautions be taken assuring the voltage applied to V_{REFH} be as noise free as possible. Any noise residing on the V_{REFH} voltage is directly transferred to the digital result.

Figure 2-11 illustrates the internal workings of the ADC voltage reference circuit.V_{REFH} must be noise filtered; a minimum configuration is shown in the figure.

2.4.9 Supply Pins V_{DDA} and V_{SSA}

Dedicated power supply pins are provided for the purposes of reducing noise coupling and to improve accuracy. The power provided to these pins is suggested to come from a low noise filtered source. Uncoupling capacitors ought to be connected between V_{DDA} and V_{SSA}.

2.5 Register Definitions

Table 2-3. ADC Memory Map

Device	Peripheral	Base Address
56F801X	ADC	$00F080

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 2-4 lists the ADC registers in ascending address order, including their acronyms and address offset of each register. ADC uses ADC*n*_BASE plus the given offset depending on the ADC being used. The ADC peripheral has 43 registers.

Table 2-4. ADC Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	CTRL1	Control Register 1	Read/Write	**Section 2.5.1**
Base + $1	CTRL2	Control Register 2	Read/Write	**Section 2.5.2** **Section 2.5.3**
Base + $2	ZXCTRL	Zero Crossing Control Register	Read/Write	**Section 2.5.4**
Base + $3	CLST1	Channel List Register 1	Read/Write	**Section 2.5.5**
Base + $4	CLST2	Channel List Register 2	Read/Write	
Base + $5	CSDIS	Sample Disable Register	Read/Write	**Section 2.5.6**
Base + $6	CSTAT	Status Register	Read/Write	**Section 2.5.7**
Base + $7	LIMSTAT	Limit Status Register	Read/Write	**Section 2.5.8**
Base + $8	ZXSTAT	Zero Crossing Status Register	Read/Write	**Section 2.5.9**
Base + $9–10	RSLT0-7	Result Registers 0-7	Read/Write	**Section 2.5.10**
Base + $11–18	LOLIMT0-7	Low Limit Registers 0-7	Read/Write	**Section 2.5.11**
Base + $19–20	HILIMT0-7	High Limit Registers 0-7	Read/Write	
Base + $21–28	OFFST0-7	Offset Registers 0-7	Read/Write	**Section 2.5.12**
Base + $29	PWR	Power Control Register	Read/Write	**Section 2.5.13**
Base + $2A	VREF	Voltage Reference Register	Read/Write	**Section 2.5.14**

Bits of each of the 43 registers are summarized in **Figure 2-12.** Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	CTRL1	R	0	STOP0	0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				0	SMODE		
		W	▨	STOP0	START0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				▨	SMODE		
$1	CTRL2 Sequential Mode	R	0	0	0	0	0	0	0	0	0	0	0	DIV				
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	DIV				
	CTRL2 Parallel Mode	R	0	STOP1	0	SYNC1	EOSIE1	0	0	0	0	0	SIMULT	DIV				
		W	▨	STOP1	START1	SYNC1	EOSIE1	▨	▨	▨	▨	▨	SIMULT	DIV				
$2	ZXCTRL	R	ZCE7		ZCE6		ZCE5		ZCE4		ZCE3		ZCE2		ZCE1		ZCE0	
		W	ZCE7		ZCE6		ZCE5		ZCE4		ZCE3		ZCE2		ZCE1		ZCE0	
$3	CLST1	R	0	SAMPLE3			0	SAMPLE2			0	SAMPLE1			0	SAMPLE0		
		W	▨	SAMPLE3			▨	SAMPLE2			▨	SAMPLE1			▨	SAMPLE0		
$4	CLST2	R	0	SAMPLE7			0	SAMPLE6			0	SAMPLE5			0	SAMPLE4		
		W	▨	SAMPLE7			▨	SAMPLE6			▨	SAMPLE5			▨	SAMPLE4		
$5	SDIS	R	0	0	0	0	0	0	0	0	DS7	DS6	DS5	DS4	DS3	DS2	DS1	DS0
		W	▨	▨	▨	▨	▨	▨	▨	▨	DS7	DS6	DS5	DS4	DS3	DS2	DS1	DS0
$6	STAT	R	CIP0	CIP1	0	EOSI1	EOSI0	ZCI	LLMTI	HLMT	RDY7	RDY6	RDY5	RDY4	RDY3	RDY2	RDY1	RDY0
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$7	LIMSTAT	R	HLS7	HLS6	HLS5	HLS4	HLS3	HLS2	HLS1	HLS0	LLS7	LLS6	LLS5	LLS4	LLS3	LLS2	LLS1	LLS0
		W	HLS7	HLS6	HLS5	HLS4	HLS3	HLS2	HLS1	HLS0	LLS7	LLS6	LLS5	LLS4	LLS3	LLS2	LLS1	LLS0
$8	ZXSTAT	R	0	0	0	0	0	0	0	0	ZCS7	ZCS6	ZCS5	ZCS4	ZCS3	ZCS2	ZCS1	ZCS0
		W	▨	▨	▨	▨	▨	▨	▨	▨	ZCS7	ZCS6	ZCS5	ZCS4	ZCS3	ZCS2	ZCS1	ZCS0
$9-$10	RSLT*n*	R	SEXT	RSLT												0	0	0
		W	SEXT	TEST_DATA												▨	▨	▨
$11-$18	LOLIM*n*	R	0	LLMT												0	0	0
		W	▨	LLMT												▨	▨	▨
$19-$20	HILIM*n*	R	0	HLMT												0	0	0
		W	▨	HLMT												▨	▨	▨
$21-$28	OFFST*n*	R	0	OFFSET												0	0	0
		W	▨	OFFSET												▨	▨	▨
$29	PWR	R	ASB	0	0	PSTS2	PSTS1	PSTS0	PUDELAY						APD	PD2	PD1	PD0
		W	ASB	▨	▨	▨	▨	▨	PUDELAY						APD	PD2	PD1	PD0
$2A	VREF	R	SEL_VREFH	SEL_VREFLO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		W	SEL_VREFH	SEL_VREFLO	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨

R	0	Read as 0
W	▨	Reserved

Figure 2-12. ADC Register Map Summary

2.5.1 Control 1 Register (CTRL1)

Bits 14, 13, 12, and 11 in CTRL1 control all types of scans except parallel scans in the B converter when SIMULT=0 in the CTRL2 register. SIMULT=0 bits 14, 13, 12, and 11 in CTRL are used to control Converter B scans in parallel scan modes while the equivalent bits in CR1 are used for Converter A.

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	STOP0	0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				0	SMODE		
Write			START0													
Reset	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1

Figure 2-13. Control 1 (CTRL1) Register

2.5.1.1 Reserved—Bit 15

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.1.2 STOP 0 (STOP0)—Bit 14

When STOP0 is asserted, the current scan is stopped and no further scans can start. Any further SYNC0 input pulses (see SYNC0 bit 12) or writes to the START0 bit are ignored until the STOP0 bit is cleared. After the ADC is in Stop mode, the result registers can be modified by the processor. Any changes to the result registers in Stop mode are treated as if the analog core supplied the data. Therefore, limit checking, zero crossing, and associated interrupts can occur if enabled. *This is not the same as the device's STOP mode.*

- 0 = Normal operation
- 1 = Stop mode

2.5.1.3 Start Conversion (START0)—Bit 13

A scan is started by writing 1 to the START0 bit. This is a *write-only* bit. Writing 1 to the START0 bit again will be ignored until the end of the current scan.

- 0 = No action
- 1 = Start command is issued

The ADC must be in a stable power configuration prior to writing the START bit. Refer to the functional description of power modes for further details.

2.5.1.4 Synchronization 0 Enable (SYNC0)—Bit 12

A conversion may be initiated by asserting a positive edge on the SYNC0 input. Any subsequent SYNC0 input pulses while the scan remains in process are ignored.

- 0 = Scan is initiated by a write to START0 bit only
- 1 = Use a SYNC0 input pulse or START0 bit to initiate a scan

The ADC must be in a stable power mode prior to SYNC0 input assertion. Refer to the functional description of power modes for further details.

In *OnCE* scan modes, only the first SYNC0 input pulse is honored. Subsequent SYNC0 input pulses are ignored until SYNC0 input is re-armed by writing to the CTRL1 register, usually by simply rewriting 1 to SYNC0. This is achieved at any time, even during the execution of the scan.

2.5.1.5 End Of Scan Interrupt Enable 0 (EOSIE0)—Bit 11

This bit enables an EOSI0 interrupt to be generated upon completion of the scan. For looping scan modes, the interrupt will trigger after the completion of each iteration of the loop.

- 0 = Interrupt disabled
- 1 = Interrupt enabled

2.5.1.6 Zero Crossing Interrupt Enable (ZCIE)—Bit 10

This bit enables the zero crossing interrupt if the current result value has a sign change from the previous result as configured by the ZXCTRL register.

- 0 = Interrupt disabled
- 1 = Interrupt enabled

2.5.1.7 Low Limit Interrupt Enable (LLMTIE)—Bit 9

This bit enables the Low Limit exceeded interrupt when the current result value is less than the Low Limit register value. The raw result value is compared to the LOLIM register, bits LLMT[11:0], before the Offset register value is subtracted.

- 0 = Interrupt disabled
- 1 = Interrupt enabled

2.5.1.8 High Limit Interrupt Enable (HLMTIE)—Bit 8

This bit enables the High Limit exceeded interrupt if the current result value is greater than the High Limit register value. The raw result value is compared to the High Limit (HILIM) register, bits HLMT[11:0], before the Offset register value is subtracted.

- 0 = Interrupt disabled
- 1 = Interrupt enabled

2.5.1.9 Channel Configure (CHNCFG)—Bits 7–4

The inputs can be configured for either single-ended or differential conversions.

Table 2-5. CHNCFG Bit Settings

Bit Settings	Inputs	Description
xxx1	ANA0–ANA1	Configured as differential pair (ANA0 is + and ANA1 is –)
xxx0		Both configured as single ended inputs
xx1x	ANA2–ANA3	Configured as differential pair (ANA2 is + and ANA3 is –)
xx0x		Both configured as single inputs
x1xx	ANB0–ANB1	Configured as differential pair (ANB0 is + and ANB1 is –)
x0xx		Both configured as single ended inputs
1xxx	ANB2–ANB3	Configured as differential pair (ANB2 is + and ANB3 is –)
0xxx		Both configured as single ended inputs

Differential measurements return the max value 32760 (= 4095 × 8) when the plus (+) input is V_{REFH} and the minus (–) input is V_{REFLO}, return 0 when the plus (+) input is at V_{REFLO} and the minus (–) input is at V_{REFLO}, and scale linearly between based on the voltage difference between the two signals. Single ended measurements return the max value 32760 when the input is at V_{REFH}, return 0 when the input is at V_{REFLO}, and scale linearly between based on the amount by which the input exceeds V_{REFLO}.

2.5.1.10 Scan Mode Control (SMODE)—Bits 2-0

SMODE controls the Scan mode of the ADC module. All scan modes make use of the eight sample slots defined by the CLST1 and CLST2 registers. A scan is the process of stepping through these sample slots, converting the analog input indicated by that slot, and storing the result. Un-required slots may be disabled by writing 1 to the appropriate bits of the SDIS register.

Input pairs ANA0-1, ANA2-3, ANB0-1, and ANB0-1 may be configured as differential pairs using the CHNCFG field. When a slot in CLST*n* refers to either member of a differential pair a differential measurement on that pair will be made, otherwise a single ended measurement will be taken on that input. The details of differential and single ended measurement are described in the description of the CHNCFG field.

SMODE determines whether the slots are used to perform a sequential scan up to eight samples or two parallel scans up to four samples. SMODE controls how these scans are initiated and terminated. It also controls whether the scans are performed once or repetitively. For more details, please see **Section 2-3.**

Parallel scans may be simultaneous (SIMULT=1) or non-simultaneous. During simultaneous parallel scans A and B converters scan synchronously using one set of shared controls (CTRL1 register). During non-simultaneous (SIMULT=0) scans the A and B converters operate asynchronously with each converter using its own independent set of controls (CTRL1 for A and CTRL2 for B). Refer to the SIMULT bit description for further details.

Note: The SIMULT bit only applies to parallel operating modes and is ignored during sequential operating modes**.**

- 000 = Once Sequential
  Upon START, or an enabled sync signal, samples are taken one at a time starting with SAMPLE0 until a first disabled sample is encountered. If no disabled sample is encountered in SDIS register, conversion concludes after SAMPLE7. If the scan is initiated by a sync signal only one scan will be completed until the converter is rearmed by writing to the CTRL1 register.
- 001 = Once Parallel
  Upon START, or an armed and enabled sync signal, converter A will capture samples 0-3 and Converter B will capture samples 4-7. By default (SIMULT=1), samples are taken simultaneously (synchronously) and scanning stops when either converter encounters a disabled sample or both converters complete all four samples. When SIMULT=0, samples are taken asynchronously and scanning stops when each converter encounters a disabled sample in its part of the SDIS register or completes all four samples. If the scan is initiated by a sync signal only one scan will be completed till the converter is rearmed by writing to the CTRL1 register. (When SIMULT=0 the B converter must be re-armed separately by writing to the CTRL2 register.)
- 010 = Loop Sequential
  Upon an initial start or enabled sync pulse, up to eight samples are taken one at a time until a disabled sample is encountered. The process repeats until the STOP0 bit is set. While a Loop mode is running, any additional start commands or sync pulses are ignored. If Auto Standby (POWER:ASB=1) or Auto Power-Down (POWER:APD=1) is the selected Power mode control, the power-up delay defined by PUDELAY will be applied only on the first conversion.

- 011= Loop Parallel
  Upon an initial start or enabled sync pulse, converter A will capture Samples0-3 and Converter B will capture Samples4-7. Each time a converter completes its current scan, it immediately restarts its scan sequence. This continues until a STOP*n* bit is asserted. While a loop is running, any additional start commands or sync pulses are ignored. By default (SIMULT=1), samples are taken simultaneously (synchronously) and scanning stops when either converter encounters a disabled sample or both converters complete all four samples. When SIMULT=0, samples are taken asynchronously and scanning stops when each converter encounters a disabled sample in its part of the SDIS register or completes all four samples. If Auto Standby or Auto Power-Down is the selected power mode control, the power-up delay defined by PUDELAY will be applied only on the first conversion.
- 100 = Triggered Sequential
  Upon START, or an enabled sync signal, samples are taken one at a time starting with SAMPLE0, until a first disabled sample is encountered. If no disabled sample is encountered, conversion concludes after SAMPLE7. If external sync is enabled new scans will be started for each sync pulse that is non-overlapping with a current scan in progress.
- 101 = Triggered Parallel (default)
  Upon START, or an enabled sync signal, Converter A will convert Samples0-3 and Converter B will convert Samples4-7 in parallel. By default (SIMULT=1), samples are taken simultaneously (synchronously) and scanning stops when either converter encounters a disabled sample or both converters complete all four samples. When SIMULT=0, samples are taken asynchronously and scanning stops when each converter encounters a disabled sample in its part of the SDIS register or completes all four samples. If external sync is enabled (SYNC0=1) new scans will be started for each sync pulse as long as the ADC has completed the previous scan (STAT:CIP*n*=0).
- 110 = Reserved use
- 111 = Reserved use

2.5.2 Control 2 Register (CTRL2) Under Sequential Scan Modes

Operating mode dependencies occur when the ADC’s scan mode (SMODE in the CTRL1 register) is set to Once Sequential, Loop Sequential, or Triggered Sequential bits 15−5 are reserved. Only the DIV field is available.

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	DIV				
Write																
Reset	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0

Figure 2-14. Control 2 (CTRL2) Register Under Sequential Scan Modes

2.5.2.1 Reserved—Bits 15–5

This bit field is reserved and should not be modified by writing.

2.5.2.2 Clock Divisor Select (DIV)—Bits 4–0

The divider circuit generates the ADC clock by dividing the system clock by $2 \times (\text{DIV}[4:0]+1)$. A DIV value must be chosen so the ADC clock does not exceed 5.33MHz. The following table shows ADC clock frequency based on the value of DIV for these various OCCS configurations.

Table 2-6. ADC Clock Frequency for Various Conversion Clock Sources

DIV	Divisor	ROSC Standby 400Khz	ROSC Normal 8Mhz	PLL 64 Mhz	External CLK
		200kHz Sys Clock	4MHz Sys Clock	32MHz Sys Clock	CLK/2 Sys Clock
0_0000	2	100K	2.00M	16.0M	CLK/4
0_0001	4	100K	1.00M	8.00M	CLK/8
0_0010	6	100K	500K	5.33M	CLK/12
0_0011	8	100K	250K	4.00M	CLK/16
0_0100	10	100K	125K	3.20M	CLK/20
—	—	—	—	—	—
—	—	—	—	—	—
1_1111	64	100K	62.5K	500K	CLK/128

2.5.3 Control 2 Register (CTRL2) Under Parallel Scan Modes

Operating mode dependencies of this register occur when the ADC's Scan mode (SMODE in the CTRL1 register) is set to Once Parallel, Loop Parallel, or Triggered Parallel bits 14–11, and 5 are no longer reserved. These bits are used to control the operation of Converter B.

By default, SIMULT=1 and Converter B operates together with Converter A. In this case bits 14, 13, 12, and 11 in CTRL2 do not affect Converter B operation. When SIMULT=0 and SMODE is a parallel scan bits 14–11 in CTRL2 along with the SYNC1 input are used to control the Converter B scan. In this case EOSIE1 enables the EOSI1 interrupt, signaling the end of a B converter scan. Also, the CIP1 bit in the STAT register is used to indicate a Converter B scan is active.

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	STOP1	0	SYNC1	EOSIE1	0	0	0	0	0	SIMULT	DIV				
Write			START1													
Reset	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0

Figure 2-15. Control 2 (CTRL2) Register Under Parallel Scan Modes

2.5.3.1 Reserved—Bit 15

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.3.2 Stop (STOP1)—Bit 14

During parallel scan modes when SIMULT=0, setting STOP1 stops parallel scans in the B Converter and prevents new ones from starting. Any further SYNC1 input pulses (please see SYNC1 bit) or writes to the START1 bit are ignored until the STOP1 bit is cleared. After the ADC is in Stop mode, the B Converter Results registers can be modified by the processor. Any changes to the Result registers in Stop mode are treated as if the analog core supplied the data. Therefore, limit checking, zero crossing, and associated interrupts can occur if enabled. *This is not the same as the device's STOP mode*.

- 0 = Normal operation
- 1 = Stop command issued

2.5.3.3 Start Conversion (START1)—Bit 13

During parallel scan modes when SIMULT=0, a B converter parallel scan is started by writing 1 to the START1 bit. This is a *write-only* bit. Writing 1 to the START1 bit again will be ignored until the end of the current scan.

- 0 = No action
- 1 = Start a B Converter parallel scan

The ADC must be in a stable power configuration prior to writing the start bit.   Refer to the functional description of power modes for further details.

2.5.3.4 SYNC1 Enable (SYNC1)—Bit 12

During parallel scan modes when SIMULT=0, setting SYNC1 to 1 permits a B Converter parallel scan to be start by asserting the SYNC1 input for at least one ADC clock cycle. Any additional SYNC1 input pulses will be ignored until the end of the scan.

- 0 = B Converter parallel scan is initiated by a write to START1 bit only
- 1 = Use a SYNC1 input pulse or START1 bit to initiate a B Converter parallel scan

The ADC must be in a stable power mode prior to SYNC1 input assertion. Please refer to the functional description of power modes for further details.

In *Once* scan modes, only a first SYNC1 input pulse is honored. Subsequent SYNC1 input pulses are ignored until the SYNC1 input is re-armed by writing to the CTRL2 Register, usually by simply rewriting 1 to SYNC1. This can be done at any time, including while the scan remains in process.

2.5.3.5 Reserved—Bits 10–6

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.3.6 End Of Scan Interrupt Enable 1 (EOSIE1)—Bit 11

During parallel scan modes when SIMULT=0, this bit enables an EOSI1 interrupt to be generated upon completion of a B Converter parallel scan. For looping Scan mode, the interrupt will trigger upon the completion of each iteration of the loop.

- 0 = Interrupt disabled
- 1 = Interrupt enabled

2.5.3.7 Simultaneous Mode (SIMULT)—Bit 5

This bit only affects parallel scan modes.

When SIMULT=1 (default value) parallel scans operate in Simultaneous mode. The scans in the A and B converter operate simultaneously and always result in pairs of simultaneous conversions in the A and B converter. START0, STOP0, SYNC0, and EOSIE0 control bits and the SYNC0 input are used to start and stop scans in both converters simultaneously. A scan ends in both converters when either converter encounters a disabled sample slot. When the parallel scan completes, the EOSI0 triggers if EOSIEN0 is set. The CIP0 status bit indicates that a parallel scan is in process.

When SIMULT=0, parallel scans in the A and B converters operate independently. The B Converter has its own independent set of the above controls (START1, STOP1, SYNC1, EOSIE1, SYNC1) designed to control its operation and report its status. Each converter′s scan continues until its sample list is exhausted (four samples) or a disabled sample its part of SDIS is encountered. For looping parallel scan mode, each converter starts its next iteration as soon as the previous iteration in that converter is complete and continues until the STOP bit for that converter is asserted.

- 0 = Parallel scans done independently
- 1 = Parallel scans done simultaneously (default)

2.5.3.8 Clock Divisor Select (DIV)—Bits 4–0

The divider circuit generates the ADC clock by dividing the system clock by 2 × (DIV[4:0]+1). A DIV value must be chosen so the ADC clock does not exceed 5.33Mhz. **Table 2-7** shows ADC clock frequency based on the value of DIV for these various OCCS configurations.

Table 2-7. ADC Clock Frequency for Various Conversion Clock Sources

DIV	Divisor	ROSC Standby 400Khz	ROSC Normal 8Mhz	PLL 64 Mhz	External CLK
		200kHz Sys Clock	4MHz Sys Clock	32MHz Sys Clock	CLK/2 Sys Clock
0_0000	2	100K	2.00M	16.0M	CLK/4
0_0001	4	100K	1.00M	8.00M	CLK/8
0_0010	6	100K	500K	5.33M	CLK/12
0_0011	8	100K	250K	4.00M	CLK/16
0_0100	10	100K	125K	3.20M	CLK/20
—	—	—	—	—	—
—	—	—	—	—	—
1_1111	64	100K	62.5K	500K	CLK/128

2.5.4 Zero Crossing Control Register (ZXCTRL)

The ADC Zero Crossing Control (ZXCTRL) register provides the ability to monitor the selected channels and determine the direction of zero crossing triggering the optional interrupt. Zero crossing logic monitors only the sign change between current and previous sample. ZCE0 bit monitors the sample stored in RSLT0, ZCE1 bit monitors RSLT1, ZCE7 bit monitors RSLT7. When the Zero Crossing is disabled for a selected result register, sign changes are not monitored or updated in the ZXSTAT register.

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ZCE7		ZCE6		ZCE5		ZCE4		ZCE3		ZCE2		ZCE1		ZCE0	
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-16. Zero Crossing Control (ZXCTRL) Register

2.5.4.1 Zero Crossing Enable *n* (ZCE*n*)—Bits 15–0

For each channel, *n*, setting the ZCE*n* field allows detection of the indicated zero crossing condition, provided the corresponding offset register (OFFST*n*) has a value *offset*, 0 < *offset* < 0x7FF8.

- 00 = Zero crossing disabled
- 01 = Zero crossing enabled for positive to negative sign change
- 10 = Zero crossing enabled for negative to positive sign change
- 11 = Zero crossing enabled for any sign change

2.5.5 Channel List 1 and 2 Registers (CLST1 and CLST2)

The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the SDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	SAMPLE3			0	SAMPLE2			0	SAMPLE1			0	SAMPLE0		
Write																
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

Figure 2-17. Channel List 1 (CLST1) Register

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	SAMPLE7			0	SAMPLE6			0	SAMPLE5			0	SAMPLE4		
Write																
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

Figure 2-18. Channel List 2 (CLST2) Register

2.5.5.1 Reserved—Bits 15, 11, 7 and 3

These bits are reserved or are not implemented. They are read as 0 and cannot be modified by writing.

2.5.5.2 SAMPLE *n* (SAMPLE4)—Bits 2, 1, and 0

The value of the SAMPLE*n* field is used to select the input channel to be sampled.

Table 2-8. ADC Input Conversion for Sample Bits

SAMPLE*n*[2:0]			ADC Input Pins Selected	
Sequential Mode	Parallel Mode			
n=0,1,2,...,7	*n*=0,1,2,3 (Conv. A)	*n*=4,5,6,7 (Conv. B)	Single Ended	Differential
000	000	▨	ANA0	ANA0+, ANA1−
001	001	▨	ANA1	
010	010	▨	ANA2	ANA2+, ANA3−
011	011	▨	ANA3	
100	▨	100	ANB0	ANB0+, ANB1−
101	▨	101	ANB1	
110	▨	110	ANB2	ANB2+, ANB3−
111	▨	111	ANB3	

In sequential modes, the sample slots are converted in order from SAMPLE0 to SAMPLE7. Analog input pins can be sampled in any order, including sampling the same input pin more than once.

In Parallel modes, Converter A processes sample slots SAMPLE0 through SAMPLE3 while Converter B processes sample slots SAMPLE4 through SAMPLE7. Since Converter A only has access to analog inputs ANA0 through ANA3, sample slots SAMPLE0-3 should only contain binary values between 000 and 011. Likewise, since Converter B only has access to analog inputs

ANB0 through ANB3, sample slots SAMPLE4-7 should only contain binary values between 100 and 111. No damage will occur if this constraint is violated but results are undefined.

When inputs are configured as differential pairs, a reference to either analog input in a differential pair by a sample slot implies a differential measurement on the pair. The details of single ended and differential measurement are described under the CHNCFG field. Sample slots are disabled using the SDIS register.

2.5.6 Sample Disable Register (SDIS)

This register is an extension to the CLST1and CLST2, providing the ability to enable only the desired samples programmed in the SAMPLE0–SAMPLE7. At reset all samples are enabled. For example, if in a Sequential mode and bit DS5 is set to 1, SAMPLE0 through SAMPLE4 are sampled. However, if Parallel mode is selected and bits DS5 or DS1 are set to 1, only SAMPLE0 and SAMPLE4 are sampled.

Base + $5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	DS7	DS6	DS5	DS4	DS3	DS2	DS1	DS0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-19. Sample Disable (SDIS) Register

2.5.6.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.6.2 Disable Sample (DS*n*)—Bits 7–0

The respective SAMPLE*n* field can be enabled or disabled where *n* = 0–7.

- 0 = Enable SAMPLE*n*
- 1 = Disable SAMPLE*n* and all subsequent samples. Which samples are actually disabled will depend on the conversion mode, sequential/parallel, and the value of SIMULT.

2.5.7 Status Register (STAT)

This register provides the current status of the ADC module. RDY*n* bits are cleared by reading their corresponding Result (RSLT*n*) registers. HLMTI and LLMTI bits are cleared by writing 1 to each asserted bit in the ADC Limit Status (LIMSTAT) register. Likewise, the ZCI bit, bit-10, is cleared by writing 1 to each asserted bit in the ADC Zero Crossing Status (ZXSTAT) register.

The EOSI*n* bits are cleared by writing 1 to them. Please see **Figure 2-19** for more information regarding the operation of interrupts.

Except for CIP0 and CIP1 all bits in the STAT Register are *sticky*. Once set to a 1 state, they require some specific action to clear them. They are not cleared automatically on the next scan sequence.

Base + $6	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Read	CIP0	CIP1	0	EOSI1	EOSI0	ZCI	LLMTI	HLMTI	RDY7	RDY6	RDY5	RDY4	RDY3	RDY2	RDY1	RDY0
Write	▨	▨	▨			▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-20. Status (STAT) Register

2.5.7.1 Conversion in Progress 0 (CIP0)—Bit 15

This bit indicates when a scan is in progress.

- 0 = Idle state
- 1 = A scan cycle is in progress. The ADC will ignore all sync pulses or start commands.

This bit supports any sequential scan or parallel scan with SIMULT=1. When executing a parallel scan with SIMULT = 0 this bit services the scan of Converter A while the CIP1 bit services the scan of Converter B.

2.5.7.2 Conversion in Progress 1 (CIP1)—Bit 14

This bit indicates when a scan is in progress.

- 0 = Idle state
- 1 = A scan cycle is in progress. The ADC will ignore all sync pulses or start commands

This refers only to a B Converter scan in non-simultaneous (SIMULT=0) parallel scan modes.

2.5.7.3 Reserved—Bit 13

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.7.4 End of Scan Interrupt 1 (EOSI1)—Bit 12

This bit indicates whether a scan of analog inputs have been completed since the last read of the STAT register or since a reset. The EOSI1 bit is cleared by writing 1 to it. This bit cannot be set by software.

- 0 = A scan cycle has not been completed, no end of scan IRQ pending
- 1 = A scan cycle has been completed, end of scan IRQ pending

In Looping Scan modes, this interrupt is triggered at the completion of each iteration of the loop. This interrupt is triggered only by the completion of a B Converter scan in non-simultaneous (SIMULT=0) Parallel Scan modes. In this case the EOSI0 interrupt is triggered when Converter A completes its scan.

2.5.7.5 End of Scan Interrupt 0 (EOSI0)—Bit 11

This bit indicates whether a scan of analog inputs has been completed since the last read of the STAT register or since a reset. The EOSI0 bit is cleared by writing 1 to it. This bit cannot be set by software. EOSI0 is the preferred bit to poll for scan completion if interrupts are not enabled.

- 0 = A scan cycle has not been completed, no end of scan IRQ pending
- 1 =A scan cycle has been completed, end of scan IRQ pending

In Looping Scan modes, this interrupt is triggered at the completion of each iteration of a loop.

This interrupt is triggered upon the completion of any sequential scan or parallel scan with SIMULT=1. When executing parallel scans with SIMULT = 0 this interrupt is triggered when Converter A completes it scan while the EOSI1interrupt services Converter B.

2.5.7.6 Zero Crossing Interrupt (ZCI)—Bit 10

This bit is asserted at the completion of an individual conversion experiencing a zero crossing enabled in ADC Zero Crossing Control (ZXCTRL) register. The bit is set as soon as an enabled zero crossing event occurs rather than at the end of the ADC scan.

The ZCI bit is cleared by writing 1 to all active ZCS[7:0] bits in the ZXSTAT register.

- 0 = No ZCI interrupt request
- 1 = Zero crossing encountered; IRQ pending if ZCIE is set

2.5.7.7 Low Limit Interrupt (LLMTI)—Bit 9

If any Low Limit (LOLIM*n*) register is enabled by having a value other than $0000, low limit checking is enabled. This bit is set at the completion of an individual conversion which may or may not be the end of a scan.

The LLMTI bit is cleared by writing 1 to all active LLS[7:0] bits in the LIMSTAT register.

- 0 = No low limit interrupt request
- 1 = Low limit exceeded, IRQ pending if LLMTIE is set

2.5.7.8 High Limit Interrupt (HLMTI)—Bit 8

If any High Limit (HILIM*n*) register is enabled by having a value other than 0x7FF8, high limit checking is enabled. This bit is set at the completion of an individual conversion which may or may not be the end of a scan.

The HLMTI bit is cleared by writing 1 to all active HLS[7:0] bits in the LIMSTAT register.

- 0 = No high limit interrupt request
- 1 = High limit exceeded, IRQ pending if HLMTIE is set

2.5.7.9 Ready Sample 7–0 (RDY*n*)—Bits 7–0

These bits indicate samples seven through zero are ready to be read. The RDY*n* bits are set as the individual channel conversions are completed and stored in a RSLT*n* register. These bits are cleared after a read from the corresponding ADC Results (RSLT*n*) Register. If polling the RDY*n* bits to determine if a particular sample is executed, care should be taken not to start a new scan until all enabled samples are completed.

Note: RDY*n* bits can be cleared when the debugger reads the corresponding Results register during a debug session.

- 0 = Sample not ready or has been read
- 1 = Sample ready to be read

Figure 2-21 illustrates how five interrupts sources are combined into three entries in the interrupt vector table.

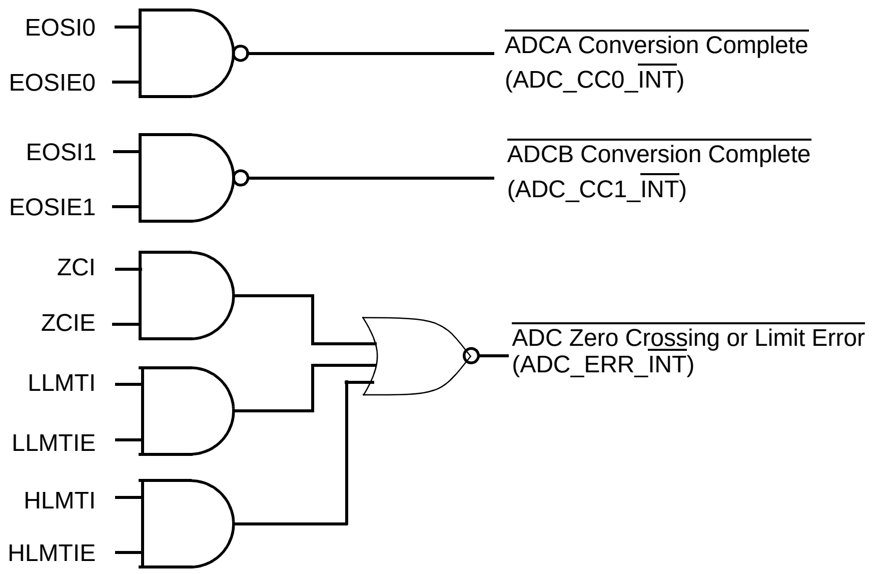

EOSI0
EOSIE0
ADCA Conversion Complete (ADC_CC0_INT)
EOSI1
EOSIE1
ADCB Conversion Complete (ADC_CC1_INT)
ZCI
ZCIE
LLMTI
LLMTIE
HLMTI
HLMTIE
ADC Zero Crossing or Limit Error (ADC_ERR_INT)

Figure 2-21. ADC Interrupt Sources

2.5.8 Limit Status Register (LIMSTAT)

The ADC Limit Status (LIMSTAT) register latches in the result of the comparison between the result of the sample in the RSLT*n* register and the respective Limit Register, HILIM*n* or LOLIM*n*.

Here is an example. If the result for RSLT0 is greater than the value programmed into the HILIM0, then set the HLS0 bit to 1. An interrupt is generated if the HLMTIE bit is set in CTRL1. These bits are *sticky*. Once set, the bits require a specific modification to clear them. They are not cleared automatically by subsequent conversions. A bit may only be cleared by writing a value of one to that specific bit.

Base + $7	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Read / **Write**	HLS7	HLS6	HLS5	HLS4	HLS3	HLS2	HLS1	HLS0	LLS7	LLS6	LLS5	LLS4	LLS3	LLS2	LLS1	LLS0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-22. Limit Status (LIMSTAT) Register

2.5.9 Zero Crossing Status Register (ZXSTAT)

The ADC Zero Crossing Status (ZXSTAT) register latches in the result of the comparison between the current result of the sample and the previous result of the same results register. For example, if the result for the channel programmed in SAMPLE0 changes sign from the previous conversion and the respective ZCE bit in ZXCTRL register is set to 11b (any edge change) then set the ZCS0 bit to 1. An interrupt is generated if the ZCIE bit is set in the CTRL1 register. These bits are *sticky*. Once set, they require a write to clear them. They are not cleared automatically by subsequent conversions. A bit may only be cleared by writing a value of 1 to that specific bit.

Base + $8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	ZCS7	ZCS6	ZCS5	ZCS4	ZCS3	ZCS2	ZCS1	ZCS0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-23. Zero Crossing Status (ZXSTAT) Register

2.5.9.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.9.2 Zero Crossing Status (ZCS[7:0])—Bits 7–0

The zero crossing condition is determined by examining the ADC value after it has been adjusted by the offset for the result register. Please see **Figure 2-7.** Each bit of the register is cleared by writing 1 to that register bit.

- 0 = a. A sign change did not occur in a comparing the current RSLT*n* value and the previous RSLT*n* value,

  or

  b. Zero crossing control is disabled for sample *n* in the ADC Zero Crossing Control (ZXCTRL) register
- 1 = In a comparison between the current channelx result and the previous channelx result, a sign change condition occurred as defined in the ADC Zero Crossing Control (ZXCTRL) register

2.5.10 Result 0-7 Registers (RSLT0–7)

The eight Result Registers contain the converted results from a scan. The SAMPLE0 result is loaded into RSLT0 Register, SAMPLE1 result in RSLT1 Register, and so on. In a simultaneous

Parallel Scan mode, the first channel pair, designated by SAMPLE0 and SAMPLE4 in register LIST1/2, is stored in RSLT0 and RSLT4, respectively.

When writing to this register, only the RSLT portion of the value written is used. This value is modified as shown in **Figure 2-7** and the result of the subtraction is stored. The SEXT bit is only set as a result of this subtraction and is not directly determined by the value written.

ADC Result Register 0 – Address:	ADC_BASE + $9
ADC Result Register 1 – Address:	ADC_BASE + $A
ADC Result Register 2 – Address:	ADC_BASE + $B
ADC Result Register 3 – Address:	ADC_BASE + $C
ADC Result Register 4 – Address:	ADC_BASE + $D
ADC Result Register 5 – Address:	ADC_BASE + $E
ADC Result Register 6 – Address:	ADC_BASE + $F
ADC Result Register 7 – Address:	ADC_BASE + $10

Base + $9-10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	SEXT	RSLT												0	0	0
Write		TEST_DATA														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-24. Result (RSLT0–7) Registers

2.5.10.1 Sign Extend (SEXT)—Bit 15

SEXT is the sign-extend bit of the result. When the SEXT bit is set to 1, it implies a negative result. When the SEXT bit is set to 0, it implies a positive result. If only positive results are required, then the respective ADC Offset (OFFST*n*) register must be set to a value of 0.

2.5.10.2 Digital Result of the Conversion (RSLT)—Bits 14–3

RSLT can be interpreted as either a signed integer or a signed fixed point (fractional) number. As a fixed point number, the RSLT can be used directly. As a signed integer, one has the option to right shift with sign extend (ASR) three places to fit it into the range [0,4095]. Or one can accept the number as presented in the register, knowing there are missing codes because the lower three LSBs are always zero.

Negative results, SEXT = 1, are always presented in twos complement format. If it is a requirement of an application, the Result registers always be positive, the Offset register (OFFST*n)* must always be set to 0.

The interpretation of the numbers programmed into the ADC Limit and Offset (LOLIM*n*, HILIM*n*, and OFFST*n*) registers should match your interpretation of the result register.

2.5.10.3 Test Data (Test_Data)—Bits 14–3

When the ADC is stopped or in Power-Down mode this field can be written by accessing the register in the memory map. Please see **Section 2.4.3** more information.

2.5.10.4 Reserved—Bits 2–0

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.11 Low and High Limit Registers (LOLIM0-7 and HILIM0-7)

Each ADC sample is compared against the values in the Limit Registers. The comparison is based upon the raw conversion value before the offset correction is applied. Refer to **Figure 2-7.** ADC Limit Registers (LOLIM*n* and HILIM*n)* correspond to Results (RSLT*n*) registers. The High Limit register is used for the comparison of *Result* > *High Limit*. The Low Limit register is used for the comparison of *Result* < *Low Limit*. The limit checking can be disabled by programming the respective limit register with 0x7FF8 for the high limit and 0x0000 for the low limit. At reset, limit checking is disabled.

ADC Low Limit Register 0 – Address:	ADC_BASE + $11
ADC Low Limit Register 1 – Address:	ADC_BASE + $12
ADC Low Limit Register 2 – Address:	ADC_BASE + $13
ADC Low Limit Register 3 – Address:	ADC_BASE + $14
ADC Low Limit Register 4 – Address:	ADC_BASE + $15
ADC Low Limit Register 5 – Address:	ADC_BASE + $16
ADC Low Limit Register 6 – Address:	ADC_BASE + $17
ADC Low Limit Register 7 – Address:	ADC_BASE + $18

Base + $11-18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	LLMT												0	0	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-25. Low Limit Register (LOLIM0–7)

ADC High Limit Register 0–Address:	ADC_BASE + $19
ADC High Limit Register 1–Address:	ADC_BASE + $1A
ADC High Limit Register 2–Address:	ADC_BASE + $1B
ADC High Limit Register 3–Address:	ADC_BASE + $1C
ADC High Limit Register 4–Address:	ADC_BASE + $1D
ADC High Limit Register 5–Address:	ADC_BASE + $1E
ADC High Limit Register 6–Address:	ADC_BASE + $1F
ADC High Limit Register 7–Address:	ADC_BASE + $20

Base + $19-20	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	HLMT												0	0	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-26. High Limit Register (HILIM0–7)

2.5.12 Offset Registers (OFFST0–7)

Value of the Offset (OFFST*n*) register is used to correct the ADC result before it is stored in the RSLT*n* registers.

ADC Offset Register 0–Address:	ADC_BASE + $21
ADC Offset Register 1–Address:	ADC_BASE + $22
ADC Offset Register 2–Address:	ADC_BASE + $23
ADC Offset Register 3–Address:	ADC_BASE + $24
ADC Offset Register 4–Address:	ADC_BASE + $25
ADC Offset Register 5–Address:	ADC_BASE + $26
ADC Offset Register 6–Address:	ADC_BASE + $27
ADC Offset Register 7–Address:	ADC_BASE + $28

Base + $21-28	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	OFFSET												0	0	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-27. Offset 0-7 (OFFST0-7) Registers

The offset value is subtracted from the ADC result. In order to obtain unsigned results, the respective Offset register should be programmed with a value of $0000, thus giving a result range of $0000 to $7FF8.

2.5.13 Power Control Register (PWR)

This register controls the power management features of the ADC module. There are manual power-down control bits for the two ADC converters and the shared voltage reference generator. There are also five distinct power modes. The following terms are used to describe power modes and their related controls.

1. Powered down state
   Each converter and the voltage reference generator can individually be put into a powered down state. When powered down, the unit consumes no power. Results of scans

referencing a powered down converter are undefined. The voltage reference generator and at least one converter must be powered up to use the ADC module.

2. Manual power-down controls
Each converter and the voltage reference generator have a manual power control bit capable of forcing that component into the power down state. Also, each converter and the voltage reference generator can be powered up/down automatically as part of ADC operation.
3. Idle state
The ADC module is idle when neither of the two converters has a scan in process.
4. Active state
The ADC module is active when at least one of the two converters has a scan in process.
5. Current mode

- Normal current mode is used to power the converters at clock rates above 100kHz.
- Standby Current mode uses less power and is engaged only when the ADC clock is at 100kHz. The current mode active does not affect the number of ADC clock cycles required to do a conversion or the accuracy of a conversion. The ADC module may change the current mode when idle as part of the power saving strategy. Both converters will be in the same current mode at all times.

In addition to the power modes, Startup delay is defined as:

- Auto Power-Down and Auto Standby Power modes cause a startup delay when the ADC module goes between the idle and active states to allow time to switch clocks or power configurations.   The number of ADC clocks used in the startup delay is defined by the PUDELAY field.

See the discussion of power modes in the Functional Description section (**Section 2-3**) for details of the five power modes and how to configure them. See the Clocks section (**Section 2.4.7**) for a more detailed description of the clocking system and the control of current mode.

Base + $29	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ASB	0	0	PSTS2	PSTS1	PSTS0	PUDELAY						APD	PD2	PD1	PD0
Write																
Reset	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	1

Figure 2-28.   Power Control (PWR) Register

2.5.13.1 Auto Standby (ASB)—Bit 15

The ASB bit selects Auto Standby mode. ASB is ignored if APD is 1. When the ADC is idle, Auto Standby mode selects the standby clock as the ADC clock source and puts the converters into Standby Current mode. At the start of any scan, the conversion clock is selected as the ADC clock and a delay of PUDELAY ADC clock cycles is imposed for current levels to stabilize. After this delay, the ADC will initiate the scan. When the ADC returns to the idle state, the standby clock is again selected and the converters revert to the standby current state.

- 0 = Auto standby mode disabled
- 1 = Auto standby mode enabled

2.5.13.2 Reserved—Bits 14–13

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

2.5.13.3 Voltage Reference Power Status 2 (PSTS2)—Bit 12

PSTS2 is a *read-only* bit. It simply reflects whether the voltage reference circuit is currently enabled.

- 0 = Voltage reference circuit is currently powered up
- 1 = Voltage reference circuit is currently powered down

2.5.13.4 Converter B Power Status 1 (PSTS1)—Bit 11

PSTS1 is a *read-only* bit. It is asserted immediately following a write of 1 to PD1. It is deasserted PUDELAY ADC clock cycles after writing 0 to PD1 if APD is 0. This bit can be read as a status bit to determine when the ADC is ready for operation. During Auto Power-Down mode, this bit indicates the current powered state of Converter B.

- 0 = ADC Converter B is currently powered up
- 1 = ADC Converter B is currently powered down

2.5.13.5 Converter A Power Status 0 (PSTS0)—Bit 10

PSTS0 is a *read-only* bit. It is asserted immediately following a write of 1 to PD0. It is deasserted PUDELAY ADC clock cycles after writing 0 to PD0 if APD is 0. This bit can be read as a status bit to determine when the ADC is ready for operation. During Auto Power-Down mode, this bit indicates the current powered state of Converter A.

- 0 = ADC Converter A is currently powered up
- 1 = ADC Converter A is currently powered down

2.5.13.6 Power-Up Delay (PUDELAY)—Bits 9–4

This 6-bit field determines the number of ADC clocks provided to power-up an ADC converter (after setting PD0 or PD1 to 0) before allowing a scan to start. It also determines the number of ADC clocks of delay provided in Auto Power-Down (APD) and Auto Standby (ASB) modes between when the ADC goes from the idle to active state and when the scan is allowed to start. The default value is 13 ADC clocks. Accuracy of the initial conversions in a scan will be degraded if PUDELAY is set to too small a value.

Note: PUDELAY defaults to a value typically sufficient for any power mode. The latency of a scan can be reduced by reducing PUDELAY to the lowest value for which accuracy is not degraded. Please refer to the *Device Data Sheet* for further details.

2.5.13.7 Auto Power-Down (APD)—Bit 3

Auto Power-Down mode powers down converters when not in use for a scan. APD takes precedence over ASB. When a scan is started in APD mode, a delay of PUDELAY ADC clock cycles is imposed during which the needed converter(s), if idle, are powered up. The ADC will then initiate a scan equivalent to when APD is not active. When the scan is completed, the converter(s) are powered down again.

- 0 = Auto Power-Down mode is not active
- 1 = Auto Power-Down mode is active

Note: If ASB or APD is asserted while a scan is in progress, that scan is unaffected and the ADC will wait to enter its low power state until after all conversions are complete and both ADCs are idle.

Note: ASB and APD are not useful in looping modes. The continuous nature of scanning means the low power state can never be entered.

2.5.13.8 Power-Down Control for Voltage Reference Circuit 2 (PD2)—Bit 2

This bit controls the power-down of the ADC's voltage reference current.

- 0 = Manually Power-Up voltage reference circuit
- 1 = Power-Down voltage reference circuit is controlled by PD0 and PD1 (default)

The voltage reference circuit is shared by both converters. When PD2=1 the voltage reference will be activated whenever PD1 or PD0 are powered up.   It is not usually necessary to modify this bit, since powering down both Converter A and Converter B will automatically power-down the voltage reference.

2.5.13.9 Manual Power-Down for Converter B (PD1)—Bit 1

This bit forces ADC Converter B to power-down.

- 0 = Power-Up ADC Converter B
- 1 = Power-Down ADC Converter B

Asserting PD1 powers down Converter B immediately. The results of a scan using Converter B will be invalid while PD1 is asserted. When PD1 is cleared, Converter B is either continuously powered up (APD = 0) or automatically powered up when needed (APD=1).

When clearing PD1 in any power mode except Auto Power-Down (APD=1), wait PUDELAY ADC clock cycles before initiating a scan to stabilize power levels within the converter. The PSTS1 bit can be polled to determine when the PUDELAY time has elapsed. Failure to follow this procedure can result in loss of accuracy of the first two samples.

2.5.13.10 Manual Power-Down for Converter A (PD0)—Bit 0

This bit forces ADC Converter A to power-down.

- 0 = Power-Up ADC Converter A
- 1 = Power-Down ADC Converter A

Asserting PD0 powers down Converter A immediately. The results of a scan using Converter A will be invalid while PD0 is asserted. When PD0 is cleared, Converter A is either continuously powered up (APD = 0) or automatically powered up when needed (APD=1).

When clearing PD0 in any power mode except Auto Power-Down (APD=1), wait PUDELAY ADC clock cycles before initiating a scan to stabilize power levels within the converter. The PSTS0 bit can be polled to determine when the PUDELAY time has elapsed. *Failure to follow this procedure can result in loss of accuracy of the first two samples.*

2.5.14 Voltage Reference Register (VREF)

In earlier series this register supported ADC calibration and had a different name. Improvements in ADC performance have eliminated the need for on-chip calibration support, hence the new name.

Base + $2A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	SEL_VREFH	SEL_VREFLO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2-29. Voltage Reference (VREF) Register

2.5.14.1 Select V_{REFH} Source (SEL_VREFH)—Bit 15

This bit selects the source of the V_{REFH} reference for conversions.

- 0 = Internal V_{DDA}
- 1 = ANA2

2.5.14.2 Select V_{REFLO} Source (SEL_VREFLO)—Bit 14

This bit selects the source of the V_{REFLO} reference for conversions.

- 0 = Internal V_{SSA}
- 1 = ANB2

2.5.14.3 Reserved—Bits 13–0

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

Chapter 3
Computer Operating Properly (COP)

Document Revision History for Chapter 3, Computer Operating Properly (COP)

Version History	Description of Change
Rev. 1	initial release
Rev. 2	**Figure 3-1**, remove the MUX in the diagram since it has only 1 input

3.1 Introduction

The Computer Operating Properly (COP) module assists software recovery from runaway code. The COP is a free-running down counter. Once enabled, it is designed to generate a reset when reaching zero. Software must periodically service the COP to clear the counter and prevent a reset.

3.2 Features

The COP module design includes these characteristics:

- Programmable time-out period = (1024 × (*n* + 1)) oscillator clock cycles, where *n* can be from $0000 to $FFFF resulting in a time-out period between 128µsec to 8.4 sec
- Programmable Wait and Stop modes
- COP timer is disabled while the device is in Debug mode, typically at a debugger break point or DEBUGHLT instruction. (For more information about the 56800E core’s debug mode, please see Chapter 11 of the 56800E 16-bit DSP core Reference Manual.)

3.3 Block Diagram

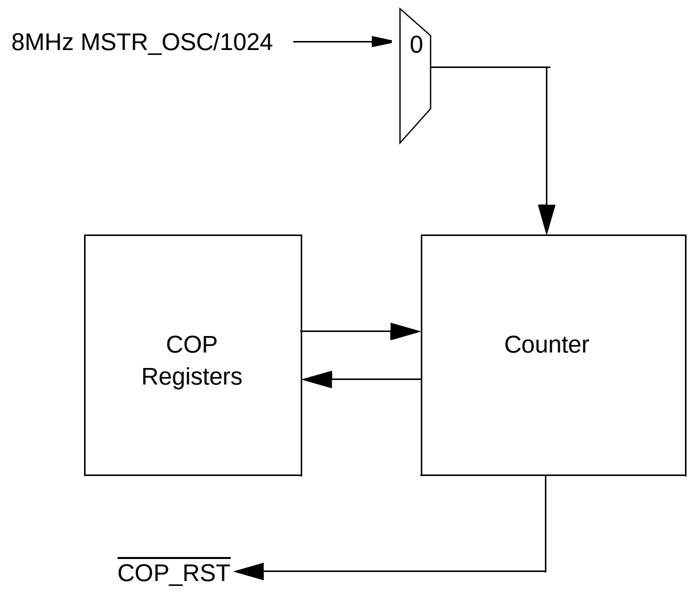


Figure 3-1. COP Module Block Diagram and Interface Signals

3.4 Functional Description

When the COP is enabled, each positive edge of the input clock causes the counter to decrease by one. If the count reaches a value of $0000, the $\overline{\text{COP\_RST}}$ signal is asserted and the chip is reset. In order for the 16-bit controller to show it is operating properly, it must perform a service routine prior to the count reaching $0000. The service routine consists of writing $5555 followed by $AAAA to COP Counter (CNTR) register.

Note: *The COP timer is stopped while the processor is in Debug mode (i.e. typically at a debugger breakpoint). If the COP is enabled, the timer will resume counting upon exiting Debug mode (typically when the debugger releases the device). The CEN bit in the COP Control (CTRL) register always reads as zero when in Debug mode, even when it has a value of one.*

3.5 Operating Modes

3.5.1 Wait Mode Operation

When entering Wait mode with both CEN and CWEN set to one, the counter continues to count down. If either CEN or CWEN is set to zero when Wait mode is entered, the counter is disabled and reloads using the value in the COP Time-Out (TOUT) register.

3.5.2 Stop Mode Operation

When entering Stop mode with both CEN and CSEN set to one, the counter continues to count down. If either CEN or CSEN is set to 0 when Stop mode is entered, the counter is disabled and reloads using the value in the TOUT register.

3.5.3 Debug Mode Operation

The COP counter is not allowed to count when the chip is in Debug mode. As a consequence, the CEN bit in the CTRL register always reads as one when the chip is in Debug mode. The actual value of CEN is unaffected, however. The CEN bit resumes its previously set value when exiting Debug mode.

3.6 Register Definitions

Table 3-1. COP Memory Map

Device	Peripheral	Base Address
56F801X	COP	$00F0E0

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 3-2 lists the COP registers in ascending address order, including their acronyms and address offset of each register. The COP peripheral has three registers.

Table 3-2. COP Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	CTRL	Control Register	Read/Write	**Section 3.6.1**
Base + $1	TOUT	Time-Out Register	Read/Write	**Section 3.6.2**
Base + $2	CNTR	Counter Register	Read/Write	**Section 3.6.3**

Bits of each of the three registers are summarized in **Figure 3-2.** Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	CTRL	R	0	0	0	0	0	0	0	0	0	0	0	0	CSEN	CWEN	CEN	CWP
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				
$1	TOUT	R	TIMEOUT															
		W																
$2	CNTR	R	COUNT															
		W	SERVICE															

R	0	Read as 0
W	▨	Reserved

Figure 3-2. COP Register Map Summary

3.6.1 Control Register (CTRL)

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	CSEN	CWEN	CEN	CWP
Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 3-3. Control (CTRL) Register

3.6.1.1 Reserved—Bits 15–5

These bits are reserved or not implemented. The bits are read as 0 and cannot be modified by writing.

3.6.1.2 Reserved—Bit 4

This is a reserved or unimplemented bit. It is read as 0. It must always be written as 0. Writing 1 to this bit will result in incorrect operation.

3.6.1.3 COP Stop Mode Enable (CSEN)—Bit 3

This bit controls the operation of the COP counter in Stop mode. This bit can only be changed when CWP is set to zero.

- 0 = CNTR will stop in Stop mode (default)
- 1 = CNTR will run in Stop mode when CEN is set to one

3.6.1.4 COP Wait Mode Enable (CWEN)—Bit 2

This bit controls the operation of the COP counter in Wait mode. This bit can only be changed when CWP is set to zero.

- 0 = CNTR will stop in Wait mode (default)
- 1 = CNTR will run in Wait mode when CEN is set to one

3.6.1.5 COP Enable (CEN)—Bit 1

This bit controls the operation of the Counter (CTNR) register. It can only be changed when CWP is set to zero. This bit always reads as zero when the chip is in Debug mode.

- 0 = CNTR is disabled (default)
- 1 = CNTR is enabled

3.6.1.6 COP Write Protect (CWP)—Bit 0

This bit controls the write protection feature of both the Control (CTRL) and Time-Out (TOUT) registers. Once set, this bit can only be cleared by resetting the device.

- 0 = CTRL and TOUT can be read and modified by writing (default)
- 1 = CTRL and TOUT are *read-only*

3.6.2 Time-Out Register (TOUT)

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	TIMEOUT															
Write																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 3-4. Time-Out (TOUT) Register

3.6.2.1 Time-Out Period (TIMEOUT)—Bits 15–0

The value in this register determines the time-out period of the COP counter. TIMEOUT should be written before enabling COP. Once COP is enabled, the recommended procedure for changing TIMEOUT is to disable the COP, write to the TOUT register, then re-enable the COP. This procedure ensures that the new TIMEOUT is loaded into the counter. Alternatively, the TOUT register can be written with a new value prior to writing the proper service patterns to the CNTR register, thereby causing the counter to reload with the new TIMEOUT value. The COP Counter is not reset by a write to the TOUT register. These bits can be changed only when CWP is set to zero.

3.6.3 Counter Register (CNTR)

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	COUNT															
Write	SERVICE															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 3-5. Counter (CNTR) Register

3.6.3.1 Count (COUNT)

This is the current value of the COP counter as it counts down from the Time-Out value to zero. A reset is issued when this count reaches zero.

3.6.3.2 Service (SERVICE)

When enabled, the COP requires a service sequence be performed periodically in order to clear its counter and prevent a reset from being issued. This routine consists of writing $5555 to the CNTR register followed by writing $AAAA before the time-out period expires. The writes to the

CNTR register must be performed in the correct order, but any number of other instructions (and writes to other registers) may be executed between the two writes.

3.7 Time-Out Specifications

The COP_PRESCALER for the 56F801X series is 1024. For the relaxation oscillator operating at 8MHz, the clock to the COP Counter will be 7.81kHz. The value of the TOUT register can be programmed from 0 to 65535 giving a time-out period range from 128µsec minimum to 8.4sec maximum.

3.8 COP After Reset

The CTRL register is cleared out of reset. Therefore, the counter is disabled by default. In addition, the TOUT register is set to its maximum value of $FFFF during reset so the counter is loaded with a maximum time-out period when reset is released.

3.9 Clocks

The COP timer base is the PLL input clock MSTR_OSC divided by a fixed prescaler value of 1024.

3.10 Interrupts

The COP module does not generate interrupts. It does, however, generate the $\overline{\text{COP\_RST}}$ signal when the counter reaches a value of $0000 causing a chip wide reset followed by the execution of the code identified by the second entry in the interrupt vector table.

3.11 Resets

Any system reset forces all registers to their reset state, clearing the $\overline{\text{COP\_RST}}$ signal if it is asserted. The counter will be loaded with its maximum value of $FFFF but will not start when reset is released because CEN is disabled by default.

Chapter 4
Inter-Integrated Circuit Interface (I^2C)

Document Revision History for Chapter 4, Inter-Integrated Circuit Interface (I2C)

Version History	Description of Change
Rev.0	Initial release

4.1 Introduction

Inter-Integrated Circuit Bus (I^{2}C) is a two-wire, bidirectional serial bus providing a simple, efficient method of data exchange between devices. Being a two-wire device, the I^{2}C Bus minimizes the need for large numbers of connections between devices, and eliminating the need for an address decoder.

This bus is suitable for applications requiring occasional communications over a short distance between a number of devices. It also provides flexibility, allowing additional devices to be connected to the bus for further expansion and system development.

The interface will operate at baud rates of up to 100kbps with maximum capacitive bus loading. With reduced bus slew rate, the device is capable of operating at higher baud rates, up to a maximum of [System] clock/20. The module can operate up to a baud rate of 400kbps provided the I^{2}C bus slew rate is less than 100ns. The maximum communication interconnect length and the number of devices able to be connected to the bus are limited by a maximum bus capacitance of 400pF in all instances.

4.2 Features

The following features are included in the design:

- Compatible with I^{2}C Bus standard
- Multi-master operation
- Software programmable for one of 256 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration Lost interrupt with automatic mode switching from master to slave
- Calling Address Identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation/detection
- Acknowledge bit generation/detection
- Bus Busy detection

4.3 Block Diagram

The block diagram of the I^{2}C module is illustrated in **Figure 4-1.**

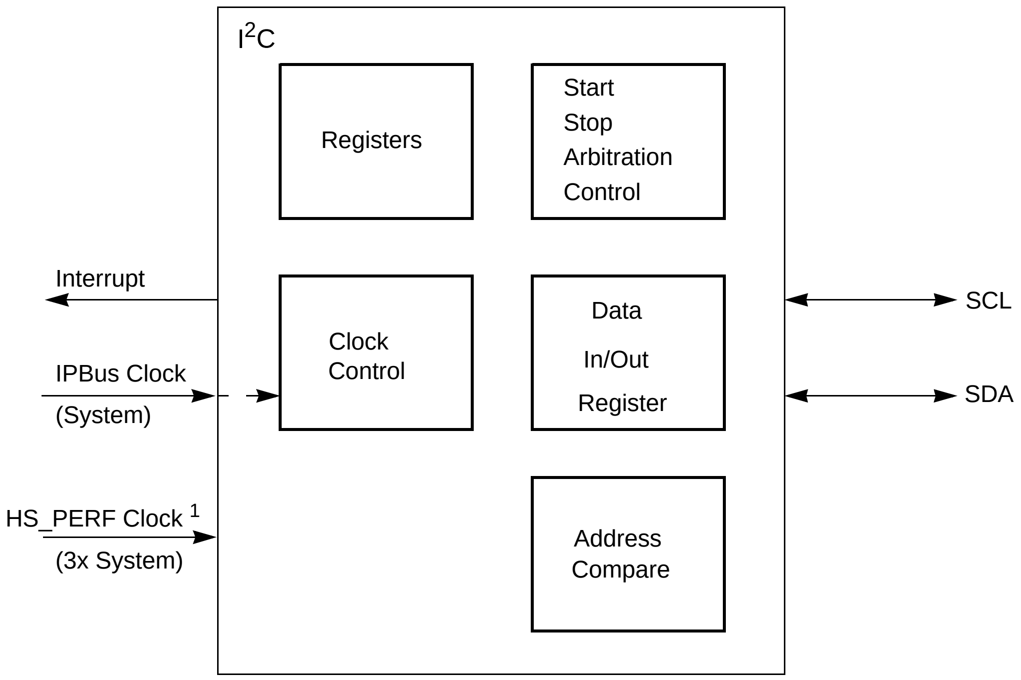


Figure 4-1. I^{2}C Block Diagram

4.3.1 Serial Clock (SCL)

This bidirectional Serial Clock (SCL) line of the module, compatible to the I^{2}C Bus specification.

4.3.2 Serial Data (SDA)

This bidirectional Serial Data (SDA) line of the module is compatible to the I^{2}C Bus specification.

1. Please refer to Figure 5-1 OCCS Block Diagram With Relaxation Oscillator.

4.4 Functional Description

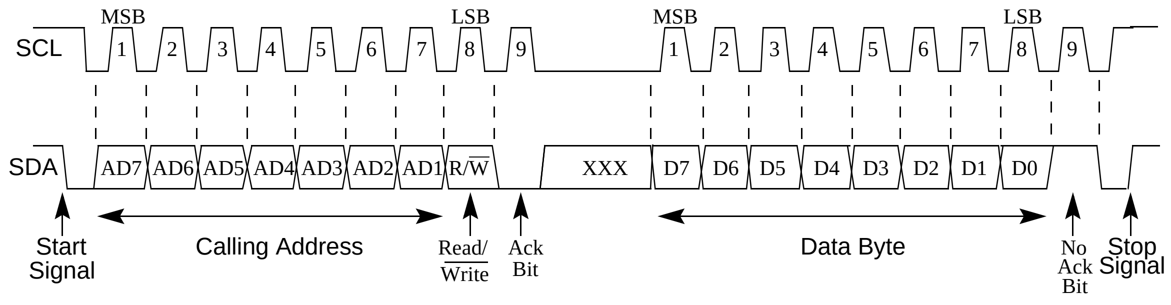
4.4.1 I-Bus Protocol

The I^{2}C Bus system uses a Serial Data (SDA) line and a Serial Clock (SCL) line for data transfer. All devices connected to it must have open drain or open collector outputs. Logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors is system dependent.

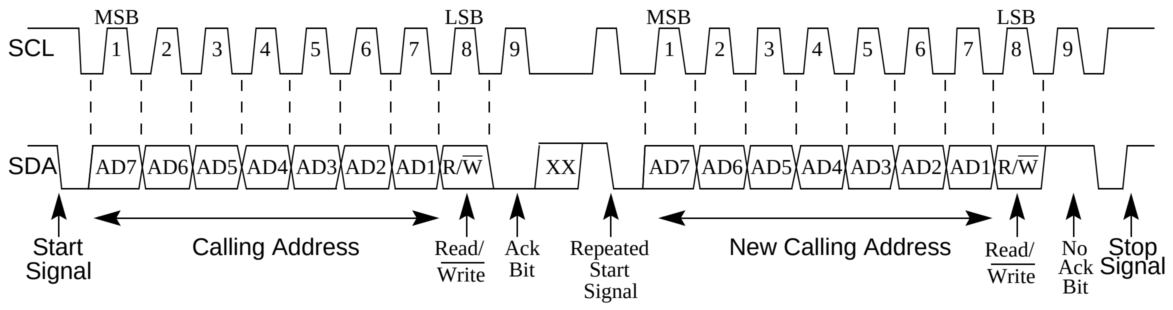
Normally a standard communication is composed of four parts:

1. Start Signal
2. Slave Address Transmission
3. Data Transfer
4. Stop Signal

They are described briefly in the following sections and illustrated in **Figure 4-2**.



a. Normal Address Plus Data Transfer



b. Use of Repeated Start Signal

Figure 4-2. I^{2}C Bus Transmission Signals

4.4.1.1 Start Signal

When the bus is free, that is no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a Start signal. A Start signal is defined, illustrated in **Figure 4-3**, as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and wakes up all slaves out of their idle states.

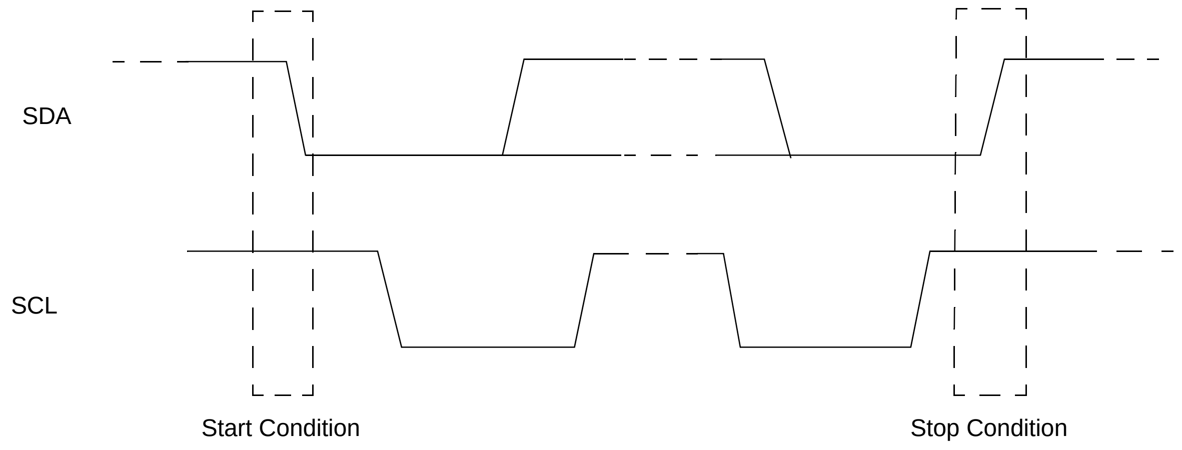


Figure 4-3. Start and Stop Conditions

4.4.1.2 Slave Address Transmission

The first byte of data transfer immediately after the Start signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a R/$\overline{\text{W}}$ bit. The R/$\overline{\text{W}}$ bit tells the slave the desired direction of data transfer.

- 0 = Write transfer, the master transmits data to the slave
- 1 = Read transfer, the slave transmits data to the master

Only the slave with a calling address matching the one transmitted by the master will respond by returning an acknowledge bit. This is achieved by pulling the SDA low at the ninth clock, illustrated in **Figure 4-2a**.

No two slaves may have the same address. A master must not transmit an address equal to its own slave address. An I^{2}C Bus device cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the I^{2}C Bus will revert to Slave mode and operate correctly even if it is being addressed by another master.

4.4.1.3 Data Transfer

Once successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/$\overline{\text{W}}$ bit sent by the calling master.

All transfers after an address cycle are referred to as data transfers even if they carry sub-address information for the slave device.

Each data byte is eight bits long. Data may be changed only while SCL line is low and must be held stable while SCL is high, illustrated in **Figure 4-2a.** There is one clock pulse on SCL for each data bit, the MSB being transferred first. Each data byte has to be followed by an acknowledge bit, which is signalled from the receiving device by pulling the SDA line low at the ninth clock. So, one complete data byte transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master, the SDA line must be left high by the slave. The master can then generate a Stop signal to abort the data transfer or a Start signal (Repeated Start) to commence a new calling.

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means *end of data* to the slave, so the slave releases the SDA line for the master to generate Stop or Start signal.

4.4.1.4 Stop Signal

The master can terminate the communication by generating a Stop signal to free the bus. However, the master may generate a Start signal followed by a calling command without generating a Stop signal first. This is called Repeated Start. A Stop signal is defined as a low-to-high transition of SDA while SCL at Logic 1. Please see **Figure 4-3.**

The master can generate a Stop even if the slave has generated an acknowledge at which point the slave must release the bus.

4.4.1.5 Repeated Start Signal

Illustrated in **Figure 4-2b,** a repeated Start signal is a Start signal generated without first generating a Stop signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in different mode (Transmit/Receive mode) without releasing the bus.

4.4.1.6 Arbitration Procedure

I^{2}C Bus is a true multi-master bus, allowing more than one master to be connected on it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure in which a bus master loses arbitration if it transmits Logic 1 while another master transmits Logic 0. The losing masters immediately switch over to Slave Receive mode and stop driving SDA output. In this case the transition from Master to Slave mode does not generate a Stop condition. Meanwhile, a status bit (IBAL) is set by hardware to indicate loss of arbitration.

4.4.2 Clock Synchronization

Since wire-AND logic is performed on SCL line, a high-to-low transition on SCL line affects all the devices connected on the bus. The devices start counting their low period and once a device's clock has gone low, it holds the SCL line low until the clock high state is reached. However, the change of low-to-high in this device clock may not change the state of the SCL line if another device clock is still within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time. Please see **Figure 4-4**. When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods. The first device to complete its high period pulls the SCL line low again.

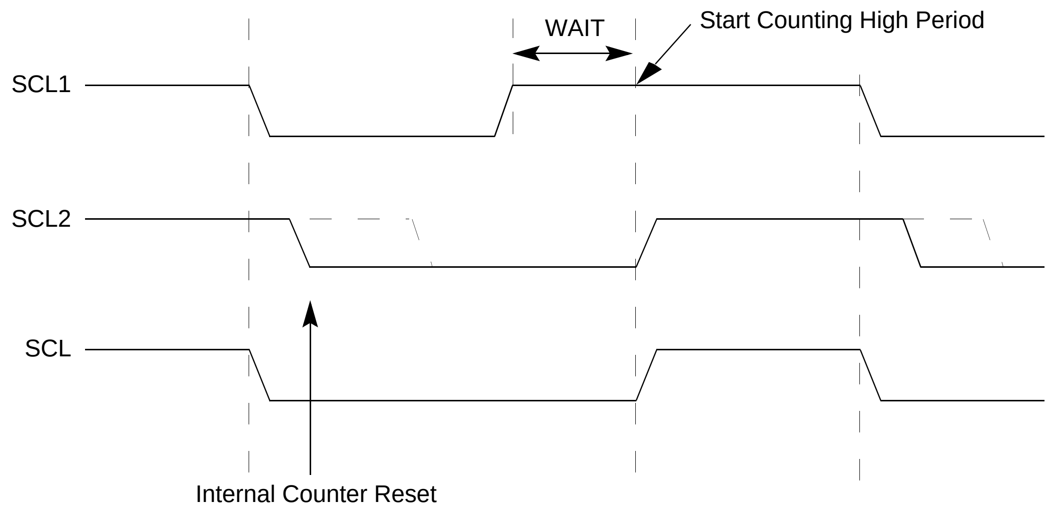


Figure 4-4. I^{2}C Bus Clock Synchronization

4.4.3 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (nine bits). In such a case, it halts the bus clock and forces the Master clock into Wait states until the slave releases the SCL line.

4.4.4 Clock Stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master has driven SCL low the slave can drive SCL low for the required period and then release it. If the slave SCL low period is greater than the master SCL low period then the resulting SCL bus signal low period is stretched.

4.5 Initialization/Application Information

4.5.1 Initialization Sequence

Reset will put the I^{2}C Bus Control Register to its default status. Before the interface can be used to transfer serial data, an initialization procedure must be carried out, as follows:

1. Update the Frequency Divider (FDIV) register and select the required division ratio to obtain SCL frequency from system clock.
2. Update the Noise Filter (NFILT) register
3. Update the I^{2}C Bus Address (ADDR) register to define its slave address.
4. Set the IBEN bit of the I^{2}C Bus Control (CTRL) register.
5. Modify the bits of the I^{2}C Bus Control (CTRL) register to select Transmit/Receive mode and interrupt enable or not.

This sequence is shown in **Figure 4-5**.

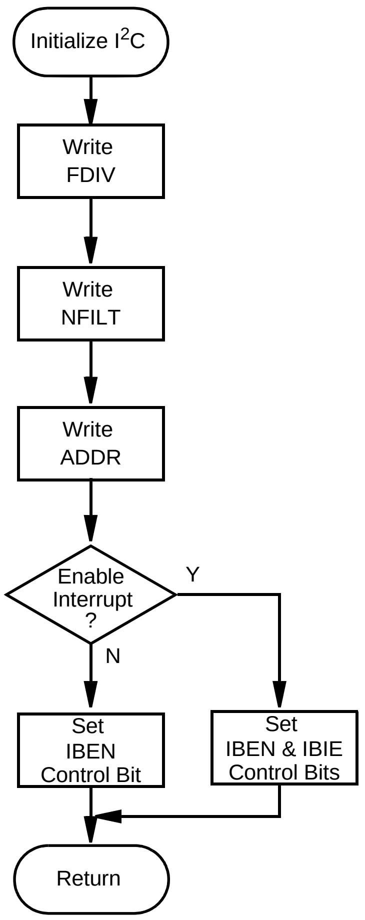


Figure 4-5. Flowchart of Typical I^2C Initialization Routine

4.5.2 Generation of Start

After completion of the initialization procedure, serial data can be transmitted by selecting the Master Transmitter mode. If the device is connected to a multi-master bus system, the state of the I^2C Bus Busy (IBB) bit must be tested to check whether the serial bus is free.

If the bus is free (IBB=0), the start condition and the first byte (the slave address) can be sent. Data written to the Data register comprises a 7-bit slave calling address and the Least Significant Bit (LSB) set to indicate the direction of transfer required from the slave. (LSB = $R/\overline{W}$.)

The bus free time, that is, the time between a Stop condition and the following Start condition, is built into the hardware generating the Start cycle. Depending on the relative frequencies of the system clock and the SCL period, it may be necessary to wait until the I^2C Bus is busy after writing the calling address to the DATA before proceeding with the following instructions.This is illustrated in **Figure 4-6**.

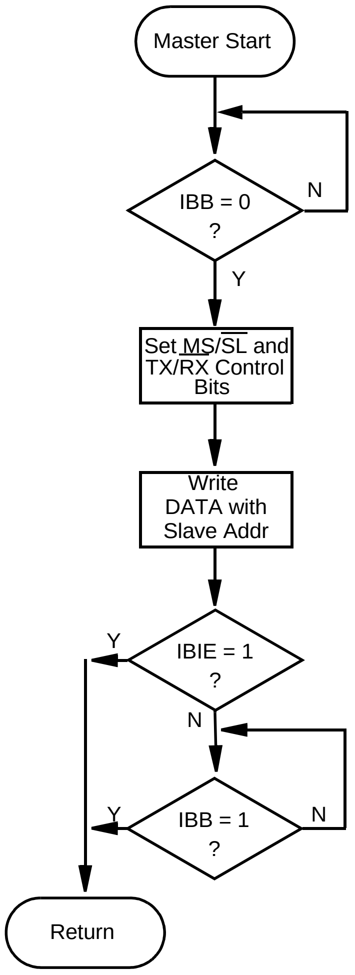

Master Start
IBB = 0 ?
N
Y
Set MS/$\overline{\text{SL}}$ and TX/RX Control Bits
Write DATA with Slave Addr
IBIE = 1 ?
Y
N
IBB = 1 ?
Y
N
Return

Figure 4-6. Flowchart of Typical I^{2}C Master Mode Start Sequence

4.5.3 Post-Transfer Software Response

Successful transmission or reception of a byte will set the Transfer Complete Flag (TCF) bit and the Interrupt Flag (IBIF) bit in the Status (STAT) Register. An interrupt will be issued only if the Interrupt Enable (IBIE) bit in the Control (CTRL) Register is set.

If the IBIE control bit is set, the software will respond with an interrupt service routine. However, if IBIE is not set, the software must poll the IBIF status bit to determine when a I^{2}C interface transfer needs servicing.

When an interrupt is detected or if the polling of IBIF status bit detects an active condition, the servicing routine *must* follow the flow indicated in **Figure 4-7**.

The Interrupt Flag (IBIF) bit is cleared by writing 1 to the IBIF bit in the IBSR status register.

The TCF bit is cleared, indicating data transfer is in progress. To clear the TCF bit, read the Data (DATA) Register in receive mode, or write the DATA Register in transmit mode.

Note: The TCF bit *should not be used as a data transfer complete flag* because the flag timing is dependent on a number of factors including the I^2C Bus frequency.

This bit may not conclusively provide an indication of a transfer complete situation. The IBIF flag is recommended to be used to determine transfer complete.

Note: At the end of the address cycle the master will always be in Transmit mode, i.e. the address is transmitted. If Master Receive mode is required, indicated by $R/\overline{W}$ bit in DATA, the $TX/\overline{RX}$ bit should be toggled at this stage.

During Slave mode address cycles (IAAS=1) the SRW bit in the Status register must be read to determine the direction of the subsequent transfer and the $TX/\overline{RX}$ bit is programmed accordingly. For Slave mode data cycles (IAAS=0) the SRW bit is not valid, the $TX/\overline{RX}$ bit in the Control register should be read to determine the direction of the current transfer.

4.5.4 Generation of Stop

A data transfer ends with a Stop signal generated by the master device. A master transmitter can simply generate a Stop signal after all the data has been transmitted.

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data, achieved by setting the Transmit Acknowledge (TXAK) bit before reading the second to last byte of data. Before reading the last byte of data, a Stop signal must be generated first.

4.5.5 Generation of Repeated Start

At the end of data transfer, if the master still wants to communicate on the bus, it can generate another Start signal followed by another slave address without first generating a Stop signal. Please see **Figure 4-7** for software flow for a repeated start.

4.5.6 Slave Mode

In the slave interrupt service routine, the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the Transmit/Receive mode select bit ($TX/\overline{RX}$ bit of CTRL) according to the R/W command bit (SRW). Writing to the CTRL Register clears the IAAS automatically.

Note: The only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred; interrupts resulting from subsequent data transfers will have IAAS cleared.

A data transfer may now be initiated by writing information to DATA, for slave transmits, or dummy reading from the DATA register, in Slave Receive mode. The slave will drive SCL low in-between byte transfers, SCL is released when the DATA register is accessed in the required mode.

In slave transmitter routine, the Received Acknowledge (RXAK) bit must be tested before transmitting the next byte of data. Setting RXAK means an *end of data* signal from the master receiver, after which it must be switched from Transmitter mode to Receiver mode by software. A dummy read then releases the SCL line so the master can generate a Stop signal.

4.5.7 Arbitration Lost

If several masters try to engage the bus simultaneously, only one master wins and the others lose arbitration. The devices which lost arbitration are immediately switched to Slave Receive mode by the hardware.

When arbitration loss occurs during an address byte, the losing master will continue to receive the address packet, so the master acts like a *slave receiver* for that address packet even though its TX/$\overline{\text{RX}}$ control bit is indicating Tx setting.

The master device losing arbitration will stop sourcing its data output to the SDA line, but will continue to generate SCL clocks until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with IBAL=1 and MS/$\overline{\text{SL}}$=0.

If one master attempts to start transmission while the bus is being engaged by another master, the hardware will inhibit the transmission switch the MS/$\overline{\text{SL}}$ bit from 1 to 0 without generating Stop condition, generate an interrupt to CPU (if enabled), and set the IBAL to indicate the attempt to engage the bus failed.

4.5.8 Receive Interrupt

The processor interrupts will occur only when the IBIE control bit is set. On an interrupt from the I^{2}C, the service routine must follow the flowchart illustrated in **Figure 4-7.**

4.5.9 Receive Polling

If the IBIE control bit is not set, the processor must poll the IBIF status bit to determine when the I^{2}C requires service. When IBIF status bit is set, the service routine must follow the flowchart illustrated in **Figure 4-7**.

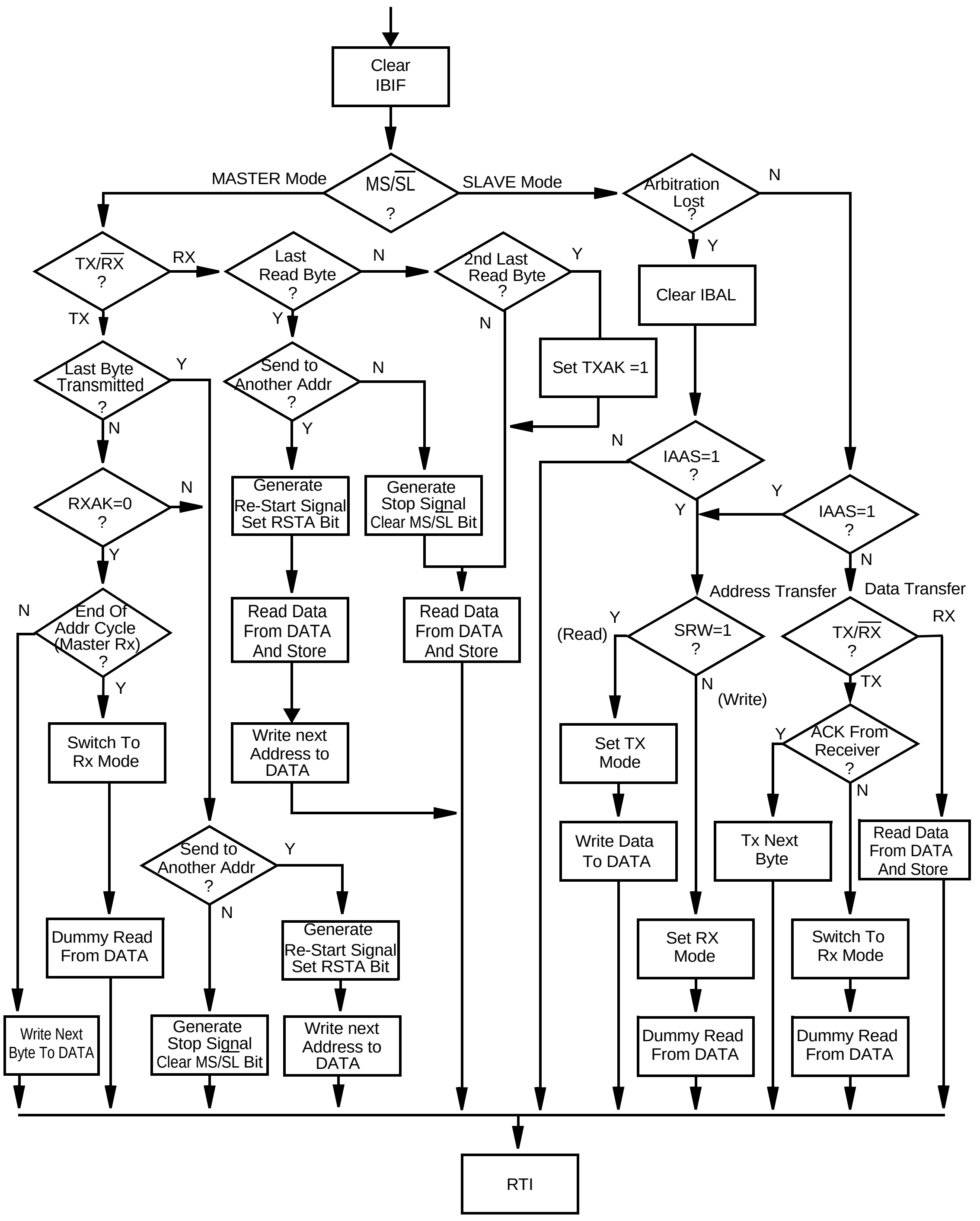


Figure 4-7. Flow-Chart of Typical I^{2}C Interrupt Routine

4.6 Register Definitions

Table 4-1. I^{2}C Memory Map

Device	Peripheral	Base Address
56F801X	I^{2}C	$00F0D0

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 4-2 lists the I^{2}C registers in ascending address order, including their acronyms and base address offset of each register. The I^{2}C peripheral has six registers.

Table 4-2. I^{2}C Bus Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	ADDR	Address Register	Read/Write	Section 4.6.1
Base + $1	FDIV	Frequency Divider Register	Read/Write	Section 4.6.2
Base + $2	CTRL	Control Register	Read/Write	Section 4.6.3
Base + $3	STAT	Status Register	Read/Write*	Section 4.6.4
Base + $4	DATA	Data I/O Register	Read/Write	Section 4.6.5
Base + $5	NFILT	Noise Filter Register	Read/Write	Section 4.6.6

* Only two bits in a field can be modified by writing. Please see Section 4.6.4.

Bits of each of the six registers are summarized in Figure 4-8. Details of each follow.

Addr. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	ADDR	R	0	0	0	0	0	0	0	0	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
		W	▨	▨	▨	▨	▨	▨	▨	▨								▨
$1	FDIV	R	0	0	0	0	0	0	0	0	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
		W	▨	▨	▨	▨	▨	▨	▨	▨								
$2	CTRL	R	0	0	0	0	0	0	0	0	IBEN	IBIE	MS/$\overline{\text{SL}}$	TX/$\overline{\text{RX}}$	TXAK	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨						RSTA	▨	▨
$3	STAT	R	0	0	0	0	0	0	0	0	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨		▨
$4	DATA	R	0	0	0	0	0	0	0	0	D7	D6	D5	D4	D3	D2	D1	D0
		W	▨	▨	▨	▨	▨	▨	▨	▨								
$5	NFILT	R	0	0	0	0	0	0	0	0	0	0	0	0	NF3	NF2	NF1	NF0
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				

R	0	Read as 0
W	▨	Reserved

Figure 4-8. I^{2}C Bus Register Map Summary

4.6.1 Address Register (ADDR)

This register contains the address the I^{2}C Bus will respond to when addressed as a slave.

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-9. Address (ADDR) Register

Note: It is not the address sent on the bus during the address transfer.

4.6.1.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

4.6.1.2 Slave Address (ADR7–ADR1)—Bits 7–1

These bits contain the specific slave address to be used by the I^{2}C Bus module. The Default mode of the I^{2}C Bus is Slave mode for an address match on the bus.

4.6.1.3 Reserved—Bit 0

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

4.6.2 Frequency Divider Register (FDIV)

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-10. Frequency Divider (FDIV) Register

4.6.2.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

4.6.2.2 Bus Clock Rate (IBC7–IBC0)—Bits 7–0

This bit field is used to prescale the clock for bit rate selection. The bit clock generator is implemented as a prescale divider (IBC7–6), prescaled Shift register (IBC5–3), select the prescaler divider and IBC2–0 select register tap point. The IBC bits are decoded to give the Tap and Prescale values provided in **Table 4-3**.

Table 4-3. I^2C Bus Tap and Prescale Values

IBC2-0 (binary)	SCL Tap	SDA Tap
000	5	1
001	6	1
010	7	2
011	8	2
100	9	3
101	10	3
110	12	4
111	15	4

IBC5-3 (binary)	SCL2Start	SCL2Stop	SCL2Tap	Tap2Tap	NfOffset
000	2	7	4	1	0
001	2	7	4	2	1
010	2	9	6	4	1
011	6	9	6	8	5
100	14	17	14	16	5
101	30	33	30	32	0
110	62	65	62	64	0
111	126	129	126	128	0

Table 4-4. Multiplier Factor

IBC7-6 (Binary)	MUL
00	01
01	02
10	04
11	Reserved

The number of clocks from the falling edge of SCL to the first tap (Tap[1]) is defined by the values provided in the SCL2Tap column of **Table 4-3**. All subsequent tap points are separated by 2^{IBC5-3}, depicted in the Tap2Tap column in **Table 4-3**. The SCL Tap is used to generate the SCL period while the SDA Tap determines the delay from the falling edge of SCL to SDA changing and the SDA hold time.

IBC7-6 defines the multiplier factor MUL. Values of MUL are provided in **Table 4-4**.

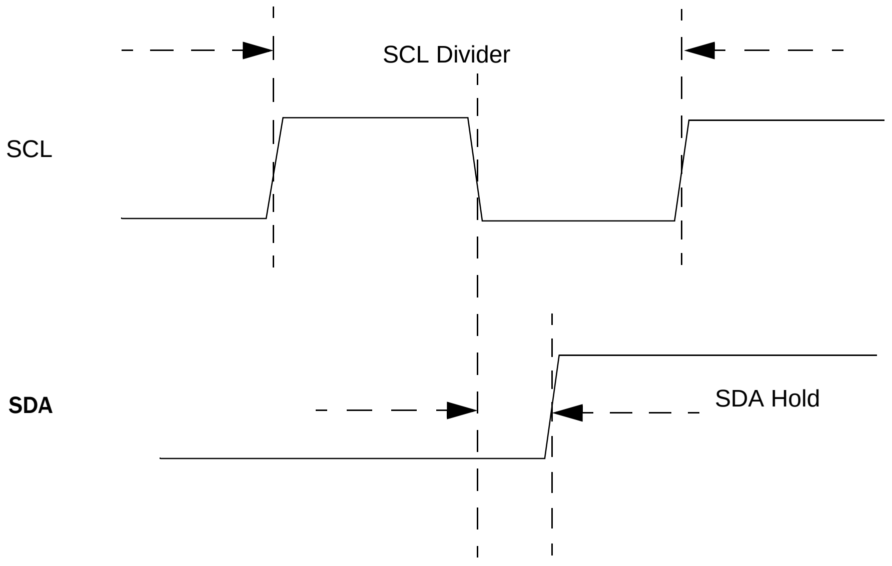


Figure 4-11. SCL Divider and SDA Hold

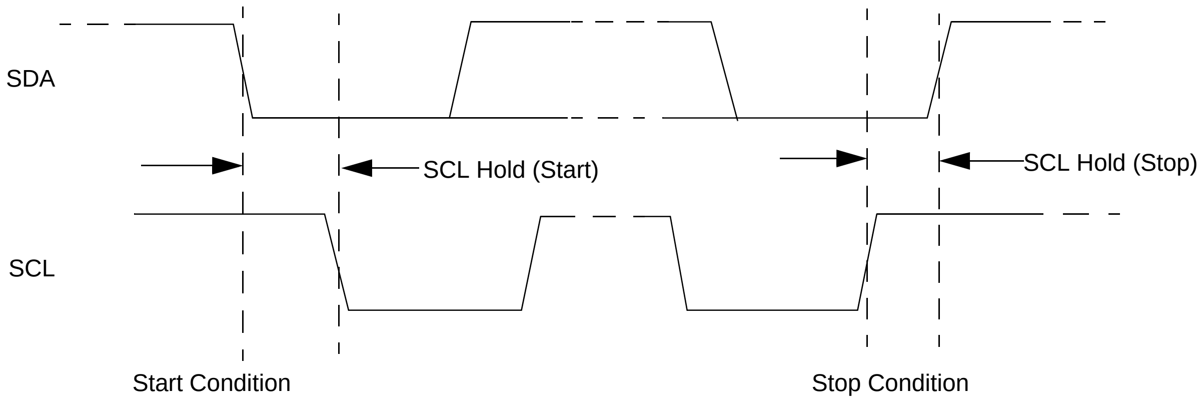


Figure 4-12. SCL Hold

4.6.2.3 SCL Divider Equation

- **Noise Filter Disabled:**

$$\text{SCL Divider} = \text{MUL} \times 2 \times \{\text{SCL2Tap} + [(\text{SCL\_Tap} - 1) \times \text{Tap2Tap}] + 2\}$$

- **Noise Filter Enabled:**

$$\text{SCL Divider} = \text{MUL} \times 2 \times \{(\text{SCL2Tap} + [(\text{SCL\_Tap} - 1) \times \text{Tap2Tap}] + 2) + [\text{Tap2Tap} \times \text{integer } \{[(\text{NF} + 1) + (F_{\text{ratio}} \times \text{MUL} \times \text{NfOffset})] / [F_{\text{ratio}} \times \text{MUL} \times \text{Tap2Tap}]\}]\}$$

NF: Programmed value. Please see **Section 4.6.6**.

F_{ratio}: Filter clock frequency/System clock frequency= 3 or 1[1]

4.6.2.4 SDA Hold Value Equation

$$\text{SDA Hold} = \text{MUL} \times \{\text{SCL2Tap} + [(\text{SDA\_Tap} - 1) \times \text{Tap2Tap}] + 3\}$$

4.6.2.5 SCL Start Hold Value Equation

$$\text{SCL Hold(Start)} = \text{MUL} \times [\text{SCL2Start} + (\text{SCL\_Tap} - 1) \times \text{Tap2Tap}]$$

4.6.2.6 SCL Stop Hold Value Equation

$$\text{SCL Hold(Stop)} = \text{MUL} \times [\text{SCL2Stop} + (\text{SCL\_Tap} - 1) \times \text{Tap2Tap}]$$

4.6.2.7 Recommended Frequency Divider Programming Values

Recommended Frequency Divider settings for System Clock frequencies for the 56F801X devices are described in **Table 4-5**. **Section 4.6.6** describes the Noise Filter requirement to be set to a count to comply with the 50ns noise suppression requirement specified in the *I^{2}C Bus Specification* for Fast mode performance. The noise filter is not required for Standard mode operation of the I^{2}C Bus: SCL frequency = 100kHz. However, to prevent potential module malfunction, it is *strongly recommended* to use the noise filter.

1. F_{ratio} = 1 if PLL is not locked or OCCS_CTRL = 01
   F_{ratio} = 3 if PLL is locked and OCCS_CTRL = 10

Table 4-5. Recommended Frequency Divider Values

System Clock (MHz)	Filter** Clock (MHz)	Noise Filter NF (HEX)	Standard Mode SCL = 100 KHz IBC7–0 (HEX)	Fast Mode SCL = 400 KHz IBC7–0 (HEX)
32	96	$5	$60	$4A
16	48	$3	$20	$0A
8	24	$2	$80	$00 * SCL = 363 KHz
4	12	$2	$0A	$00 * SCL = 181 KHz
2	6	$1	$00	$00 * SCL = 100 KHz
1	3	$1	$00 * SCL = 50 KHz	$00 * SCL = 50 KHz

* Expected SCL frequency
** Filter Clock Frequency = 3x System Clock when PLL is locked and OCCS_CTRL = 10, otherwise the Filter Clock Frequency = System Clock.

4.6.3 Control Register (CTRL)

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	IBEN	IBIE	MS/$\overline{\text{SL}}$	TX/$\overline{\text{RX}}$	TXAK	0	0	0
Write														RSTA		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-13. Control (CTRL) Register

4.6.3.1 Reserved—Bits15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

4.6.3.2 I-Bus Enable (IBEN)—Bit 7

This bit controls the software reset of the entire I^{2}C Bus module.

- 0 = The module is reset and disabled. This is the power-on reset situation. When low, the interface is held in reset but registers can still be accessed.
- 1 = The I^{2}C Bus module is enabled.This bit must be set before any other CTRL register bits have any effect.

If the I^{2}C Bus module is enabled in the middle of a byte transfer the interface behaves as set forth below:

- Slave mode ignores the current transfer on the bus and starts operating whenever a subsequent start condition is detected.Master mode will not be aware the bus is busy, hence if a start cycle is initiated, the current bus cycle may become corrupt. This action would ultimately result in either the current bus master or the I^{2}C Bus module losing arbitration, after which bus operation would return to normal.

4.6.3.3 I-Bus Interrupt Enable (IBIE)—Bit 6

- 0 = Interrupts from the I^{2}C Bus module are disabled. This does not clear any currently pending interrupt condition
- 1 = Interrupts from the I^{2}C Bus module are enabled. An I^{2}C Bus interrupt occurs provided the IBIF bit in the Status Register is also set.

4.6.3.4 Master/Slave Mode Select (MS/$\overline{\text{SL}}$)—Bit 5

This bit is cleared upon reset. When this bit is changed from 0 to 1, a Start signal is generated on the bus, and the Master mode is selected. When this bit is changed from 1 to 0, a Stop signal is generated and the Operation mode changes from Master to Slave. A Stop signal should only be generated if the IBIF flag is set. MS/$\overline{\text{SL}}$ is cleared without generating a Stop signal when the master loses arbitration.

- 0 = Slave mode
- 1 = Master mode

4.6.3.5 Transmit/Receive Mode Select (TX/$\overline{\text{RX}}$)—Bit 4

This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the Status Register. In Master mode, this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit will always be high.

- 0 = Receive
- 1 = Transmit

4.6.3.6 Transmit Acknowledge Enable (TXAK)—Bit 3

This bit specifies the value driven onto SDA during acknowledge cycles for both master and slave receivers. The I^{2}C module will always acknowledge address matches, provided it is enabled, regardless of the value of TXAK. Values written to this bit are only used when the I^{2}C Bus is a receiver, not a transmitter.

- 0 = An acknowledge signal is sent to the bus at the ninth clock bit after receiving one byte data
- 1 = No acknowledge signal response is sent (i.e., acknowledge bit = 1)

4.6.3.7 Repeat Start (RSTA)—Bit 2

Writing 1 to this bit generates a repeated Start condition on the bus, provided it is the current bus master. This bit will always be read as a low. Attempting a repeated start at the wrong time, if the bus is owned by another master, will result in loss of arbitration.

- 0 = No action
- 1 = Generate repeat Start cycle

4.6.3.8 Reserved—Bits 1–0

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

4.6.4 Status Register (STAT)

This status register is *read-only* with exception of bits one (IBIF) and four (IBAL). Both are cleared by software.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
Write																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Figure 4-14. Status (STAT) Register

4.6.4.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

4.6.4.2 Transfer Complete Flag (TCF)—Bit 7

While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the ninth clock of a byte transfer. This bit is only valid during or immediately following a transfer to/from the I^{2}C module.

- 0 = Transfer in progress
- 1 = Transfer complete

4.6.4.3 I-Bus Addressed As Slave (IAAS)—Bit 6

When its own specific address (I^2C Bus Address register) is matched with the calling address, this bit is set before the ACK bit. The Core is interrupted provided the IBIE is set. At this point the CPU needs to check the SRW bit and set its TX/$\overline{\text{RX}}$ mode accordingly. Writing to the I^2C Bus Control register clears this bit.

- 0 = Not addressed or cleared by a write to the Control register
- 1 = Addressed as a slave

4.6.4.4 I-Bus Busy (IBB)—Bit 5

This bit indicates the status of the bus. When a Start signal is detected, the IBB is set. If a Stop signal is detected, IBB is cleared and the bus enters an idle state.

- 0 = Bus is idle
- 1 = Bus is busy

4.6.4.5 I-Bus Arbitration Lost (IBAL)—Bit 4

The Arbitration Lost (IBAL) bit is set by hardware when the arbitration procedure is lost. Arbitration is lost in the following circumstances:

1. SDA sampled low when the Master drives a high during an Address or Data Transmit cycle.

   IBAL status bit asserts after the ACK clock cycle.

2. SDA sampled low when the Master drives a high during the acknowledge bit of a Data Receive cycle.

   IBAL status bit asserts after the ACK clock cycle.

3. A Start cycle is attempted when the bus is busy.

   IBAL status bit asserts after start cycle is attempted.

4. A repeated Start cycle is requested in Slave mode.

   IBAL status bit asserts after repeat start is attempted.

5. A Stop condition is detected when the Master did not request it.

   IBAL status bit asserts after stop condition is detected.

This bit must be cleared by software by writing 1 to it. A write of 0 has no effect on this bit.

4.6.4.6 Reserved—Bit 3

This bit is reserved or not implemented. A read operation on this bit will return 0. It cannot be modified by writing.

4.6.4.7 Slave Read/Write (SRW)—Bit 2

When IAAS is set this *read-only* bit indicates the value of the R/W command bit of the calling address sent from the master.

This bit is only valid when the I-Bus is in Slave mode, a complete address transfer has occurred with an address match and no other transfers are initiated.

Checking this bit, the CPU can select Slave Transmit/Receive mode according to the command of the Master.

- 0 = Slave receive, Master writing to Slave
- 1 = Slave transmit, Master reading from Slave

4.6.4.8 I-Bus Interrupt (IBIF)—Bit 1

The IBIF bit is set when one of the following conditions occurs:

- Arbitration Lost (IBAL bit set)
- Byte transfer complete (TCF bit set)
- Addressed as slave (IAAS bit set)

It will cause a processor interrupt request if the IBIE bit is set. This bit must be cleared by software by writing 1 to it. A write of 0 has no effect on this bit.

4.6.4.9 Received Acknowledge (RXAK)—Bit 0

The value of SDA during the acknowledge bit of a bus cycle. If the received acknowledge bit (RXAK) is low, it indicates an acknowledge signal was received after the completion of eight bits data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the ninth clock.

- 0 = Acknowledge received
- 1 = No acknowledge received

CAUTION: This bit is only valid when TCF status bit is 1, indicating transfer has completed.

4.6.5 Data I/O Register (DATA)

In Master Transmit mode, when data is written to the I^{2}C Data I/O (DATA) register a data transfer is initiated. The Most Significant Bit (MSB) is sent first. In Master receive mode, reading this register initiates next byte data receiving. In Slave mode, the same functions are available after an address match occurred.

Note: The TX/$\overline{\text{RX}}$ bit in the CTRL register must correctly reflect the desired direction of transfer in Master and Slave modes for the transmission to begin. For instance, if the I^{2}C is configured for master transmit, but a master receive is desired, reading the DATA register will not initiate the receive.

Reading the DATA register returns the last byte received while the I^{2}C is configured in either master receive or slave receive modes. The DATA register does not reflect every byte transmitted on the I^{2}C Bus, nor can software verify a byte was written to the DATA register correctly by reading it back.

In Master Transmit mode, the first byte of data written to DATA register following assertion of MS/$\overline{\text{SL}}$ is used for the address transfer and should comprise of the calling address (in position D7-D1) concatenated with the required R/$\overline{\text{W}}$ bit (in position D0).

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	D7	D6	D5	D4	D3	D2	D1	D0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-15. Data I/O (DATA) Register

4.6.5.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. These bits will always read as 0 and cannot be modified by writing.

4.6.5.2 Data (D7–D0)—Bits 7–0

Please refer to **Section 4.6.5** for additional information.

4.6.6 Noise Filter Register (NFILT)

The Noise Filter is inserted to intercept and process the SCL and SDA input signals before the module processes the interface sequence. The Noise filter samples received signals independently with a filter clock. When an input signal changes state, the Noise Filter will pass the state change after the signal is stable (not changed) for NF counts. If the input signal returns to the original state prior to the NF counts, the counter clears, not changing the state sent to the module. This filters any false glitches as specified in the *I^{2}C Bus Specification* for Fast mode performance.

The addition of this filter alters the computation of the SCL Divider algorithm. Please see **Section 4.6.2.3** for computation of SCL Divider with Noise Filter active.

The Filter Clock frequency and NF counts dictate the size of glitch suppression. The size of glitch suppressed is less than NF/(Filter Clock frequency). The filter can be completely bypassed by setting the Noise Filter to Hex 0.

Base + $5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	NF3	NF2	NF1	NF0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-16. Noise Filter (NFILT) Register

4.6.6.1 Reserved—Bits 15–4

This bit field is reserved or not implemented. These bits will always read 0 and cannot be modified by writing.

4.6.6.2 Noise Filter Count (NF3-NF0)—Bits 3-0

Please refer to **Section 4.6.6** for additional information.

4.7 Interrupts

I^{2}C uses only one interrupt vector. See each device data sheet for the location of this vector in the Interrupt Vector Table.

Table 4-6. Interrupt Summary

Interrupt	Offset	Vector	Priority	Source	Description
I^{2}C Interrupt	—	—	—	IBAL, TCF, IAAS, IBIF bits in STAT register	An interrupt issued when Arbitration lost, Transfer Complete, or Address Detected occurs.

4.7.1 Interrupt Description

Internally, there are three types of interrupt conditions in the I^{2}C. The interrupt service routine can determine the interrupt type by reading the Status Register.

1. Arbitration Lost condition (IBAL bit set)
2. Byte Transfer condition (TCF bit set)
3. Address Detect condition (IAAS bit set)

The I^{2}C interrupts are enabled by the IBIE bit in the I^{2}C CTRL Register. They must be cleared by writing 1 to the IBIF bit in the interrupt service routine.

4.8 Resets

The reset state of each individual bit is listed within the Register Definition **Section 4.6,** detailing the registers and their bit fields.

Chapter 5
On-Chip Clock Synthesis (OCCS)

Document Revision History for Chapter 5, On-Chip Clock Synthesis (OCCS)

Version History	Description of Change
Rev. 0	Initial Release
Rev. 5	Corrected the address offsets In **Table 5-3** as follows SHUTDN (was Base + $3, is Base + $4) OCTRL (was Base + $4, is Base + $5)

5.1 Introduction

This module provides the 2× system clock frequency to the System Integration Module (SIM), used to generate the various chip clocks. This module also produces the OSC_CLK signal signals.

The On-Chip Clock Synthesis (OCCS) module allows product design using an internal relaxation oscillator to run 56800E at user selectable frequencies up to 32Mhz.

5.2 Features

The OCCS module interfaces with the oscillator and PLL, as well as having an on-chip relaxation oscillator. The OCCS module characteristics include:

- Internal relaxation oscillator
- Ability to power-down internal relaxation oscillator
- Ability to put the internal relaxation oscillator into a Standby mode
- Three-bit postscaler provides control for the PLL output
- Ability to power-down the internal PLL
- Provides 2× master clock frequency and OSC_CLK signals
- Provides 3× High Speed Peripheral Clock to Timer and PWM modules
- Safety shutdown feature available in the event the PLL reference clock disappears
- Can be driven from an external clock source

The clock generation module provides the programming interface for both the PLL and relaxation oscillator.

5.3 Block Diagram

Figure 5-1 illustrates the clock generation module.

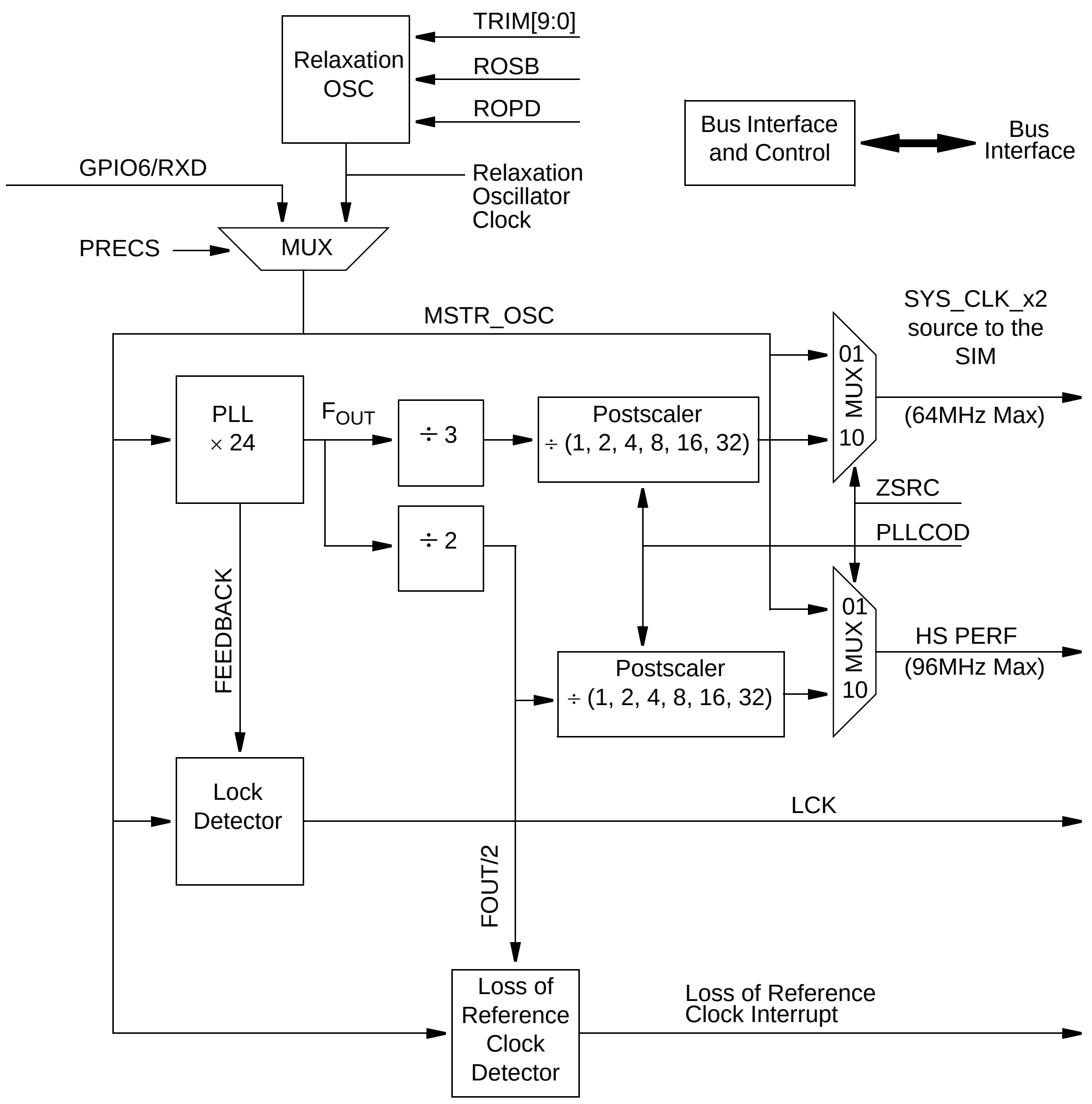

TRIM[9:0]
ROSB
ROPD
Relaxation OSC
Bus Interface and Control
Bus Interface
GPIO6/RXD
Relaxation Oscillator Clock
PRECS
MUX
MSTR_OSC
SYS_CLK_x2 source to the SIM
(64MHz Max)
PLL × 24
FOUT
÷ 3
Postscaler ÷ (1, 2, 4, 8, 16, 32)
01
MUX
10
ZSRC
÷ 2
PLLCOD
FEEDBACK
01
MUX
10
HS PERF
(96MHz Max)
Postscaler ÷ (1, 2, 4, 8, 16, 32)
Lock Detector
LCK
FOUT/2
Loss of Reference Clock Detector
Loss of Reference Clock Interrupt

Figure 5-1. OCCS Block Diagram

5.4 Functional Description

A block diagram of the OCCS module is illustrated in **Figure 5-1**. Possible clock source choices are:

- Internal relaxation oscillator
- External clock source

Each of these clock sources can be selected to drive the remainder of the clock generation circuitry. This circuitry allows direct use of the clock, or the clock can be used as an input to the PLL. The PLL will generate a higher frequency clock for use within the chip. This allows two final clock output selections:

- Direct oscillator output
- Postscaler output

The clock multiplexer (ZSRC MUX) selects the direct clock on power-up. A different clock source can be selected by writing to the PLL Control (CTRL) register. Once a new clock source is selected, it will be activated within four periods of the current IPBus clock.

Transitions to/from prescaler to postscaler frequencies are guaranteed to be glitch free. Before switching to the PLL, the PLL must be locked. The PLL Status (STAT) register shows the status of the core clock source. Because the synchronizing circuit changes modes to avoid any glitches, the STAT register ZCLOCK Source (ZSRC) will show overlapping modes as an intermediate step. After PLL lock is detected the core clock can be switched to the PLL by writing to the ZSRC bits in the CTRL register.

Frequencies going out of the PLL are controlled by the postscaler, and the divide-by ratio within the PLL. For proper operation of the PLL, keep the VCO within the PLL in its operational range of 60MHz to 68MHz. The output of the VCO is depicted as F_{OUT} in **Figure 5-1**. The input frequency multiplied by the divide-by ratio is the frequency at which the VCO is running.

The PLL lock time is 10ms or less when coming from a powered down state to a power up state. It is recommended when powering down, or powering up, the PLL be deselected as the clocking source. Only after lock is achieved should the PLL be used as a valid clocking source.

Table 5-1 provides possible clock sources and configurations.

Table 5-1. Clock Choices

Clock Source	Selected Clock	Configuration Steps (After Reset)
Relaxation Oscillator	Postscaler	1. Change TRIM as needed to obtain the desired clock rate 2. Change PLLCOD if desired 3. Enable the PLL (PLL=0) 4. Wait for PLL lock (LCK1=1 and LCK0=1) 5. Change ZSRC to select the postscaler clock (ZSRC=10)
External Clock Source	Postscaler	1. The clock source should be changed to the external source (PRECS=1) 2. The Relaxation Oscillator should be powered down (ROPD=1) to conserve power 3. Change PLLCOD if desired 4. Enable the PLL (PLLD=0) 5. Wait for PLL lock (LCK1=1 and LCK0=1) 6. Change ZSRC to select the postscaler clock (ZSRC=10)

5.5 Relaxation Oscillator

5.5.1 Trimming Frequency on the Relaxation Oscillator

The Relaxation Oscillator frequency will vary as much as ± 20 percent due to process, temperature, and voltage dependencies. These dependencies are in the voltage and current references, the offset of the comparators, and the internal capacitor. Voltage and temperature dependencies are designed to be a maximum of approximately two percent error. The process dependencies account for the rest.

Fortunately for an individual part, process dependencies are constant. An individual part can operate at approximately two percent variance from its adjusted operating point over the entire specification range of the application. If the unadjusted operating point can be changed, the entire variance can be limited to two percent.

The method of changing the unadjusted operating point is by changing the size of the capacitor. This capacitor value is controlled by the Trim Factor (TRIM) in the OCTRL register. The default value for TRIM is $200. Each unit added or removed will adjust the output period by about 0.078 percent of the unadjusted period (adding to TRIM will increase the clock period, decreasing the frequency). With TRIM containing 10-bits, the clock period of the Relaxation Oscillator Clock can be changed to ± 40 percent of its unadjusted value, enough to cancel the process variability mentioned before.

The best way to trim the Internal Clock is to use the Factory Trim value stored in the Flash Module OPT1 register. Please see **Section 6.7.8** in the Flash Module chapter. At boot-up simply copy this value into the TRIM field of the OCTRL register.

5.6 External Reference

If higher clock precision is required, the chip can be operated from an external clock source.

5.6.1 Switching Clock Sources

To robustly switch between the Relaxation Oscillator Clock and the External Oscillator Clock, the changeover switch assumes the clocks are completely asynchronous, so a synchronizing circuit is required to make the transition. When the Prescaler Clock Select (PRECS) is changed, the switch will continue to operate off the original clock for between one and two cycles as the select input is transitioned through one side of the synchronizer. Next, the output will be held low for between one and two cycles of the new clock as the select input transitions through the other side. Then the output starts switching at the new clock's frequency. This transition guarantees no glitches will be seen on the output even though the select input may change asynchronously to the clocks. The unpredictability of the transition period is a necessary result of the asynchronicity. The switch automatically selects Relaxation Oscillator Clock during reset.

Switching from the Relaxation Oscillator Clock to the Crystal Oscillator Clock source or vice-versa requires both clock sources to be enabled and stable. A simple flow requires:

- If switching to the external reference, be certain it was enabled via GPIO.
- If switching to the Relaxation Oscillator, be certain it is powered up, e.g. ROPD is clear.
- Wait for a few cycles for the clock to become active.
- Switch clocks
- Wait for PRECS bit in STAT register to change to the desired value
- Disable previous clock source, e.g. power-down the Internal Relaxation Oscillator if the clock is selected.

The key point to remember in this flow is the clock source should not be switched unless the desired clock is on and stable.

When a new controller core clock is selected, the clock generation module will synchronize the request and select the new clock. The Status (STAT) register shows the status of the core clock source. Since the synchronizing circuit changes modes as to avoid any glitches, the PRECS and ZSRCS bits in CTRL register will show overlapping modes as an intermediate step.

5.7 PLL Frequency Lock Detector Block

This digital block monitors the Voltage Controlled Oscillator (VCO) output clock and sets the LCK bit field in the Status (STAT) register based on its frequency accuracy. The lock detector is enabled with the LCKON bit of the Control (CTRL) register, as well as the PLL_PDN bit of the CTRL register. Once enabled, the detector starts two counters whose outputs are periodically compared. The input clocks to these counters are the VCO output clock divided by the value in the PLLDB field of the CTRL register, called feedback, and the PLL input clock, shown as FREF in **Figure 5-1.** The period of the pulses being compared cover one whole period of each clock. This is due to the feedback clock not guaranteeing a 50 percentage duty cycle. The design of this block was accomplished with the assumption the feedback clock transitions high on the rising edge of FREF.

Feedback and FREF clocks are compared after 16, 32, and 64 cycles. If, after 32 cycles, the clocks match, the LCK0 bit is set to 1. If, after 64 cycles of FREF, there is the same number of FREF clocks as feedback clocks, the LCK1 bit is also set. The LCK bit stay set until:

- Clocks fail to match
- On reset caused by LCKON, PLL_PDN
- Chip_level reset

When the circuit sets the LCK1, the two counters are reset and start the count again. The lock detector is designed so if LCK1 is reset to zero because clocks did not match, LCK0 can stay high. This provides the processor the accuracy of the two clocks with respect to each other.

5.8 Loss of Reference Clock Detector

The Loss of Reference Clock Detector is designed to generate an interrupt when the reference clock to the PLL is interrupted. An LOR interrupt should occur after LORTP × 10 × PLL-output-clock-time-periods. **Figure 5-2** illustrates the general operation of the LOR detector. The operation of the LOR detector relies on the phase lock loop continually running for a time after its reference clock is disturbed. This provides time for detection of the problem and an orderly shutdown.

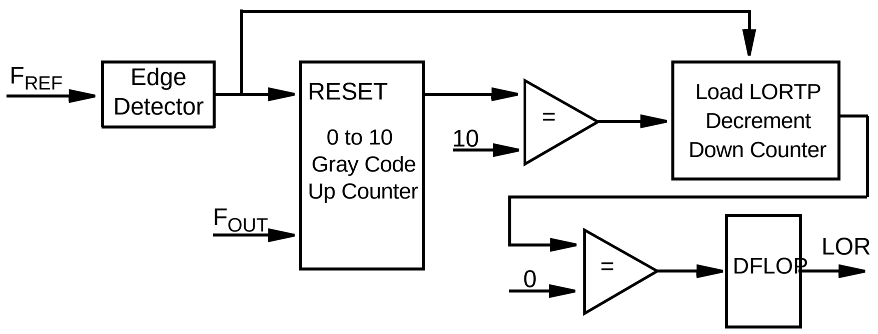

F_{REF}
Edge Detector
RESET
0 to 10 Gray Code Up Counter
F_{OUT}
10
=
Load LORTP Decrement Down Counter
0
=
DFLOP
LOR

Figure 5-2. Simplified Block Diagram, Loss of Reference Clock Detector

5.9 Operating Modes

Either an external crystal, or an external frequency source can be used to provide a reference clock (SYS_CLK_2×) to the 56F801X family parts SIM.

The 2× System Clock source output from the OCCS can be described with one of the following two equations:

$$2\times \text{ system frequency} = \text{oscillator frequency}$$

$$2\times \text{ system frequency} = (\text{oscillator frequency X } 8) / (\text{postscaler})$$

Where:

postscaler = 1, 2, 4, 8, 16, or 32 PLL output divider

The SIM is responsible for further dividing these frequencies by two, insuring a 50 percent duty cycle in the system clock output.

The on-chip synthesis module of the 56F801X family parts has the following registers:

- PLL Control (CTRL) register
- Divide-By (DIVBY) register
- PLL Status (STAT) register
- Shutdown (SHUTDN) register
- Oscillator Control (OCTRL) register

For more information on these registers, please refer to register descriptions in **Section 5.10.**

5.9.1 Internal Clock Source

The Relaxation Oscillator is optimized for accuracy and programmability while providing several different power-saving configurations, accommodating different operating conditions. The internal oscillator has very little variability with temperature and voltage, but it does vary as much as ± 20 percent as a function of wafer fabrication process. It also is very fast in reaching a stable frequency (well under 1μsec). Under typical conditions the circuit provides an 8MHz clock at the center of its tuning range. The tuning range is controlled by 10 bits, with each tuning bit providing a binary weighted change from the previous bit. The maximum tuning step size is 40 percent while the minimum tuning step size is .08 percent. To optimize power, the architecture supports a standby state and a power-down state. During the reset sequence, the internal oscillator will be enabled by GPIO6/RXD. Application code can then switch to the external source and power-down the internal oscillator if desired.

5.9.2 External Clock Source

The recommended method of connecting an External Clock is illustrated in **Figure 5-3**. The External Clock Source is connected to GPIO6/RXD.

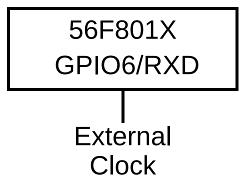


Figure 5-3. Connecting an External Clock Signal Using GPIO6/RXD

5.10 Register Definitions

Table 5-2. OCCS Memory Map

Device	Peripheral	Base Address
56F801X	OCCS	$00F0F0

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 5-3 lists the OCCS registers in ascending address order, including their acronyms and address offset of each register. The OCCS has five registers.

Table 5-3. OCCS Register Summary

Address Offset	Address Acronym	Register Name	Access Type	Chapter Location
Base + $0	CTRL	Control Register	Read/Write	**Section 5.10.1**
Base + $1	DIVBY	Divide-By Register	Read/Write	**Section 5.10.2**
Base + $2	STAT	Status Register	Read/Write	**Section 5.10.3**
Base + $4	SHUTDN	Shutdown Register	Read/Write	**Section 5.10.4**
Base + $5	OCTRL	Oscillator Control Register	Read/Write	**Section 5.10.5**

Bit fields of each of the five registers are illustrated in **Figure 5-4**. Details of each follow.

Add. Offset	Register Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	CTRL	R	PLLIE1		PLLIE0		LOCIE	0	0	0	LCKON	CHPMPTRI	0	PLLPD	0	PRECS	ZSRC	
		W						▨	▨	▨			▨		▨			
$1	DIVBY	R	LORTP				0	PLLCOD			0	0	0	0	0	0	0	0
		W					▨				▨	▨	▨	▨	▨	▨	▨	▨
$2	STAT	R	LOLI1	LOLI0	LOCI	0	0	0	0	0	0	LCK1	LCK0	PLL PDN	0	0	ZSRCS	
		W				▨	▨	▨	▨	▨	▨				▨	▨		
$4	SHUTDN	R	SHUTDOWN															
		W																
$5	OCTRL	R	ROPD	ROSB	0	0	0	0	TRIM									
		W			▨	▨	▨	▨										

R	0	Read as 0
W	▨	Reserved

Figure 5-4. OCCS Register Map Summary

5.10.1 Control Register (CTRL)

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	PLLIE1		PLLIE0		LOCIE	0	0	0	LCKON	CHPMPTRI	0	PLLPD	0	PRECS	ZSRC	
Write						▨	▨	▨			▨		▨			
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Figure 5-5. Control (CTRL) Register

5.10.1.1 PLL Interrupt Enable 1 (PLLIE1)—Bits 15–14

An optional interrupt can be generated when the PLL Lock (LCK1) status bit in the PLL Status (STAT) register changes:

- 00 = Disable interrupt
- 01 = Enable interrupt on any rising edge of LCK1
- 10 = Enable interrupt on falling edge of LCK1
- 11 = Enable interrupt on any edge change of LCK1

5.10.1.2 PLL Interrupt Enable 0 (PLLIE0)—Bits 13–12

An optional interrupt can be generated if the PLL Lock (LCK0) status bit in the PLL Status (STAT) register changes:

- 00 = Disable interrupt
- 01 = Enable interrupt on any rising edge of LCK0
- 10 = Enable interrupt on falling edge of LCK0
- 11 = Enable interrupt on any edge change of LCK0

5.10.1.3 Loss of Reference Clock Interrupt Enable (LOCIE)—Bit 11

Loss of the reference clock circuit monitors the output of the selected clock source. In the event of reference clock loss, an interrupt can be generated.

- 0 = Interrupt disabled
- 1 = Interrupt enabled

5.10.1.4 Reserved—Bits 10–8

This bit field is reserved or not implemented. It is read and written as 0.

5.10.1.5 Lock Detector On (LCKON)—Bit 7

- 0 = Lock detector disabled
- 1 = Lock detector enabled

5.10.1.6 Charge Pump Tri-State (CHPMPTRI)—Bit 6

During normal chip operation the CHPMPTRI bit should be set to a value of zero. In the event of loss of reference clock, the CHPMPTRI bit must be set to a value of one.

- 0 = Normal operation

- 1 = Isolates the charge pump from the loop filter allowing the PLL output to slowly drift, thereby providing enough time to shutdown the chip. Activating this bit will render the PLL inoperable and should not be executed during standard operation of the chip.

Additionally, this bit is used for isolating the charge pump from the loop filter so the filter voltage can be driven from an external source during bench test evaluations.

5.10.1.7 Reserved—Bit 5

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.1.8 PLL Power-Down (PLLPD)—Bit 4

The PLL can be turned off by setting the PLLPD bit. There is a four IPBus clock delay from changing the bit to signaling the PLL. When the PLL is powered down, the gear shifting logic automatically switches to ZSRC=1, preventing loss of reference clock to the core.

- 0 = PLL enabled
- 1 = PLL powered down

5.10.1.9 Reserved—Bits 3

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.1.10 Prescaler Clock Select (PRECS)—Bit 2

This bit is used to select between the external clock source or the Relaxation Oscillator.

- 0 = Internal Relaxation Oscillator selected (reset value)
- 1 = External reference selected

Note: This bit should not be set unless the external reference (CLKIN) is enabled.

5.10.1.11 Clock Source (ZSRC)—Bits 1–0

The Clock Source (ZSRC) determines the SYS_CLK_×2 source to the SIM module, generating divided down versions of this signal for use by memories and the IPBus. ZSRC is automatically set to one during Stop mode, or when PLLPD is set, preventing loss of the reference clock to the core. For the 20 family parts, ZSRC may have the following values.

- 00 = Reserved
- 01 = MSTR_OSC
- 10 = Postscaler output
- 11 = Reserved

5.10.2 Divide-By Register (DIVBY)

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	LORTP				0	PLLCOD			0	0	0	0	0	0	0	0
Write																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5-6. Divide-By (DIVBY) Register

5.10.2.1 Loss of Reference Clock Timer Period (LORTP)—Bits 15–12

These bits control the amount of time required for the loss of reference clock interrupt to be generated. This failure detection time is LORTP × 10 × PLL-output-clock-time-period.

The *Loss of Reference Clock Detector* block counts FOUT/2 clocks continuously, illustrated in **Figure 5-1**. The MSTR_OSC clock input resets this counter. If the counter ever exceeds LORTP × 10 the *loss of reference clock interrupt* is generated.

Note: When programming the PLLCOD field be sure to not zero out the LORTP field. A zero value for LORTP will preemptively set the LOCI bit in the PLL Status register, rendering the Loss of Clock Interrupt useless.

5.10.2.2 Reserved—Bit 11

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.2.3 PLL Clock Out Divide or Postscaler (PLLCOD)—Bits 10–8

The PLL output clock can be divided down by a 2-bit postscaler, defined in **Table 5-4**. The output of the postscaler is a selectable clock source for the core as determined by the ZSRC bit in the CTRL register.

Table 5-4. PLLCOD Values and System Clock Speed

PLL Settings	Divide-By Value	System Clock Speed (MHz) (Assuming ZSRC = 10)
000	1	32
001	2	16
010	4	8
011	8	4
100	16	2
101	32	1
11x	32	1

5.10.2.4 Reserved—Bits 7–0

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.3 Status Register (STAT)

Loss of Lock Interrupt is cleared by writing 1 to the respective LOLI bit in the STAT register. The Loss of Reference Clock Interrupt is cleared by writing 1 to the LOCI bit in the STAT register, or disabling the interrupt in the CTRL register. A PLL interrupt is generated if any of the LOLI or LOCI bits are set and the respective interrupt enable is set in the CTRL register.

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	LOLI1	LOLI0	LOCI	0	0	0	0	0	0	LCK1	LCK0	PLLPDN	0	0	ZSRCS	
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Figure 5-7. Status (STAT) Register

5.10.3.1 Loss of Lock Interrupt 1 (LOLI1)—Bit 15

LOLI1 displays the status of the lock detector state from LCK1 circuit. The interrupt is cleared by writing 1 to LOLI1.

- 0 = PLL locked
- 1 = PLL unlocked

5.10.3.2 Loss of Lock Interrupt 0 (LOLI0)—Bit 14

LOLI0 shows the status of the lock detector state from LCK0 circuit. The interrupt is cleared by writing 1 to LOLI0.

- 0 = PLL locked
- 1 = PLL unlocked

5.10.3.3 Loss of Reference Clock Interrupt (LOCI)—Bit 13

LOCI shows the status of the reference clock detection circuit.

The interrupt can be cleared by writing 1 to LOCI, but this is not recommended. A set LOCI bit indicates a loss of reference clock. Nominally the application should immediately set the Charge Pump Tri-State (CHPMPTRI) bit in the Control register and put the device into a safe mode (peripheral shutdown followed by STOP) using the remaining good clocks cycles provided.

- 0 = Oscillator clock normal
- 1 = Lost oscillator clock

5.10.3.4 Reserved—Bits 12–7

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.3.5 Loss of Lock 1 (LCK1)—Bit 6

- 0 = PLL is unlocked
- 1 = PLL is locked (fine)

5.10.3.6 Loss of Lock 0 (LCK0)—Bit 5

- 0 = PLL is unlocked
- 1 = PLL is locked (coarse)

5.10.3.7 PLL Power-Down (PLLPDN)—Bit 4

PLL power-down status is delayed by four IPBus clocks from the PLLPD bit in the CTRL register.

- 0 = PLL not powered down
- 1 = PLL powered down

5.10.3.8 Reserved—Bits 3–2

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.3.9 Clock Source Status (ZSRCS)—Bit 1–0

ZSRCS indicates the current SYS_CLK_x2 clock source. Since the synchronizing circuit switches the system clock source, ZSRCS takes more than one IPBus clock to indicate the new selection.

- 00 = Synchronizing in progress
- 01 = Prescaler output
- 10 = Postscaler output
- 11 = Synchronizing in progress

5.10.4 Shutdown Register (SHUTDN)

This register can be used to shutdown all system clocks, including SYS_CLK_×2, SYS_CLK.

Caution: *This is a terminal condition.* The *only* recovery is to assert the $\overline{\text{RST}}$.

This function is intended to place the device in a known state in the following specific circumstance:

- The Loss of Reference (LOR) interrupt is, and continues to be, asserted AND
- The CHPMPTRI bit in the CTRL register is set, AND
- The value 0×DEAD is written to this register

This procedure occurs when the chip clock source or crystal input dies, an LOR interrupt is issued, and the PLL is placed in a mode where it ignores the reference clock input and continues to run open loop. The PLL is designed to run at the target frequency for at least 1000 cycles. During this time, the LOR routine can shutdown the system in a controlled manner, terminating in actually shutting down the system clocks as well.

A write of 0×DEAD, or any other value, to this register will have no effect unless the first two conditions above are met. If the LOR is clear, or CHPMPTRI is not set, no action is taken and the register is not written.

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	SHUTDOWN															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5-8. Shutdown (SHUTDN) Register

Clock shutdown may take several cycles, so the write to this register should be followed by an idle loop, simply branching to itself.

5.10.5 Oscillator Control Register (OCTRL)

This register control aspects of the Relaxation Oscillator is illustrated in **Figure 5-9**.

Base + $5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ROPD	ROSB	0	0	0	0	TRIM									
Write																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Figure 5-9. Oscillator Control (OSCTL) Register

5.10.5.1 Relaxation Oscillator Power-Down (ROPD)—Bit 15

This bit operates the power-down usage of the crystal oscillator. It may be power-down the Relaxation Oscillator if the external reference is being used. In order to prevent loss of clock to the core or the PLL, this bit should only be asserted if the clock source is changed to the external source by setting the PRECS bit in the CTRL register.

- 0 = Relaxation Oscillator enabled
- 1 = Relaxation Oscillator powered down

5.10.5.2 Relaxation Oscillator Standby (ROSB)—Bit 14

This bit controls the power usage and gross frequency of the Relaxation Oscillator. It is reset to the more accurate but higher power state.

- 0 = Normal mode. The Relaxation Oscillator output frequency is 8MHz.
- 1 = Standby mode. The Relaxation Oscillator output frequency is reduced to 400KHz (± 50 percent). The PLL should be disabled in this mode and MSTR_OSC should be selected as the output clock.

5.10.5.3 Reserved—Bits 13–10

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

5.10.5.4 Relaxation Oscillator Trim (TRIM)—Bits 9–0

These bits change the size of the internal capacitor used by the Relaxation Oscillator. By testing the frequency of the internal clock and incrementing or decrementing this factor accordingly, the accuracy of the internal clock can be improved by 40 percent. Incrementing these bits by one increases the clock period by 0.078 percent of the unadjusted value. Decrementing this register by one decreases the clock period by 0.078 percent. Reset sets these bits to $200, centering the range of possible adjustment.

5.11 Clocks

Table 5-5 summarizes the various clock signals running around the OCCS module. All clocks are ultimately derived from the oscillator output.

Table 5-5. Clock Summary

Clock	Source	Characteristics
IPBus Clock	IPBus Bridge	Derived from SYS_CLK and has the same frequency. Used for all peripheral register reads and writes.
SYS_CLK_x2	This Module	Primary source for all on-chip clocks excluding the oscillator clock used by the FlexCAN module. This signal is divided by two in the SIM to generate the master system frequency.
Oscillator Output	0scillator	This is the starting point for all clocks.
Feedback	PLL	Feedback pin of the PLL.
F_{OUT}	This Module	Output of the PLL.
Postscaler Output	This Module	May be used as a source for SYS_CLK_x2.

5.12 Interrupts

The interrupts listed in Table 5-6 are ORed into a single processor core interrupt. The ISR servicing this interrupt should check status bits in STAT register to determine which interrupt fired.

Table 5-6. Interrupt Summary

Interrupt	Source	Description	Reference
LOLI1	STAT	Lock 1 Interrupt	Section 5.10.3.1
LOLI0	STAT	Lock 2 Interrupt	Section 5.10.3.2
LOCI	STAT	Loss of Reference Clock Interrupt	Section 5.10.3.3

Chapter 6
Flash Memory (FM)

Document Revision History for Chapter 6, Flash Memory (FM)

Version History	Description of Change
Rev. 0	Initial release
Rev. 1	Cross reference error: **Figure 6-1**. Protection Word should cross referenence **Figure 6.7.4**
Rev. 2	Corrected FAQ number in **Table 6-3** footnote to FAQ 25630, see *freescale.com*.

6.1 Introduction

The 28 Flash Memory (FM) module is a non-volatile memory module, operating with the core and IP Buses, providing program Flash memory.

The Program Flash Memory serves as electrically erasable and programmable memory (EEPROM). It is ideal for program storage for single-chip applications, and supports field reprogramming without requiring external programming voltage sources.

The programming voltage required to program and erase Flash is generated internally by on-chip charge pumps. Program and erase operations are performed by a command driven interface from the DSC core controller using a state machine internal to the module.

6.2 Features

- 12K or 16K bytes of Program Flash Memory
- 32MHz single cycle operation for all Program Flash accesses. Worst case conditions are specified to be 150°C (T_J) and 2.25V.
- Automated program and erase operation
- Interrupts on command completion, command buffer empty, and access error
- Fast page erase
- Single power supply program and erase
- Security feature
- Sector protection system
- The Flash Memory (FM) supports byte and word/read operations by the host Digital Signal Controller (DSC)
- Code integrity check using built-in data signature calculation

6.3 Block Diagram

The Flash Memory (FM) demonstrated in **Figure 6-1,** contains Flash array Blocks, the 28 program buses, IP Interface Control, Flash Interface, and Register Blocks. The Flash block is an array of electrically erasable and programmable, non-volatile memory for use as program (P:) memory.

Program Memory of 64k to 16k bytes in length is constructed from a 64k, 32k, or 16k byte block. Read access occurs within one cycle with no penalty.

A 64k block is organized as 1024 rows of 64 bytes each. A 32K block is organized as 512 rows of 64 bytes each. A 16K block is organized as 256 rows of 64 bytes each.

An erase page contains eight rows of 64 bytes giving 512 bytes. The erase operation also supports mass erase of an entire block. An erased bit reads as one and a programmed bit reads as zero.

All registers are memory mapped for easy access to control program and erase operations.

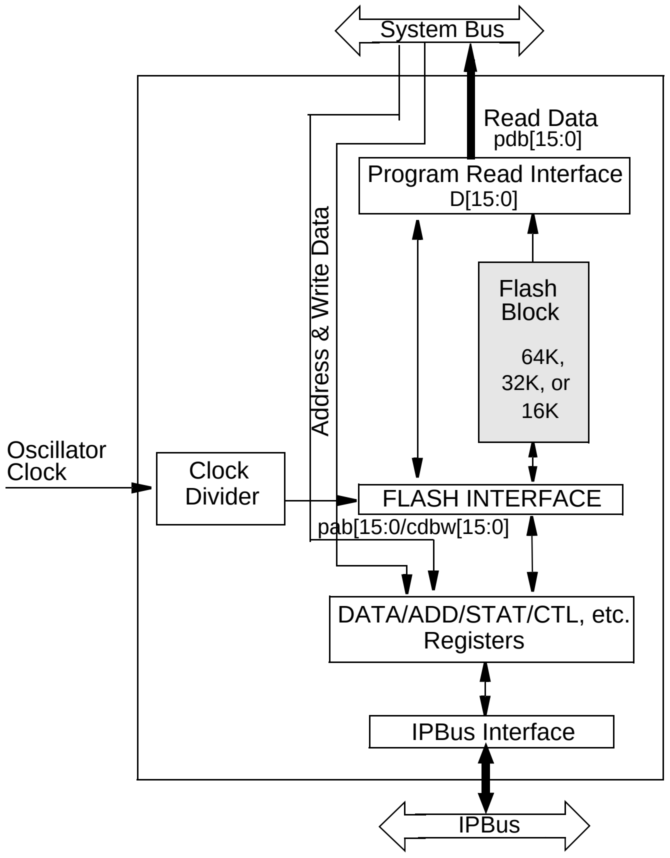


Figure 6-1. Flash Block Diagram

6.4 Memory Map

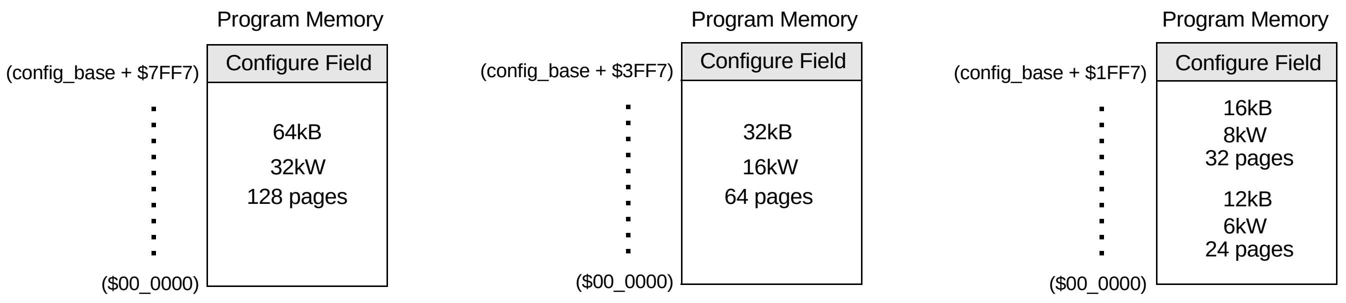


Figure 6-2. Flash Array Memory Map

The configuration field contains data to facilitate both security and protection features. Protection refers to undesired core access while security refers to undesired external access. The configuration field is composed of the top nine words of program Flash Memory, containing information determining the module’s protection and access restriction scheme out of reset. The protection word in the configuration field is transferred into the Protection (PROT) register at reset. The security words in the configuration field are transferred into the Security (SECLO, SECHI) registers upon reset. A description of each value in the configuration field is provided in **Table 6-1.** For further details refer to the Protection and Security register descriptions.

Table 6-1. Flash Memory Configuration Field

Address			Description	Word Name
64k	32k	16k		
$7FFF	$3FFF	$1FFF	Back Door Comparison Key 3	BACK_KEY_3_VALUE
$7FFE	$3FFE	$1FFE	Back Door Comparison Key 2	BACK_KEY_2_VALUE
$7FFD	$3FFD	$1FFD	Back Door Comparison Key 1	BACK_KEY_1_VALUE
$7FFC	$3FFC	$1FFC	Back Door Comparison Key 0	BACK_KEY_0_VALUE
$7FFB	$3FFB	$1FFB	Not Used	Not Used
$7FFA	$3FFA	$1FFA	Protection Word (See **Section 6.7.4**)	PROT_VALUE
$7FF9	$3FF9	$1FF9	Not Used	Not Used
$7FF8	$3FF8	$1FF8	Security Word Upper (See **Section 6.7.3**)	SECH_VALUE
$7FF7	$3FF7	$1FF7	Security Word Lower (See **Section 6.7.3**)	SECL_VALUE

Flash Memory also contains a set of Control and Status registers located data memory space at the FM register base register address. A summary of these registers is provided in **Table 6-2.**

Table 6-2. Flash Memory Register Address Map

Offset From Register Base Address[1]	Bits 15-8	Bits 7-0
Base + $0	RESERVED	CLKD[2]
Base + $1	CNFG	
Base + $2	RESERVED	
Base + $3	SECHI	
Base + $4	SECLO	
Base + $5 – $F	RESERVED	
Base + $10	PROT	
Base + $11 – $12	RESERVED	
Base + $13	RESERVED	USTAT
Base + $14	RESERVED	CMD
Base + $15 – $17	RESERVED	
Base + $18	DATA	
Base + $19 – $1A	RESERVED	
Base + $1B	OPT1	
Base + $1C	RESERVED	
Base + $1D	TSTSIG	
Base + $1E – $3C	RESERVED	

6.5 Operating Modes

This section details the modes of operation of Flash Memory. Table 6-3 shows all of the available user commands.

6.5.1 Read Operation

The FM module provides transparent read access of all memory locations in its Flash array. Each read is achieved without wait states, providing memory access without delays.

6.5.2 Write Operation

There are two types of write access supported by the FM module. Normal write access, where a value is *written* to a memory location within the program Flash memory array, is part of a programming attempt. Writing the data must be followed by writing a program command to the FM module's Command (CMD) register.

Erased Flash is all ones, so any programming attempt will write only the required zeros to create the desired bit pattern. It is up to the user to verify the erased state of the destination in Flash before programming it.

If the KEYACC bit is set in the Configuration (CNFG) register, all writes to the Flash are interpreted as attempts to unlock program Flash by matching the 64-bit key string found in the four word memory. For more information about this write mode, please see **Section 6.6.1**, Back Door Access.

6.5.3 Program and Erase Operation

Both write and read operations are used for the program and erase algorithms described in this section. These algorithms are controlled by the module’s state machine.

The Flash Command (CMD) register operates as a buffer and a register (two-stage FIFO). Consequently, a new command, along with the necessary data and address, can be stored in the buffer while the previous command is still in progress. This feature increases the rate at which the FM state machine receives commands, reduces command processing time, and expedites program/erase cycles. Buffer Empty as well as Command Completion are signaled by flags in the Flash Status (STAT) register. Interrupts are generated when enabled.

6.5.4 User Mode Commands

Table 6-3 summarizes the valid Flash User commands.

Table 6-3. Flash User Mode Commands

FM Command	Meaning	Description	Array Write Address/Data Usage
$05	Erase Verify	Verify the Flash array is erased. If the array is verified to be blank, the BLANK bit will set in the USTAT register, **Figure 6-12**, upon command completion.	The address must be any address within the array. Data is ignored.
$06	Calculate Data Signature	Calculate a signature over a user-specified range of words in the Flash array. The resulting signature is returned in the DATA register, **Figure 6-17**, at the completion of the command. The new signature can be compared with a previously calculated signature (performed by user)[1] to verify program Flash integrity.	The array write address specifies the starting address and data specifies the number of words to calculate the signature over.
$20	Program	Program 16-bit word. A program is only possible when the protection bit for that sector is not set.	The address and data specifies the address and value to program.
$40	Page Erase	Erase a specific page (sector) of the Flash array. A page erase is only possible when the PROTECT bit for the selected page is not set.	The address must be any address within the page to be erased. Data is ignored.
$41	Mass Erase	Erase all Flash memory. A mass erase is only possible when no PROTECT bits are set.	The address must be any address within the array. Data is ignored.

Table 6-3. Flash User Mode Commands (Continued)

FM Command	Meaning	Description	Array Write Address/Data Usage
$66	Calculate IFR Block Signature	Calculate a signature over the Flash Information Row (IFR) of the Flash array. The resulting signature is returned in the DATA register, **Figure 6-14**, at the completion of the command. The entire IFR row 1 is compressed except the last word containing the expected compression result. The IFR Block Signature calculated at the factory is available, for comparison to the new value, in the TSTSIG register. If the value in DATA differs with that found in the TSTSIG register then Flash programming parameters stored in the IFR Block have been corrupted.	The address must be any address within the array. Data is ignored.

1. PERL script exists which can be used to calculate the signature. See FAQ 25630 on *freescale.com*.

6.5.5 Command Sequence Operation

This section describes how to perform a command sequence. This algorithm involves the use of an Array Write operation described above, as well as read/writes of FM registers. These algorithms are controlled by a state machine whose time base (FCLK) is derived from the FM system clock using a programmable prescaler and divider controlled by fields in the CLKDIV register. Please see **Section 6.5.5.1**.

Parameters required for timed events in program and erase algorithms are loaded from the Flash Information Row (IFR). *Buffer Empty* as well as *Command Complete* are signalled by flags in the Status register. Interrupts are generated if enabled.

While the algorithm can be used to process commands to completion one at a time, the Command register operates as a buffer and a register (two-stage FIFO), so a new command along with the necessary data and address can be stored in the buffer while the previous command is still in progress. This technique can be applied to all commands except calculate signature commands.

6.5.5.1 Writing the CLKDIV Register

Prior to the execution of any Flash module command, the Flash module Clock Divider (CLKDIV) register must be initialized. *The values of this register determine the speed of the internal Flash Clock (FCLK). FCLK must be in the range of 150kHz ≤ FCLK ≤ 200kHz for proper operation of the Flash module.* (Running FCLK too slowly wears out the module, while running it too fast under programs Flash leading to bit errors.)

Table 6-4 provides CLKDIV register values for various operating calculations. The table shows values when a external clock is used as well as cases where the on-chip Relaxation Oscillator (ROSC) is used. The OCCS parameters required for each clock setup are shown in the columns for PRECS, PLLCOD, and ZSRC. The CTRL register in OCCS contains PRECS and ZSRC. PLLCOD is located in the DIVBY register in the OCCS peripheral.

The parameter T_{BUS} represents the largest period of the device's system clock, measured in microseconds. If $T_{BUS} > 1\mu sec$, it is not possible to execute Flash module commands. Reading Flash is still possible, but writing (programming new values) or erasing Flash is not possible until $T_{BUS} \leq 1\mu sec$.

The DIVLD bit in the CLKDIV register shows if the register has been initialized since the last reset. Applications should check this bit before initiating any Flash module commands.

Table 6-4. Flash Module Clock Parameters for Various Operating Conditions

Input Clock *MSTR_OSC*	Freq (MHz)	Prescaler Clock *PRECS*	PLL Divider *PLLCOD*	Output Clock Source	*ZSRC*	Sys_Clk (MHz)	T_{BUS} (usec)	T_{BUS} <=1us ?	PRDIV8	DIV	FCLK
External	1	1	n/a	MSTR_OSC	01	0.5	2	NO	~~0~~	~~5~~	~~166.7~~
External	2	1	n/a	MSTR_OSC	01	1	1	YES	0	10	181.8
External	4	1	n/a	MSTR_OSC	01	2	0.5	YES	0	20	190.5
ROSC	0.4	0	n/a	MSTR_OSC	01	0.2	5	NO	~~0~~	~~2~~	~~133.3~~
ROSC	8	0	n/a	MSTR_OSC	01	4	0.25	YES	0	40	195.1
ROSC	8	0	32	Postscaler	10	1	1	YES	0	41	190.5
ROSC	8	0	16	Postscaler	10	2	0.5	YES	0	40	195.1
ROSC	8	0	8	Postscaler	10	4	0.25	YES	0	40	195.1
ROSC	8	0	4	Postscaler	10	8	0.125	YES	0	40	195.1
ROSC	8	0	2	Postscaler	10	16	0.0625	YES	0	40	195.1
ROSC	8	0	1	Postscaler	10	32	0.03125	YES	0	40	195.1
External	16	1	n/a	MSTR_OSC	01	8	0.125	YES	1	10	181.8
External	32	1	n/a	MSTR_OSC	01	16	0.0625	YES	1	20	190.5
External	64	1	n/a	MSTR_OSC	01	32	0.03125	YES	1	40	195.1

A spreadsheet tool is available in the Freescale FAQs for this family. This spreadsheet, illustrated in **Figure 6-3**, allows the recalculation of PRDIV8 and DIV values for operating scenarios not included in **Figure 6-4**. Simply search for FAQ number 25464 on the Freescale website.

Frequency of MSTR_OSC Clock (MHz)	Fosc	8	**<- Enter**
Frequency of System Clock (MHz)	Fsys	32	**<- Enter**
System clock period (µSeconds)	T_{BUS}	0.03125	
Is $T_{BUS} \leq = 1$µSecond?	Continue?	**Continue**	
Enable Prescaler Divide By 8	**PRDIV8**	**0**	**<- Use**
Output Stage Frequency After Divider (MHz)	Fadj	8	
Provisional Divider Value	Prov DIV	39	
Provisional Flash Module Clock (kHz)	Prov Fclk	200	
Final Divider Value	**DIV**	**40**	**<- Use**
Final Flash Module Clock (kHz)	**Fclk**	**195.122**	

Figure 6-3. FM Clock Parameter Tool

6.5.5.2 Command Sequence Protocol

A Command State Machine supervises command processing. To prepare for a command execution, the CBEIF flag in the USTAT register should be tested, ensuring that the address, data, and command buffers are empty. If the CBEIF flag is set, the command sequence can be started.

You must follow the command write sequence below to successfully program Flash.

Note: The command write sequence must execute from RAM.

Intermediate writes to the FM are not permitted between the three steps.

Command Write Sequences (See Figure 6-4):

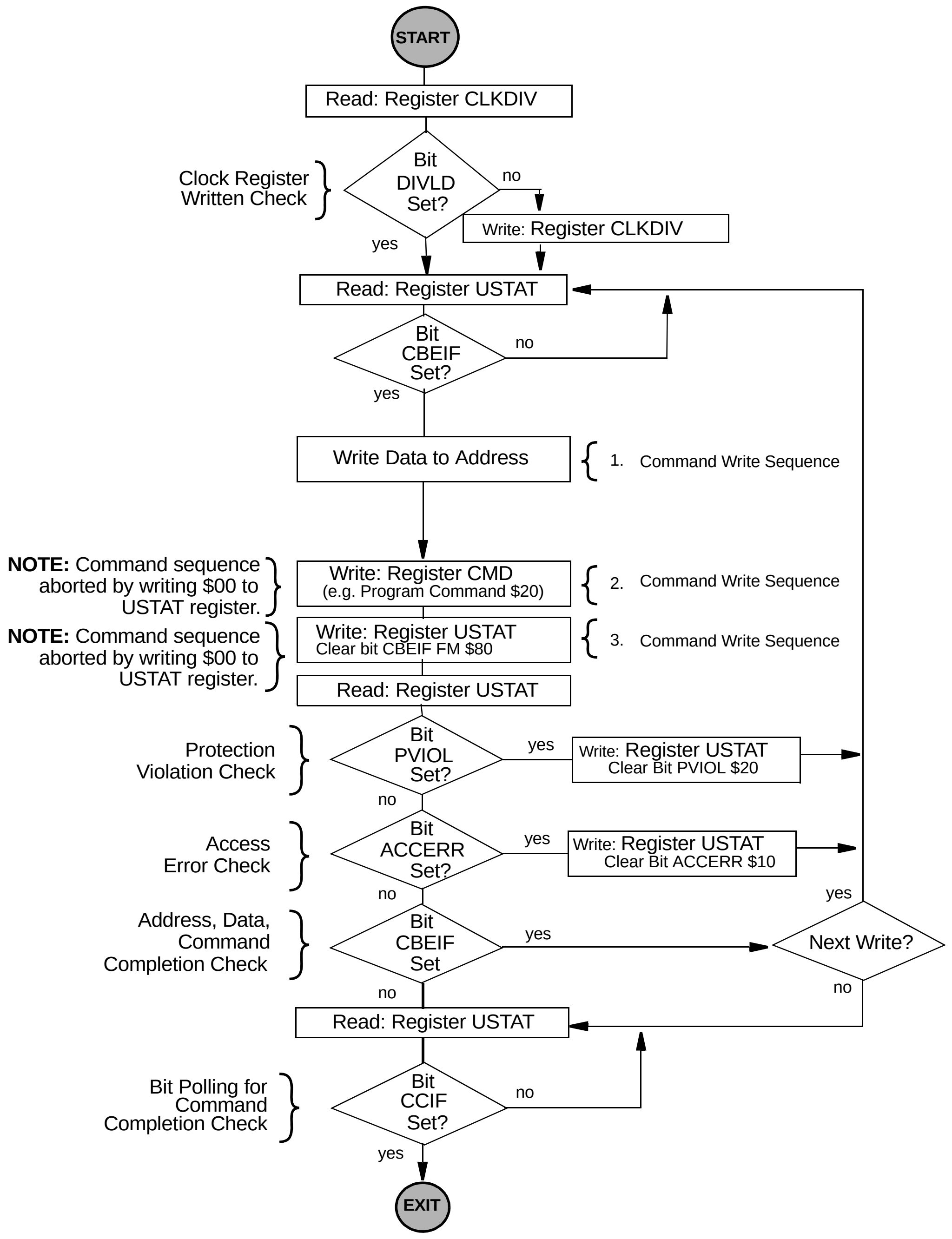
1. Write the desired value to the address in Flash memory you wish to program.
2. Write the numeric code for the command to the Command Buffer (CMD) register. Flash module commands are described in **Table 6-3**.
3. To launch the command, clear the CBEIF flag by writing 1. After the CBEIF flag is cleared, the CCIF flag in the STAT register will be cleared by hardware, indicating the command was successfully launched. The CBEIF flag will be set again indicating the address, data and command buffers are ready for a new Command Write sequence to begin.

The completion of the command operation is indicated by the CCIF flag setting.

The Command State Machine will flag errors in Program or Erase Write sequences by means of the Access Error (ACCERR) and Protection Violation (PVIOL) flags in the USTAT register. An erroneous Command Write sequence will abort, setting the appropriate flag. If set, the ACCERR or PVIOL flags must be cleared before starting another Command Write sequence.

Note: By writing zero to the CBEIF flag, the command sequence can be aborted after the word write to the Flash address space, or after writing a command to the CMD register and before the command is launched. The ACCERR flag is set on aborted commands and must be cleared before a new command is launched.

A flow chart of the entire command sequence is provided in **Figure 6-4**.



START
Read: Register CLKDIV
Clock Register Written Check
Bit DIVLD Set?
no
Write: Register CLKDIV
yes
Read: Register USTAT
Bit CBEIF Set?
no
yes
Write Data to Address
1. Command Write Sequence
NOTE: Command sequence aborted by writing $00 to USTAT register.
Write: Register CMD (e.g. Program Command $20)
2. Command Write Sequence
NOTE: Command sequence aborted by writing $00 to USTAT register.
Write: Register USTAT Clear bit CBEIF FM $80
3. Command Write Sequence
Read: Register USTAT
Protection Violation Check
Bit PVIOL Set?
yes
Write: Register USTAT Clear Bit PVIOL $20
no
Access Error Check
Bit ACCERR Set?
yes
Write: Register USTAT Clear Bit ACCERR $10
no
Address, Data, Command Completion Check
Bit CBEIF Set
yes
Next Write?
yes
no
no
Read: Register USTAT
Bit Polling for Command Completion Check
Bit CCIF Set?
no
yes
EXIT

Figure 6-4. Command Sequence Flow Chart[1]

6.5.5.3 Flash User Mode Illegal Operations

The ACCERR flag is set during the Command Write sequence if any of the following illegal operations are performed. These operations will cause the Command State Machine to immediately abort. Writes to the Flash Memory Address space refer to writes through the 16-bit controller core buses, not the IP register bus. The following are illegal operations and will have undetermined results:

- Writing to the Flash address space before initializing CLKD
- Writing to the Flash address space while CBEIF in USTAT is not set
- Writing a second word to the Flash address space
- Writing an invalid user command to the CMD register
- Writing to any Flash register other than the CMD register after writing a word to the Flash address space
- Writing a second command to the CMD register before executing the previously written command
- Writing to any FM register other than the USTAT register (to clear CBEIF) after writing to the CMD register
- The part enters Stop or Wait modes and a program or erase command is in progress; the command is aborted
- Aborting a Command sequence by writing 0 to the CBEIF bit in USTAT register after a word write to the Flash Address space, or after writing a command to the CMD register, and before the command is launched
- Writing to the Array while a Calculate Signature command is running (either IFR or main array)

The Protection Violation (PVIOL) flag is set during the Command/Write sequence after the word write to the Flash address space if any of the following illegal operations are performed. Such operations will cause the Command sequence to immediately abort:

1. Writing a Flash address to program a protected area
2. Page erase of a protected area
3. Writing a Mass erase command to the CMD register while any protection is enabled for that block. Please see **Section 6.7.4.**

If a Flash block is read during execution of an algorithm on that block (e.g., CCIF is low), the read will return non valid data and the ACCERR bit in the USTAT register will not be set.

1. Code must reside in RAM

6.5.5.4 Affects of Wait/Stop Modes

If a command is active (CCIF=0) when the controller enters Wait or Stop mode, the command will be aborted, and the data being programmed or erased is lost. The high voltage circuitry to the Flash will be switched off and a pending command (CBEIF=0) will not be executed once the controller exits Wait or Stop mode. The CCIF and ACCERR flags in the USTAT register will be set if a command is active when the 16-bit controller enters Wait or Stop mode.

WARNING:

As active commands are immediately aborted when the 16-bit controller enters Wait mode, it is *strongly* recommended not to execute the Wait instruction during program and erase execution.

6.6 Flash Security Operation

Flash security provides a means to protect the embedded code within the Flash array from unauthorized external access. The state of Flash Security is reflected in the state of the SECSTAT bit in the SECHI register. The value of the SECSTAT bit at reset is determined by the values in the security words stored in the Flash configuration field. Thus, by appropriately programming the configuration field, the part can be forced into Secure mode at reset or power-up.

Flash security prevents unauthorized external access by the JTAG/EONCE port. Once Flash Security is set, an external user is unable to view or change embedded software and is thus unable to introduce code sequences to undo security or export code.

There are three methods of disabling Flash Security at Run Time:

1. Executing a back door access scheme built into the Application
2. Pass an erase verify check
3. Execute the JTAG lockout recovery routine to mass erase the Flash

Only the first method preserves the content of Flash memory.

6.6.1 Back Door Access

Flash Security can be disabled at run-time by following these directions. The KEYEN bit in the SECHI register must be set to enable Back Door Key access.

1. Set the KEYACC bit in the Configuration (CNFG) register
2. Write the correct four word (64-bit) Back Door Comparison Key to the Flash Memory Configuration field. Please see **Table 6-1.** For example, the addresses for a 16kB block would be: $1FFC – $1FFF. This operation must be composed of four word writes started with the smallest address. For example, write to address $1FFC, $1FFD, $1FFE and

$1FFF in that order for a 16kB block. The four write cycles can be separated by any number of other operations.

3. Clear the KEYACC bit

If all four words written match the Flash content in the configuration field, security is bypassed until the next reset. In the unprotected state the DSC core has full control of Flash memory.

The value of the Flash Security words ($1FF7 – $1FF8), is not changed by the Back Door method of unsecuring the device. After the next reset sequence, the device is secured again and the same back door key is in effect, unless the configuration field is changed by program or erase. Flash Security, defined by the Flash Security words in **Table 6-6,** must be changed directly by reprogramming the Flash Security words in memory when the highest sector is unprotected.

The Back Door method of unsecuring the device has no effect on the program and erase protections defined in the Protection (PROT) register.

6.6.2 JTAG Lockout Recovery

To unsecure a secured FM:

- Mass erase the Flash memory via a sequence of JTAG commands, and then on the next reset, the part will be unsecured.
- See the device *Data Sheet* section on security features.

The JTAG lockout recovery sequence is initiated by activating input signal *jtag_lockout_recovery_sec*, facilitating the load of *jtag_fm_data[6:0]* into the CLKDIV register from the JTAG register within the platform. A Mass Erase Command Write sequence is then executed. *If the Mass Erase is successful*, the next reset results in an unsecured part.

6.6.3 Erase Verify Check

A secured module can be unsecured by verifying the Flash array is blank. If required, the Mass Erase command sequence can be executed on the Flash array. The Erase Verify command sequence must then be executed on the Flash array. The Flash module will be unsecured if the erase verify command determines the entire Flash array is blank. After the next reset sequence, the security state of the Flash module is determined again by the lower Flash Security word in the configuration field.

6.7 Register Descriptions

Device	Peripheral	Base Address
56F801X	FM	$00F400

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 6-5 lists the Flash registers in ascending address, including their acronyms and address offset of each register. The Flash peripheral has 11 registers.

Table 6-5. Flash Registers

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	CLKDIV	Clock Divider Register	Read/Write	**Section 6.7.1**
Base + $1	CNFG	Configuration Register	Read/Write	**Section 6.7.2**
Base + $3	SECHI	Security High Half Register	*Read-Only*	**Section 6.7.3**
Base + $4	SECLO	Security Low Half Register	*Read-Only*	
Base + $10	PROT	Protection Register	Read/Write	**Section 6.7.4**
Base + $13	USTAT	User Status Register	Read/Write	**Section 6.7.5**
Base + $14	CMD	Command Register	Read/Write	**Section 6.7.6**
Base + $18	DATA	Data Buffer Register	*Read-Only*	**Section 6.7.7**
Base + $1B	OPT1	Optional Data 1 Register	*Read-Only*	**Section 6.7.8**
Base + $1D	TSTSIG	Test Array Signature Register	*Read-Only*	**Section 6.7.9**

Bits of each of the 11 registers are summarized in **Figure 6-5**. Details of each follow.

Addr. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	CLKDIV	R	0	0	0	0	0	0	0	0	DIVLD	PRDIV8	DIV					
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨							
$1	CNFG	R	0	0	0	0	0	LOCK	0	AEIE	CBEIE	CCIE	KEY ACC	0	0	0	0	0
		W	▨	▨	▨	▨	▨		▨					▨	▨	▨	▨	▨
$3	SECHI	R	KEYEN	SEC STAT	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$4	SECLO	R	SECURITY															
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$10	PROT	R	PROTECT															
		W																
$13	USTAT	R	0	0	0	0	0	0	0	0	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨		▨			▨		▨	▨
$14	CMD	R	0	0	0	0	0	0	0	0	0	COMMAND						
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨							
$18	DATA	R	DATA															
		W																
$1B	OPT1	R	0	0	0	0	0	0	RELAXATION OSCILLATOR TRIM VALUE (ROTV)									
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$1D	TSTSIG	R	TEST SIGNATURE (TSTSIG)															
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨

R	0	Read as 0
W	▨	Reserved

Figure 6-5. Flash Register Map Summary

6.7.1 Clock Divider Register (CLKDIV)

The Flash Memory Clock Divider (CLKDIV) register controls the period of the FCLK used for timed events in program and erase algorithms within the Flash Interface (FI). While the FI operates at system bus frequency, FCLK must operate within the 150-200kHz range. FCLK is generated by dividing the Oscillator Clock (MSTR_OSC in OCCS) by a prescaler and a divider. Please see **Section 6.5.5.1**.

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	DIVLD	PRDIV8	DIV					
Write	▨	▨	▨	▨	▨	▨	▨	▨	▨							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6-6. Clock Divider (CLKDIV) Register

6.7.1.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

6.7.1.2 Clock Divider Loaded (DIVLD)—Bit 7

This is a status-only bit. Writing has no effect.

- 0 = Register has not been written
- 1 = Register has received writing since the last reset

6.7.1.3 Enable Prescaler By 8 (PRDIV8)—Bit 6

This is a read and write bit.

- 0 = Prescaler divides oscillator clock by 1
- 1 = Prescaler divides oscillator clock by 8

6.7.1.4 Clock Divider (DIV)—Bits 5–0

The Clock Divider register bits PRDIV8 and DIV must be set with appropriate values before programming or erasing the FM array. Because FCLK is re-timed into the system clock domain, the values of PRDIV8 and DIV are affected by the system bus frequency as well. Refer to the Functional Description of Writing the CLKDIV register located in **Section 6.5.5.1** for the detailed algorithm to determine the settings for DIV and PRDIV8.

Note: Unlike the 56F8300 Series, bits 6–0 are not write-protected once written. These bits can be modified even after their initialization.

6.7.2 Configuration Register (CNFG)

The Flash Memory Configuration (CNFG) register configures and controls the operation of the FM array.

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	LOCK	0	AEIE	CBEIE	CCIE	KEYACC	0	0	0	0	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6-7. Configuration (CNFG) Register

6.7.2.1 Reserved—Bits 15–11

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

6.7.2.2 Write Lock Control (LOCK)—Bit 10

This bit can always be read. Once set, this bit cannot be cleared except by reset. This bit provides additional security for the Flash array by disabling writes to the protection register.

- 0 = The PROT register can be modified by writing
- 1 = The PROT register is write-locked

6.7.2.3 Reserved—Bit 9

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

6.7.2.4 Access Error Interrupt Enable (AEIE)—Bit 8

This read/write bit enables an interrupt in case of an error accessing the Flash.

- 0 = ACCERR interrupts disabled
- 1 = An interrupt is requested whenever the ACCERR flag is set

6.7.2.5 Command Buffer Empty Interrupt Enable (CBEIE)—Bit 7

This read/write bit enables an interrupt in case of an empty command buffer in the Flash.

- 0 = Command Buffer Empty interrupts disabled
- 1 = An interrupt is requested whenever the CBEIF flag is set

6.7.2.6 Command Complete Interrupt Enable (CCIE)—Bit 6

This read/write bit enables an interrupt in case of all commands being completed in the Flash.

- 0 = Command complete interrupts disabled
- 1 = An interrupt is requested whenever the CCIF flag is set

6.7.2.7 Enable Security Key Writing (KEYACC)—Bit 5

This bit can be read; however, it can receive writing if the KEYEN bit in the SECHI register is set.

- 0 = Flash writes are interpreted as the start of a command sequence, e.g. program or erase
- 1 = Writes to Flash array are interpreted as keys to open the back door

6.7.2.8 Reserved—Bits 4–0

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

6.7.3 Security Registers (SECHI and SECLO)

The SECHI and SECLO registers store the Flash security word, defined in **Table 6-1**.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	KEYEN	SECSTAT	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Write																
Reset	F[1]	F[2]	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. Reset state loaded from Flash array during reset.
2. Reset state determined by security state of module.

Figure 6-8. Security High (SECHI) Register

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	SECURITY															
Write																
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

Figure 6-9. Security Low (SECLO) Register

Bits 15 and 14 of the SECHI register, illustrated in **Figure 6-8**, and each of the 16 bits of SECLO register, illustrated in **Figure 6-8**, can be read; however, neither register can be modified by writing.

The SECLO register is initialized at reset using SECLO_VALUE from the Flash Configuration field. Likewise, the SECHI register is initialized at reset using SECHI_VALUE from the Flash configuration field. The Flash configuration field in the Flash array is described in **Table 6-1**. Value *F* in the reset value shown in **Figure 6-8** indicates bits copied directly from the corresponding Flash Memory Configuration Field in the top nine words of Flash Memory.

The reset value of SECSTAT is determined by the value loaded into the SECURITY field at reset as described in the following four sections. Flash Security can be enabled or disabled during

Read Only Run-Time, discussed in **Section 6.6**, and the current state of Flash Security determines the state of SECSTAT.

6.7.3.1 Enable Back Door Key to Security (KEYEN)—Bit 15

This is a *read-only* bit.

- 0 = Back door to Flash is disabled
- 1 = Back door to Flash is enabled

6.7.3.2 Security Status (SECSTAT)—Bit 14

This is a *read-only* bit.

- 0 = Flash Security is disabled
- 1 = Flash Security is enabled

6.7.3.3 Reserved—Bit 13–0

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

6.7.3.4 Security (SECURITY)—Bits 15–0

This is a *read-only* register. Value loaded into the Security (SECURITY) bit field from the configuration field at reset in turn determines the state of Flash Security at reset (SECSTAT). **Table 6-6** outlines the single code enabling the security feature in the Flash Memory.

Table 6-6. Security States

SECURITY	Description
$E70A	Flash Secured[1] (SECSTAT set at reset)
All other combinations	Flash Unsecured (SECSTAT cleared at reset)

1. The $E70A value was selected because it represents an illegal instruction on the 56800E core, making it unlikely user compiled code accidentally programmed in the security configuration field location would unintentionally secure the Flash.

WARNING:

To perform product analysis when security is enabled, either security must be disabled by the back door key, or the array must be totally erased either by performing the lockout recovery sequence or by performing a mass erase followed by a read verify.

6.7.4 Protection Register (PROT)

This register defines which Flash pages or sectors are protected against program and erase.

Base + $10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	PROTECT															
Write																
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

Figure 6-10. Protection (PROT) Register

This register may always be read, but may be modified by writing only when LOCK=0.

This register value is reset to the PROT_VALUE as defined in the Flash Memory Configuration field; however, this is only possible if the LOCK bit in CNFG is zero. To change the Flash protection loaded on reset, the sector [15] of program Flash must first be unprotected as just described above, then the protection word in the Configuration Field at addresses defined in **Table 6-1** must be programmed with the desired value.

PROT Bit *N* = Protection Field for Sector *N*

PROT Bit 0 $\rightarrow$ Sector 0

PROT Bit 1 $\rightarrow$ Sector 1

PROT Bit 15 $\rightarrow$ Sector 15

On part configurations for which the first quarter of the Flash array is not implemented, bits 0–3 of the PROT register are reserved and read as 0. Bits 4–15 continue to define protection status for the remaining 12 sectors of the Flash array.

- PROTECT[M] = 0: Array sector M is not protected
- PROTECT[M] = 1: Array sector M is protected

The PROT register controls the protection of 16 sectors in a 64K byte section of program memory. **Figure 6-11** outlines the association between each bit in the PROT register and the corresponding Flash Memory sector within the Program Flash.

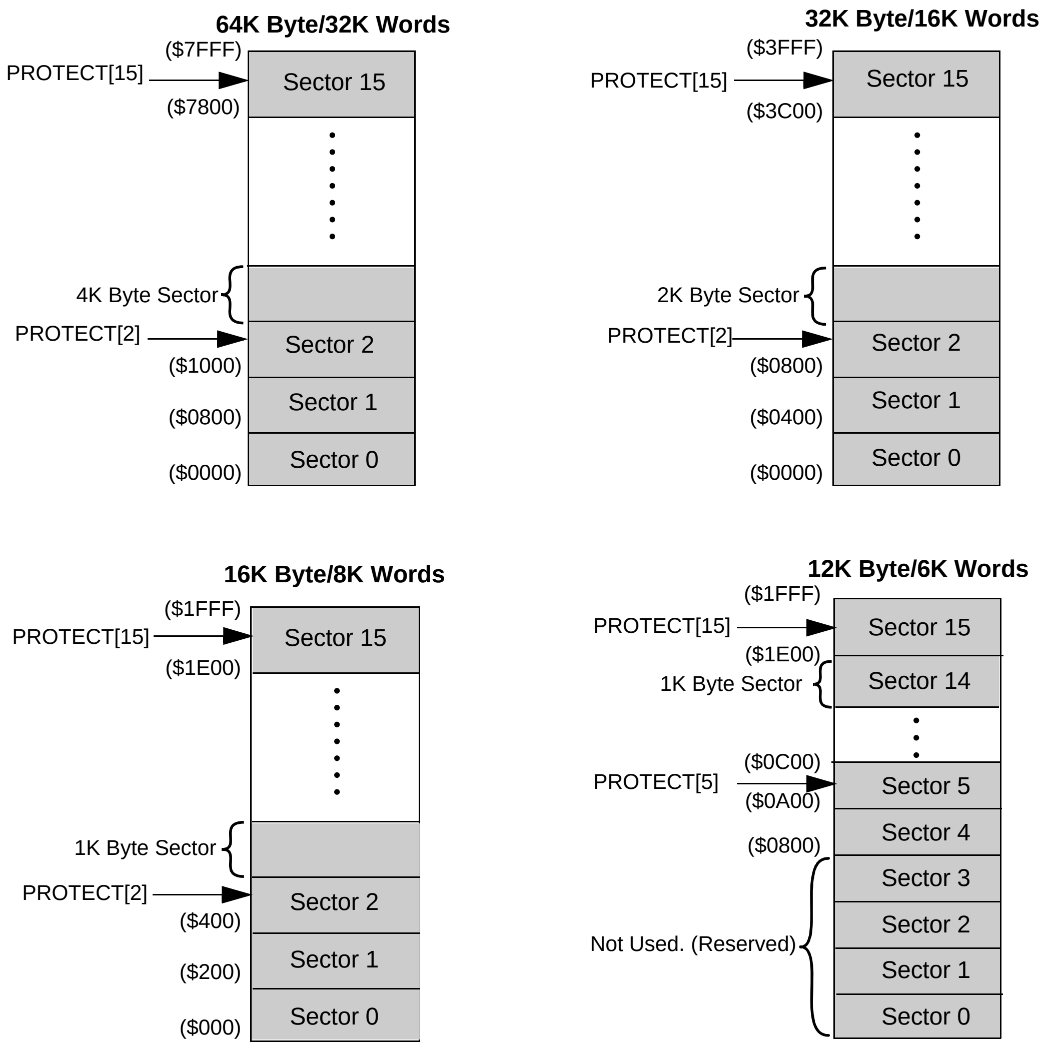


Figure 6-11. Protection Diagram

6.7.5 User Status Register (USTAT)

The USTAT register defines the Flash state machine command status and Flash array access, protection and blank verify status.

Note: Only one bit should be cleared at a time in the USTAT register. This is due to the nature of the state machine clearing the register bits.

Base + $13	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
Write																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Figure 6-12. User Status (USTAT) Register

6.7.5.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is not available to be read or written.

6.7.5.2 Command Buffer Empty Interrupt Flag (CBEIF)—Bit 7

The CBEIF flag indicates the address, data, and command buffers are empty, allowing a new command sequence to be started. The CBEIF flag is cleared by writing 1. Clearing the flag results in the CMD register being transferred to the internal state machine for launch of a command.

Writing 0 has no effect on CBEIF, but it can be used to abort a command sequence. The CBEIF bit can generate an interrupt if the CBEIE bit in the CNFG register is set. While the CBEIF flag is clear the CMD register cannot be modified by writing.

- 0 = Command buffer is full
- 1 = Command buffer is ready to accept a new command

6.7.5.3 Command Complete Interrupt Flag (CCIF)—Bit 6

The CCIF flag indicates there are no other pending commands. The CCIF flag is set and cleared automatically upon start and completion of a command. Writing to CCIF has no effect. The CCIF bit can generate an interrupt if the CCIE bit in the CNFG register is set.

- 0 = Command in progress
- 1 = All commands are completed

6.7.5.4 Protection Violation (PVIOL)—Bit 5

The PVIOL flag indicates an attempt was made to program or erase an address in a protected Flash memory area. The PVIOL flag is cleared by writing 1. Writing 0 has no effect on PVIOL. While the PVIOL flag is set, it is not possible to launch another command.

- 0 = No failure
- 1 = A protection violation has occurred

6.7.5.5 Access Error (ACCERR)—Bit 4

The ACCERR flag indicates an illegal access to the FM array or registers caused by a bad program or erase sequence. The ACCERR flag is cleared by writing 1. Writing 0 to ACCERR bit has no effect. While the ACCERR flag is set, it is not possible to launch another command. The ACCERR relates to FM array writes from the 56F801XE core buses and will not be set by writing directly to the data and address registers from the IPBuses. Please see **Section 6.5.5.3** to set ACCERR flag details.

- 0 = No failure
- 1 = Access error has occurred

6.7.5.6 Flash Block Verified Erased (BLANK)—Bit 2

The BLANK flag indicates an Erase Verify command (RDARY1) checked the Flash block and found it to be blank. The BLANK flag is cleared by writing 1. Writing 0 has no effect.

- 0 = If an Erase Verify command is requested, and the CCIF flag is set, a zero in BLANK indicates the block is not erased.
- 1 = Flash block verifies as erased

6.7.6 Command Register (CMD)

This register defines the Flash Command to be executed.

Base + $14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	COMMAND						
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6-13. Command (CMD) Register

6.7.6.1 Reserved—Bits 15–7

This bit field is reserved or not implemented. It is not available to be read or written, except bit 7 which can be read as 0.

6.7.6.2 Command (COMMAND)—Bits 6–0

Valid User mode commands are shown in **Table 6-7**. Writing a command in User mode other than those listed in **Table 6-7** will cause the ACCERR flag in the USTAT register to be set. Please see **Table 6-3** for a description of the commands.

Table 6-7. Command User Mode Commands

Command	Description
$05	Erase Verify (All Ones)
$06	Calculate Data Signature
$20	Word Program
$40	Page Erase
$41	Mass Erase
$66	Calculate IFR Block Signature

6.7.7 Data Register (DATA)

Read access is permitted to the DATA register.

Base + $18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DATA															
Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6-14. Data (DATA) Register

6.7.7.1 Data Buffer (DATA)—Bits 15–0

The results of the Calculate Data Signature and Calculate IFR Block Signature commands are available in this register after execution of these commands.

6.7.8 Option Data 1 Register (OPT1)

The Flash Optional Data 1 (OPT1) register provides a user accessible copy of data stored in the IFR block.

Base + $1B	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	RELAXATION OSCILLATOR TRIM VALUE (ROTV)									
Write																
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

Figure 6-15. Optional Data 1 (OPT1) Register

6.7.8.1 Reserved—Bits 15–10

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

6.7.8.2 Relaxation Oscillator Trim Value (ROTV)—Bits 9–0

This bit field contains a factory-measured trim value for the relaxation oscillator. Copy this value into the Trim Field of the OCCS OCTRL register.

6.7.9 Test Array Signature Register (TSTSIG)

The TSTSIG register stores the Flash Information Block Signature, generated by the Calculate IFR Block Signature command during factory test. The value in the TSTSIG register is compared to the result of the Calculate IFR Block Signature throughout the life of the part to confirm data used by the user commands and internal module adjustments has not been compromised.

Base + $1D	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	TEST SIGNATURE															
Write																
Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

Figure 6-16. Test Array Signature (TSTSIG) Register

6.8 Interrupts

The Flash Memory module generates an interrupt when all Flash commands are completed, or when the address, data, and command buffers are empty. Interrupts can also be generated when the ACCERR bit is set.

Table 6-8. Flash Memory Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable
Flash Command, Data and Address Buffer empty	CBEIF (USTAT)	CBEIE (CNFG)
All commands are completed on Flash	CCIF (USTAT)	CCIE (CNFG)
ACCERR generated	ACCERR (USTAT)	AEIE (CNFG)

Note: Vector addresses and their relative interrupt priority are determined at the 16-bit controller level.

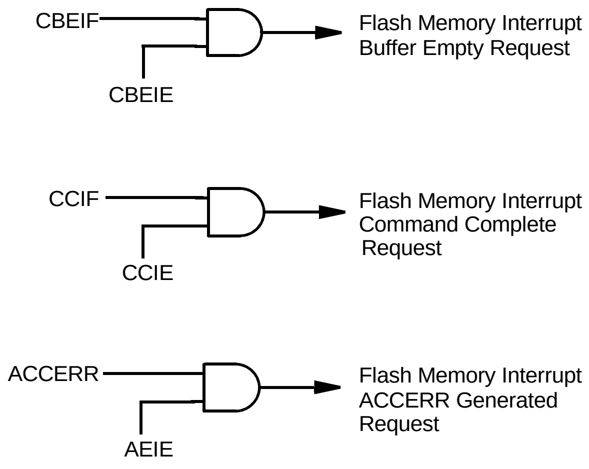

CBEIF
CBEIE
Flash Memory Interrupt Buffer Empty Request
CCIF
CCIE
Flash Memory Interrupt Command Complete Request
ACCERR
AEIE
Flash Memory Interrupt ACCERR Generated Request

Figure 6-17. Interrupt Implementation

6.9 Resets

The FM module uses the early reset (16 clocks in advance of the core’s reset signal) signal supplied by the SIM module to load the reset state of bit fields within the PROT and SECHI registers with data from the Flash Memory area. The early reset signal is also used to trigger the read of the Security word and the enabling of chip security if it is so configured. Please see **Section 6.6** for more details. The Flash array is not accessible for any operations, via the address and data buses, during early reset. If a reset occurs while any command is in progress, that command will be immediately aborted. The state of the word being programmed or the page/block being erased is not guaranteed.

Chapter 7
General Purpose Input/Output (GPIO)

Document Revision History for Chapter 7, General Purpose Input/Output (GPIO)

Version History	Description of Change
Rev. 0	Initial release
Rev. 1	Clarifications throughout chapter.

7.1 Introduction

The General Purpose Input/Output (GPIO) module allows direct read or write access to pin values, as well as the ability to use a pin as an external interrupt. GPIO pins can also be multiplexed with other peripherals on the chip as well. The *Device Data Sheet* specifies the assigned GPIO ports and its multiplexed pin package.

A GPIO pin may be configured in three ways:

1. As GPIO input with, or without, pull-up
2. As GPIO output with Push-Pull mode or open drain mode
3. As a peripheral pin when multiplexed with another module

GPIOs are placed on the chip in groups of one to 16 bits, called ports, which are designated as port A, B, C, etc. Please refer to the *Device Data Sheet* for the specific definition of each of the GPIO ports on the chip. For purposes of illustration, a port width of 16 bits is assumed throughout this chapter.

7.2 Features

The GPIO module design includes these characteristics:

- Individual control for each pin to be in either Peripheral or GPIO mode
- Individual Input/Output direction control for each pin in GPIO mode
- Individual Pull-Up Enable Control for each input pin in either Peripheral or GPIO mode
- Individual output Push-Pull mode or Open Drain mode control for output each pin in either Peripheral or GPIO mode
- Individual output drive strength control for each pin
- Ability to monitor pin logic values even when GPIO are not enabled by using the RDATA register
- Interrupt assert capability
- 5V tolerant

7.3 Block Diagram

Figure7-1 illustrates the logic associated with a single multiplexed device pin. Each pin may be configured as either Peripheral or GPIO mode. An input edge detection feature and a pull-up enable feature are provided independent of the operating mode configuration.

When the pin is configured for GPIO mode, the corresponding PE bit in the Peripheral Enable (PEREN) register must be set to 0. At this time the pin may be configured as either an input or output by setting the corresponding DD bit in the Data Direction (DDIR) register to 0 for input or 1 for output. When configured as an input, the logic level on the pin can be determined by reading the Data (DATA) register. When configured as an output, any data written to the DATA register will be reflected on the output pin.

When the pin is configured for Peripheral mode, the corresponding PE bit in the PEREN register will be set to 1. In this configuration the corresponding peripheral function will drive the functionality of the pin. For example, the SCI peripheral requires two pins (one transmit and one receive) for correct operation. The two corresponding PE bits must be set to enable these pins for SCI operation. See the specific peripheral section for more details.

As can be seen in the block diagram, the following features are available independent of the Peripheral or GPIO mode configuration.

- The edge detection feature is available to provide detection and interrupt notification of rising or falling edge signals. The active state of the input signal is configured in the Interrupt Edge Polarity (IEPOL) register and interrupt notification is enabled in the Interrupt Enable (IEN) register. The Interrupt Pending (IPEND) and Interrupt Edge (IEDGE) registers work in concert to provide detection status and interrupt clearing capability respectively. The Interrupt Assert (IASSRT) register provides software interrupt generation as well as a way to simulate external interrupts. See **Figure7.7** for more details.
- While configured as an input, the pins may be internally pulled up by enabling the corresponding bit in the Pull-Up Enable (PUPEN) register. The typical value of internal pull-up is about 110KΩ.
- While configured as an output, the pin may be placed in either Push/Pull or Open Drain mode through the Push/Pull Output Mode Control (PPOUTM) register.
- While configured as an output, the output drive capability of the pin may be configured to source either 4mA or 8mA through the Drive Strength Control (DRIVE) register.
- The logic level of the pin may be sampled at any time by reading the Raw Data (RDATA) register. This reading is independent of the input/output configuration of the pin. This feature my be required during the system verification or debugging of the system.
- Capable of accepting 5V inputs or driving 5V outputs when configured as Open Drain mode with external pull-ups.

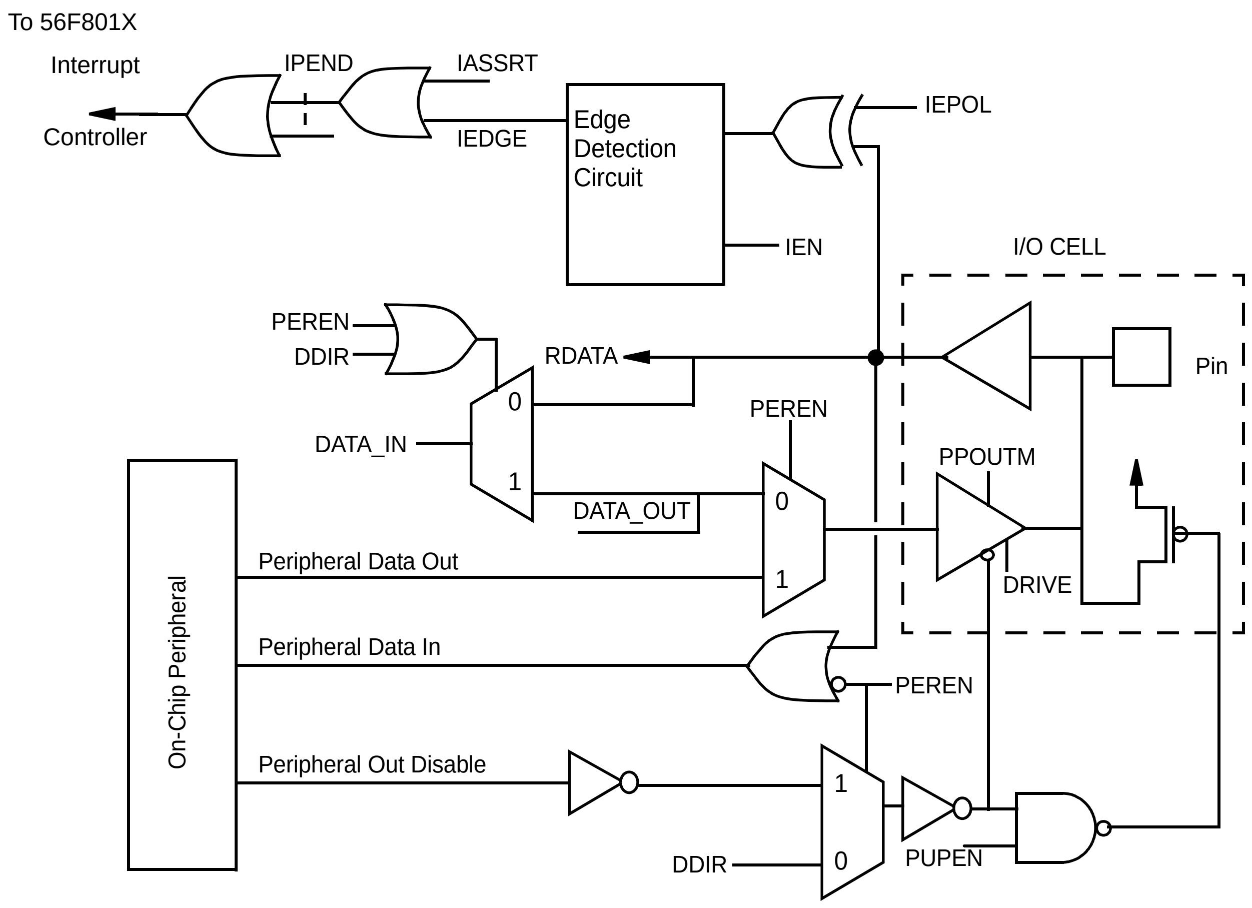


Figure 7-1. Bit-Slice View of the GPIO Logic

7.4 Operating Modes

The GPIO module contains two major modes of operation:

- Peripheral Mode—The peripheral module controls the output/input enable and any output data is supplied by the peripheral.
- GPIO Mode—In this mode, the GPIO module controls the input/output direction of each pin any input data can be read from the DATA register. If the pin is configured as input, and any data written to the DATA register is reflected on the output pin if it is configured as output.

Pull-up enables are controlled by the PUPEN register when a pin is configured as a peripheral input or GPIO input. See **Table 7-2** for more detail.

7.5 Register Definitions

Note: Please refer to the *Device Data Sheet* for the base address of this peripheral.

For illustration purposes, a 16-pin wide GPIO port is assumed for all registers in this chapter. Each register bit performs an identical function for each of the GPIO pins controlled by a specific GPIO port. Besides pin widths, the only other differences between GPIO ports relate to the initial operating conditions at reset. Some GPIO ports are initialized in GPIO mode as default, while others are not. The same is true with pull-up resistors. Some may be enabled, others not.

Note: Please be certain to consider the specific chip's availability on GPIO ports. Not all GPIO ports are available on all chips.

A register address is the sum of a base address and an address offset. Please see the *Device Data Sheet* for base addresses. Each GPIO module has 12 registers.

Register names, addresses, descriptions, and chapter locations are provided in **Table 7-1.**

Table 7-1. GPIO Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	PUPEN	Pull-Up Enable Register	Read/Write	**Section 7.5.1**
Base + $1	DATA	Data Register	Read/Write	**Section 7.5.2**
Base + $2	DDIR	Data Direction Register	Read/Write	**Section 7.5.3**
Base + $3	PEREN	Peripheral Enable Register	Read/Write	**Section 7.5.4**
Base + $4	IASSRT	Interrupt Assert Register	Read/Write	**Section 7.5.5**
Base + $5	IEN	Interrupt Enable Register	Read/Write	**Section 7.5.6**
Base + $6	IEPOL	Interrupt Edge Polarity Register	Read/Write	**Section 7.5.7**
Base + $7	IPEND	Interrupt Pending Register	*Read-Only*	**Section 7.5.8**
Base + $8	IEDGE	Interrupt Edge Sensitive Register	Read/Write	**Section 7.5.9**
Base + $9	PPOUTM	Push-Pull Output Mode Register	Read/Write	**Section 7.5.10**
Base + $A	RDATA	Provides an unclocked version of the data values currently present on each GPIO pin even when not in GPIO mode.	*Read -Only*	**Section 7.5.11**
Base + $B	DRIVE	Drive Strength Register	Read/Write	**Section 7.5.12**

Bits of each of the 12 registers are summarized in **Figure 7-2.** Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	GPIO*n*_PUPEN	R	PU															
		W																
$1	GPIO*n*_DATA	R	D															
		W																
$2	GPIO*n*_DDIR	R	DD															
		W																
$3	GPIO*n*_PEREN	R	PE															
		W																
$4	GPIO*n*_IASSRT	R	IA															
		W																
$5	GPIO*n*_IEN	R	IEN															
		W																
$6	GPIO*n*_IEPOL	R	IPOL															
		W																
$7	GPIO*n*_IPEND	R	IPR															
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$8	GPIOn_IEDGE	R	IES															
		W																
$9	GPIO*n*_PPOUTM	R	OEN															
		W																
$A	GPIO*n*_RDATA	R	RAWDATA															
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$B	GPIO*n*_DRIVE	R	DRIVE															
		W																

R	0	Read as 0
W	▨	Reserved

Note: Assumes 16-bit GPIO. This information maybe different depending on the specific part. Please see the device Data Sheet for actual value.

Figure 7-2. GPIO Register Map Summary

7.5.1 Pull-Up Enable Register (PUPEN)

This read/write register enables and disables the pull-up on each GPIO pin. This configuration affects the pin only when it is functioning as an input which may function as an input in both GPIO mode and Peripheral mode. See **Table 7-2** for active conditions.

This pull-up is intended *only* to drive an undriven input pin to a known state. It is characteristically a very weak pull-up and it typically will not meet the pull-up requirements of an active circuit.

Note: The reset value may be different depending on the specific device. Please see the *Device Data Sheet* for actual value.

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	PU															
Write																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-3. Pull-Up Enable (PUPEN) Register

- 0 = Pull-Up is disabled
- 1 = Pull-Up is enabled (See **Figure7-2** for additional conditions)

Table 7-2 provides the state of the pin and PUPEN register as a function of current peripheral state and control register values.

Table 7-2. GPIO Pull-Up Enable Functionality

PEREN	Pin Operating Mode	PUPEN	DDIR	Pin State	Pin Pull-Up
0	GPIO	0	0	Input	Disabled
0	GPIO	0	1	Output	Disabled
0	GPIO	1	0	Input	Enabled
0	GPIO	1	1	Output	Disabled
1	Peripheral	x	x	Output	Disabled
1	Peripheral	0	x	Input	Disabled
1	Peripheral	1	x	Input	Enabled

When the Peripheral Enable (PEREN) register is zero, the Pull-Up Enable (PUPEN) register and the Data Direction (DDIR) register control the pin pull-up. When the PEREN register is one, the pin pull-up is controlled by the PUPEN and peripheral pin function. The PUPEN value is only recognized when the pin is functioning as an input.

7.5.2 Data Register (DATA)

This read/write register holds data coming either from the pin or the IPBus when pins are configured as GPIO. When pins are configured as GPIO and input, logic level on the pin can be determined by reading the DATA register value. When pins are configured as GPIO and output, value written to the DATA register will be reflected on the output pin.

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	D															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-4. Data (DATA) Register

7.5.3 Data Direction Register (DDIR)

When the pin is configured as GPIO mode (PE*n*=0), this read/write register configures the state of the pin as either an input or output. When DDIR is set to 0, the pin is an input. When DDIR is set to 1, the pin is an output.

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DD															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-5. Data Direction (DDIR) Register

- 0 = Pin is an input
- 1 = Pin is an output

7.5.4 Peripheral Enable Register (PEREN)

This read/write register determines the GPIO configuration. When PEREN value is one, the peripheral owns the pin. This ownership includes configuring the pin as a required input, or output depending on the status of the peripheral output enable and includes data transfers from the pin to the peripheral. Please see **Table 7-2** for additional information. When the PEREN value is zero, the GPIO module owns the pin, controlling the direction of the data flow via the DDIR. Regardless peripheral mode or GPIO mode, input pull-up, output drive strength, and output push/pull /open drain selection are controlled by corresponding registers in the GPIO module.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	PE															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-6. Peripheral Enable (PEREN) Register

Note: The reset value may be different depending upon the specific device. Please see the *Device Data Sheet* for actual value.

- 0 = Pin is for GPIO
- 1 = Pin is for Peripheral

7.5.5 Interrupt Assert Register (IASSRT)

This read/write register is typically used for software testing, but it also provides a software interrupt compatibility. An interrupt is asserted when any IA bit is set to 1. The register is cleared by writing 0s. Interrupts will be generated continually until this bit is cleared.

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	IA															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-7. Interrupt Assert (IASSRT) Register

- 0 = Deassert software interrupt
- 1 = Assert software interrupt

7.5.6 Interrupt Enable Register (IEN)

This read/write register enables or disables the interrupt for each GPIO pin. Set a bit to 1 to enable interrupts for the associated GPIO pin. Interrupts are recorded in the IPEND register, if edge transition is detected.

Base + $5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	IEN															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-8. Interrupt Enable (IEN) Register

- 0 = Edge interrupt is disabled
- 1 = Edge interrupt is enabled

7.5.7 Interrupt Edge Polarity Register (IEPOL)

This read/write register controls the edge polarity of any external interrupts enabled by an IEN bit set to 1. When this register is set to 1, the falling edge causes the interrupt. When this register is set to 0, the rising edge causes the interrupt.

Base + $6	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	IPOL															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-9. Interrupt Edge Polarity (IEPOL) Register

- 0 = Rising edge generates the interrupt
- 1 = Falling edge generate the interrupt

7.5.8 Interrupt Pending Register (IPEND)

This *read-only* register records any incoming interrupts. This register is read to determine which pin or bit of IASSRT register caused the interrupt. For external interrupts, the IPEND is cleared by writing 1s into the IEDGE register. For software interrupts, the IPEND is cleared by writing 0s into the IASSRT register.

Base + $7	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	IPR															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-10. Interrupt Pending Register (IPEND)

- 0 = Interrupt cleared
- 1 = Interrupt pending

7.5.9 Interrupt Edge Sensitive Register (IEDGE)

When an edge is detected by the edge detector circuit, and IEN is set to 1, IEDGE records the interrupt. The corresponding bit in the IPEND register is also set. This read/write register clears the corresponding IPR bit field by writing 1 to the corresponding bit in the IEDGE register. Writing 0 to an IEDGE bit is ignored.

Base + $8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	IES															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-11. Interrupt Edge Sensitive (IEDGE) Register

- 0 = No edge seen
- 1 = Edge detected

7.5.10 Push/Pull Output Mode Control Register (PPOUTM)

This read/write register explicitly sets each output driver to either Push/Pull or Open Drain mode, independent of perhipheral or GPIO mode configuration of the pin.

Note: The Open Drain mode can be used to tri-state any pin on the GPIO port without switching that pin to input mode. This is useful for some applications, including a keypad interface. These bits default to Push/Pull mode upon reset, giving all GPIO ports the same default functionality in this regard.

Base + $9	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	OEN															
Write																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-12. Push/Pull Output Mode Control (PPOUTM) Register

- 0 = Open Drain mode
- 1 = Push/Pull mode

7.5.11 Raw Data Register (RDATA)

This *read-only* register allows the controller direct access to the logic values on each GPIO pin even when pins are not in the GPIO mode. Values are not locked and are subject to change at any time. The reset state is unknown. It is recommended to read several times to assure a stable value.

Base + $A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	RAWDATA															
Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Figure 7-13. Raw Data (RDATA) Register

- 0 = Logic 0 present on GPIO pin
- 1 = Logic 1 present on GPIO pin

7.5.12 Drive Strength Control Register (DRIVE)

This read/write register can be used to explicitly set the drive strength of each output driver, independent of the Peripheral or GPIO mode configuration of the pin. This feature provides an additional design variable with the ability to control the effects of electromagnetic interference (EMI) due to digital switching. For capacitive loads, higher drive strengths improve low to high transition rates but increase EMI. Lower drive strengths reduce EMI and circuit board trace width requirements. These bits default to 4mA upon reset, providing all GPIO ports with the same default functionality.

Base + $B	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DRIVE															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-14. Drive Strength Control Register (DRIVE)

- 0 = 4mA drive strength outputs
- 1 = 8mA drive strength outputs

7.6 Clocks and Resets

The module runs at standard IPBus speeds and will assume reset states as defined in each chip specification. Reset occurs whenever any source of system reset occurs (Power-On Reset (POR), External Reset (RESET), or COP or, software reset).

7.7 Interrupts

The GPIO has two types of interrupts:

1. Software interrupt

   The Interrupt Assert (IASSRT) register is used to generate the software interrupt. It can be generated by writing 1s to IASSRT. The Interrupt Pending (IPEND) register will record the value of IASSRT. The IPEND register can be cleared by writing 0s into the IASSRT during the IASSRT testing.

Note: When testing the IASSRT, the Interrupt Edge Polarity (IEPOL) register, Interrupt Edge Sensitive (IEDGE) register, and the Interrupt Enable (IEN) register must be zero to guarantee the interrupt registered in the IPEND is due to IASSRT only.

2. Hardware interrupt from the pin

   When a GPIO pin is used as an external interrupt source, its IEN bit is set to 1 and the IASSRT register must be set to 0. The IEPOL register must be set to 1 for an falling edge interrupt and zero for a rising edge interrupt. When the signal at the pin transitions from high to low or low to high the value is seen at the IEDGE register and recorded by the IPEND register. Please see **Table 7-3**. The IPEND is cleared by writing 1s into the IEDGE.

The interrupt signals in each port are ORed together, presenting only a single interrupt per port to the interrupt controller. The interrupt service routine must then check the contents of the IPEND register to determine which pin(s) caused the interrupt.

External interrupt sources do not need to remain asserted because the detection mechanism is edge sensitive.

Table 7-3. GPIO Interrupt Assert Functionality

IPOLR	Interrupt Asserted	Remark
0	Rising Edge	If the IEN is set to 1, as the pin goes from low to high an interrupt will be recorded by the IPEND register.
1	Falling Edge	If the IEN is set to 1, as the pin goes from high to low an interrupt will be recorded by the IPEND register.

Chapter 8
Joint Test Action Group Port (JTAG)

Document Revision History for Chapter 8, Joint Test Action Group Port (JTAG)

Version History	Description of Change
Rev 0	Initial release

8.1 Introduction

Because this device also has a 56800E core containing its own Test Access Port (TAP), or *Core TAP*, a TAP Linking Module (TLM) is included to manage the TAP access. Normal operation of this part will use the Chip TAP as the Master TAP controller, thereby disabling the 56800E TAP (Core TAP) controller. This chapter discusses the Master TAP only.

8.2 Features

TAP characteristics include:

- Provide a means of accessing the EOnCE module controller and circuits to control a target system
- Query the IDCODE from any TAP in the system
- Force test data onto the peripheral outputs while replacing its Boundary Scan Register (BSR) with a single bit register
- Enable/disable pull-up devices on peripheral boundary scan pins

8.3 Block Diagram

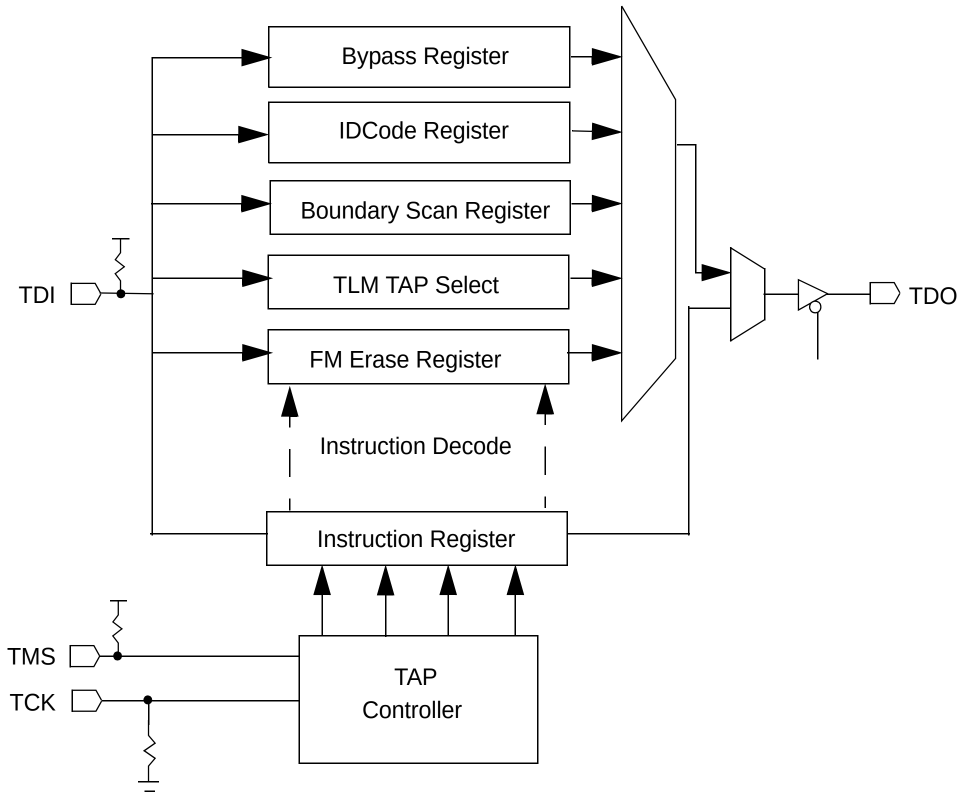


Figure 8-1. JTAG Block Diagram

8.4 Functional Description

The Master TAP consists of a synchronous finite 16-bit state machine, an eight-bit instruction register, a bypass register, and an identification code register.

8.4.1 JTAG Port Architecture

The TAP Controller is a simple state machine used to sequence the JTAG port through its varied operations:

- Serially shift in or out a JTAG port command
- Update and decode the JTAG port Instruction Register (IR)
- Serially input or output a data value
- Update a JTAG port or EOnCE module register

Note: The JTAG port supervises the shifting of data into and out of the EOnCE module through the TDI and TDO pins respectively. In this case, the shifting is guided by the same controller used when shifting JTAG information.

A block diagram of the JTAG port is provided in **Figure 8-1**. The JTAG port has four read/write registers:

1. Instruction Register (JTAGIR)
2. Chip Identification (CID) register
3. Bypass Register (JTAGBR)
4. Boundary Scan Register (BSR)

Access to the EOnCE registers is described in the *56800E Data Sheet*.

8.4.2 Master TAP Instructions

The eight-bit Master TAP Instruction register is in support of all JTAG functions. It is described in **Table 8-1**. This register includes all IEEE 1149.1 required instructions plus several additional instruction registers accommodating debug and BIST testing.

Table 8-1. Master TAP Instructions Opcode

Instruction	Target Register	Opcode
BYPASS	BYPASS	11111111
IDCODE	IDCODE	00000010
	Reserved	00000011
	Reserved	00000100
TLM_SEL	TLM	00000101
Lock Out Recovery (Flash_Erase)	FLASH_ERASE	00001000

8.4.2.1 Bypass Instruction (BYPASS)

The BYPASS instruction is a required JTAG instruction, selecting the TAP Bypass register. This register is a single stage shift register providing a serial path between the TDI and the TDO pins illustrated in **Figure 8-2**. This instruction enhances test efficiency by shortening the overall path between TDI and TDO when no test operation of a component is required.

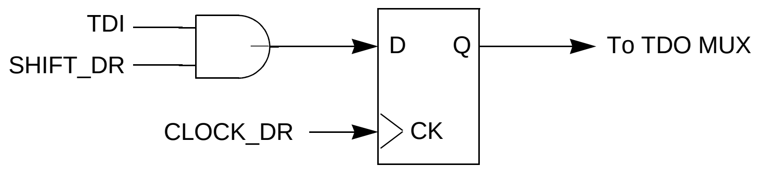


Figure 8-2. Bypass Register Diagram

8.4.2.2 IDCODE

The IDCODE instruction is an optional JTAG instruction enabling the Chip Identification (CID) register between TDI and TDO. This 32-bit register identifies the manufacturer, part and version numbers.

8.4.2.3 TLM_SEL

The TLM_SEL instruction is a user-defined JTAG instruction, disabling the Master TAP and enabling the TAP Linking Module, or TLM. The TLM then selects the 56800E core TAP or the Master TAP as the enabled TAP.

8.5 TAP Controller

The TAP Controller is a synchronous 16-bit finite state machine illustrated in **Figure 8-3**. TAP Controller responds to changes at the TMS and TCK pins. Transitions from one state to another occur on the rising edge of TCK. The value shown adjacent to each state transition represents the signal present on TMS at the time of a rising edge of TCK.

The TDO pin remains in the high impedance state except during the Shift-DR and Shift-IR TAP Controller states. In Shift-DR and Shift-IR controller states, TDO updates on the falling edge of TCK. TDI is sampled on the rising edge of TCK.

The TAP Controller executes the last instruction decoded until a new instruction is entered at the Update-IR state, or Test-Logic-Reset is entered.

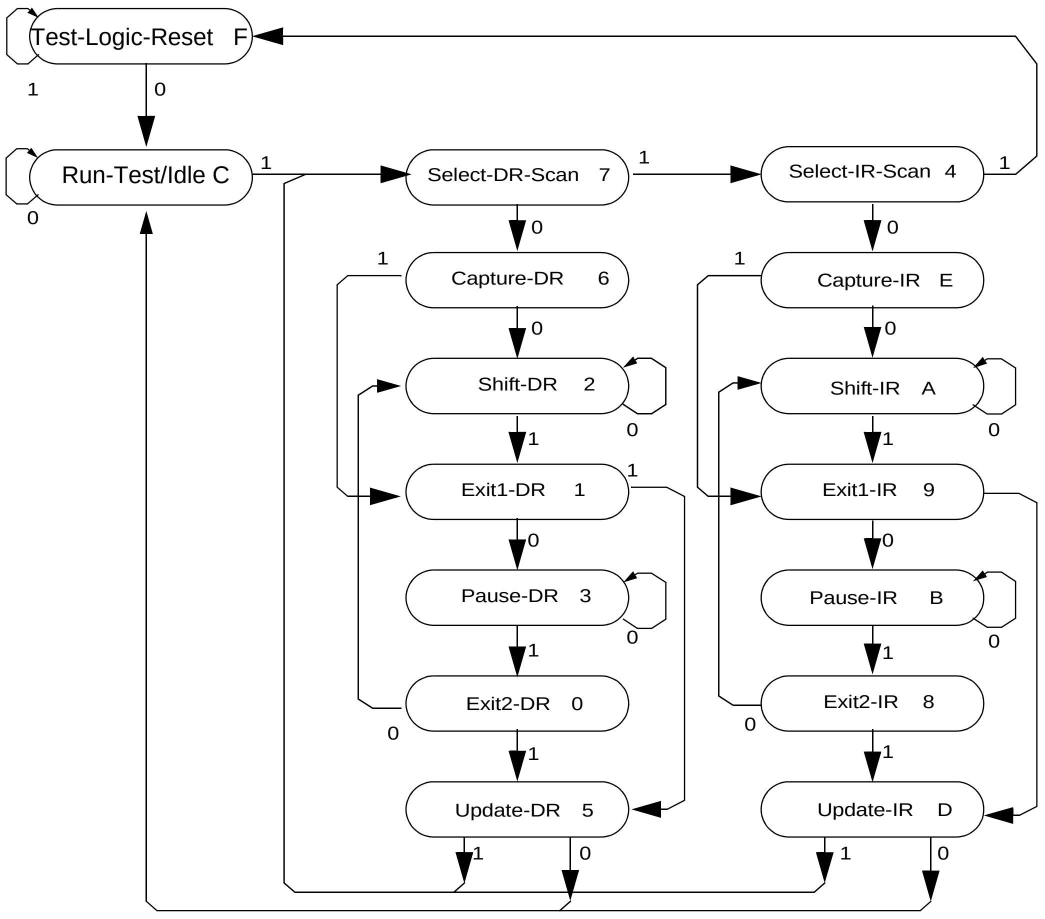


Figure 8-3. TAP Controller State Diagram

8.5.1 Operation

All state transitions of the TAP Controller occur based on the value of TMS at the time of a rising edge of TCK. Actions of the instructions occur on the falling edge of TCK in each controller state illustrated in **Figure 8-3**.

8.5.1.1 Test Logic Reset (pstate = F)

During Test-Logic-Reset, all JTAG test logic is disabled so the chip can operate in Normal mode. This is achieved by initializing the Instruction Register (IR) with the IDCODE instruction. By holding TMS high for five rising edges of TCK, the device always remains in Test-Logic-Reset no matter what state the TAP Controller was in previously.

8.5.1.2 Run-Test-Idle (pstate = C)

Run-Test-Idle is a controller state between scan operations. When EOnCE is entered, the controller remains in Run-Test-Idle mode as long as TMS is held low. When TMS is high and a rising edge of TCK occurs, the controller moves to the Select-DR state.

8.5.1.3 Select Data Register (pstate = 7)

The Select-Data register state is a temporary state. In this state, all Test Data registers selected by the current instruction retain their previous states. If TMS is held low and a rising edge of TCK occurs when the controller is in this state, the controller moves into the Capture-DR state and a scan sequence for the selected test date register is initiated. If TMS is held high and a rising edge of TCK occurs, the controller moves to the Select-IR state.

8.5.1.4 Select Instruction Register (pstate = 4)

The Select-Instruction register state is a temporary state. In this state, all Test Data registers selected by the current instruction retain their previous states. If TMS is held low and a rising edge of TCK occurs when the controller is in this state, the controller moves into the Capture-IR state and a scan sequence for the instruction register is initiated. If TMS is held high and a rising edge of TCK occurs, the controller moves to the Test-Logic-Reset state.

8.5.1.5 Capture Data Register (pstate = 6)

In this controller state, data may be parallel loaded into test registers selected by the current instruction on the rising edge of TCK. If a Test Data register selected by the current instruction does not have a parallel input, the register retains its previous value.

8.5.1.6 Shift Data Register (pstate = 2)

In this controller state, the Test Data register is connected between TDI and TDO. This data is then shifted one stage towards its serial output on each rising edge of TCK. The TAP Controller remains in this state while TMS is held at low. When 1 is applied to TMS and a positive edge of TCK occurs, the controller will move to the Exit1-DR state.

8.5.1.7 Exit1 Data Register (pstate = 1)

This is a temporary controller state. If TMS is held high, and a rising edge is applied to TCK while in this state, the controller advances to the Update-DR state. This terminates the scanning process.

8.5.1.8 Pause Data Register (pstate = 3)

This controller state permits shifting of the Test Data register in the serial path between TDI and TDO to be temporarily halted. All Test Data registers selected by the current instruction retain their previous state unchanged. The controller remains in this state while TMS is held low. When TMS goes high and a rising edge is applied to TCK, the controller advances to the Exit2-DR state.

8.5.1.9 Exit2 Data Register (pstate = 0)

This is a temporary controller state. If TMS is held high, and a rising edge is applied to TCK while it is in this state, the scanning process terminates and the TAP Controller advances to the Update-DR state. If TMS is held low and a rising edge of TCK occurs, the controller advances to the Shift-DR state.

8.5.1.10 Update Data Register (pstate = 5)

All Boundary Scan registers contain a two-stage data register. It isolates the shifting and capturing of data on the peripheral from what is applied to internal logic during Scan mode. This register is the second stage, or parallel output, and applies a stimulus to internal logic. Data is latched on the parallel output of these Test Data registers from the Shift register path on the falling edge of TCK in the Update-DR state. On a rising edge of TCK, the controller advances to the Select_DR state if TMS is held high or the Run-Test-Idle state if TMS is held low.

8.5.1.11 Capture Instruction Register (pstate = E)

When the TAP Controller is in this state and a rising edge of TCK occurs, the controller advances to the Exit1-IR state if TMS is held at one, or the Shift-IR state if TMS is held at zero.

8.5.1.12 Shift Instruction Register (pstate = A)

In this controller state, the Shift register contained in the instruction register is connected between TDI and TDO and shifts data one stage toward its serial output on each rising edge of TCK. When the TAP Controller is in this state and a rising edge of TCK occurs, the controller advances to the Exit1-IR state if TMS is held at one or remains in the Shift-IR state if TMS is held at zero.

8.5.1.13 Exit1 Instruction Register (pstate = 9)

This is a temporary controller state. If TMS is held high, and a rising edge is applied to TCK while in this state, the controller advances to the Update-IR state. This terminates the scanning process. If TMS is held low and a rising edge of TCK occurs the controller advances to the Pause-IR state.

8.5.1.14 Pause Instruction Register (pstate = B)

This controller state allows shifting of the instruction register in the serial path between TDI and TDO to be temporarily halted. All Test Data registers selected by the current instruction retain their previous state unchanged. The controller remains in this state while TMS is held low. When TMS goes high and a rising edge is applied to TCK, the controller advances to the Exit2-IR state.

8.5.1.15 Exit2 Instruction Register (pstate = 8)

This is a temporary controller state. If TMS is held high, and a rising edge is applied to TCK while in this state, the scanning process terminates and the TAP Controller advances to the Update-IR state. If TMS is held low and a rising edge of TCK occurs, the controller advances to the Shift-IR state.

8.5.1.16 Update Instruction Register (pstate = D)

During this state, the instruction shifted into the Instruction register is latched from the Shift register path on the falling edge of TCK and into the instruction latch. It becomes the current instruction. On a rising edge of TCK, the controller advances to the Selector state if TMS is held high, or the Run-Test-Idle state if TMS is held low.

8.6 Memory Map

JTAG has no memory mapped registers.

8.7 Pin Description

The signal summaries for the JTAG are located in Table 8-2.

Table 8-2. JTAG Pin Description

Pin Name	Pin Description
TCK	**Test Clock Input**—This input pin provides the clock to synchronize the test logic and shift serial data to and from all TAP Controllers and the TLM. If the EOnCE module is not being accessed using the Master or 56800E core TAP Controllers, the maximum TCK frequency is 1/4 the maximum frequency for the 56800E core. When accessing the EOnCE module through the 56800E core TAP Controller, the maximum frequency for TCK is 1/8 the maximum frequency for the 56800E core. The TCK pin has a pull down non-disabled resistor.
TDI	**Test Data Input**—This input pin provides a serial input data stream to the TAP and the TLM. It is sampled on the rising edge of TCK. TDI has an on-chip pull-up resistor which can be disabled through PUPEN register in the GPIO module.
TMS	**Test Mode Select Input**—This input pin is used to sequence the TAP Controller's TLM state machine. It is sampled on the rising edge of TCK. TMS has an on-chip pull-up resistor which can be disabled through PUPEN register in the GPIO module.
TDO	**Test Data Output**—This tri-state output pin provides a serial output data stream from the Master TAP, or 56800E core TAP Controller. It is driven in the Shift-IR and Shift-DR controller states of the TAP Controller state machines. Output data changes on the falling edge of TCK.

8.8 Clocks

8.8.1 TCK

This is the sole clock used by the Master TAP module. If the EOnCE module is not being accessed using the Master or 56800E core TAP controllers, the maximum TCK frequency is one-quarter the maximum frequency for the 56800E core. When accessing the EOnCE module through the 56800E core TAP controller, the maximum frequency for TCK is one eighth the maximum frequency for the 56800E core.

Table 8-3. Clock Summary

Clock	Priority	Source	Characteristics
TCK	1	External	This user provided clock shifts data and control the state machine.

8.9 Interrupts

This module has no interrupt capabilities.

Chapter 9
Power Supervisor (PS)

Document Revision History for Chapter 9, Power Supervisor (PS)

Version History	Description of Change
Rev.0	Initial release

9.1 Introduction

This chapter details the on-chip Power Supervisor (PS) module. Its function ensures the chip is operated only within required voltage ranges while assisting in a orderly shutdown of the chip in the event the power supply is interrupted, or there is a brownout, or a voltage sag.

9.2 Features

The Power Supervisor (PS) monitors on-chip voltages (digital 3.3V and 2.5V rails), providing the following qualities:

- Power-On Reset (POR)
  - Holds device in reset until:
    - V_{DD} core voltage exceeds 1.8V
    - regulator voltages have risen above LVI thresholds (2.2V and 2.7V)
- Core Low Voltage Interrupt (LVI22)
  - generated when the 2.5V rail drops below 2.2V
- I/O Low Voltage Interrupt (LVI27)
  - generated when the 3.3V rail drops below 2.7V

Working under the assumption power supply voltages move relatively slowly with respect to the system clocks, low voltage interrupts should provide ample warning of impending problems as well as time to accomplish an orderly shutdown of the chip be accomplished.

9.3 Block Diagram

The basic Power-On Reset (POR) and low voltage detect module is illustrated in **Figure 9-1.** The POR circuit is designed to assert the internal reset from V_{DD} = 0V to V_{DD} = 1.8V. $\overline{\text{POR}}$ switching high indicates the 3.3V and 2.5V rails are above their minimum thresh holds. POR is asserted when the core voltage drops below 1.8V.

The deglitch blocks are essentially strings of four flops in series. The outputs of all four must indicate an active interrupt before the output switches to the active state. This is intended to prevent the LVI circuitry from responding to momentary glitches brought about as a result of normal operation. Once set, the sticky status bits LVIS27S, LVIS22S, and LVI must be explicitly reset by writing one.

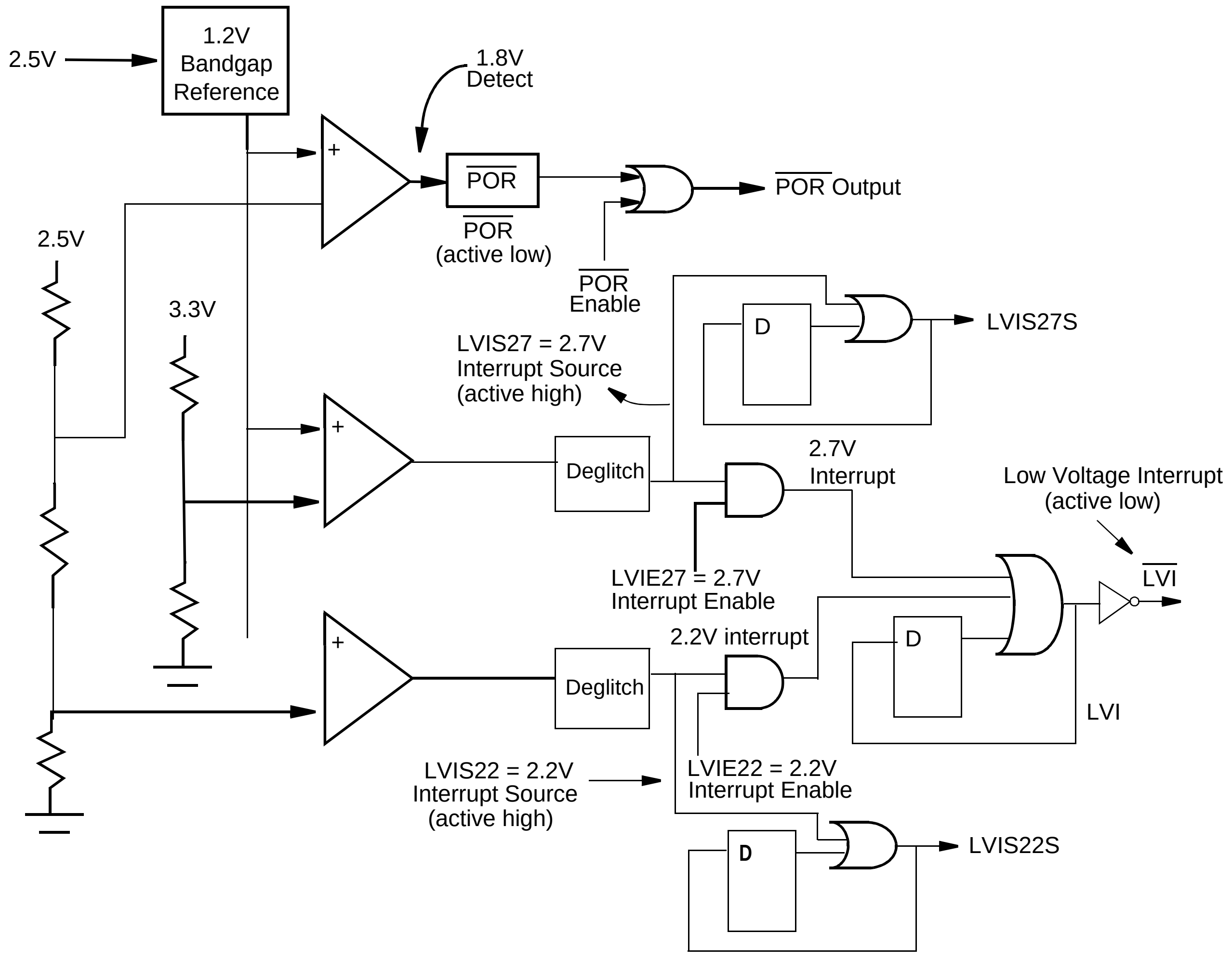

2.5V
1.2V Bandgap Reference
1.8V Detect
POR
POR (active low)
POR Output
POR Enable
2.5V
3.3V
LVIS27S
LVIS27 = 2.7V Interrupt Source (active high)
Deglitch
2.7V Interrupt
Low Voltage Interrupt (active low)
LVI
LVIE27 = 2.7V Interrupt Enable
2.2V interrupt
Deglitch
LVI
LVIS22 = 2.2V Interrupt Source (active high)
LVIE22 = 2.2V Interrupt Enable
LVIS22S
D
D
D

Figure 9-1. Power Supervisor Block Diagram

9.4 Functional Description

The Power-On Reset ($\overline{\text{POR}}$) signal is always enabled and the low voltage interrupts are disabled out of reset. As the device powers up, the voltage regulator circuit will provide 3.3V for the I/O circuitry and 2.5V for the core. When the core voltage exceeds 1.8V and the LVI (2.2V and 2.7V) thresholds are exceeded, the device is released from reset 64 clock cycles later as illustrated in **Figure 9-2.** At this point, the LVIs should be enabled and the PLL used to increase the system clock frequency.

For most initializations, waiting for the PLL to lock represents the largest component in the start-up time line. Hence, it is prudent to check the LVI status bits again just after PLL lock is achieved.

Figure 9-2 illustrates operation of the $\overline{\text{POR}}$ versus low voltage detect circuits. The LVI should be explicitly cleared, and disabled by the interrupt service routine responsible for shutting down the part when a low voltage is detected.

Low voltage interrupts are masked upon $\overline{\text{POR}}$. They must be explicitly enabled and disabled thereafter. Low voltage status bits in the Power Supervisor Status Register (STAT) are always active, independent of whether low voltage interrupts are enabled. LVIs include roughly 50-100mV of hysteresis.

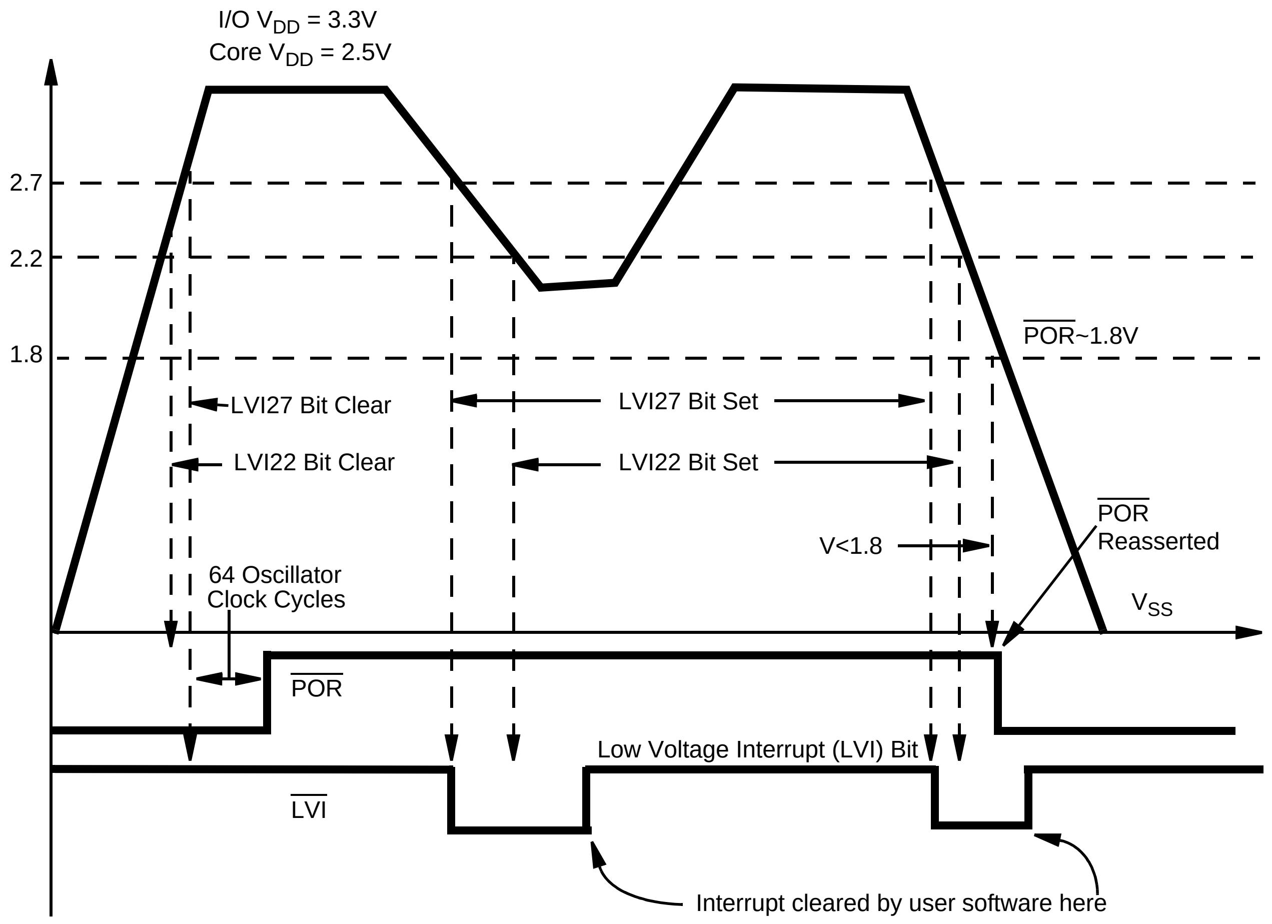


Figure 9-2. $\overline{\text{POR}}$ Vs. Low-Voltage Interrupts

9.5 Register Definitions

Table 9-1. PS Memory Map

Device	Peripheral	Address
56F801X	PS	$00F160

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level. The Low Power Voltage Supervisor (PS) module has two registers.

Table 9-2. Power Supervisor Register Summary

Register Address	Register Acronym	Register Name	Access Type	Chapter Location
Base + $0	CTRL	Control Register	Read/Write	**Section 9.5.1**
Base + $1	STAT	Status Register	Read/Write	**Section 9.5.2**

Bit fields of each of the two PS registers are illustrated in **Figure 9-3** and **Figure 9-4**. Details of each follow.

9.5.1 Control Register (CTRL)

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LVIE27	LVIE22
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 9-3. Control (CTRL) Register

9.5.1.1 Reserved—Bits 15–2

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

9.5.1.2 2.7V Low Voltage Interrupt Enable (LVIE27)—Bit 1

- 0 = Interrupt is disabled
- 1 = Interrupt is enabled

9.5.1.3 2.2V Low Voltage Interrupt Enable (LVIE22)—Bit 0

- 0 = Interrupt is disabled
- 1 = Interrupt is enabled

9.5.2 Status Register (STAT)

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	LVI	LVIS27S	LVIS22S	LVIS27	LVIS22
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 9-4. Status (STAT) Register

9.5.2.1 Reserved—Bits 15–5

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

9.5.2.2 Low Voltage Interrupt (LVI)—Bit 4

LVI is a sticky status bit derived from the conditions where either of the 2.2 or 2.7 LVI thresholds have been met and associated interrupts are enabled.

LVI = (LVIS27 AND LVIE27) OR (LVIS22 AND LVIE22)

This bit may be cleared by writing one to it. Writing zero has no effect. It may be set again in the very next clock cycle after being cleared if the voltages are still below the thresholds.

- 0 = Interrupt not asserted
- 1 = Interrupt asserted

9.5.2.3 Sticky 2.7V Low Voltage Interrupt Source (LVIS27S)—Bit 3

When this bit is set it must be cleared explicitly by writing one to it. Writing zero has no effect.

- 0 = 2.7V interrupt threshold has not been passed
- 1 = 3.3V supply has dropped below the 2.7V interrupt threshold

Bit LVIS27S may be set again in the very next clock cycle after being cleared if the condition causing the supply voltage to drop still persists.

9.5.2.4 Sticky 2.2V Low Voltage Interrupt Source (LVIS22S)—Bit 2

When this bit is set, it must be cleared explicitly by writing one to it. Writing zero has no effect.

- 0 = 2.2V interrupt threshold has not been passed
- 1 = 2.5V supply dropped below the 2.2V interrupt threshold

Bit LVIS22S may be set again in the very next clock cycle after being cleared if the condition causing the supply voltage to drop still persists.

9.5.2.5 Non-Sticky 2.7V Low Voltage Interrupt Source (LVIS27)—Bit 1

This *read-only* bit resets itself if the supply voltage raises above the comparator threshold point. This bit cannot be modified.

- 0 = 2.7V interrupt threshold has not been passed
- 1 = 3.3V supply has dropped below the 2.7V interrupt threshold

9.5.2.6 Non-Sticky 2.2V Low Voltage Interrupt Source (LVIS22)—Bit 0

This *read-only* bit resets itself if the supply voltage raises above the comparator threshold point. This bit cannot be modified.

- 0 = 2.2V interrupt threshold has not been passed
- 1 = 2.5V supply is below the 2.2V interrupt threshold

9.6 Suggestions for LVI Interrupt Service Routines

The presence of one or more LVI bits high in the Power Supervisor Status (STAT) register indicates normal operation of the chip is no longer possible. If the 2.7V interrupt has occurred, the operation of the chip I/O is questionable. Peripherals will operate correctly but output signal levels will not meet requirements. If the 2.2V Interrupt has occurred, the core's operation at high speed is questionable.

Suggested Handling of the LVI Interrupt:

```
While (LVIS22S or LVIS27S) bit is set
{
    If (LVIS22S bit is set) then
    {
          Clear LVIS22S bit
          Reduce the core operating frequency to 8MHz
          Shutdown ALL peripherals in an orderly manner.
          While (LVIS22S bit is set) then
          {
          Clear LVIS22S bit
          Delay four clock cycles
          }
          Restore Core Operating Frequency
    }
    If (LVIS27S bit is set) then
    {
          Clear LVIS27S bit
          Shutdown I/O peripherals in an orderly manner
          While ((LVIS27S bit is set) and (LVIS22S bit is not set)) then
          {
          Clear LVIS27S bit
          Delay four clock cycles
          }
     }
}
Restore ALL peripherals to operation
Return from ISR
```

Chapter 10
Pulse Width Modulator (PWM)

Document Revision History for Chapter 10, Pulse Width Modulator (PWM)

Version History	Description of Change
Rev. 0	Initial release
Rev. 1	Added the word *two* to Note, below **Figure 10-13**, should read: “. . . output pin is delayed by two PWM Operation Clock cycles for. . .“Added to Note prior to **Section 10.4.6**, should read” “. . . new value does not take effect *until the next PWM period* so ICCn bits take . . ." **Section 10.10.4.3** corrected bit name in text from OUTCTRLn to *OUTCTLn* Sec 10.10.8.2 in two places, deleted System Clock and replaced with *PWM Operation Clock* cycle

10.1 Introduction

This chapter describes the Pulse Width Modulator (PWM) module. The PWM can be configured as three complementary pairs, six independent PWM signals or their combinations, such as one complementary and four independent. Both Edge- and Center-Aligned synchronous pulse width control, from zero to 100 percent modulation, are supported.

A 15-bit common PWM counter is applied to all six channels. PWM resolution is one clock period for Edge-Aligned operation and two clock periods for Center-Aligned operation. The clock period is dependent on clock source frequency at either System Clock or 3× System Clock. and a programmable prescaler.

When generating complementary PWM signals, the module features automatic deadtime insertion to PWM output pairs. Each PWM output can be controlled by PWM generator, timer, conversion results of ADC, GPIO pins, or software manually and separate top and bottom output polarity control. Asymmetric PWM output is able to change PWM duty cycle alternatively at every half cycle without software involvement.

10.2 Features

- PWM Operation Clock runs at either System Clock or 3× System Clock
- Six PWM signals
  - all independent
  - complementary pairs
  - mix independent and complementary
- Features of complementary channel operation
  - separate deadtime insertions for rising and falling edges
  - separate top and bottom pulse width correction via software
  - asymmetric PWM output within Center Align operation
  - separate top and bottom polarity control
- Edge- or Center-Aligned PWM signals
- 15 bits of resolution
- Half-cycle reload capability
- Integral reload rates from 1 to 16
- Individual software controlled PWM output
- Programmable fault protection
- PWM compare output polarity control
- PWM output polarity control
- Push-Pull and Open Drain modes are available on PWM pins

- Write-protected registers
- Selectable PWM supply source for each complementary PWM signal pair
  - — PWM generator
  - — external GPIO pin
  - — internal timer channel
  - — ADC conversion result, taking into account values set in ADC high and low limit registers

If all three PWM pairs are driven by any one of the above sources, the following features are disabled:

- PWM sync pulse is not applicable
- PWM reload registers will have no effect

10.3 Block Diagram

The PWM block diagram is illustrated in **Figure 10-1**.

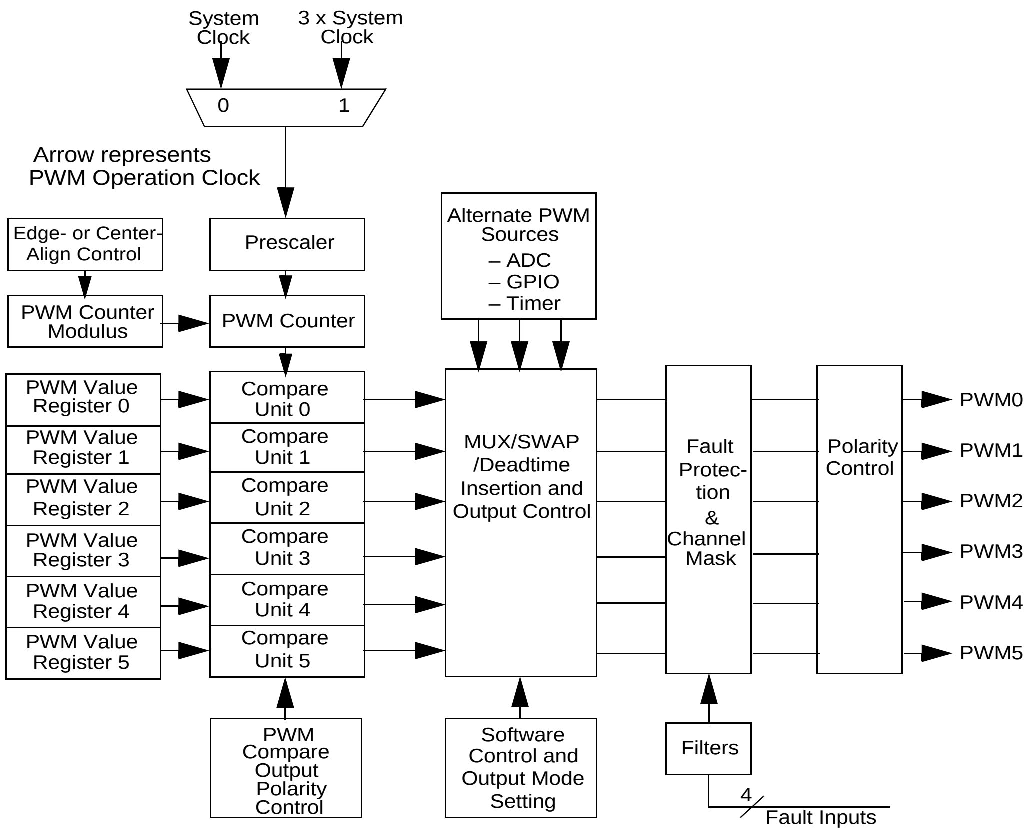


Figure 10-1. PWM Block Diagram

10.4 Functional Description

10.4.1 Prescaler

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the PWM Operation Clock frequency by one, two, four, or eight. The prescaler bits, PRSC0 and PRSC1 in the Control (CTRL) register, select the prescaler divisor. This prescaler is buffered and will not be used by the PWM generator until the LDOK bit is set and a new PWM reload cycle begins.

10.4.2 Generator

The PWM generator contains a 15-bit up/down PWM counter producing output signals with software selectable alignment, period, duty cycle, and the inversion of PWM signal generation.

10.4.2.1 Alignment and Compare Output Polarity

The Edge-Align (EDG) bit in the Configure (CNFG) register selects either Center-Aligned or Edge-Aligned PWM generator outputs.

PWM compare output polarity is selected by the CINV*n* bit field in the Source Control (SCTRL) register. Please see the output operations in **Figure 10-3** and **Figure 10-3.**

- The PWM compare output is driven to high state when the value of PWM Value (VAL0-5) register is greater than the value of PWM counter, and PWM compare is counting downwards if the corresponding channel CINV*x*=0. Or, the PWM compare output is driven to low state if the corresponding channel CINV*x*=1.
- The PWM compare output is driven to low state when the value of PWM Value (VAL0-5) register matches the value of PWM counter, and PWM counter is counting upwards if the corresponding channel CINV*x*=0. Or, the PWM compare output is driven to high state if the corresponding channel CINV*x*=1.

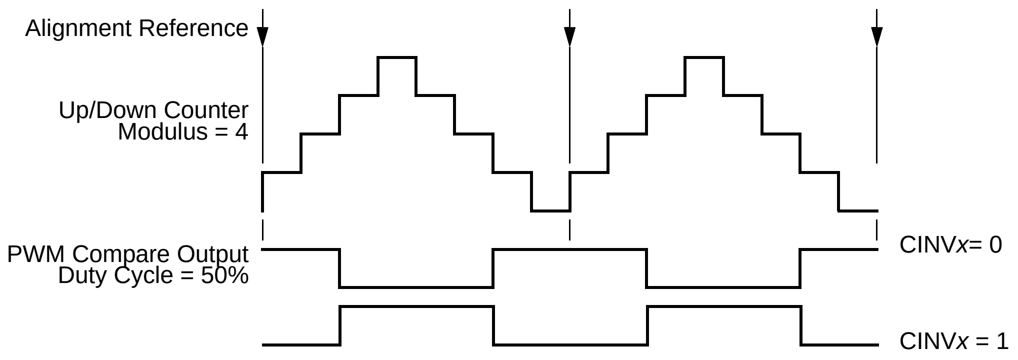


Figure 10-2. Center-Aligned PWM Output

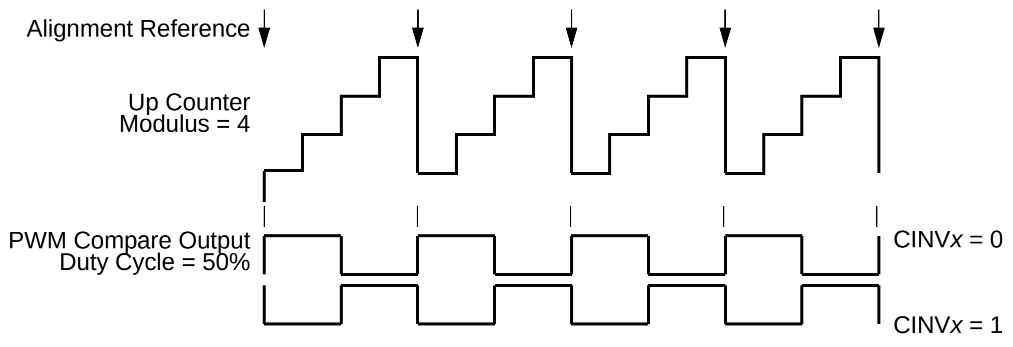


Figure 10-3. Edge-Aligned PWM Output

Note: Because of the equals-comparator architecture of this PWM, the modulus=0 case is considered illegal. However, the deadtime constraints and fault conditions will still be guaranteed.

10.4.2.2 Period

The PWM period is determined by the value written to the Counter Modulo (CMOD) register. The PWM counter is an up/down counter in a Center-Aligned operation. In this mode the PWM highest output resolution is two 3× System Clock cycles if the PWM clock inputs from 3× System Clock. The modulus is one-half of the PWM output period in PWM clock cycles.

$$\text{PWM period} = (\text{PWM modulus}) \times (\text{PWM clock period}) \times 2$$

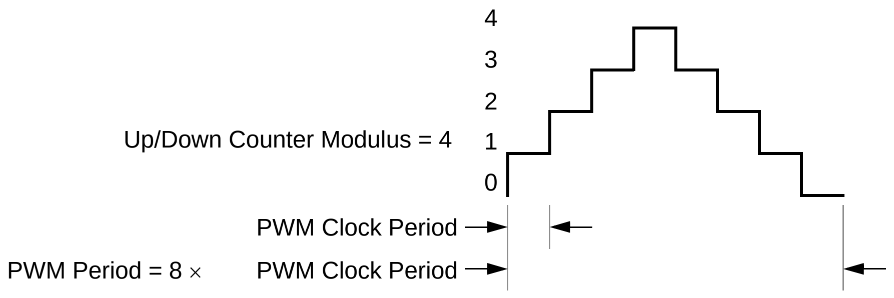


Figure 10-4. Center-Aligned PWM Period

The PWM counter is an up-counter during an Edge-Aligned operation. In this mode, the PWM highest output resolution is one 3× System Clock cycle if the PWM clock inputs from 3× System Clock. The modulus is the period of the PWM output in PWM clock cycles.

$$\text{PWM period} = (\text{PWM modulus}) \times (\text{PWM clock period})$$

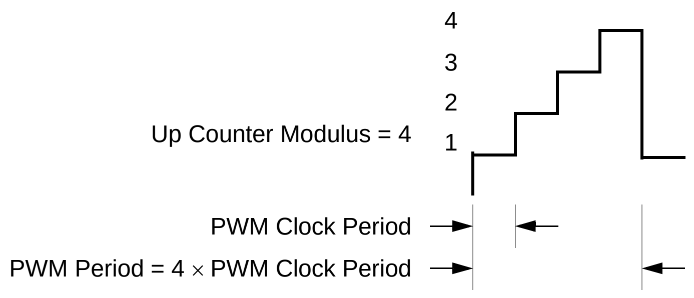


Figure 10-5. Edge-Aligned PWM Period

10.4.2.3 Pulse Width Duty Cycle

The signed 16-bit number written to the PWM value registers is the pulse width in PWM clock periods of the PWM prescaler output (or period minus the pulse width if CINV*x*=1).

$$\text{Duty Cycle} = \frac{\text{PWM value}}{\text{Modulus}} \times 100$$

Note: A PWM value less than, or equal to zero deactivates the PWM output for the entire PWM period. A PWM value greater than, or equal to the modulus activates the PWM output for the entire PWM period when CINV*x*=0, and vice versa if CINV*x*=1.

Table 10-1. PWM Value and Underflow Conditions

PWMVAL*n*	Condition	PWM Value Used
$0000–$7FFF	Normal	Value in Registers
$8000–$FFFF	Underflow	$0000

A Center-Aligned operation is illustrated in **Figure 10-6**. The pulse width is twice the value written to the PWM Value register with Center-Aligned output in PWM clock cycles.

$$\text{PWM pulse width} = (\text{PWM value}) \times (\text{PWM clock period}) \times 2$$

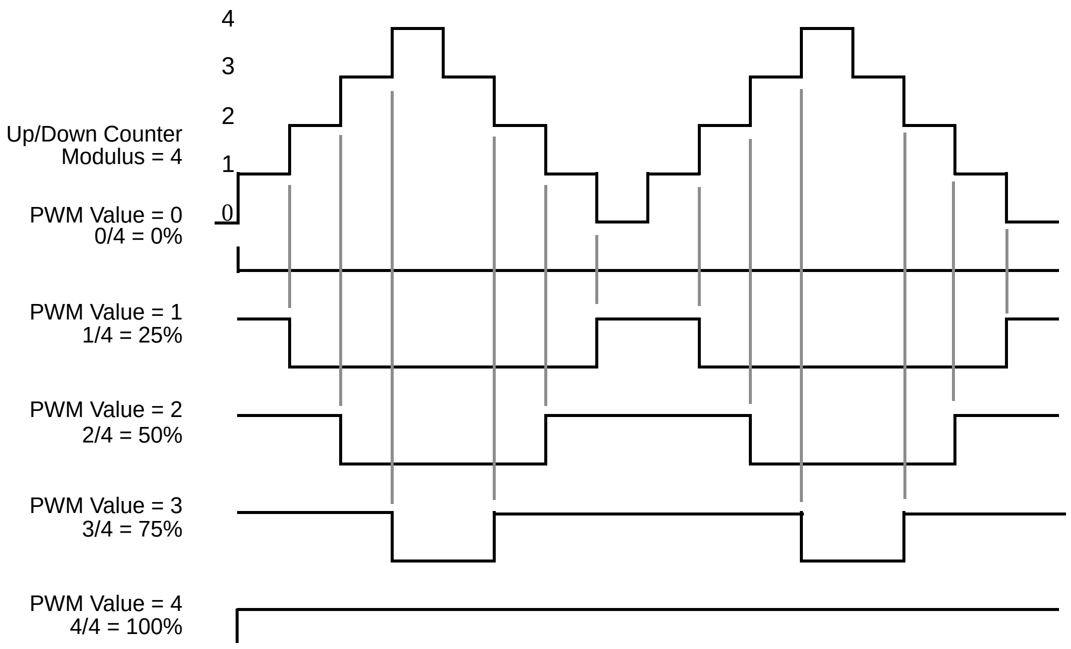


Figure 10-6. Center-Aligned PWM Pulse Width

An Edge-Aligned operation is illustrated in **Figure 10-7**. The pulse width is the value written to the PWM Value register with Edge-Aligned output in PWM clock cycles.

$$\text{PWM pulse width} = (\text{PWM value}) \times (\text{PWM clock period})$$

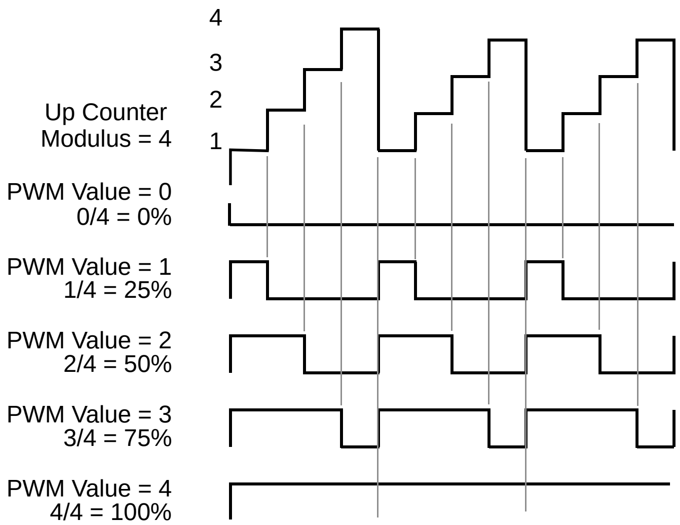


Figure 10-7. Edge-Aligned PWM Pulse Width

10.4.3 Independent or Complementary Channel Operation

In the CNFG register, writing Logic 1 to the Independent (INDEP*nn*) or complement pair operation bit configures a pair of the PWM outputs as two independent PWM channels. Each PWM output has its own PWM value register operating independently of the other channels in independent channel operation.

Writing Logic 0 to the INDEP*nn* bit configures the PWM output as a pair of complementary channels. The PWM pins are paired in complementary channel operation, illustrated in **Figure 10-8**.

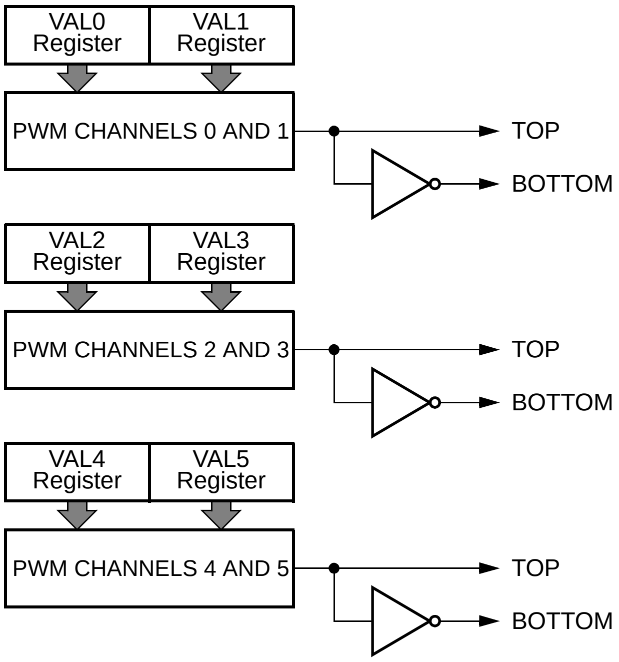


Figure 10-8. Complementary Channel Pairs

The complementary channel operation drives top and bottom transistors in an inverter circuit, such as the one in **Figure 10-9**.

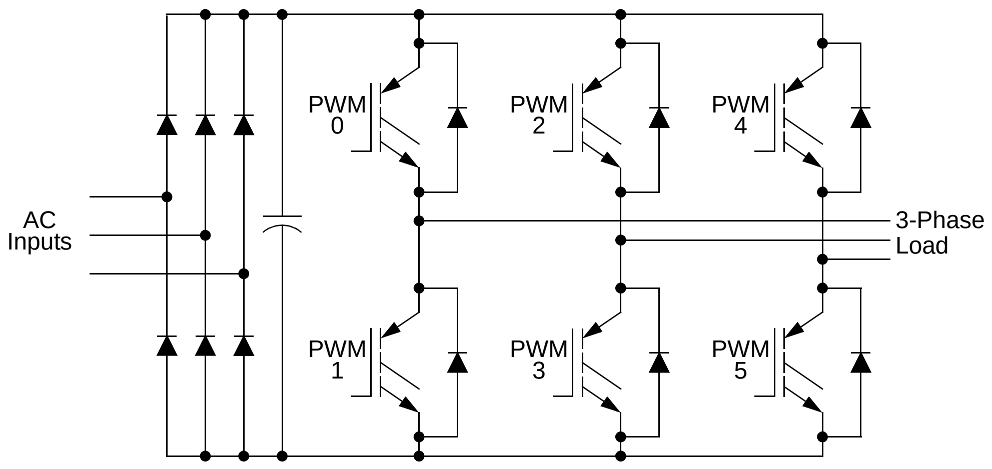


Figure 10-9. Typical 3-Phase Inverter

In complementary channel operation, there are three additional features:

1. Deadtime insertion
2. Separate top and bottom pulse width correction for distortions caused by deadtime inserted and reactive load characteristics
3. Separate top and bottom output polarity control

10.4.4 Deadtime Generators

While in the Complementary mode, each PWM pair can be used to drive top/bottom transistors, illustrated in **Figure 10-8** and **Figure 10-9.** Ideally, the PWM pairs are an inversion of each other. When the top PWM channel is active, the bottom PWM channel is inactive and vice versa.

To avoid short circuiting between top and bottom transistor, there must be no overlap of conducting intervals between top and bottom transistor. But the transistor’s characteristics make its switching-off time longer than switching-on time. To avoid the conducting overlap of top and bottom transistors, deadtime may be operationally inserted in the switching period.

Deadtime generators automatically insert software-selectable activation delays into each pair of PWM outputs during switching. The PWM Deadtime (DTIM1) registers specify the number of PWM clock cycles to use for deadtime delay. Every time the PWM generator output changes state, deadtime is inserted. DTIM0 controls deadtime during Low State to High State transitions, while DTIM1 controls deadtime during High State to Low State transitions. Deadtime forces both PWM outputs in the pair to the inactive state. A method of correcting this inserted deadtime, adding to or subtracting from the PWM value used, is discussed subsequently.

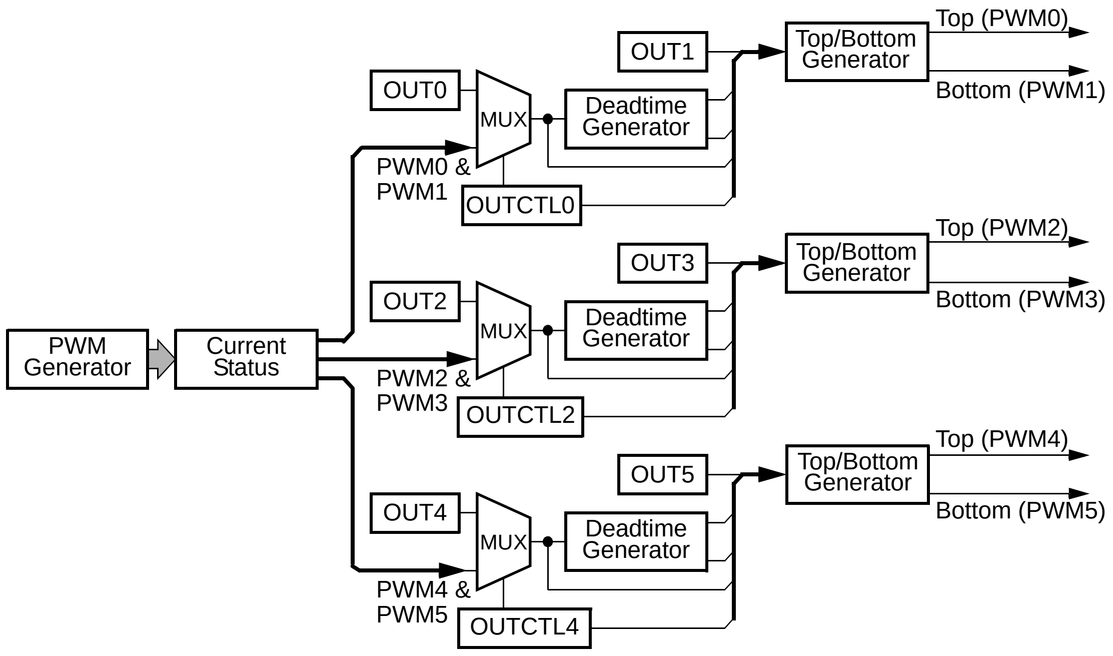


Figure 10-10. Deadtime Generators

Figure 10-10, Figure 10-11, and Figure 10-12 illustrate deadtime insertion in different operation conditions.

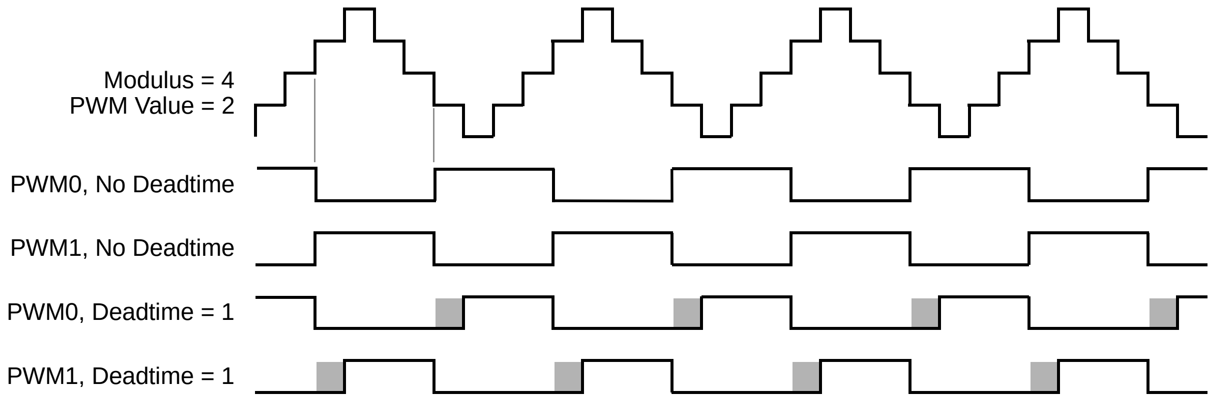


Figure 10-11. Deadtime Insertion, Center Alignment

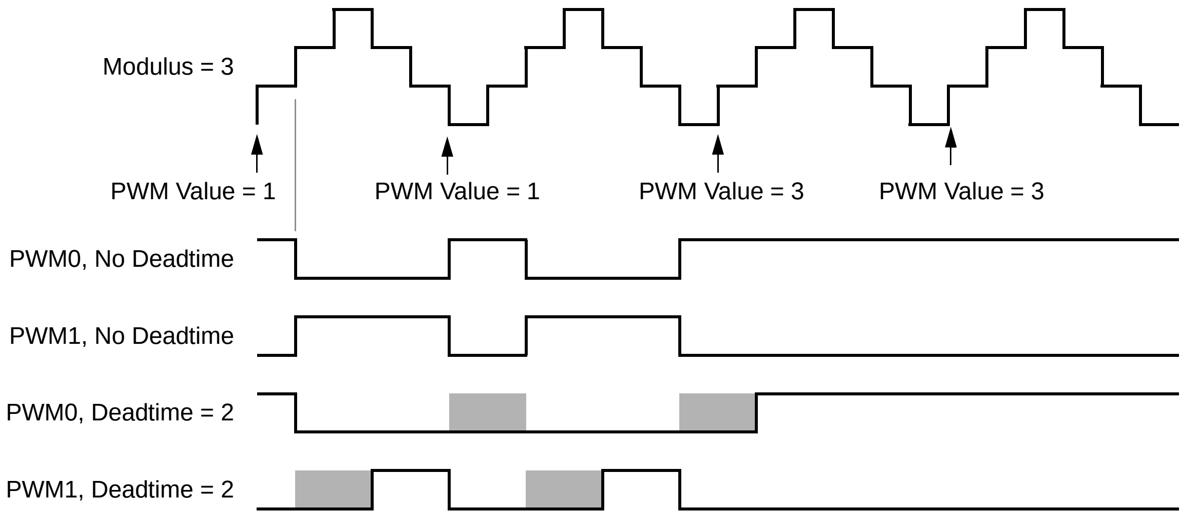


Figure 10-12. Deadtime at Duty Cycle Boundaries

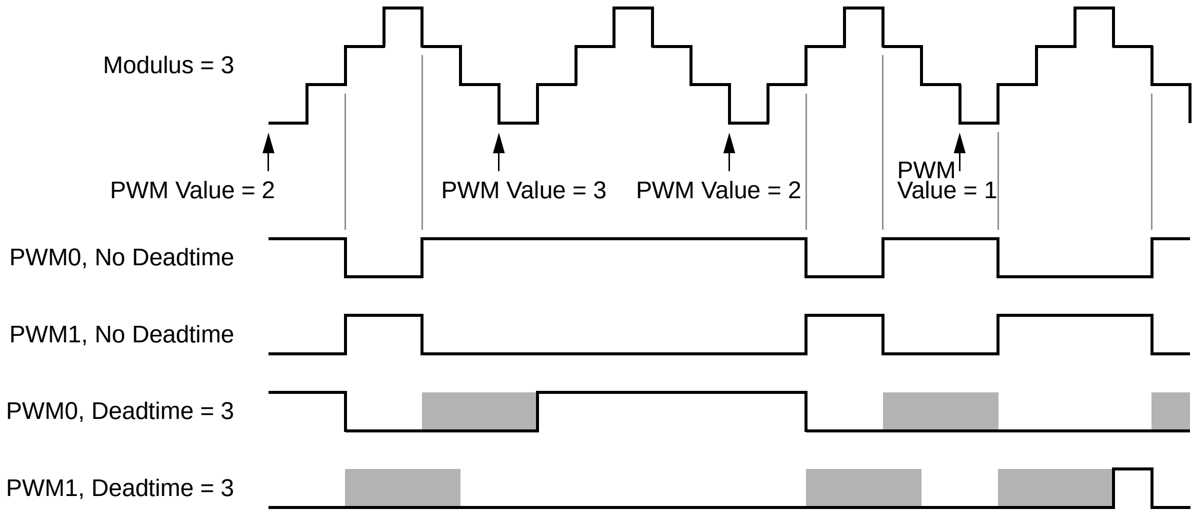


Figure 10-13. Deadtime and Small Pulse Widths

Note: The waveform at the output pin is delayed by two PWM Operation Clock cycles for deadtime insertion.

10.4.4.1 Top/Bottom Deadtime Correction

In the Complementary mode, either the top or the bottom transistor controls the output voltage. However, deadtime has to be inserted to avoid overlap of conducting interval between the top and bottom transitions. Both transistors in Complementary mode are off during deadtime inserted, allowing the output voltage to be determined by the current direction of load and introduce distortion in the output voltage. Please refer to **Figure 10-14.** The distortion typically manifests itself as poor output waveforms with visible glitches and harmonics.

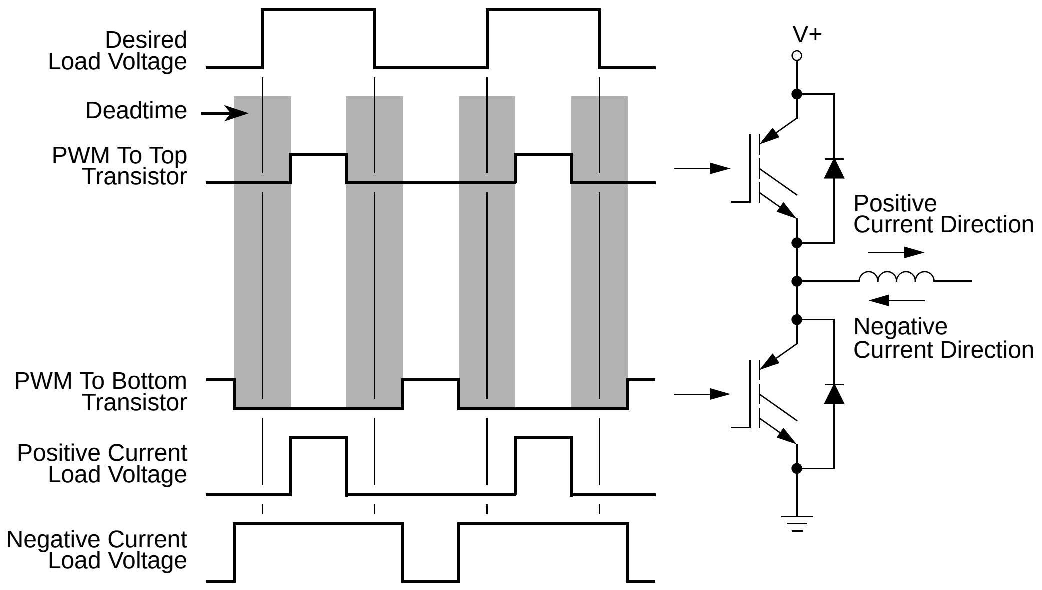

Desired Load Voltage
Deadtime
PWM To Top Transistor
PWM To Bottom Transistor
Positive Current Load Voltage
Negative Current Load Voltage
V+
Positive Current Direction
Negative Current Direction

Figure 10-14. Deadtime Distortion

Load inductance distorts output voltage by keeping current flowing through the anti-body diode of transistor during deadtime. This deadtime current flow creates a output voltage varying with current direction. With a positive current flow, the output voltage during deadtime is equal to the bottom supply voltage, putting the top transistor in control. With a negative current flow, the output voltage during deadtime is equal to the top supply voltage, putting the bottom transistor in control. This results in the original pulse widths shortened by deadtime insertion, the averaged output will be less than desired value. However, when deadtime is inserted, it creates a distortion in load current waveform. This distortion is aggravated by dissimilar turn-on and turn-off delays of each of the transistors. By giving the PWM information regarding which transistor is controlling at a given time, distortion can be corrected.

For a typical circuit in complementary channel operation, only one of the transistors will be effective in controlling the output voltage at any given time. This depends on the direction of the load current for that pair. Please see **Figure 10-14.** To correct distortion one of two different factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the software is responsible for calculating both compensated PWM values prior to placing them in an odd/even numbered PWM register pair. Either the odd or the even PWM Value (VAL*n*) registers control the pulse width at any given time. For a given PWM pair, whether the odd or even VAL register is active depends on either:

- The state of the odd/even correction bit (IPOL*n*) if ICC bits in the ICCTRL register are set to zeros
- The direction of PWM counter if ICC bits in the ICCTRL register are set to ones

To correct deadtime distortion, software can decrease or increase the value in the appropriate VAL register.

- In Edge-Aligned operation, decreasing or increasing the PWM value by a correction value equal to the deadtime typically compensates for deadtime distortion.
- In Center-Aligned operation, decreasing or increasing the PWM value by a correction value equal to one-half the deadtime typically compensates for deadtime distortion.

Note: Assumes the user will provide current sensing circuitry to detect the current direction.

The IPOL0–IPOL2 bits in Control (CTRL) register select either the odd or the even PWM value registers to use in the next PWM cycle in Complementary mode if corresponding ICC*n* bit is 0.

Table 10-2. Top/Bottom Manual Correction

Bit	Logic State	Output Control
IPOL0	0	VAL0 Controls PWM0/PWM1 Pair
	1	VAL1 Controls PWM0/PWM1 Pair
IPOL1	0	VAL2 Controls PWM2/PWM3 Pair
	1	VAL3 Controls PWM2/PWM3 Pair
IPOL2	0	VAL4 Controls PWM4/PWM5 Pair
	1	VAL5 Controls PWM4/PWM5 Pair

Note: IPOL*n* bits are buffered allowing only one PWM register to be used per PWM cycle. If an IPOL*n* bit changes during a PWM period, the new value does not take effect until the next PWM period. IPOL*n* bits take effect at the end of each PWM cycle regardless of the state of the Load Okay (LDOK) bit.

Corrected local voltage waveforms are illustrated in **Figure 10-15** and **Figure 10-16.**

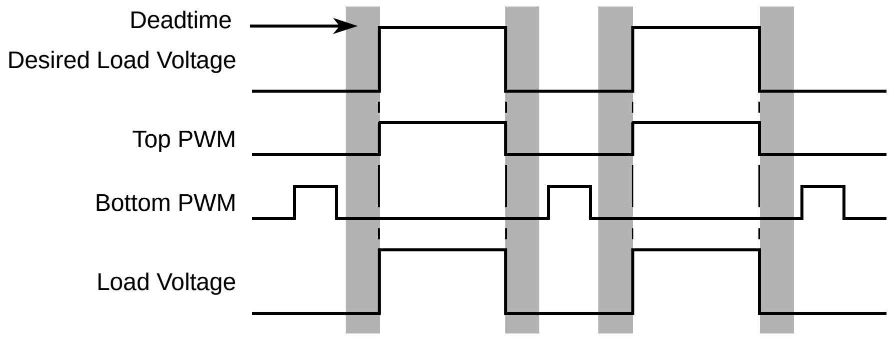


Figure 10-15. Correction with Positive Current

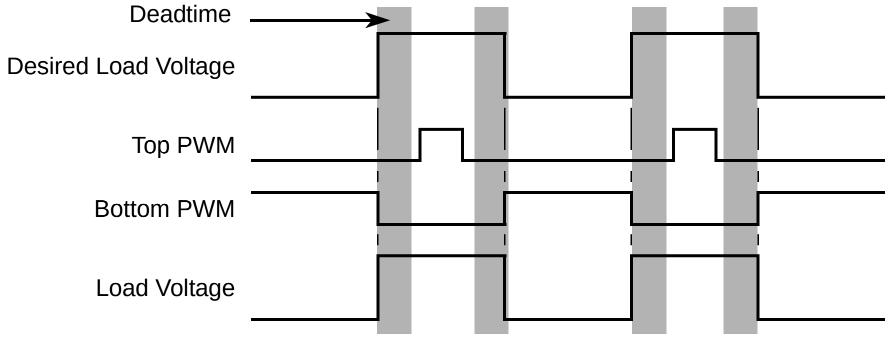


Figure 10-16. Correction with Negative Current

10.4.5 Asymmetric PWM Output

In Complementary mode with Center-Align operation, the PWM duty cycle is able to change alternatively at every half cycle. The count direction of the PWM counter selects either the odd or the even PWM value registers to use in the PWM cycle. For counting up, select even PWM value registers to use in the PWM cycle. For counting down, select odd PWM value registers to use in the PWM cycle.

Table 10-3. Top/Bottom Corrections Selected by ICC*n* Bits

Bit	Logic State	Output Control
ICC0	0	IPOL0 Controls PWM0/PWM1 Pair
	1	PWM Count Direction Controls PWM0/PWM1 Pair
ICC1	0	IPOL1 Controls PWM2/PWM3 Pair
	1	PWM Count Direction Controls PWM2/PWM3 Pair
ICC2	0	IPOL2 Controls PWM4/PWM5 Pair
	1	PWM Count Direction Controls PWM4/PWM5 Pair

Note: If an **ICC*n*** bit in ICCTRL register changes during a PWM period, the new value does not take effect until the next PWM period so ICC*n* bits take effect at the end of each PWM cycle regardless of the state of the Load Okay (LDOK) bit.

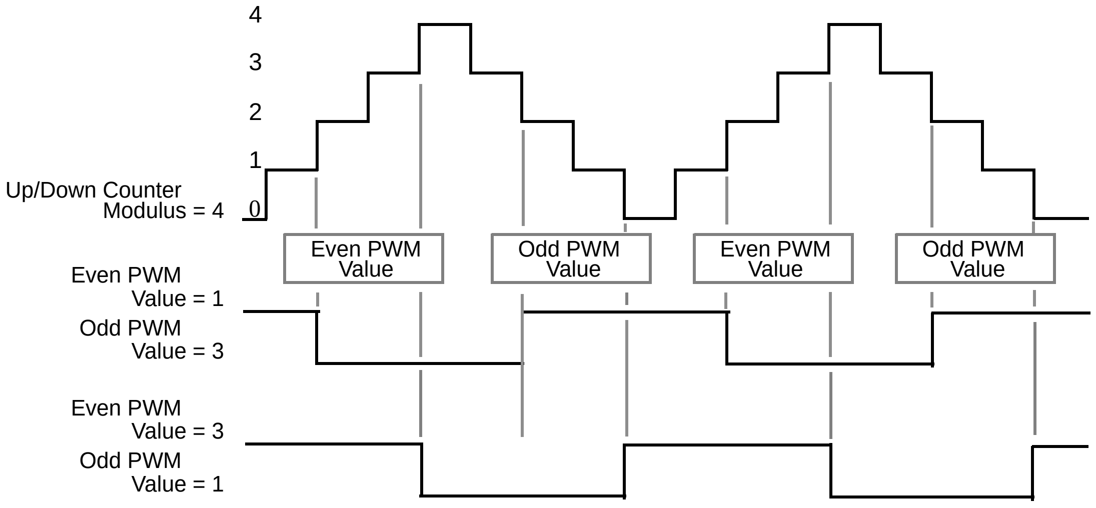


Figure 10-17. Asymmetric Waveform - Phase Shift PWM Output

10.4.6 PWM Output Polarity

Positive polarity means when the PWM is active its output is high. Conversely, *negative* polarity means when the PWM is active its output is low.

Output polarity of the PWMs is determined by two options:

1. TOPNEG*nn* controls the polarity of PWM0, PWM2 and PWM4 outputs, which typically drive the top transistors of the pair. When TOPNEG*nn* is set these outputs are active-low.
2. BOTNEG*nn* controls the polarity of PWM1, PWM3 and PWM5 outputs, which typically drive the bottom transistors of the pair. When BOTNEG*nn* is set these outputs are active-low.

Both TOPNEG*nn* and BOTNEG*nn* bits are in the Configure (CNFG) register. Please see **Figure 10-18**.

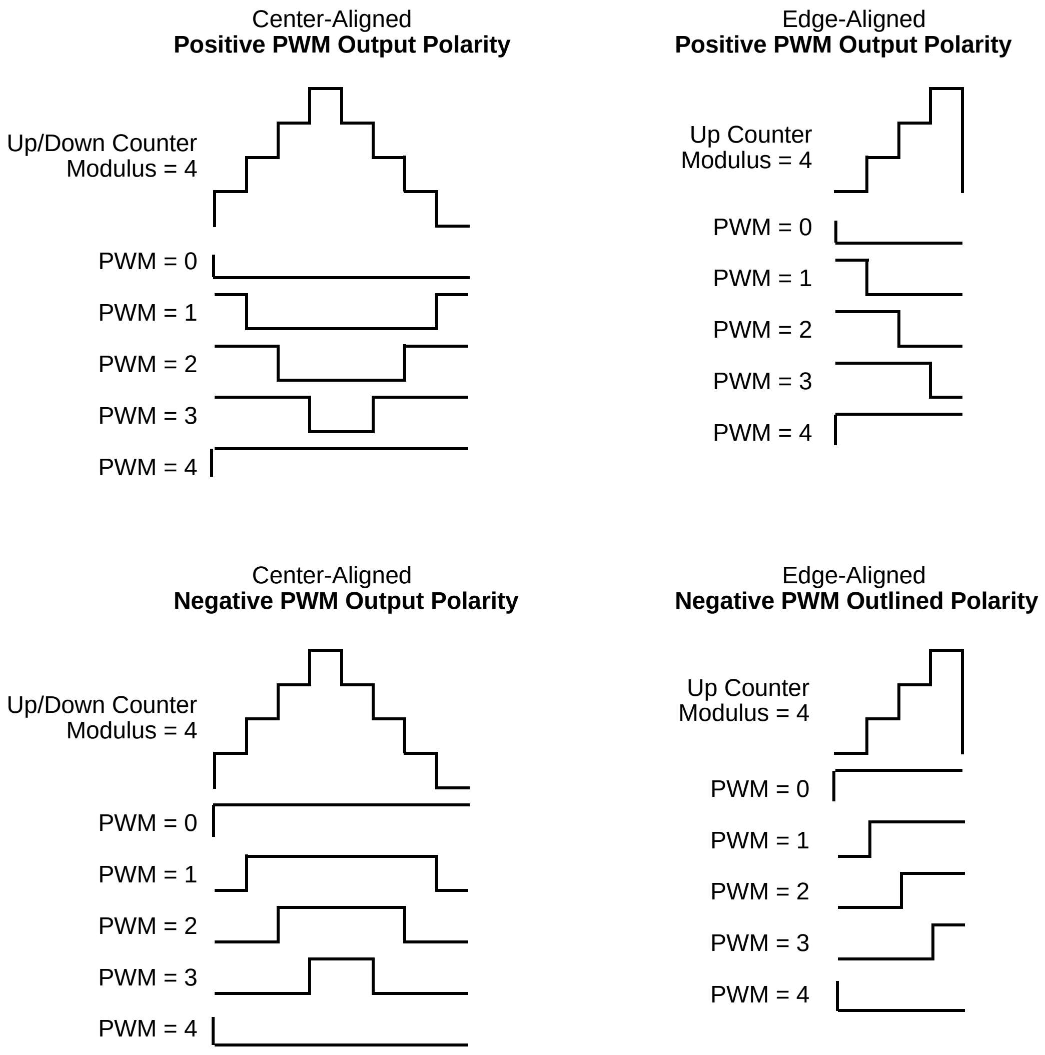


Figure 10-18. PWM Output Polarity

10.5 Software Output Control

Setting Output Control Enable (OUTCTRL*n*) bit, the PWM outputs are driven by software rather than by the PWM generator.

In an Independent mode, with OUTCTRL*n*=1, the output bit OUT*n*, controls the PWM*n* channel. Setting and clearing the OUT*n* bit activates and deactivates the corresponding PWM channel.

The OUTCTRL*n* and OUT*n* bits are in the PWM Output Control (OUT) register.

During software output control, TOPNEG*nn* and BOTNEG*nn* still control output polarity.

In complementary channel operation odd and even OUTCTRL*n* must be identical and switched concurrently for proper operation. The even-numbered OUT*n* bits replace the PWM generator outputs. The deadtime generators inserts deadtime whenever an even OUT*n* bit toggles. Deadtime is not inserted when the odd OUT*n* bit toggles. The even OUT*n* bit controls complementary channel pairs when the odd OUT*n* bit is set. However, the even OUT*n* bit still controls complementary channel pairs with odd PWM*n* deactivated if the odd OUT*n* bit is cleared. In other words, setting the odd OUT*n* bit makes its corresponding PWM*n* the complement of its even pair, while clearing the odd OUT*n* bit deactivates the odd PWM*n*. Please refer to **Figure 10-19.**

Setting the OUTCTL*n* bits do not disable the PWM generators. They continue to run, but no longer control the output pins. When the OUTCTL*n* bits are cleared, the outputs of the PWM generator takes control of PWM outputs at the beginning of the next PWM cycle. Please refer to **Figure 10-19.**

Software can drive the PWM outputs, even when the PWM Enable (PWMEN) bit is set to zero.

Note: Avoid an unexpected deadtime insertion by clearing the OUT*n* bits before setting and after clearing the OUTCTL*n* bits.

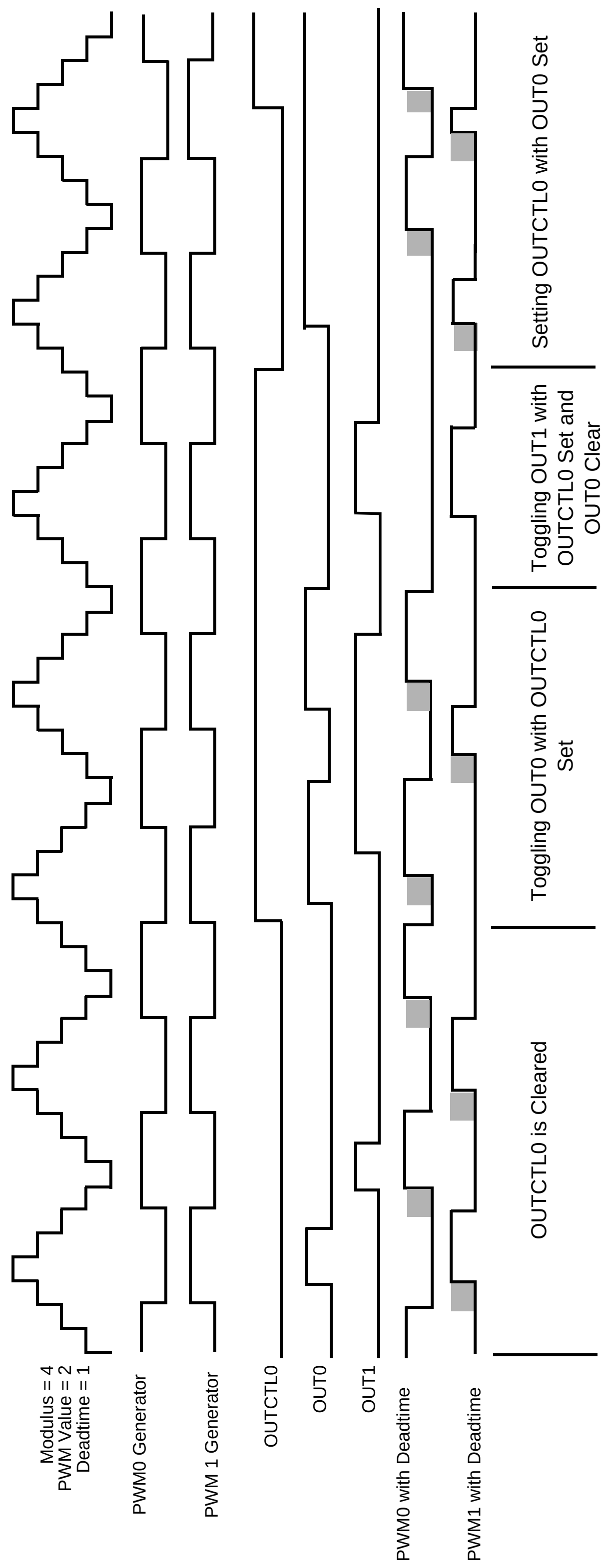


Figure 10-19. Software Output Control in Complementary Mode

10.6 Generator Loading

10.6.1 Load Enable

The Load Okay (LDOK) bit enables loading the PWM generator with:

- A prescaler divisor from the PRSC1 and PRSC0 bits in the Control (CTRL) register
- A PWM period from the PWM Counter Modulus (CMOD) registers
- A PWM pulse width from the all PWM Value (VAL*n*) registers

LDOK prevents reloading of these PWM parameters simultaneously. Setting LDOK allows the prescale bits, CMOD and VAL*n* registers to be loaded into a set of buffers. The loaded buffers are used by the PWM generator at the beginning of the next PWM reload cycle. Set LDOK by reading it, and then writing a Logic 1 to it. After loading, LDOK is automatically cleared.

10.6.2 Load Frequency

The LDFQ3, LDFQ2, LDFQ1, and LDFQ0 bits in the CTRL register select an integral loading frequency of one to 16-PWM reload opportunities. The LDFQ bits take effect at every PWM reload opportunity, regardless of the state of the LDOK bit. The HALF bit in the CTRL register controls half-cycle reloads for Center-Aligned PWMs. If the HALF bit is set, a reload opportunity occurs at both beginning of the PWM cycle and at the PWM half cycle. If the HALF bit is not set, a reload opportunity occurs only at the beginning of the cycle. Reload opportunities can only occur at the beginning of a PWM cycle in Edge-Aligned mode.

Note: Loading a new modulus on a half cycle will force the counter to the new modulus value *minus one* count on the next PWM Clock cycle. Half cycle reloads are only changes reload rate in Center-Aligned mode. Enabling or disabling half cycle reloads in Edge-Aligned mode will have no effect on the reload rate.

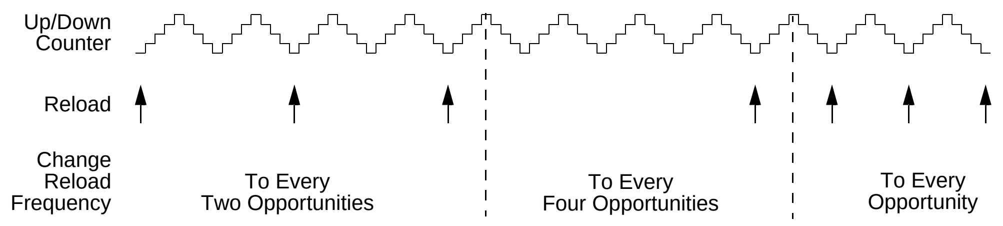


Figure 10-20. Full Cycle Reload Frequency Change

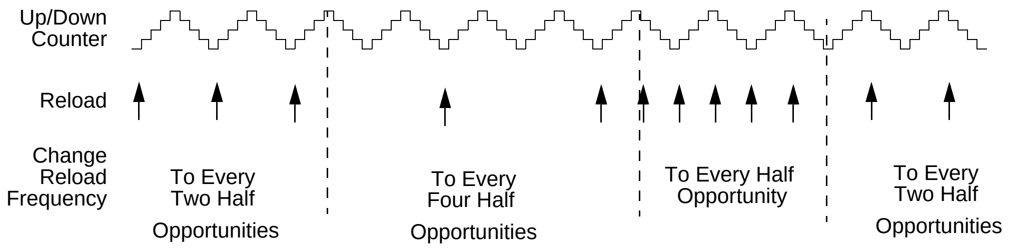

Up/Down Counter
Reload
Change Reload Frequency
To Every Two Half Opportunities
To Every Four Half Opportunities
To Every Half Opportunity
To Every Two Half Opportunities

Figure 10-21. Half Cycle Reload Frequency Change

10.6.3 Reload Flag

At every reload opportunity the PWM Reload Flag (PWMF) bit in the CTRL register is set regardless of the state of the LDOK bit. If the PWM Reload Interrupt Enable (PWMRIE) bit is set, the PWMF flag generates a core interrupt request allowing software to calculate new PWM parameters in real time. When PWMRIE is not set, reloads still occur at the selected reload rate without generating interrupt requests. Clear the PWMF bit by reading it then write a Logic 0 to it.

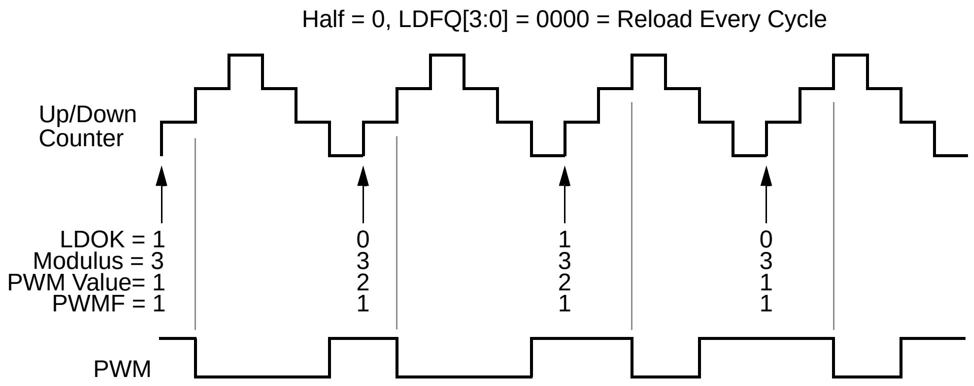


Figure 10-22. Full-Cycle Center-Aligned PWM Value Loading

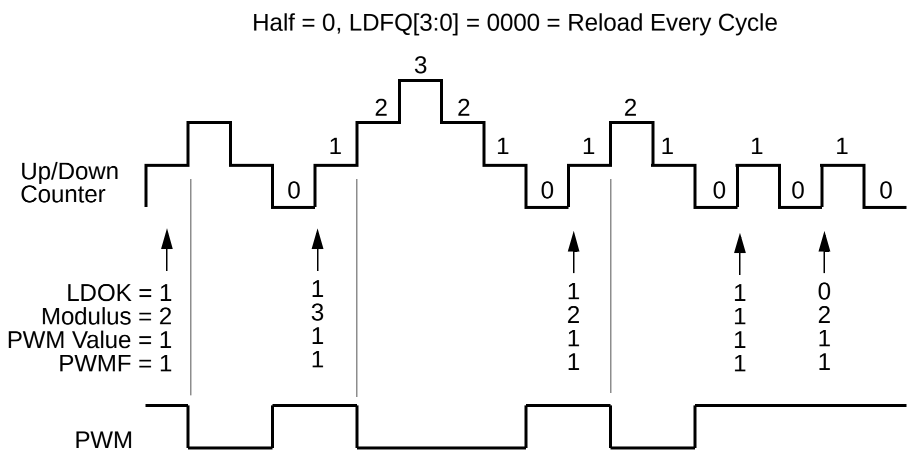


Figure 10-23. Full-Cycle Center-Aligned Modulus Loading

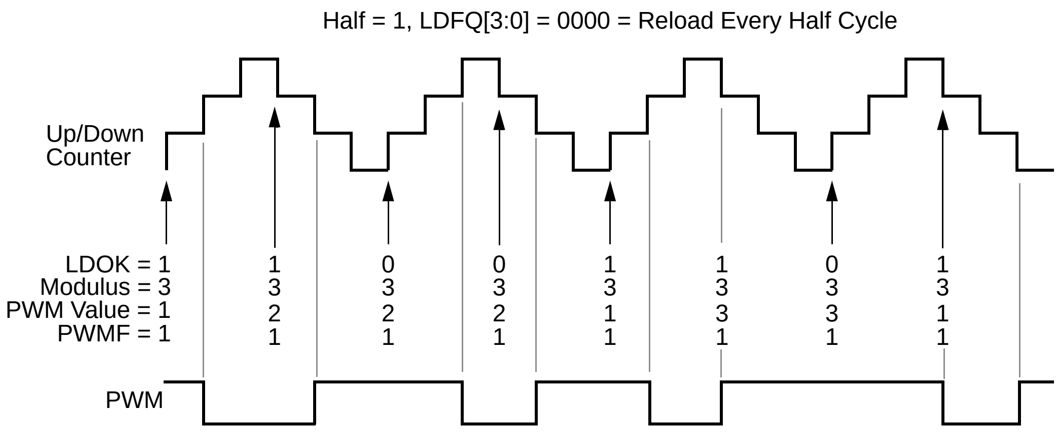


Figure 10-24. Half-Cycle Center-Aligned PWM Value Loading

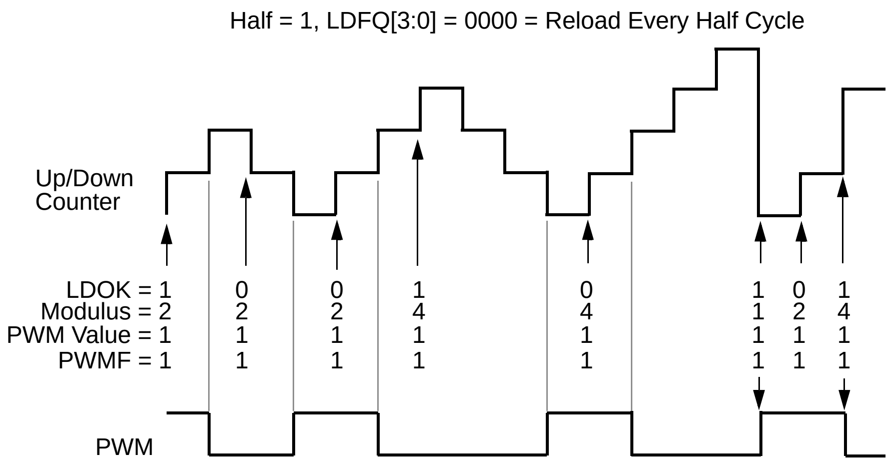


Figure 10-25. Half-Cycle Center-Aligned Modulus Loading

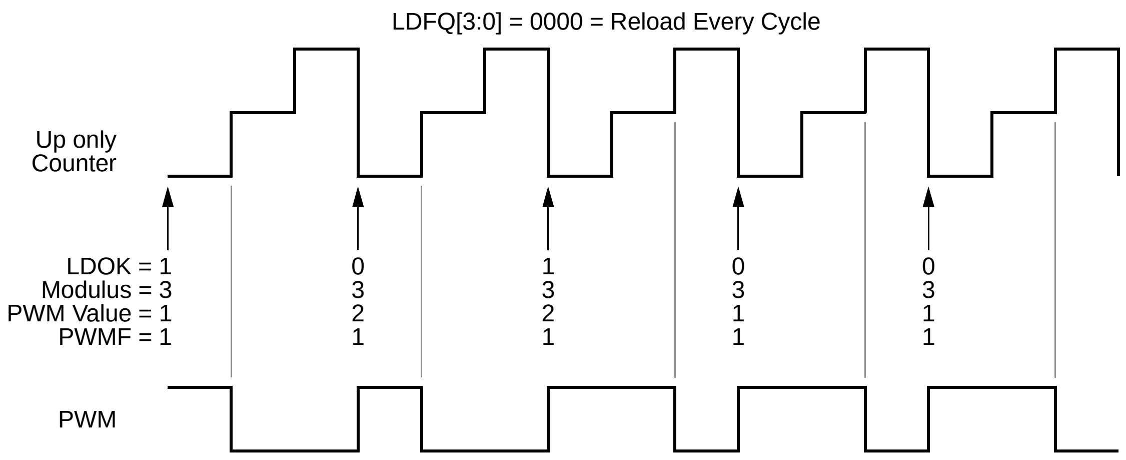


Figure 10-26. Edge-Aligned PWM Value Loading

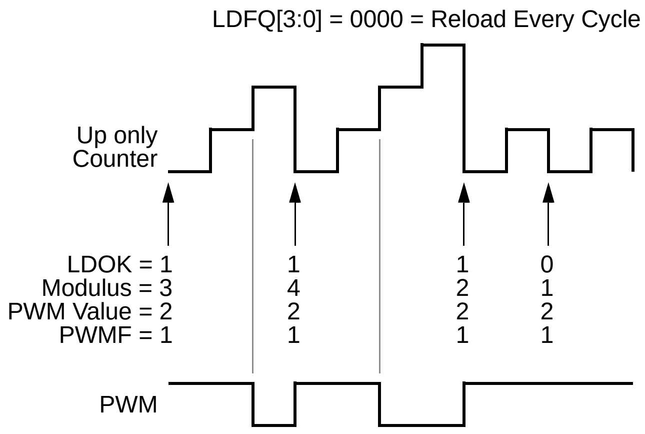


Figure 10-27. Edge-Aligned Modulus Loading

10.6.4 Synchronization Output

The PWM uses reload events to output a synchronization pulse, which can be used as an input to the Timer module. A high-true pulse occurs for each PWM cycle start of the PWM, regardless of the state of the LDOK bit and load frequency.

10.6.5 Initialization

Initialize all registers and set the Load Okay (LDOK) bit before setting the ENABLE (PWMEN) bit. With LDOK set, setting the PWMEN bit is first set, a reload will immediately occur, thereby setting the PWMF bit. The PWMF bit generates an interrupt request if the PWMRIE bit is set. In complementary channel operation, the combination of IPOL*n* bits and ICC*n* bits determine the even or odd numbered PWM Value registers control the outputs for the first PWM cycle.

Note: Even if LDOK is not set, setting PWMEN also sets the PWMF bit. To prevent a core interrupt request, clear the PWMRIE bit before setting PWMEN bit.

Setting PWMEN bit for the first time after reset without first setting LDOK loads a prescaler divisor of one, a PWM value of $0000, and an unknown modulus. If the LDOK bit is not set after the PWMEN bit is cleared, then set (without a RESET) the value last loaded will be used in the PWM generated. If the Deadtime register is changed after PWMEN or OUTCTL*n* bits are set, an improper deadtime insertion will occur.

Initializing the deadtime register after setting PWMEN or OUTCTL*n* can cause an improper deadtime insertion. However, the deadtime can never be shorter than the specified value.

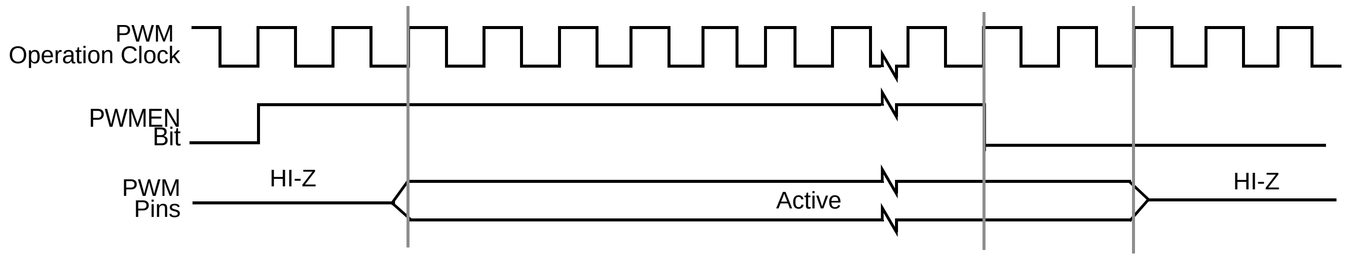


Figure 10-28. PWMEN and PWM Pins in Independent Operation (OUTCTL0–5 = 0)

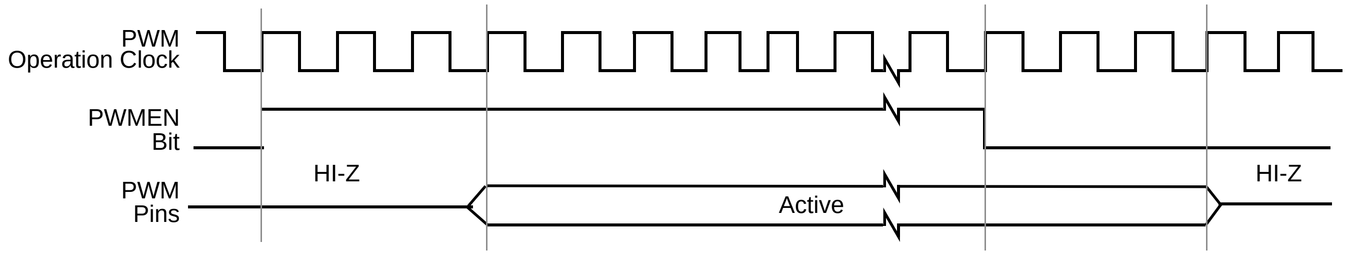


Figure 10-29. PWMEN and PWM Pins in Complement Operation (OUTCTL0, 2, 4 = 0)

When the PWMEN bit is cleared:

- The PWM*n* pins will be in their inactive status unless OUTCTL*n*=1
- The PWM counter is cleared and does not count
- The PWM generator forces its outputs to zero
- The PWMF and pending interrupt requests are not cleared
- All fault circuitry remains active
- Software output control remains active if OUTCTL*n*=1
- Deadtime insertion continues during software output control

10.7 Fault Protection

Fault protection can disable any combination of PWM pins. Faults are generated by either a Logic 1 or Logic 0, determined by the fault polarity control bits in the fault control (FCTRL) register on any of the FAULT pins. Each FAULT pin can be mapped arbitrarily to any of the PWM pins. When fault protection hardware disables PWM pins, the PWM generator continues to run, only the output pins are deactivated. The fault decoder disables PWM pins selected by the fault logic and the disable mapping register. Please see **Figure 10-30**. Each bank of four bits in the disable mapping registers (DMAP1–2) control the mapping for a single PWM pin. Please refer to **Table 10-4**. The fault protection is enabled even when the PWM is not enabled; therefore, if a fault is latched in, it must be cleared prior to enabling the PWM to prevent an unexpected interrupt. Please see **Section 10.10.3.4**.

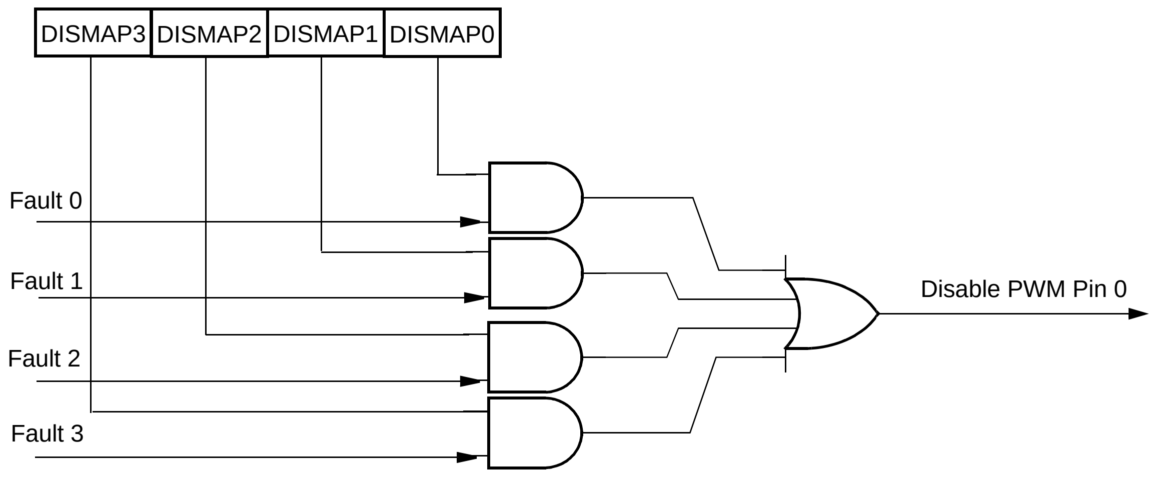


Figure 10-30. Fault Decoder for PWM 0

Table 10-4. Fault Mapping

PWM Pin	Controlling Register Bits
PWM0	DISMAP3–DISMAP0
PWM1	DISMAP7–DISMAP4
PWM2	DISMAP11–DISMAP8
PWM3	DISMAP15–DISMAP12
PWM4	DISMAP19–DISMAP16
PWM5	DISMAP23–DISMAP20

Note: For parts with less than four fault pins, the same controls apply. The unavailable DISMAP field bits should be set to zero. For example, if Fault 3 is not available as an input, set DISMAP3=0.

10.7.1 Fault Pin Filter

Each fault pin has a filter to test for fault conditions. A fault input transition to a high state is not declared until the input is sampled high on two consecutive PWM Operation Clocks. Only then FFLAG*n* and FPIN*n* are set. The FPIN*n* bit will remain set until the Fault input is detected low on two consecutive PWM Operation Clocks. Clear FFLAG*n* by writing a Logic 1 to the corresponding Fault Acknowledge (FTACK*n*) bit. If the FIE*n*, FAULT*n* pin interrupt enable bit is set, the FFLAG*n* flag generates an interrupt request. The interrupt request latch remains set until one of the following actions occur:

- Software clears the FFLAG*n* flag by writing a Logic 1 to the FTACK*n* bit
- Software clears the FIE*n* bit by writing a Logic 0 to it
- A reset occurs

10.7.2 Automatic Fault Clearing

In Automatic mode, when FMODE*n* is set, disabled PWM pins are enabled when the FAULT*n* pin returns to Logic 0 and a new PWM half cycle begins. Please refer to **Figure 10-31.** Clearing the FFLAG*n* flag does not affect disabled PWM pins when FMODE*n* is set.

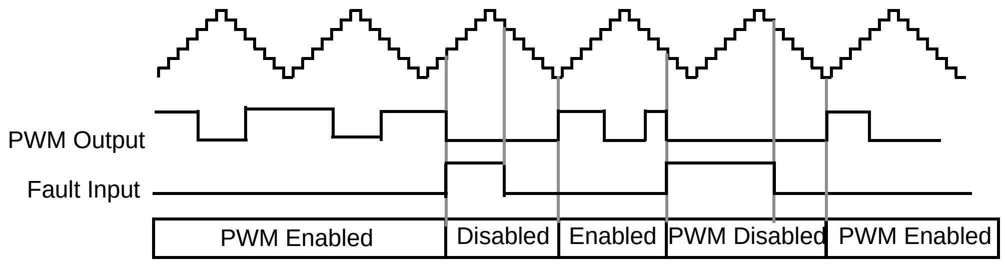


Figure 10-31. Automatic Fault Clearing

10.7.3 Manual Fault Clearing

In Manual mode, the fault pins are grouped in pairs, each pair sharing common functionality. A fault condition on Fault pins 0 and 2 can be cleared by software clearing the corresponding FFLAG bit, allowing the PWM(s) to enable at the next PWM half cycle regardless of the logic level at the Fault pin. The PWM outputs will remain enabled even if the logic level of the fault pin is still high. The fault pin must go low and then back high in order to register a new fault and disable the PWM outputs. **Figure 10-32.** A fault condition on Fault pins 1 and 3 can only be cleared by software clearing corresponding FFLAG*n* bit, allowing the PWM(s) to enable if a Logic Low at the Fault pin is detected at the start of the next PWM half cycle boundary. Please see **Figure 10-33**.

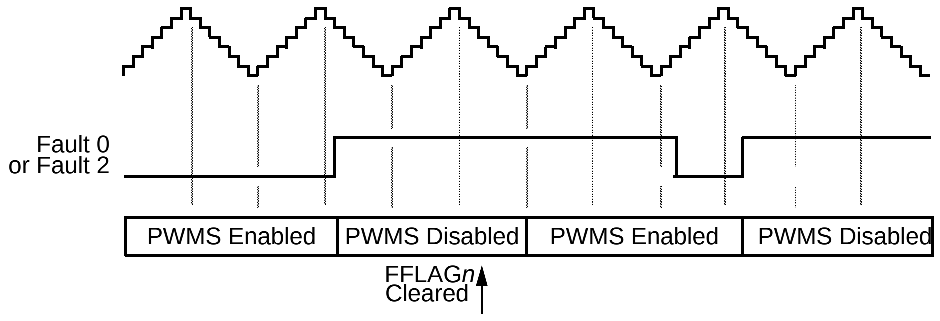


Figure 10-32. Manual Fault Clearing (Example 1)

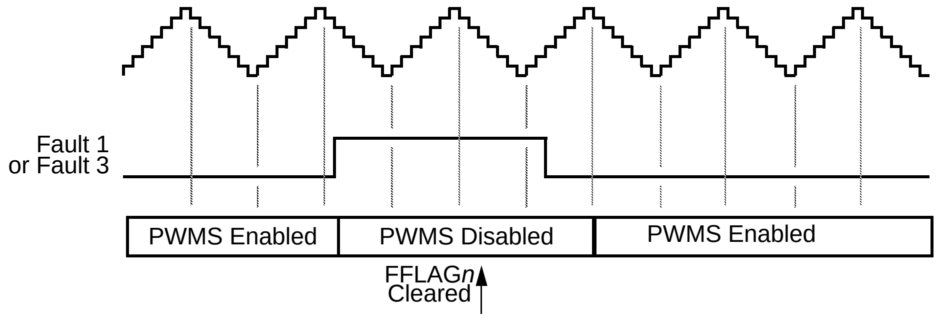


Figure 10-33. Manual Fault Clearing (Example 2)

Note: PWM half-cycle boundaries occur at both the PWM cycle start and when the counter equals the modulus, so in Edge-Aligned operation full cycles and half cycles are equal.

Note: Fault protection also applies during software output control when the OUTCTL*n* bits are set. Fault clearing still occurs at half PWM cycle boundaries while the PWM generator is engaged where PWMEN=1. However, the OUT*n* bits can also control the PWM pins while the PWM generator is off where PWMEN=0. Thus, fault clearing occurs at PWM Operation Clock cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

10.8 Operating Modes

Exercise care when using this module in certain chip operating modes. Some applications require regular software updates for proper operation. Failure to use caution could result in damaging the circuit. Because of this, PWM outputs are placed in their inactive states in Stop mode, and optionally under Wait and EOnCE modes. PWM outputs will be reactivated (assuming they were active to begin with) when these modes are exited.

Table 10-5. Modes When PWM Operation is Restricted

Mode	Description
Stop	PWM Outputs are Disabled
Wait	PWM Outputs are Disabled as a Function of the CNFG WAIT_EN Bit
EOnCE	PWM Outputs are Disabled as a Function of the CNFG DBG_EN Bit

10.9 Pin Descriptions

The Pulse Width Modulator (PWM) is capable of having the following external pins.

10.9.1 PWM0–PWM5 Pins—(PWM0–5)

PWM0–PWM5 are output pins of the six PWM channels.

10.9.2 FAULT0–FAULT3 Pins—(FAULT0–3)

FAULT0–FAULT3 are input pins for disabling selected PWM outputs.

10.10 Register Definitions

Table 10-6. PWM Memory Map

Device	Peripheral	Base Address
56F801X	PWM	$00F040

The address of a register is the sum of a base address and an address offset. The base address is defined at the core level and the address offset is defined at the module level.

Table 10-7. PWM Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	CTRL	Control Register	Read/Write	Section 10.10.1
Base + $1	FCTRL	Fault Control Register	Read/Write	Section 10.10.2

Table 10-7. PWM Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $2	FLTACK	Fault Status/Acknowledge Reg.	Read/Write	**Section 10.10.3**
Base + $3	OUT	Output Control Register	Read/Write	**Section 10.10.4**
Base + $4	CNTR	Counter Register	*Read-Only*	**Section 10.10.5**
Base + $5	CMOD	Counter Modulo Register	Read/Write	**Section 10.10.6**
Base + $6 to $B	VAL0-5	Value Register 0-5	Read/Write	**Section 10.10.7**
Base + $C to $D	DTIM0-1	Deadtime Register 0-1	Read/Write	**Section 10.10.8**
Base + $E to $F	DMAP1-2	Disable Mapping Register 1-2	Read/Write	**Section 10.10.9**
Base + $10	CNFG	Configure Register	Read/Write	**Section 10.10.10**
Base + $11	CCTRL	Channel Control Register	Read/Write	**Section 10.10.11**
Base + $12	PORT	Port Register	Read/Write	**Section 10.10.12**
Base + $13	ICCTRL	Internal Correction Control Reg.	Read/Write	**Section 10.10.13**
Base + $14	SCTRL	Source Control Register	Read/Write	**Section 10.10.14**

Bit fields of each of the 19 registers are illustrated in **Figure 10-34.** Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	CTRL	R	LDFQ				HALF	IPOL2	IPOL1	IPOL0	PRSC		PWMRIE	PWMF	▨	▨	LDOK	PWMEN
		W													0	0		
$1	FCTRL	R	0	0	0	0	0	0	0	0	FIE3	FMODE3	FIE2	FMODE2	FIE1	FMODE1	FIE0	FMODE0
		W	▨	▨	▨	▨	▨	▨	▨	▨								
$2	FLTACK	R	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1	FPIN0	FFLAG0	▨	▨	▨	▨	▨	▨	▨	▨
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	FTACK3	▨	FTACK2	▨	FTACK1	▨	FTACK0
$3	OUT	R	PAD_EN	0	OUT CTL5	OUT CTL4	OUT CTL3	OUT CTL2	OUT CTL1	OUT CTL0	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
		W		▨							▨	▨						
$4	CNTR	R	0	CR														
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$5	CMOD	R	0	PWMCM														
		W	▨															
$6-$B	VAL0-5	R	PWMVAL															
		W																
$C	DTIM0	R	0	0	0	0	PWMDT0											
		W	▨	▨	▨	▨												
$D	DTIM1	R	0	0	0	0	PWMDT1											
		W	▨	▨	▨	▨												
$E	DMAP1	R	DISMAP1[15:0]															
		W																
$F	DMAP2	R	0	0	0	0	0	0	0	0	DISMAP2[23:16]							
		W	▨	▨	▨	▨	▨	▨	▨	▨								
$10	CNFG	R	0	DBG_EN	WAIT_EN	EDG	0	TOPNEG 45	TOPNEG 23	TOPNEG 01	0	BOTNEG 45	BOTNEG 23	BOTNEG 01	INDEP 45	NDEP 23	NDEP 01	WP
		W	▨				▨				▨							
$11	CCTRL	R	ENHA	nBX	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	0	0	VLMODE		0	SWP 45	SWP 23	SWP 01
		W									▨	▨			▨			
$12	PORT	R	0	0	0	0	0	0	0	0	0	0	0	0	PORT			
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$13	ICCTRL	R	0	0	0	0	0	0	0	0	0	0	0	0	0	ICC2	ICC1	ICC0
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨			
$14	SCTRL	R	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
		W	▨	▨														

R	0	Read as 0
W	▨	Reserved

Figure 10-34. PWM Register Map Summary

10.10.1 Control Register (CTRL)

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	LDFQ				HALF	IPOL2	IPOL1	IPOL0	PRSC		PWMRIE	PWMF	▨	▨	LDOK	PWMEN
Write													0	0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-35. Control (CTRL) Register

10.10.1.1 Load Frequency (LDFQ)—Bits 15–12

These buffered read/write bits select the PWM load frequency according to **Table 10-8**. Reset clears the LDFQ bit field, selecting loading every PWM opportunity. The occurrence of a PWM opportunity is determined by the half bit.

Note: The LDFQ*n* bit field takes effect when the current load cycle is complete, regardless of the state of the Load Okay (LDOK) bit. Reading the LDFQ*n* bit field reads the buffered values and not necessarily the values currently in effect.

Table 10-8. PWM Reload Frequency

LDFQ	PWM Reload Frequency	LDFQ	PWM Reload Frequency
0000	Every PWM Opportunity	1000	Every 9 PWM Opportunities
0001	Every 2 PWM Opportunities	1001	Every 10 PWM Opportunities
0010	Every 3 PWM Opportunities	1010	Every 11 PWM Opportunities
0011	Every 4 PWM Opportunities	1011	Every 12 PWM Opportunities
0100	Every 5 PWM Opportunities	1100	Every 13 PWM Opportunities
0101	Every 6 PWM Opportunities	1101	Every 14 PWM Opportunities
0110	Every 7 PWM Opportunities	1110	Every 15 PWM Opportunities
0111	Every 8 PWM Opportunities	1111	Every 16 PWM Opportunities

10.10.1.2 Half Cycle Reload (HALF)—Bit 11

This read/write bit enables half-cycle reloads in Center-Aligned PWM mode. This bit has no effect on Edge-Aligned PWMs.

- 0 = Half cycle reloads disabled
- 1 = Half cycle reloads enabled

10.10.1.3 Current Polarity 2 (IPOL2)—Bit 10

This buffered read/write bit selects the PWM value register for the PWM4 and PWM5 pins in complementary mode.

- 0 = VAL4 register in next PWM cycle
- 1 = VAL5 register in next PWM cycle

Note: The IPOL*n* bits take effect at the beginning of the next load cycle regardless of the state of the LDOK bit. Reading the IPOL*n* bits reads the buffered values and not necessarily the values currently in effect.

10.10.1.4 Current Polarity 1 (IPOL1)—Bit 9

This buffered read/write bit selects the VAL*n* register for the PWM2 and PWM3 pins in complementary mode.

- 0 = VAL2 register in next PWM cycle
- 1 = VAL3 register in next PWM cycle

10.10.1.5 Current Polarity 0 (IPOL0)—Bit 8

This buffered read/write bit selects the VAL*n* register for the PWM0 and PWM1 pins in complementary mode.

- 0 = VAL0 register in next PWM cycle
- 1 = VAL1 register in next PWM cycle

10.10.1.6 Prescaler (PRSC)—Bits 7–6

These buffered read/write bits select the PWM clock frequency illustrated in **Table 10-9**.

Table 10-9. PWM Prescaler

PRSC	PWM Clock Frequency
00	PWM Operation Clock Frequency
01	PWM Operation Clock Frequency /2
10	PWM Operation Clock Frequency /4
11	PWM Operation Clock Frequency /8

Note: Reading the PRSC*n* bits reads the buffered values and not necessarily the values currently in effect. The PRSC*n* bits take effect at the beginning of the next PWM cycle and only when the LDOK bit is set.

Note: PWM Operation Clock frequency is at either System Clock frequency or 3× System Clock frequency.

10.10.1.7 PWM Reload Interrupt Enable (PWMRIE)—Bit 5

This read/write bit enables the PWMF flag to generate interrupt request.

- 0 = PWMF interrupt disabled
- 1 = PWMF interrupt enabled

10.10.1.8 PWM Reload Flag (PWMF)—Bit 4

This read/write flag is set at the beginning of every reload cycle regardless of the state of the LDOK bit. Clear PWMF by reading it, and then write a Logic 0 to it. If another reload occurs before the clearing sequence is complete, writing Logic 0 to the PWMF bit. If another reload occurs before the clearing sequence is complete, writing Logic 0 to the PWMF has no effect.

- 0 = No new reload cycle since last PWMF clearing
- 1 = New reload cycle since last PWMF clearing

Note: Clearing PWMF clears pending PWMF interrupt requests.

10.10.1.9 Reserved—Bits 3–2

This bit field is reserved or not implemented. *It must be written as zero in all conditions.* Writing any number may result in unexpected PWM duty cycles.

10.10.1.10 Load Okay (LDOK)—Bit 1

This read/write bit loads the prescaler bits of CTRL and the entire CMOD and VAL*n* registers into a set of buffers. The buffered prescaler divisor, PWM counter modulus value, and PWM pulse width take effect at the next PWM reload. Set LDOK by reading it, then write a Logic 1 to it. LDOK is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a Logic 0 to it.

- 0 = No action is taken
- 1 = Load prescaler, PWM modulus and PWM values into buffers

10.10.1.11 PWM Enable (PWMEN)—Bit 0

This read/write bit enables the PWM generator. When PWMEN =0, the PWM outputs are in their inactive states unless OUTCTL*n* =1.

- 0 = PWM generator and PWM outputs disabled unless OUTCTL*n* =1
- 1 = PWM generator and PWM outputs enabled

Note: PWM pins outputs are in tri-state if Output Pad Enable (PAD_EN) bit in OUT register is cleared.

10.10.2 Fault Control Register (FCTRL)

Base+ $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	FIE3	FMODE3	FIE2	FMODE2	FIE1	FMODE1	FIE0	FMODE0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-36. Fault Control (FCTRL) Register

10.10.2.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.2.2 FAULT*n* Pin Interrupt Enable (FIE*n*)—Bits 7, 5, 3, 1

These read/write bits enable the interrupt request generated by the FAULT*n* pin.

- 0 = FAULT*n* interrupt disabled
- 1 = FAULT*n* interrupt enabled

Note: The fault protection circuit is independent of the FIE*n* bits and is always active. If a fault is detected, the PWM pins are disabled according to the PWM disable mapping register.

10.10.2.3 FAULT*n* Pin Clearing Mode (FMODE*n*)—Bits 6, 4, 2, 0

These read/write bits select automatic or manual clearing of FAULT*n* pin faults.

- 0 = Manual fault clearing of FAULT*n* pin faults
- 1 = Automatic fault clearing of FAULT*n* pin faults

10.10.3 Fault Status and Acknowledge Register (FLTACK)

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1	FPIN0	FFLAG0								
Write										FTACK3		FTACK2		FTACK1		FTACK0
Reset	U	0	U	0	U	0	U	0	0	0	0	0	0	0	0	0

Figure 10-37. Fault Status and Acknowledge (FLTACK) Register

10.10.3.1 FAULT*n* Pin (FPIN*n*)—Bits 15, 13, 11, 9

These *read-only* bits reflect the current state of the filtered FAULT*n* bit. A reset has no effect on FPIN*n*.

- 0 = Logic 0 on the FAULT*n* bit
- 1 = Logic 1 on the FAULT*n* bit

10.10.3.2 FAULT*n* Pin Flag (FFLAG*n*)—Bits 14, 12, 10, 8

These *read-only* flags are set within two PWM Operation Clock cycles after a rising edge on the FAULT*n* pin. Clear FFLAG*n* by writing a Logic 1 to the FTACK*n* bits in the FLTACK register. A reset clears FFLAG*n*.

- 0 = No fault on the FAULT*n* bit
- 1 = Fault on the FAULT*n* bit

10.10.3.3 Reserved—Bit 7, 5, 3, 1

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.3.4 FAULT*n* Pin Acknowledge (FTACK*n*)—Bits 6, 4, 2, 0

Writing a Logic 1 to FTACK*n* clears FFLAG*n*. Writing a Logic 0 has no effect. Read these bits as 0. Reset clears FTACK*n*. The fault protection is enabled even when the PWM is not enabled; therefore a fault is latched in, requiring it to be cleared to prevent an interrupt when the PWM is enabled.

10.10.4 Output Control Register (OUT)

This is a read/write register.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	PAD_EN	0	OUT CTL5	OUT CTL4	OUT CTL3	OUT CTL2	OUT CTL1	OUT CTL0	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-38. Output Control (OUT) Register

10.10.4.1 Output Pad Enable (PAD_EN)—Bit 15

The PWM*n* output pads can be enabled or disabled by setting the PAD_EN bit. The power-up default has the pads disabled. This bit does not affect the functionality of the PWM, so the PWM can be energized with the output pads disabled. This enable is to power-up with a safe default value for the PWM drivers.

- 0 = Output pads disabled
- 1 = Output pads enabled

10.10.4.2 Reserved—Bit 14

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.4.3 Output Control Enables (OUTCTL5–0)—Bits 13–8

These read/write bits enable software control of their corresponding PWM pin. When OUTCTL*n* is set, the OUT*n* bit activates and deactivates the PWM*n* output. A reset clears the OUTCTL bits. When operating the PWM in Complementary mode, these bits must be switched in pairs for proper operation. Please see **Section 10.5** for details.

- 0 = Software control disabled
- 1 = Software control enabled

10.10.4.4 Reserved—Bits 7–6

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.4.5 Output Control (OUT5–0)—Bits 5–0

When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins, outlined in **Table 10-10**.

Table 10-10. Software Output Control

Out*n* Bit	Complementary Channel Operation	Independent Channel Operation
OUT0	1—PWM0 is Active 0—PWM0 is Inactive	1—PWM0 is Active 0—PWM0 is Inactive
OUT1	1—PWM1 is Complement of PWM 0 0—PWM1 is Inactive	1—PWM1 is Active 0—PWM1 is Inactive
OUT2	1—PWM2 is Active 0—PWM2 is Inactive	1—PWM2 is Active 0—PWM2 is Inactive
OUT3	1—PWM3 is Complement of PWM 2 0—PWM3 is Inactive	1—PWM3 is Active 0—PWM3 is Inactive
OUT4	1—PWM4 is Active 0—PWM4 is Inactive	1—PWM4 is Active 0—PWM4 is Inactive
OUT5	1—PWM5 is Complement of PWM 4 0—PWM5 is Inactive	1—PWM5 is Active 0—PWM5 is Inactive

10.10.5 Counter Register (CNTR)

This *read-only* register displays the state of the 15-bit PWM counter.

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	CNT														
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-39. Counter (CNTR) Register

10.10.5.1 Reserved—Bit 15

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.5.2 Counter (CNT)—Bits 14–0

This *read-only* bit field displays the state of the 15-bit PWM counter.

10.10.6 Counter Modulo Register (CMOD)

Base + $5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	CM														
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-40. Counter Modulo (CMOD) Register

10.10.6.1 Reserved—Bit 15

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.6.2 Counter Modulo (CM)—Bits 14–0

This 15-bit unsigned value written to this buffered read/write register defines the PWM period in the PWM clock periods.

Note: The PWM counter modulo register is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading CMOD reads the value in a buffer. It is not necessarily the value the PWM generator is currently using.

10.10.7 Value Registers (VAL0–5)

The 16-bit signed value in these buffered, read/write registers is the PWM pulse width in PWM clock periods for each of the output channels.

Base +	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	VAL															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-41. Value (VAL0-5) Registers

10.10.7.1 Value (VAL)—Bits 15–0

The PWM Value registers are buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading VAL*n* register reads the value in a buffer and not necessarily the value the PWM generator is currently using.

A PWM value less than, or equal to zero, deactivates the PWM output for the entire PWM period. A PWM value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. Please see **Table 10-1**.

Note: The terms activate and deactivate refer to the high and low logic states of the PWM outputs.

10.10.8 Deadtime Register 0–1 (DTIM0–1)

The PWMDT0 field is used to insert the deadtime during 0 to 1 transitions of the even PWM output. The PWMDT1 field is used to insert the deadtime 0 to 1 transitions of the odd PWM output.

Base + $C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	PWMDT0											
Write																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Figure 10-42. Deadtime 0 (DTIM0) Register

Base + $D	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	PWMDT01											
Write																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Figure 10-43. Deadtime 1 (DTIM1) Register

10.10.8.1 Reserved—Bits 15–12

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.8.2 Deadtime (PWMDT0–1)—Bits 11–0

The 12-bit value written to this write-protectable register is the number of PWM clock cycles in complementary channel operation. A reset sets the Deadtime (DTIM) register to a default value of 0×0FFF, selecting a deadtime of 4096-PWM clock cycles minus one PWM Operation Clock cycle. This register is write protected after the Write Protect (WP) bit in the PWM configuration register is set. Please refer to **Section 10.10.10.**

Note: Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT=P × PWMDT - 1 PWM Clock where DT is deadtime, P is the prescaler value, PWMDT is the programmed value of deadtime. For example: if the prescaler is programmed for a divide-by-two and PWMDT is set to five, then P=2 and the deadtime value is equal to:

$$DT = 2 \times 5 - 1 = 9 \text{ PWM Clock cycles}$$

A special case exists when the P=1, DT=PWMDT.

10.10.9 Disable Mapping Registers 1–2 (DMAP1–2)

These *write-protectable* registers determine which PWM pins are disabled by the fault protection inputs, provided in **Table 10-4**. Reset sets all of the bits used in the DMAP1-2 registers. These registers are write-protected after the WP bit in the CNFG register is set. Reserved bits 15-8 in the DMAP2 register cannot be modified. The bits are read as 0.

Base + $E	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DISMAP1[15:0]															
Write																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 10-44. Disable Mapping 1 (DMAP1) Register

Base + $F	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	DISMAP2 [23:16]							
Write																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Figure 10-45. Disable Mapping 2 (DMAP2) Register

10.10.10 Configure Register (CNFG)

This *write-protectable* register contains the configuration bits determining PWM modes of operation detailed below. This register cannot be modified after the WP bit is set.

Base + $10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	DBG_EN	WAIT_EN	EDG	0	TOPNEG 45	TOPNEG 23	TOPNEG 01	0	BOTNEG 45	BOTNEG 23	BOTNEG 01	INDEP 45	INDEP 23	INDEP 01	WP
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-46. Configure (CNFG) Register

10.10.10.1 Reserved—Bit 15

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.10.2 Debug Enable (DBG_EN)—Bit 14

When this *write-protectable* bit is set to 1, the PWM will continue to run while the chip is in EOnCE Debug mode. If the device enters the EOnCE mode and this bit is 0, the PWM outputs will be switched to their inactive state until the EOnCE mode is exited. At that point the PWM pins will resume operation as programmed in the PWM registers.

Note: PWM parameter updates will not occur in the EOncE Debug mode.

10.10.10.3 Wait Enable (WAIT_EN)—Bit 13

When this *write-protectable* bit is set to 1, the PWM will continue to run while the chip is in the Wait mode. In this mode, the peripheral clock continues to run; however, the core clock does not. If the device enters the Wait mode and this bit is 0, the PWM outputs will be switched to their inactive state until the Wait mode is exited. At the point of exit the PWM pins will resume operation as programmed in the PWM registers.

Note: PWM parameter updates will not occur in Wait mode.

10.10.10.4 Edge-Aligned or Center-Aligned PWMs (EDG)—Bit 12

This *write-protectable* bit determines whether all PWM channels will use Edge-Aligned or Center-Aligned wave forms.

- 0 = Center-Aligned PWMs
- 1 = Edge-Aligned PWMs

10.10.10.5 Reserved—Bit 11

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.10.6 Top-Side PWM Polarity (TOPNEG)—Bits 10–8

This *write-protectable* bit determines the polarity for the top-side PWMs.

- 0 = Positive top-side polarity
- 1 = Negative top-side polarity

10.10.10.7 Bottom-Side PWM Polarity (BOTNEG)—Bits 6–4

This *write-protectable* bit determines the polarity for the bottom-side PWMs.

- 0 = Positive bottom-side polarity
- 1 = Negative bottom-side polarity

10.10.10.8 Independent or Complement Pair Operation (INDEP)—Bits 3–1

This *write-protectable* bit determines if the PWM channels will be independent PWMs or complementary PWM pairs.

- 0 = Complementary PWM pair
- 1 = Independent PWMs

Note: Each pair of PWM channels can be configured: Channel 0-1, Channel 2-3, and Channel 4-5.

10.10.10.9 Write Protect (WP)—Bit 0

This bit enables write-protection for all write-protectable registers. While clear, *WP allows write-protected registers and bits to be written*. When set, WP prevents writes to write-protectable registers and bits. Once set, WP can be cleared only by a reset. *Write-protectable* registers and bits include: DMAP1–2, DTIM, CNFG, and the ENHA bit in the CCTRL register. The VLMODE, SWP45, SWP23, and SWP01 bits in the CCTRL are protected when the Enable Hardware Acceleration (ENHA) bit is set to zero in the CCTRL. ENHA is in turn, protected by setting the WP bit in the CNFG.

- 0 = Write-protectable registers may be written
- 1 = Write-protectable registers are write protected

Note: The write to CNFG setting the WP bit is the last write accepted to the register until reset.

10.10.11 Channel Control Register (CCTRL)

Base + $11	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ENHA	nBX	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	0	0	VLMODE		0	SWP45	SWP23	SWP01
Write									▨	▨			▨			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-47. Channel Control (CCTRL) Register

This *write-protectable* register contains the configuration bits determining PWM modes of operation detailed below. The ENHA bit cannot be modified after the WP bit in the CNFG register is set. In turn, ENHA provides protection for the nBX, VLMODE, SWP45, SWP23 and SWP01 bits. Mask bits 0-5 are not write-protectable.

10.10.11.1 Enable Hardware Acceleration (ENHA)—Bit 15

This bit enables writing to the nBX, VLMODE, SWP45, SWP23, and SWP01 bits. The bit is *write-protectable* by the WP bit in the CNFG register.

- 0 = Disable writing to nBX, VLMODE, SWP45, SWP23, and SWP01 bits
- 1 = Enable writing to nBX, VLMODE, SWP45, SWP23, and SWP12 bits

10.10.11.2 56F80x Compatibility (nBX)—Bit 14

This bit is used to enable/disable improved SWAP and MASK operations. If it is cleared, SWAP/MASK operates identical to the 56F80x version of this module. See 56F80x User's Manual for details. If is set, SWAP and MASK operations are described in this manual which are not compatible features with 56F80 and are not supported.

- 0 = SWAP and MASK provide 56F80x compatible operation
- 1 = SWAP*n* and MASK*n* provide new functionality described in this manual *(Recommended)*

This bit is write-protected when ENHA is zero.

Note: Unless expecting SWAP/MASK operate identical to 56F80x version of this module, nBX=1 mode is highly recommended.

10.10.11.3 Mask (MSK5–0)—Bits 13–8

These six bits determine the mask for each of the PWM logical channels.

- 0 = Unmasked
- 1 = Masked, channel deactivated

10.10.11.4 Reserved—Bits 7–6

These bits are reserved or not implemented. They are read as 0 and cannot be modified by writing.

10.10.11.5 Value Register Load Mode (VLMODE)—Bits 5–4

These two bits determine the way the Value registers are being loaded. These bits are write-protected when ENHA is zero.

- 00 = Each VAL*n* register is accessed independently
- 01 = Writing to VAL0 register also writes to registers VAL1-5
- 10 = Writing to VAL0 register also writes to registers VAL1-3
- 11 = Reserved

10.10.11.6 Reserved—Bit 3

This bit is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.11.7 Swap45 (SWP45)—Bit 2

This bit is write-protected when ENHA is zero.

- 0 = No swap
- 1 = Channels four and five are swapped

10.10.11.8 Swap23 (SWP23)—Bit 1

This bit is write-protected when ENHA is zero.

- 0 = No swap
- 1 = Channels two and three are swapped

10.10.11.9 Swap01 (SWP01)—Bit 0

This bit is write-protected when ENHA is zero.

- 0 = No swap
- 1 = Channels zero and one are swapped

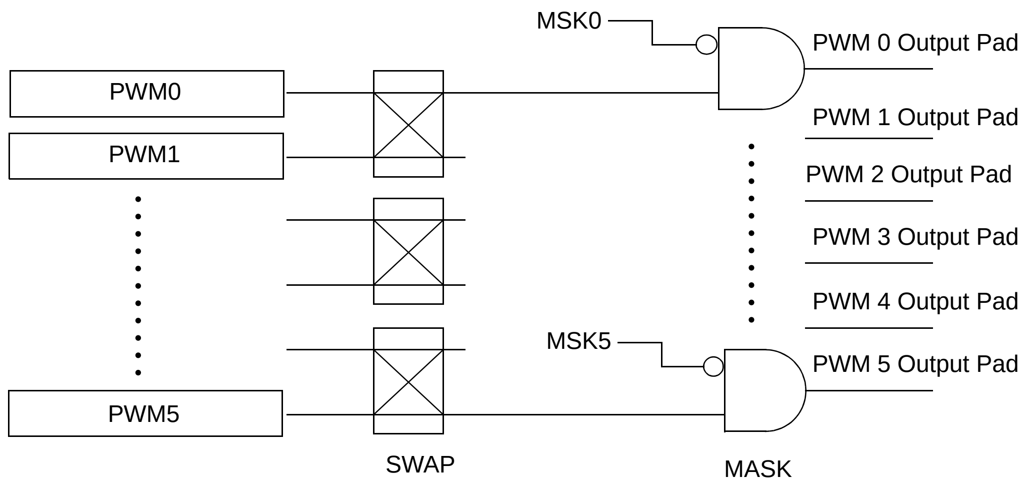


Figure 10-48. Channel Swapping

10.10.12 Port Register (PORT)

This register contains values of the four fault inputs, bits 3, 2, 1, and 0. This register may be read while the PWM is active.

Base + $12	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	PORT			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	U	U	U	U

Figure 10-49. Port (PORT) Register

10.10.12.1 Reserved—Bits 15–4

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.12.2 Port (PORT)—Bits 3–0

These are *read-only* bits; therefore, any writes to them are not affected.

10.10.13 Internal Correction Control Register (ICCTRL)

These bits only apply in Center-Aligned operation during Complementary mode. These control bits determine whether values set in the IPOL*n* bits control which Val*n* register is used, or PWM count direction controls which PWM value register is used.

Base + $13	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	0	ICC2	ICC1	ICC0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-50. Internal Correction Control (ICCTRL) Register

10.10.13.1 Reserved—Bits 15–3

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.13.2 Internal Current Control 2 (ICC2)—Bit 2

This bit controls PWM4/PWM5 pair.

- 0 = Either VAL4 or VAL5 registers use is determined by IPOL2 setting.
- 1 = Use VAL4 register when the PWM counter is counting up. Use VAL5 register when counting down.

10.10.13.3 Internal Current Control 1 (ICC1)—Bit 1

This bit controls PWM2/PWM3 pair.

- 0 = Either VAL2 or VAL3 registers use is determined by IPOL1 setting.
- 1 = Use VAL2 register when the PWM counter is counting up. Use VAL3 register when counting down

10.10.13.4 Internal Current Control 0 (ICC0)—Bit 0

This bit controls PWM0/PWM1 pair.

- 0 = Either VAL0 or VAL1 registers use is determined by the IPOL0 setting.
- 1 = Use VAL0 register when the PWM counter is counting up. Use VAL1 register when counting down

10.10.14 Source Control Register (SCTRL)

This register contains control bits to determine the source signals for the complementary PWM outputs. This register is affected by the WP bit in the CNFG. It can only receive writing when the WP bit is clear.

Base + $14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10-51. Source Control (SCTRL) Register

10.10.14.1 Reserved—Bits 15–14

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

10.10.14.2 PWM Compare Output Polarity Invert 5 (CINV5)—Bit 13

This bit controls the polarity of PWM compare output 5. Please see the output operations in **Figure 10-2** and **Figure 10-3**.

- 0 = The PWM compare output is driven to high state when the value of VAL5 register is greater than the value of PWM counter.
- 1 = The PWM compare output is driven to low state when the value of VAL5 register is less than the value of PWM counter.

10.10.14.3 PWM Compare Output Polarity Invert 4 (CINV4)—Bit 12

This bit controls the polarity of PWM compare output 4. Please see the output operations in **Figure 10-2** and **Figure 10-3**.

- 0 = The PWM compare output is driven to high state when the value of VAL4 register is greater than the value of PWM counter.
- 1 = The PWM compare output is driven to low state when the value of VAL4 register is less than the value of PWM counter.

10.10.14.4 PWM Compare Output Polarity Invert 3 (CINV3)—11

This bit controls the polarity of PWM compare output 3. Please see the output operations in **Figure 10-2** and **Figure 10-3**.

- 0 = The PWM compare output is driven to high state when the value of VAL3 register is greater than the value of PWM counter.
- 1 = The PWM compare output is driven to low state when the value of VAL3 register is less than the value of PWM counter.

10.10.14.5 PWM Compare Output Polarity Invert 2 (CINV2)—Bit 10

This bit controls the polarity of PWM compare output 2. Please see the output operations in **Figure 10-2** and **Figure 10-3**.

- 0 = The PWM compare output is driven to high state when the value of VAL2 register is greater than the value of PWM counter.
- 1 = The PWM compare output is driven to low state when the value of VAL2 register is less than the value of PWM counter.

10.10.14.6 PWM Compare Output Polarity Invert 1 (CINV1)—Bit 9

This bit controls the polarity of PWM compare output 1. Please see the output operations in **Figure 10-2** and **Figure 10-3**.

- 0 = The PWM compare output is driven to high state when the value of VAL1 register is greater than the value of PWM counter.
- 1 = The PWM compare output is driven to low state when the value of VAL1 register is less than the value of PWM counter.

10.10.14.7 PWM Compare Output Polarity Invert 0 (CINV0)—Bit 8

This bit controls the polarity of PWM compare output 0. Please see the output operations in **Figure 10-2** and **Figure 10-3**.

- 0 = The PWM compare output is driven to high state when the value of VAL0 register is greater than the value of PWM counter.
- 1 = The PWM compare output is driven to low state when the value of VAL0 register is less than the value of PWM counter.

10.10.14.8 PWM 2 Sources (SRC2)—Bits 7–5

This bit field controls PWM5/PWM4 pair. Make sure OUTCTL4 and OUTCTL5 (bits 12 and 13 of the OUT register) are set when using these bits.

- 000 = Use PWM generator as PWM source (operation is consistent with 80x and 83xx devices).
- 001 = Use ADC SAMPLE2 result as PWM source.
  - — If the ADC conversion result in SAMPLE2 is greater than the value programmed into the High Limit register HLMT2, then PWM4 is set to 0 and PWM5 is set to 1
  - — If the ADC conversion result in SAMPLE2 is less than the value programmed into the Low Limit register LLMT2, the PWM4 is set to 1 and the PWM5 is set to 0
- 010 = Use a GPIO input as PWM source. The specific GPIO pin to be used is specified in the device data sheet.
- 011 = Use the output of TMR module as PWM source. The specific timer channel to be used is specified in the device data sheet.
- 1xx = Use the value selected in SRC0 as the PWM source.

10.10.14.9 PWM 1 Source (SRC1)—Bits 4–2

This bit field controls PWM2/PWM3 pair. Make sure OUTCTL2 and OUTCTL3 (bits 10 and 11 of the OUT register) are set when using these bits.

- 000 = Use PWM generator as PWM source (operation is consistent with 80x and 83xx devices).
- 001 = Use ADC SAMPLE1 result as PWM source.
  - — If the ADC conversion result in SAMPLE1 is greater than the value programmed into the High Limit register HILIM1, then PWM2 is set to 0 and PWM3 is set to 1
  - — If the ADC conversion result in SAMPLE1 is less than the value programmed into the Low Limit register LOLIM1, the PWM2 is set to 1 and the PWM3 is set to 0
- 010 = Use a GPIO input as PWM source. The specific GPIO pin to be used is specified in the device data sheet.

- 011 = Use the output of TMR module as PWM source. The specific timer channel to be used is specified in the device data sheet.
- 1xx = Use the value selected in SRC0 as the PWM source.

10.10.14.10 PWM 0 Source (SRC0)—Bits 1–0

This bit field controls PWM0/PWM1 pair. Make sure OUTCTL0 and OUTCTL1 (bits 8 and 9 of the OUT register) are set when using these bits.

- 00 = Use PWM generator as PWM source (operation is consistent with 80x and 83xx devices).
- 01 = Use ADC SAMPLE0 result as PWM source.
  - If the ADC conversion result in SAMPLE0 is greater than the value programmed into the High Limit register HILIM0, then PWM0 is set to 0 and PWM1 is set to 1
  - If the ADC conversion result in SAMPLE0 is less than the value programmed into the Low Limit register LOLIM0, the PWM0 is set to 1 and the PWM1 is set to 0
- 10 = Use a GPIO input as PWM source. The specific GPIO pin to be used is specified in the device data sheet.
- 11 = Use the output of TMR module as PWM source. The specific timer channel to be used is specified in the device data sheet.

10.11 Clocks

The PWM Operation Clock is the only clock required by this module. Along with the PWM prescaler, it determines the amount of time allocated for a single PWM bit value.

10.12 Interrupts

Five PWM sources can generate two interrupt requests to the core:

- Reload Flag (PWMF)—PWMF is set at the beginning of every reload cycle. The Reload Interrupt Enable (PWMRIE) bit enables PWMF to generate core interrupt requests. PWMF and PWMRIE are in PWM Control (CTRL) register.
- Fault Flags (FFLAG0–FFLAG3)—The FFLAG*n* bit is set when a Logic 1 occurs on the FAULT*n* pin. The fault pin interrupt enable bits, FIE0–FIE3, enable the FFLAG*n* flags to generate core interrupt requests. FFLAG0–FFLAG3 are in the Fault Status Acknowledge (FLTACK) register. FIE0–FIE3 are in the Fault Control (FCTL) register.

10.13 Resets

All PWM registers are reset to their default values upon any system reset.

Chapter 11
Serial Communications Interface (SCI)

Document Revision History for Chapter 11, Serial Communications Interface (SCI)

Version History	Description of Change
Rev. 0	Initial release

11.1 Introduction

This chapter describes the Serial Communications Interface (SCI) module. The module allows asynchronous serial communications with peripheral devices and other controllers.

11.2 Features

- Full-duplex or Single-Wire Operation
- Standard mark/space Non-Return-to-Zero (NRZ) format
- 13-bit baud rate selection
- Programmable 8- or 9-bit data format
- Separately enabled transmitter and receiver
- Separate receiver and transmitter CPU interrupt requests
- Programmable polarity for transmitter and receiver
- Two receiver wake-up methods:
  - Idle line
  - Address mark
- Interrupt-driven operation with seven flags:
  - Transmitter empty
  - Transmitter idle
  - Receiver full
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error
  - LIN sync error
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection

11.3 Block Diagram

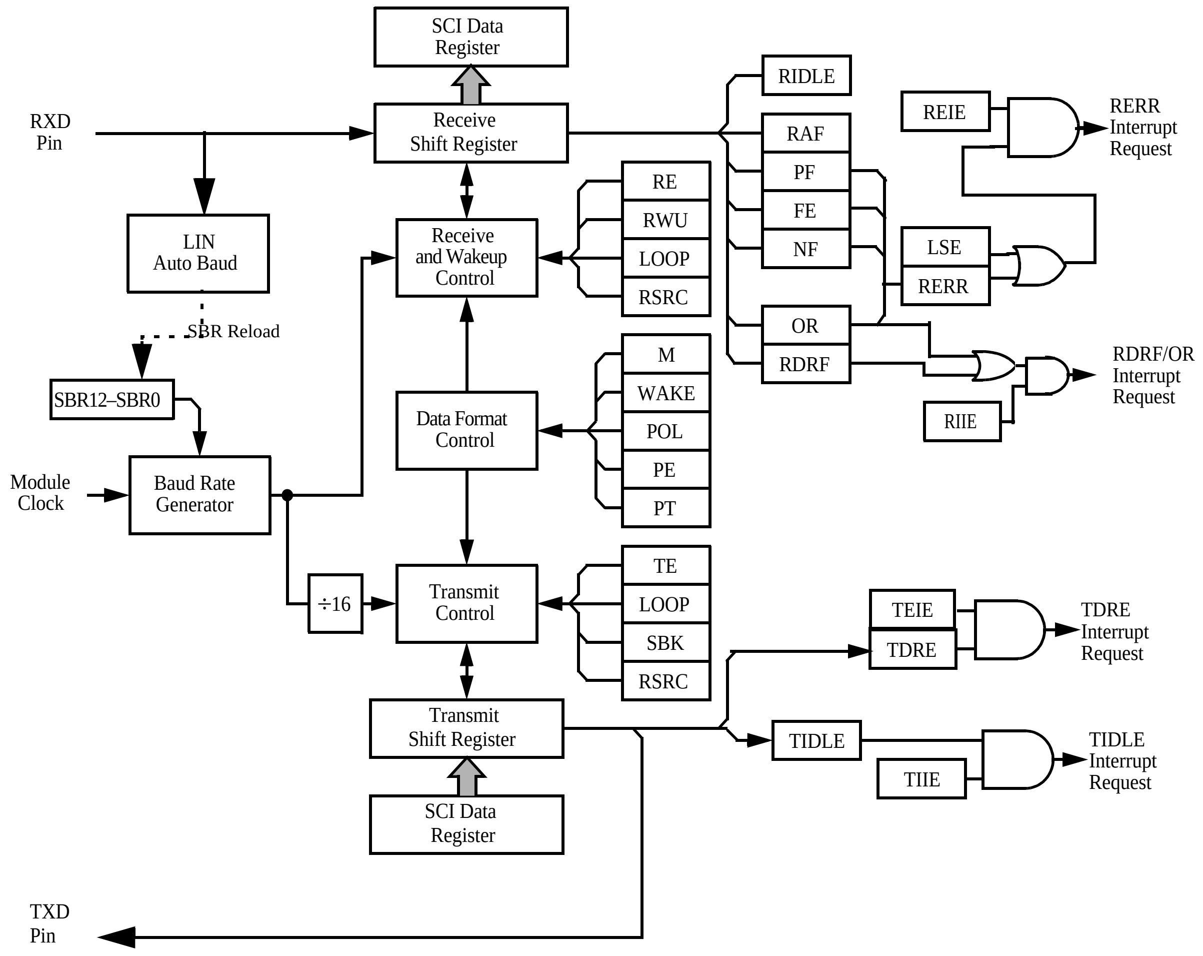


Figure 11-1. SCI Block Diagram

11.4 Functional Description

Figure 11-1 explains the SCI module structure. The SCI allows full duplex, asynchronous, Non-Return-to-Zero (NRZ) serial communication between the controller and remote devices, including other controllers. The SCI transmitter and receiver operate independently although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

When initializing the SCI, be certain to set the proper peripheral enable bits in the General Purpose Input/Output (GPIO) registers as well as any pull-up enables.

To initialize the SCI for full duplex operation:

- Initialize software flags.
- Set each bit of the SCI CTRL1 register as 0:

  LOOP=0, SWAI=0, RSRC=0, M=0, WAKE=0, POL=0, PE=0, PT=0, TEIE=0 TITE=0, RFIE=0, REIE=0, TE=0, RE=0, RWU=0, and SBK=0

- If the LIN Slave mode is desired, enable that bit in the SCI CTRL2 register.
- Set the prescaler bits.
- Enable transmitter and receiver.

11.4.1 Data Frame Format

The SCI uses the standard NRZ mark/space data frame format illustrated in Figure 11-2.

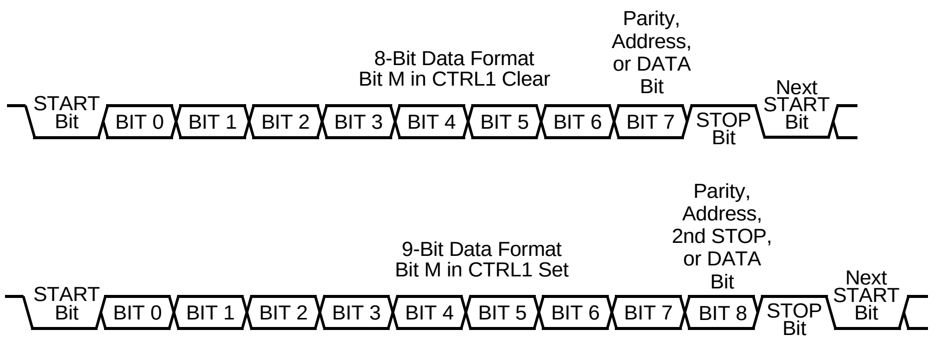


Figure 11-2. SCI Data Frame Formats

Each data character is contained in a frame including a START bit, eight or nine data bits, and a STOP bit. Clearing the MODE (M) bit in the CTRL1 register configures the SCI for 8-bit data characters. A frame with eight data bits has a total of 10 bits. Formats are provided in Table 11-1.

Table 11-1. Example 8-Bit Data Frame Formats

Start Bit	Data Bits	Address Bit	Parity Bit	Stop Bit
1	8	0	0	1
1	7	0	1	1
1	7	1[1]	0	1

1. The address bit identifies the frame as an address character. Please see Section 11.4.4.6, Receiver Wake-Up

Setting the M bit configures the SCI for 9-bit data characters. A frame with nine data bits has a total of 11 bits. Formats are provided in **Table 11-2**.

Table 11-2. Example 9-Bit Data Frame Formats

Start Bit	Data Bits	Address Bit	Parity Bit	Stop Bit
1	9	0	0	1
1	8	0	0	2
1	8	0	1	1
1	8	1[1]	0	1

1. The address bit identifies the frame as an address character. Please see Section 11.4.4.6, Receiver Wake-Up

11.4.2 Baud Rate Generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value of 0 to 8191 written to the SBR bits determines the module clock divisor. The SBR bits are in the SCI Baud Rate (RATE) register. The baud rate clock is synchronized with the IPbus clock and drives the receiver. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to two sources of error:

1. The Integer division of the module clock may not give the exact target frequency.
2. Synchronization with the bus clock can cause phase shift.

Table 11-3 lists examples of achieving target baud rates with a module clock frequency of 32MHz.

Table 11-3. Example Baud Rates (Module Clock = 32MHz)

SBR Bits	Receiver Clock (Hz)	Transmitter Clock (Hz)	Target Baud Rate	Error (%)
17	1,882,352	117,647.0	115,200	2.12
35	914,286	57,142.8	57,600	-0.79
52	615,385	38,461.5	38,400	0.16
104	307,692	19,230.8	19,200	0.16
208	153,846	9,615.4	9,600	0.16
417	76,739	4,796.2	4,800	-0.08
833	38,415	2,401.0	2,400	0.04
1667	19,196	1,199.8	1,200	0.02
3333	9,601	600.1	600	0.01

Note: Maximum baud rate is module clock rate divided by 16. System overhead may preclude processing the data at this speed.

11.4.3 Transmitter

Figure 11-3 illustrates the SCI transmitter block diagram. Detailed discussion of the transmitter is in the following sections.

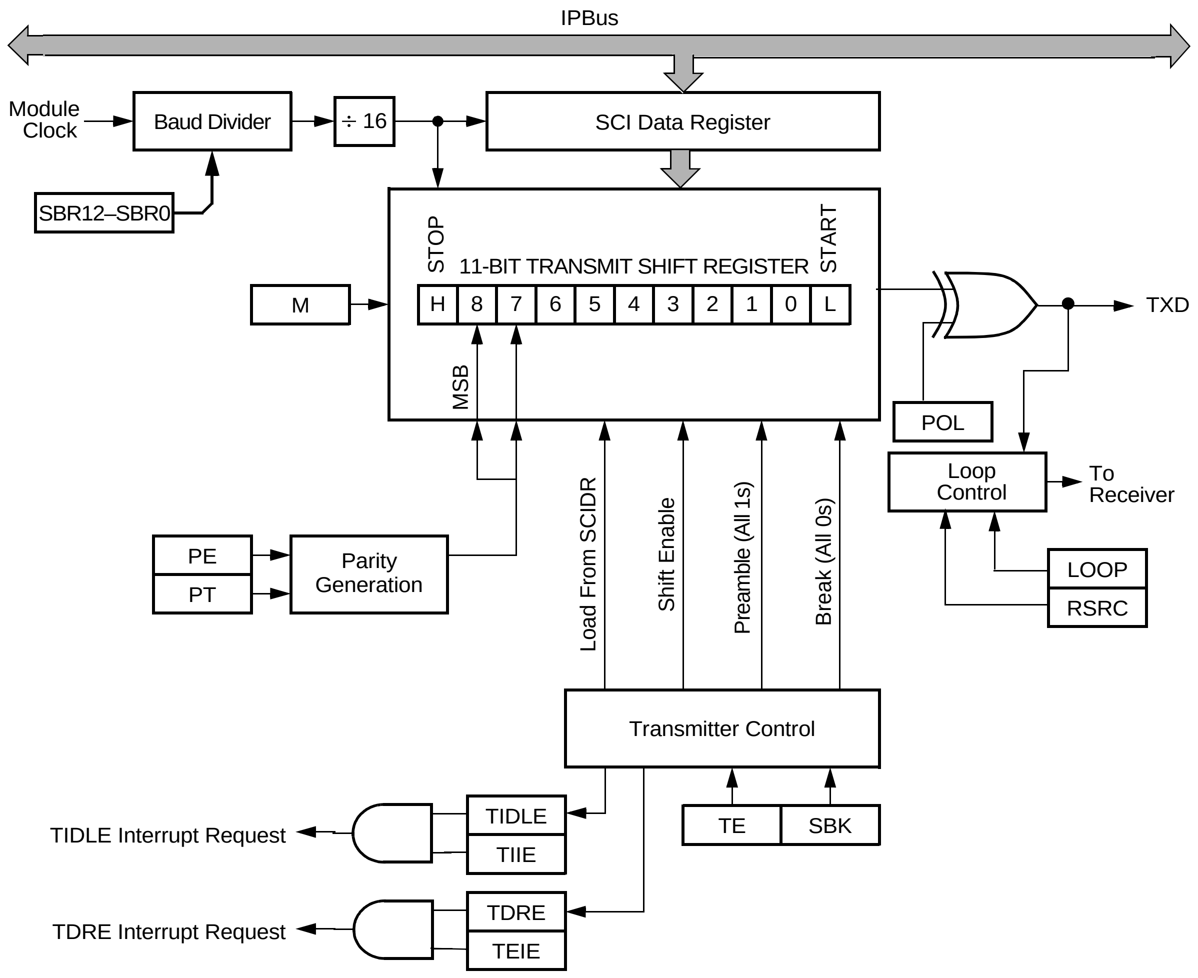


Figure 11-3. SCI Transmitter Block Diagram

11.4.3.1 Character Length

The SCI transmitter can accommodate either 8- or 9-bit data characters. The state of the M bit in the SCI Control 1 (CTRL1) register determines the length of data characters.

11.4.3.2 Character Transmission

During an SCI transmission, the Transmit Shift register moves a frame out to the TXD pin. The Data (DATA) register is the *write-only* buffer between the internal data bus and the Transmit Shift register.

To initiate a SCI transmission:

1. Enable the transmitter by writing a Logic 1 to the Transmitter Enable (TE) bit in the SCI Control 1 (CTRL1) register.
2. Wait for the Transmit Data Register Empty (TDRE) flag to be set.
3. Clear the TDRE flag, by first reading the SCI Status (STAT) register, and then writing to the SCI Data (DATA) register.
4. Repeat Step 2 and 3 for each subsequent transmission.

Writing the TE bit from 0 to 1 automatically loads the Transmit Shift register with a preamble of 10 Logic 1s (if M=0) or 11 Logic 1s (if M=1). After the preamble shifts out, control logic automatically transfers the data from the SCI DATA register into the Transmit Shift register. A Logic 0 START bit automatically goes into the Least Significant Bit (LSB) position of the Transmit Shift register. A Logic 1 STOP bit goes into the Most Significant Bit (MSB) position of the frame.

Hardware supports odd or even parity. When parity is enabled, the MSB of the data character is replaced by the parity bit.

The Transmit Data Register Empty (TDRE) flag in the SCI STAT register becomes set when the DATA register transfers a character to the Transmit Shift register. The TDRE flag indicates the DATA register can accept new transmit data. If the TEIE bit in the CTRL1 register is also set, the TDRE flag generates a transmitter empty interrupt request.

When the SCI Transmit Shift register is not transmitting a frame and TE=1, the TXD pin goes to the idle condition, Logic 1. If software clears the TE bit in the CTRL1 register, the transmitter relinquishes control of the port I/O pin upon completion of the current transmission. This action causes the TXD pin to go into a HighZ state.

If software clears TE while a transmission is in progress (TIDLE=0), the frame in the Transmit Shift register continues to shift out. Then transmission stops even if there is data pending in the SCI DATA register. To avoid accidentally cutting off the last frame in a message, always wait for TDRE to go high after the last frame before clearing TE.

To separate messages with preambles having minimum idle line time, use this sequence between messages:

1. Write the last character of the first message to the DATA register.
2. Wait for the TDRE flag to go high, indicating the transfer of the last frame to the Transmit Shift register.
3. Queue a preamble by clearing, then set the TE bit.
4. Write the first character of the second message to DATA register.

11.4.3.3 Break Characters

Writing Logic 1 to the Send Break (SBK) bit in the CTRL1 register loads the Transmit Shift register with a break character. A break character contains all Logic 0s without START, STOP, or PARITY bits. Break character length depends on the MODE (M) bit in the CTRL1 register. As long as SBK is at Logic 1, transmitter logic continuously loads break characters into the Transmit Shift register. After software clears the SBK bit, the Transmit Shift register finishes transmitting the last break character subsequently transmitting at least one Logic 1. The automatic Logic 1 at the end of the last break character guarantees the recognition of the START bit of the next frame.

The SCI recognizes a break character when a start bit is followed by eight or nine Logic 0 data bits and a Logic 0 where the STOP bit should be. Receiving a break character has these effects on SCI registers:

- Sets the Framing Error (FE) flag
- Sets the Receive Data Register Full (RDRF) flag
- Clears the SCI Data (DATA) register
- May set the Overrun (OR) flag, Noise Flag (NF), Parity Error (PE) flag, or the Receiver Active Flag (RAF). Please see SCISR in **Section 11.6.4.**

11.4.3.4 Preambles

A preamble contains all Logic 1s with no START, STOP, or PARITY bit. Preamble length depends on the M bit in the CTRL1 register. The preamble is a synchronizing mechanism initiating the first transmission begun after modifying the TE bit from zero to one.

If the TE bit is cleared during a transmission, the TXD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues a preamble to be sent after the frame currently being transmitted.

Note: Toggle the TE bit for a queued preamble when the TDRE flag becomes set and immediately before writing the next character to the DATA register.

When queueing a preamble, return the TE bit to Logic 1 before the STOP bit of the current frame shifts out to the TXD pin. Setting TE after the STOP bit appears on TXD causes data previously written to the DATA register to be lost.

11.4.4 Receiver

Figure 11-4 illustrates the SCI receiver block diagram. Detailed discussion of the receiver function is in the following paragraphs.

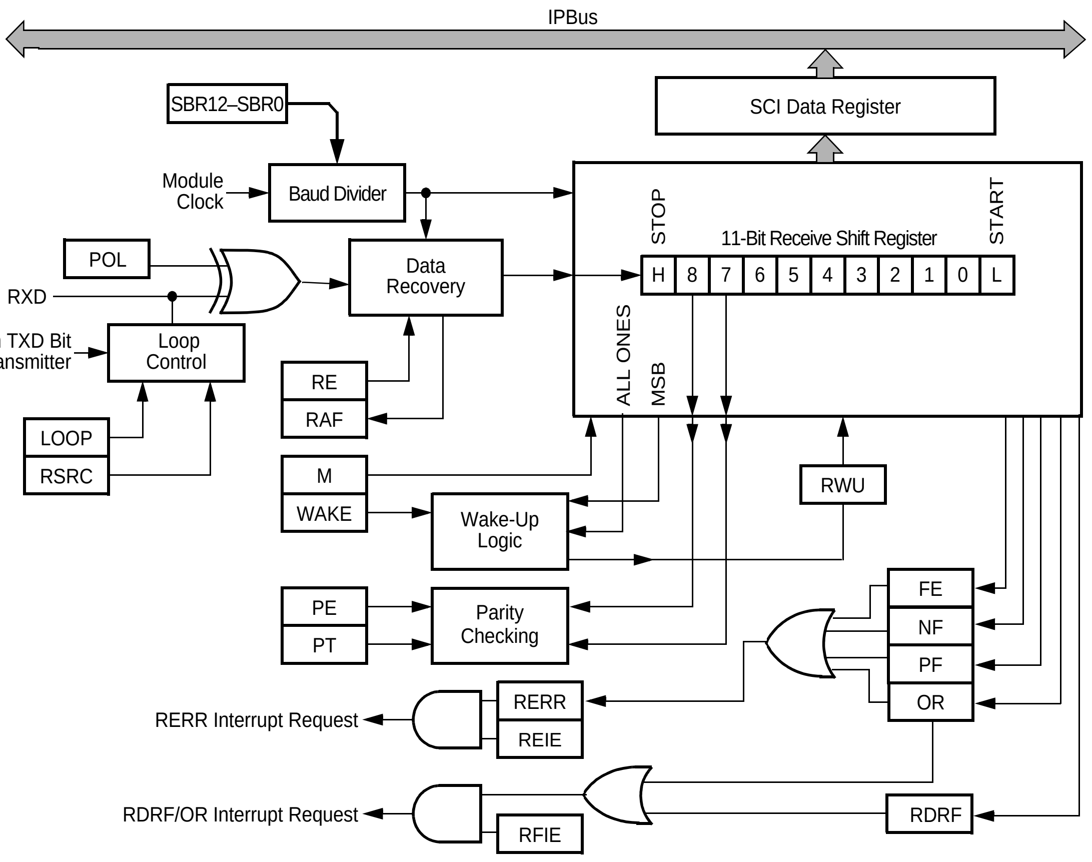


Figure 11-4. SCI Receiver Block Diagram

11.4.4.1 Character Length

The SCI transmitter can accommodate either 8- or 9-bit data characters. The state of the M bit in the CTRL1 register determines the length of data characters.

11.4.4.2 Character Reception

During an SCI reception, the Receive Shift register moves a frame in from the RXD pin. The DATA Register is the read-only buffer between the internal data bus and the Receive Shift register.

After complete frame shifts into the Receive Shift register, the data portion of the frame transfers to the DATA register. The Receive Data Register Full (RDRF) flag in the STAT register is set, indicating the received character can be read. If the Receiver Full Interrupt Enable (RFIE) bit in the CTRL1 register is also set, the RDRF flag generates an RDRF interrupt request.

11.4.4.3 Data Sampling

The receiver samples the RXD pin at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust the baud rate mismatch, the RT clock, illustrated in **Figure 11-5**, is resynchronized:

- After every START bit
- After the receiver detects a data bit change from Logic 1 to Logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid Logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid Logic 0)

To locate the START bit, data recovery logic does an asynchronous search for a Logic 0 preceded by three Logic 1s. When the falling edge of a possible START bit occurs, the RT clock begins to count to 16.

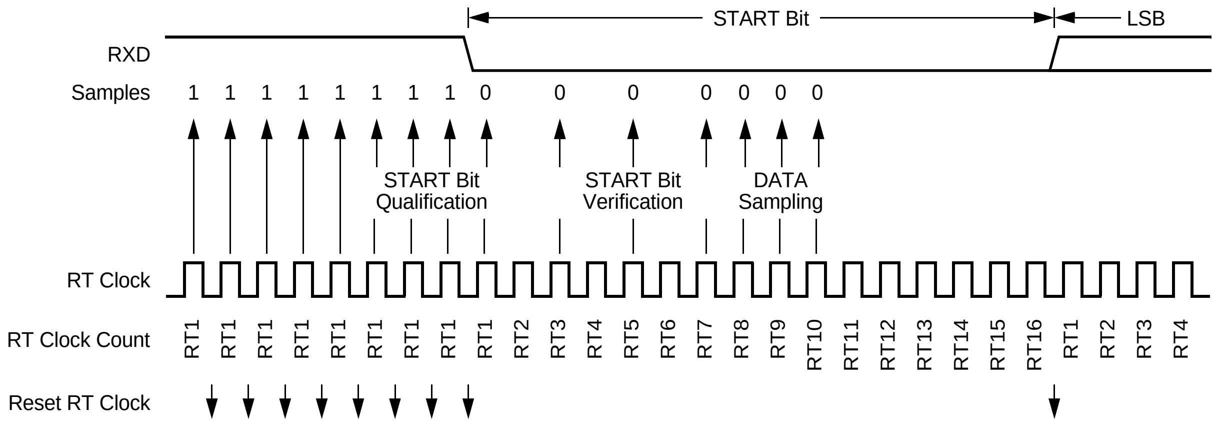


Figure 11-5. Receiver Data Sampling

To verify the START bit and detect noise, data recovery logic takes samples at RT3, RT5, and RT7. **Table 11-4** summarizes the results of the START bit verification samples.

Table 11-4. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If START bit verification is not successful, the RT clock is reset and a new search for a START bit begins.

To determine the value of a data bit and to detect noise, data recovery logic takes samples at RT8, RT9, and RT10. **Table 11-5** summarizes the results of DATA bit samples.

Table 11-5. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

Note: The RT8, RT9, and RT10 samples do not affect START bit verification. If any or all of the RT8, RT9, and RT10 START bit samples are Logic 1s following a successful START bit verification, the Noise Flag (NF) is set and the receiver assumes the bit is a START bit (Logic 0).

To verify a STOP bit and to detect noise, data recovery logic takes samples at RT8, RT9, and RT10. **Table 11-6** summarizes the results of the STOP bit samples.

Table 11-6. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In **Figure 11-6** the verification samples RT3 and RT5 determine the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the START bit search begins again. The noise flag is not set because the noise occurred before the START bit was found.

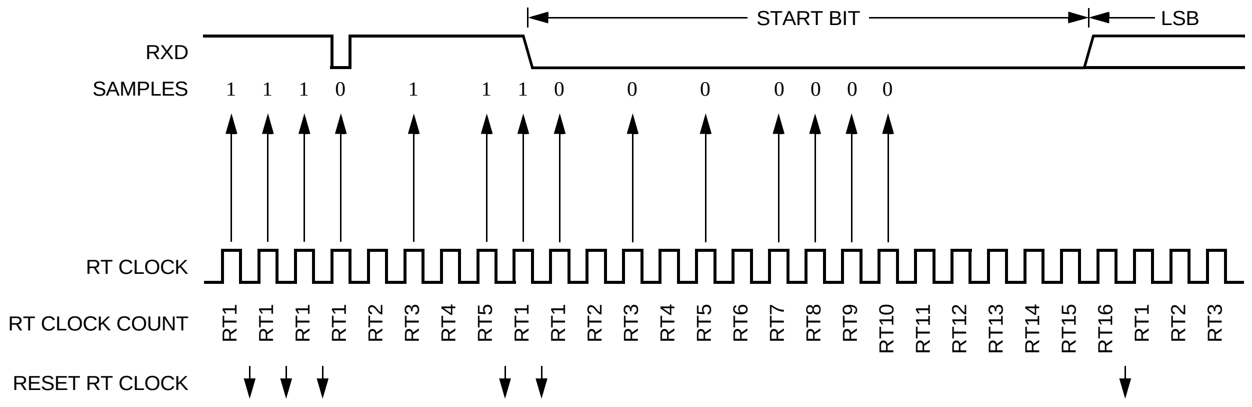

START BIT
LSB
RXD
SAMPLES
1 1 1 0 1 1 1 0 0 0 0 0 0 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3
RESET RT CLOCK

Figure 11-6. Start Bit Search Example 1

In **Figure 11-7** noise is perceived as the beginning of a start bit because the verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

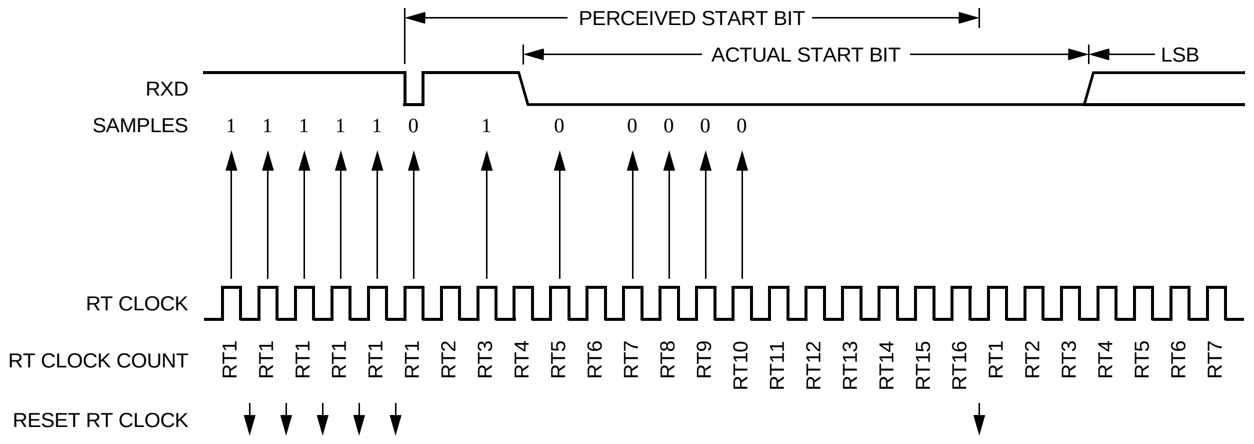

PERCEIVED START BIT
ACTUAL START BIT
LSB
RXD
SAMPLES
1 1 1 1 1 0 1 0 0 0 0 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3 RT4 RT5 RT6 RT7
RESET RT CLOCK

Figure 11-7. Start Bit Search Example 2

In **Figure 11-8** a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

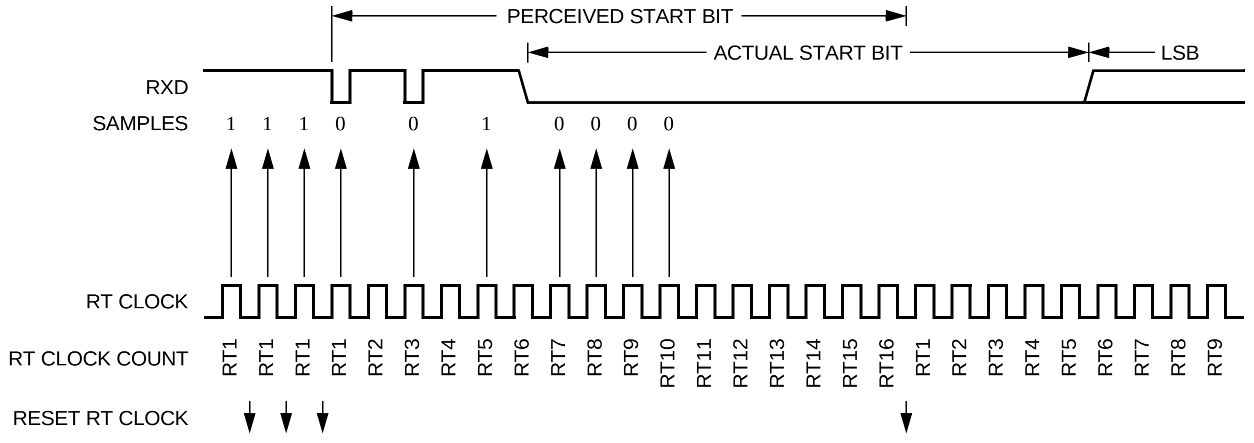

PERCEIVED START BIT
ACTUAL START BIT
LSB
RXD
SAMPLES
1 1 1 0 0 1 0 0 0 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9
RESET RT CLOCK

Figure 11-8. Start Bit Search Example 3

Figure 11-9 illustrates the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

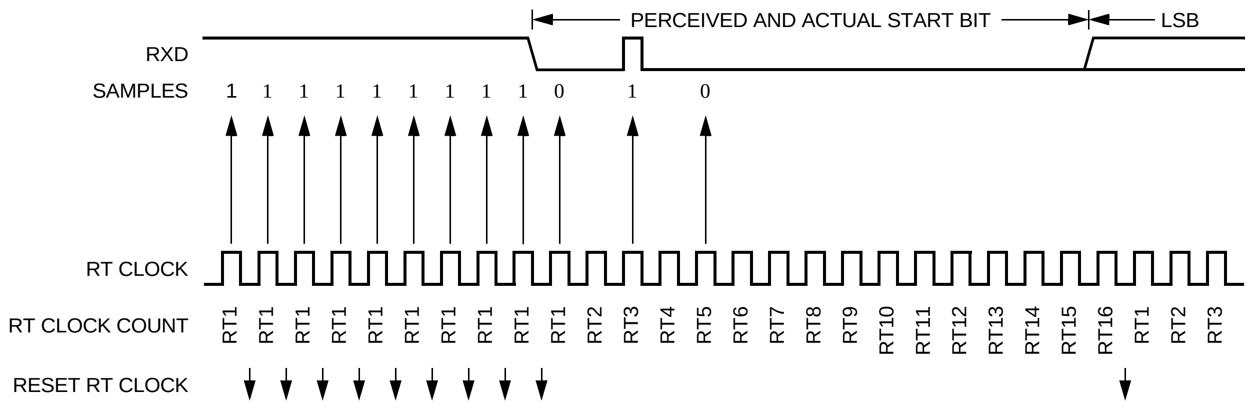

PERCEIVED AND ACTUAL START BIT
LSB
RXD
SAMPLES
1 1 1 1 1 1 1 1 1 0 1 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3
RESET RT CLOCK

Figure 11-9. Start Bit Search Example 4

Figure 11-10 delineates a burst of noise near the beginning of the start bit, resetting the RT clock. The sample after the reset is low but is not preceded by three high samples qualifying as a falling edge. Depending on the timing of the START bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

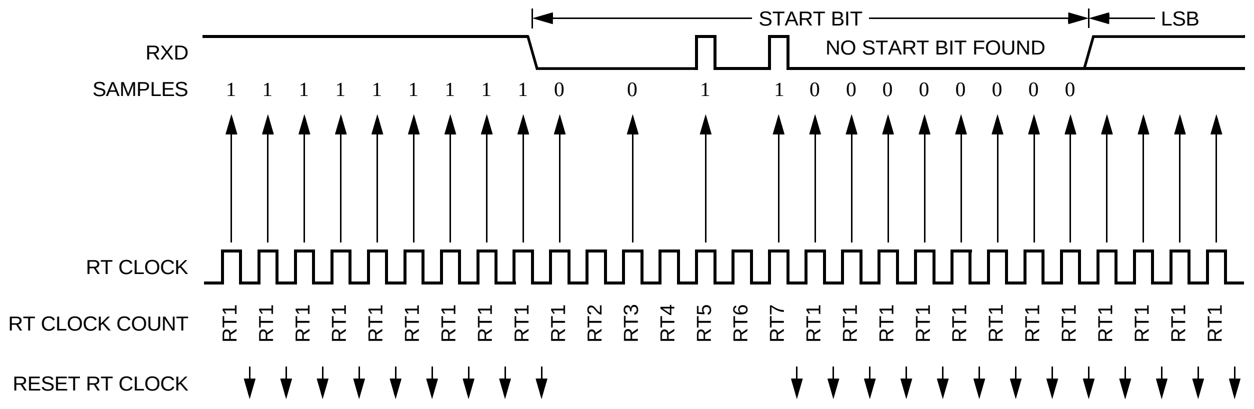


Figure 11-10. Start Bit Search Example 5

In Figure 11-11 a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

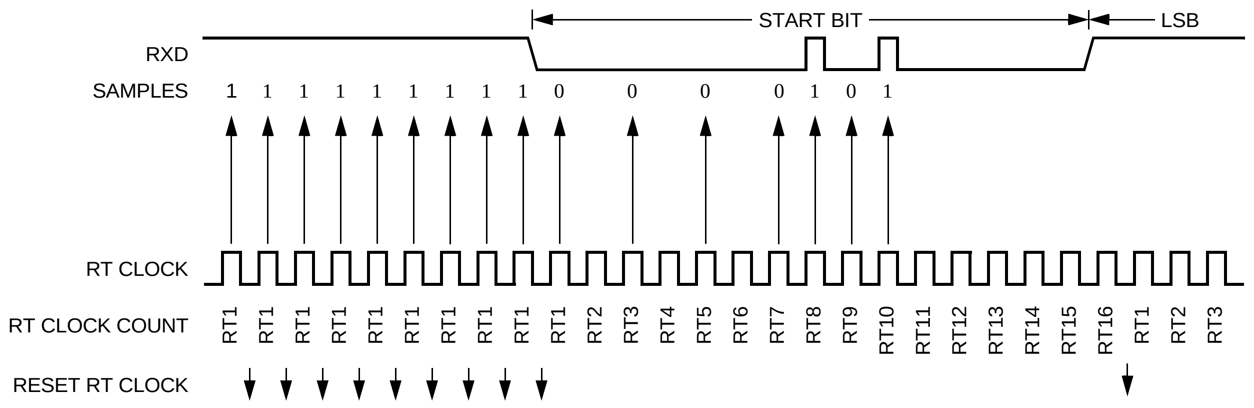

START BIT
LSB
RXD
SAMPLES
1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 1
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3
RESET RT CLOCK

Figure 11-11. Start Bit Search Example 6

11.4.4.4 Framing Errors

If the data recovery logic does not detect a Logic 1 where the STOP bit should be in an incoming frame, it sets the Framing Error (FE) flag in STAT register. A break character has no STOP bit. The FE flag is set concurrently with the RDRF flag. The FE flag inhibits further data reception until it is cleared.

11.4.4.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three STOP bit data samples to fall outside the actual STOP bit. Then a noise error occurs. If more than one of the samples is outside the STOP bit, a framing error occurs. In most applications, the baud rate tolerance is much more than the degree of misalignment likely to occur.

As the receiver samples an incoming frame, it resynchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames corrects misalignments between transmitter bit times and receiver bit times.

11.4.4.5.1 Slow Data Tolerance

Figure 11-12 illustrates how much a slow received frame can be misaligned without causing a noise or framing error. The slow STOP bit begins at RT8 instead of RT1, but it arrives in time for the STOP bit data samples at RT8, RT9, and RT10.

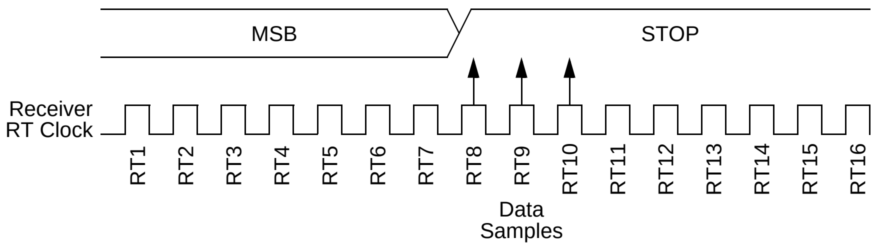


Figure 11-12. Slow Data

For an 8-bit data character, data sampling of the STOP bit takes the receiver:

$$9\text{-bit} \times 16\text{ RT cycles} + 10\text{ RT cycles} = 154\text{ RT cycles}$$

With the misaligned character, illustrated in **Figure 11-12,** the receiver counts 154 RT cycles at the point when the count of the transmitting device is:

$$9\text{-bit} \times 16\text{ RT cycles} + 3\text{ RT cycles} = 147\text{ RT cycles}$$

The maximum percentage difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$\left|\frac{154 - 147}{154}\right| \times 100 = 4.54\%$$

For a 9-bit data character, data sampling of the STOP bit takes the receiver:

$$10\text{-bit} \times 16\text{ RT cycles} + 10\text{ RT cycles} = 170\text{ RT cycles}$$

With the misaligned character illustrated in **Figure 11-12,** the receiver counts 170 RT cycles at the point when the count of the transmitting device is:

$$10\text{-bit} \times 16\text{ RT cycles} + 3\text{ RT cycles} = 163\text{ RT cycles}$$

With the misaligned character delineated in **Figure 11-12**, the receiver counts 170 RT cycles at the point when the count of the transmitting device is:

$$10\text{ bit times} \times 16\text{ RT cycles} + 3\text{ RT cycles} = 163\text{ RT cycles.}$$

The maximum percentage difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$\left|\frac{170 - 163}{170}\right| \times 100 = 4.12\%$$

11.4.4.5.2 Fast Data Tolerance

Figure 11-13 illustrates how much a fast received frame can be misaligned without causing a noise error or a framing error. The fast STOP bit ends at RT10 instead of RT16 but it is still sampled at RT8, RT9, and RT10.

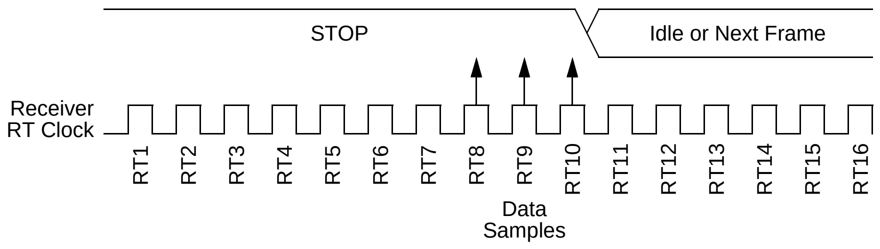


Figure 11-13. Fast Data

For an 8-bit data character, data sampling of the STOP bit takes the receiver:

$$9\text{-bit} \times 16 \text{ RT cycles} + 10 \text{ RT cycles} = 154 \text{ RT cycles}$$

With the misaligned character, illustrated in **Figure 11-13,** the receiver counts 154 RT cycles at the point when the count of the transmitting device is:

$$10\text{-bit} \times 16 \text{ RT cycles} = 160 \text{ RT cycles}$$

The maximum percentage difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$\left|\frac{154 - 160}{154}\right| \times 100 = 3.90\%$$

For a 9-bit (all 0s or 1s) data character, data sampling of the STOP bit takes the receiver:

$$10\text{-bit} \times 16 \text{ RT cycles} + 10 \text{ RT cycles} = 170 \text{ RT cycles}$$

With the misaligned character, illustrated in **Figure 11-13,** the receiver counts 170 RT cycles at the point when the count of the transmitting device is:

$$11\text{-bit} \times 16 \text{ RT cycles} = 176 \text{ RT cycles}$$

The maximum percentage difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$\left|\frac{170 - 176}{170}\right| \times 100 = 3.53\%$$

11.4.4.6 Receiver Wake-Up

In order for the SCI to ignore transmissions intended only for other receivers in multiple receiver systems, the receiver can be put into a standby state. Setting the Receiver Wake-Up (RWU) bit in the CTRL1 register puts the receiver into a standby state while receiver interrupts are disabled.

The transmitting device can address messages to selected receivers by including addressing information in the initial frame or frames of each message.

The WAKE bit in the CTRL1 register determines how the SCI is brought out of the standby state to process an incoming message. The WAKE bit enables either Idle Input Line Wake-Up or Address Mark Wake-Up:

- Idle Input Line Wake-Up (WAKE=0)—In this wake-up method, an idle condition on the RXD pin clears the RWU bit, waking up the SCI. The initial frame, or frames of every message contain addressing information. All receivers evaluate the addressing information. Receivers of the message then process the following frames. Any receiver a message does not address can set its RWU bit, returning to the standby state. The RWU bit remains set and the receiver remains on standby until another preamble appears on the RXD pin.

  Idle line wake-up require messages be separated by at least one preamble and no message contains preambles.

  The preamble waking a receiver does not set the receiver idle bit (RIDLE), nor the Receive Data Register Full (RDRF) flag.

- Address Mark Wake-Up (WAKE=1)—In this wake-up method, a Logic 1 in the MSB position of a frame clears the RWU bit, awakening the SCI. The Logic 1 in the MSB position marks a frame as an address frame containing addressing information. All receivers evaluate the addressing information. Receivers of the message then process the following frames. Any receiver a message does not address can set its RWU bit, returning to the standby state. The RWU bit remains set and the receiver remains on standby until another address frame appears on the RXD pin.

  The Logic 1 MSB of an address frame clears the receiver's RWU bit before the STOP bit is received and sets the RDRF flag.

  Address mark wake-up allows messages to contain preambles but requires the MSB be reserved for use in address frames.

Note: With the WAKE bit clear, setting the RWU bit after the RXD pin was idle can cause the receiver to wake-up immediately.

11.5 Operating Modes

11.5.1 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI, making it available as a General-Purpose I/O (GPIO) pin. The SCI uses the TXD pin for both receiving and transmitting. Please see **Figure 11-14**.

Setting the TE bit in the CTRL1 register configures TXD as the output for transmitted data. Clearing the TE bit configures TXD as the input for received data.

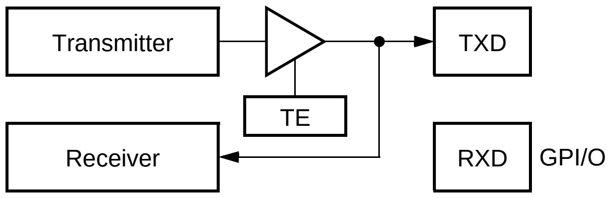


Figure 11-14. Single-Wire Operation (LOOPS = 1, RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the Receiver Source (RSRC) bit in the CTRL1 register. Setting the LOOPS bit disables the path from the RXD pin to the receiver. Setting the RSRC bit connects the receiver input to the output of the TXD pin driver.

11.5.2 Loop Operation

In Loop Operation, the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI and is available as a GPIO pin. Please see **Figure 11-15**.

Setting the TE bit in the CTRL1 register connects the transmitter output to the TXD pin. Clearing the TE bit disconnects the transmitter output from the TXD pin.

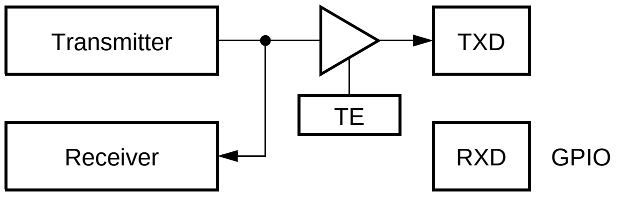


Figure 11-15. Loop Operation (LOOP = 1, RSRC = 0)

Enable Loop Operation by setting the LOOPS bit and clearing the RSRC bit in the CTRL1 register. Setting the LOOPS bit disables the path from the RXD pin to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. To enable Loop Operation both the Transmitter Enable (TE) and Receiver Enable (RE) bits in the CTRL1 register must be set.

Table 11-7. Loop Functions

Loop	RSRC	Function
0	x	Normal operation
1	0	Loop mode with internal TXD fed back to RXD
1	1	Single-wire mode with TXD output fed back to RXD

11.5.3 LIN Slave Operation

LIN slave operation occurs when the LIN MODE bit is set in CTRL2 register. The receiver searches for a break character (a start bit, eight bits of data all 0, and a 0 in the STOP bit location). Once a break character is detected (11 consecutive samples of Logic 0), the next field to be received is the synch field. The synch field is a word with 0×55 data which produces an alternating 0 and 1 pattern. The receiver detects the falling edge at the beginning of the START bit and begins counting system clocks until the falling edge at the beginning of data bit seven is detected at which time it stops counting. This count is divided by eight (for the passed 8-bit periods) and further divided by 16 to provide a new SBR value. If the data value of the synch field is 0×55, this new SBR value is placed in the Baud Rate register and the slave is considered synced to the master and further data words will be received properly. If the data value of the synch field is not 0×55, the LIN sync error (LSE) bit is set in the STAT register and subsequent received data bytes should be ignored. The receiver will resume searching for a break character as above.

In order for the break character to be successfully detected, the initial baud rate for this slave device must be within 15% of the nominal baud rate of the LIN master device.

11.5.4 Low-Power Options

11.5.4.1 Run Mode

Clearing the Transmitter Enable (TE) or Receiver Enable (RE) bits in the CTRL1 register reduces power consumption in Run mode. SCI registers are still accessible when TE or RE is cleared, but clocks to the core of the SCI are disabled.

11.5.4.2 Wait Mode

SCI operation in Wait mode depends on the state of the SWAI bit in the CTRL1 register.

- If SWAI is clear, the SCI operates normally when the CPU is in Wait mode.

- If SWAI is set, SCI clock generation ceases and the SCI module enters a power-conservation state when the CPU is in Wait mode. In this condition, SCI registers are not accessible. Setting SWAI does not affect the states of the Receiver Enable (RE) nor the Transmitter Enable (TE) bits.

  If SWAI is set, any transmission or reception in progress stops at Wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the DSP out of Wait mode. Exiting Wait mode by reset aborts any transmission or reception in progress, resetting the SCI.

11.5.4.3 Stop Mode

The SCI is inactive in Stop mode for reduced power consumption. The stop instruction does not affect SCI register states. SCI operation resumes after an external interrupt brings the CPU out of Stop mode. Exiting Stop mode by reset aborts any transmission or reception in progress and resets the SCI.

11.6 Register Definitions

Table 11-8. SCI Memory Map

Device	Peripheral	Base Address
56F801X	SCI	$00F0B0

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 11-9 lists the SCI registers in ascending address order, including their acronyms and address offset of each register. The SCI peripheral has five registers.

Table 11-9. SCI Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	RATE	Baud Rate Register	Read/Write	**Section 11.6.1**
Base + $1	CTRL1	Control Register1	Read/Write	**Section 11.6.2**
Base + $2	CTRL2	Control Register 2	Read/Write	**Section 11.6.3**
Base + $3	STAT	Status Register	*Read-Only*	**Section 11.6.4**
Base + $4	DATA	Data Register	Read/Write	**Section 11.6.5**

Bits of each of the five registers are summarized in **Figure 11-16.** Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	RATE	R	0	0	0	SBR												
		W	▨	▨	▨													
$1	CTRL1	R	LOOP	SWAI	RSRC	M	WAKE	POL	PE	PT	TEIE	TIIE	RFIE	REIE	TE	RE	RWU	SBK
		W																
$2	CTRL2	R	0	0	0	0	0	0	0	0	0	0	0	0	LIN MODE	0	0	0
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨	▨
$3	STAT	R	TDRE	TIDLE	RDRF	RIDLE	OR	NF	FE	PF	0	0	0	0	LSE	0	0	RAF
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$4	DATA	R	0	0	0	0	0	0	0	RECEIVE DATA								
		W	▨	▨	▨	▨	▨	▨	▨	TRANSMIT DATA								

R	0	Read as 0
W	▨	Reserved

Figure 11-16. SCI Register Map Summary

11.6.1 Baud Rate Register (RATE)

This register can be read at any time. Bits 12 through 0 can be written at any time, but bits 15 through 13 are reserved.

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	SBR												
Write	▨	▨	▨													
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Figure 11-17. Baud Rate (RATE) Register

The count in this register determines the baud rate of the SCI. The formula for calculating baud rate is:

$$\text{SCI Baud Rate} = \frac{\text{IPBus Clock}}{16 \times \text{SBR}}$$

Please see **Section 11.4.2** for more details and examples.

Note: The baud rate generator is disabled until the TE or the RE bits are set for the first time after reset. The baud rate generator is disabled when SBR=0.

11.6.1.1 Reserved—Bits 15–13

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

11.6.1.2 SCI Baud Rate (SBR)—Bits 12–0

Contents of the Baud Rate register has a value of 1 to 8191.

11.6.2 Control 1 Register (CTRL1)

The SCI Control 1 (CTRL1) register can be read and written at anytime.

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	LOOP	SWAI	RSRC	M	WAKE	POL	PE	PT	TEIE	TIIE	RFIE	REIE	TE	RE	RWU	SBK
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-18. Control 1 (CTRL1) Register

11.6.2.1 Loop Select (LOOP)—Bit 15

This bit enables loop operation. Please see **Section 11.5.2.** The loop operation disconnects the RXD pin from the SCI and the transmitter output goes into the receiver input.

- 0 = Normal operation enabled
- 1 = Loop operation enabled

11.6.2.2 Stop in Wait Mode (SWAI)—Bit 14

The SWAI bit disables the SCI in the Wait mode. Please see **Section 11.5.4.2.**

- 0 = SCI enabled in Wait mode
- 1 = SCI disabled in Wait mode

11.6.2.3 Receiver Source (RSRC)—Bit 13

When LOOP=1, the RSRC bit determines the internal feedback path for the receiver. See **Section 11.5.2** and **Table 11-7** for more details.

- 0 = Receiver input internally connected to transmitter output
- 1 = Receiver input connected to TXD pin

11.6.2.4 Data Format Mode (M)—Bit 12

This bit determines whether data characters are eight or nine bits long.

- 0 = One START bit, eight data bits, one STOP bit
- 1 = One START bit, nine data bits, one STOP bit

11.6.2.5 Wake-Up Condition (WAKE)—Bit 11

This bit determines which condition wakes up the SCI.

- 0 = Idle Line Wake-Up
- 1 = Address Mark Wake-Up (a Logic 1 in the MSB position of a receive data character)

Note: Address Mark Wake-Up is not a valid option when the parity function is enabled (PE=1) because the ADDRESS bit and the PARITY bit both occupy the MSB.

11.6.2.6 Polarity (POL)—Bit 10

This bit determines whether to invert the data as it goes from the transmitter to the TXD pin and from the RXD pin to the receiver. All bits, START, DATA, and STOP, are inverted as they leave the Transmit Shift register and before they enter the Receive Shift register.

- 0 = Doesn’t invert transmit and receive data bits (Normal mode)
- 1 = Invert transmit and receive data bits (Inverted mode)

Note: It is recommended the POL bit be toggled only when both TE and RE=0.

11.6.2.7 Parity Enable (PE)—Bit 9

This bit enables the parity function. When enabled, the parity function replaces the MSB of the data character with a parity bit.

- 0 = Parity function disabled
- 1 = Parity function enabled

Note: Address Mark Wake-Up (WAKE=1) is not a valid option when the parity function is enabled because the ADDRESS bit and the PARITY bit both occupy the MSB.

11.6.2.8 Parity Type (PT)—Bit 8

This bit determines whether the SCI generates and checks for even or odd parity of the data bits. With *even parity*, an *even* number of *ones clear* the PARITY bit while an *odd* number of *ones*, *sets* the PARITY bit. However, with *odd parity,* an *odd* number of *ones clear* the PARITY bit while an *even* number of *ones, sets* the PARITY bit.

- 0 = Even parity
- 1 = Odd parity

11.6.2.9 Transmitter Empty Interrupt Enable (TEIE)—Bit 7

This bit enables the TDRE flag to generate interrupt requests.

- 0 = TDRE interrupt requests disabled
- 1 = TDRE interrupt requests enabled

11.6.2.10 Transmitter Idle Interrupt Enable (TIIE)—Bit 6

This bit enables the TIDLE flag to generate interrupt requests.

- 0 = TIDLE interrupt requests disabled
- 1 = TIDLE interrupt requests enabled

11.6.2.11 Receiver Full Interrupt Enable (RFIE)—Bit 5

This bit enables the RDRF flag or the OR flag to generate interrupt requests.

- 0 = RDRF and OR interrupt requests disabled
- 1 = RDRF and OR interrupt requests enabled

11.6.2.12 Receive Error Interrupt Enable (REIE)—Bit 4

This bit enables the Receive Error (RE) flags (NF, PF, FE, and OR) to generate interrupt requests. The status bits can be checked during the error interrupt process.

- 0 = Error interrupt requests disabled
- 1 = Error interrupt requests enabled

11.6.2.13 Transmitter Enable (TE)—Bit 3

This bit enables the SCI transmitter and configures the TXD pin as the SCI transmitter output. The TE bit can be used to queue an idle preamble.

- 0 = Transmitter disabled
- 1 = Transmitter enabled

11.6.2.14 Receiver Enable (RE)—Bit 2

This bit enables the SCI Receiver.

- 0 = Receiver disabled
- 1 = Receiver enabled

11.6.2.15 Receiver Wake-Up (RWU)—Bit 1

This bit enables the wake-up function, inhibiting further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU. Please refer to **Section 11.4.4.6** for a description of Receive Wake-Up.

- 0 = Normal operation
- 1 = Standby state

11.6.2.16 Send Break (SBK)—Bit 0

Setting SBK sends one break character (all Logic 0s, include START, DATA, STOP). As long as SBK is set, transmitter sends uninterrupted break characters.

- 0 = No break characters
- 1 = Transmit break characters

11.6.3 Control 2 Register (CTRL2)

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	LIN MODE	0	0	0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-19. Control 2 (CTRL2) Register

11.6.3.1 Reserved—Bits 15–4

This bit field is reserved or not implemented. It is read as 0 and can be modified by writing.

11.6.3.2 Local Interconnect Network Mode (LIN MODE)—Bit 3

This bit should only be used in Local Interconnect Network (LIN) applications.

- 0 = The LIN auto baud feature is disabled and the SBR register will maintain whatever value the processor writes to it.
- 1 = Enable a search for the break followed by sync char (0x55) from the master LIN device. When the break is detected the following sync character will be used to measure the baud rate of the transmitting master and the SBR register will be automatically reloaded with the value needed to *match* that baud rate.

Note: During initialization the SBR register should be loaded to a value that is within 15 percent of the actual master data rate, otherwise 0x00 data may be misinterpreted as a break.

Note: If the first character following a break is not the LIN sync character (0x55) the SBR will NOT be adjusted.

11.6.3.3 Reserved—Bits 2–0

This bit field is reserved or not implemented. It is read as 0 and can be modified by writing.

11.6.4 Status Register (STAT)

This register can be read at anytime; however, it cannot be modified by writing. Writes clear flags.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	TDRE	TIDLE	RDRF	RIDLE	OR	NF	FE	PF	0	0	0	0	LSE	0	0	RAF
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-20. Status (STAT) Register

11.6.4.1 Transmit Data Register Empty Flag (TDRE)—Bit 15

This bit is set when the Transmit Shift register receives a character from the DATA register. Clear TDRE by reading the STAT register, then write to the DATA register.

- 0 = No character transferred to Transmit Shift register
- 1 = Character transferred to Transmit Shift register

11.6.4.2 Transmitter Idle Flag (TIDLE)—Bit 14

This bit is set when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TIDLE is set, the TXD pin becomes idle (Logic 1). Clear TIDLE by reading the STAT register, then write to the DATA register.

- 0 = Transmission in progress
- 1 = No transmission in progress

11.6.4.3 Receive Data Register Full Flag (RDRF)—Bit 13

This bit is set when the data in the Receive Shift register transfers to the DATA register. Clear RDRF by reading the STAT register, then read the DATA register.

- 0 = Data not available in the DATA register
- 1 = Received data available in the DATA register

11.6.4.4 Receiver Idle Line Flag (RIDLE)—Bit 12

This bit is set when ten consecutive Logic 1s (if M=0) or eleven consecutive Logic 1s (if M=1) appear on the receiver input. Once the RIDLE flag is cleared by the receiver detecting a Logic 0, a valid frame must again set the RDRF flag before an idle condition can set the RIDLE flag.

- 0 = Receiver input is either active now or has never become active since the RIDLE flag was last cleared by reset
- 1 = Receiver input has become idle (after receiving a valid frame)

Note: When the Receiver Wake-Up (RWU) bit is set, an idle line condition does not set the RIDLE flag.

11.6.4.5 Overrun Flag (OR)—Bit 11

This bit is set when software fails to read the DATA register before the Receive Shift register receives the next frame. The data in the Shift register is lost, but the data already in the DATA register is not affected. Clear OR by reading the STAT register, then write the STAT register with any value.

- 0 = No overrun
- 1 = Overrun

11.6.4.6 Noise Flag (NF)—Bit 10

This bit is set when the SCI detects noise on the receiver input. The NF bit is set during the same cycle as the RDRF flag, but it is not set in the case of an overrun. Clear NF by reading the STAT register, then write the STAT register with any value.

- 0 = No noise
- 1 = Noise

11.6.4.7 Framing Error Flag (FE)—Bit 9

This bit is set when a Logic 0 is accepted as the STOP bit. The FE bit is set during the same cycle as the RDRF flag but it is not set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading the STAT register, then write the STAT register with any value.

- 0 = No framing error
- 1 = Framing error

11.6.4.8 Parity Error Flag (PF)—Bit 8

This bit is set when the Parity Enable (PE) bit is set and the parity of the received data does not match its parity bit. Clear PF by reading the STAT register, then write the STAT register with any value.

- 0 = No parity error
- 1 = Parity error

11.6.4.9 Reserved—Bits 7–4

These bits are reserved or not implemented. They are read as 0 and cannot be modified by writing.

11.6.4.10 Local Interconnect Network Sync Error (LSE)—Bit 3

This bit is only active when LIN MODE is set. When LSE is set, an RERR interrupt will occur if REIE is set.

LSE is set when a LIN sync search detects a non-sync character (anything other than 0x55). Having this bit set indicates either a protocol error was detected from the Master LIN device or there is a gross mis-match in data rates. This bit is cleared by reading the STAT register with LSE set and then writing the STAT register with any value.

- 0 = No error occurred since the SCI LIN MODE was enabled or the bit was last cleared.
- 1 = A sync error prevented loading of the SBR with a revised value after the break was detected.

11.6.4.11 Reserved—Bits 2–1

These bits are reserved or not implemented. They are read as 0 and cannot be modified by writing.

11.6.4.12 Receiver Active Flag (RAF)—Bit 0

This bit is set when the receiver detects a Logic 0 during the RT1 time period of the START bit search. RAF is cleared when the receiver detects false start bits (usually from noise or baud rate mismatch) or when the receiver detects a preamble.

- 0 = No reception in progress
- 1 = Reception in progress

11.6.5 Data Register (DATA)

The DATA register can be read and modified at any time. Reading accesses the SCI Receive Data register. Writing to the register accesses the SCI Transmit Data register.

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	RECEIVE DATA								
Write								TRANSMIT DATA								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11-21. Data (DATA) Register

11.6.5.1 Reserved—Bits 15–9

These bits are reserved or not implemented. They are read as 0 and cannot be modified by writing.

11.6.5.2 Receive/Transmit Data—Bits 8–0

Writing to these bits loads the transmit data. Reading these bits accesses the receive data.

Note: When configured for 8-bit data, bits 7 to 0 contain the data received.

11.7 Clocks

All timing is derived from the IPBus clock, the main clock for this module. See **Section 11.4.2** for a description of how the data rate is determined.

11.8 Interrupts

11.8.1 Description of Interrupt Operation

Figure 11-22. SCI Interrupt Summary

Interrupt	Source	Interrupt Vector	Description
TDRE	Transmitter	SCI_Vector_Base + $0	Transmit Data Register Empty Interrupt
TIDLE		SCI_Vector_Base + $1	Transmit Idle Interrupt
RERR	Receiver	SCI_Vector_Base + $3	Receive Error (FE, NF, PF, LSE, or OR) Interrupt
RDRF/OR		SCI_Vector_Base + $4	Receive Data Register Full/Overrun interrupt

Figure 11-23. SCI Interrupt Sources

Flag	Source	Local Enable	Description
TDRE	Transmitter	TEIE	Transmit Data Register Empty Interrupt
TIDLE		TIIE	Transmit Idle Interrupt
RDRF	Receiver	RFIE	Receive Data Register Full/Overrun interrupt
OR			
FE	Receiver	REIE	Receive Error (FE, NF, PF, LSE, or OR) Interrupt
PE			
NF			
OR			
LSE			

11.8.1.1 Transmitter Empty Interrupt

This interrupt is enabled by setting the TEIE bit of the CTRL1 register. When this interrupt is enabled an interrupt is generated when data is transferred from the DATA register to the Transmit Shift register. The interrupt service routine should read the STAT register, verifying the TDRE bit is set, and then write the next data to be transmitted to the DATA register, clearing the TDRE bit.

11.8.1.2 Transmitter Idle Interrupt

This interrupt is enabled by setting the TIIE bit of the CTRL1 register. This interrupt indicates the TDRE flag is set and the transmitter is no longer sending data, preamble, or break characters. The interrupt service routine should read the STAT register, verifying the TIDLE bit is set and then initiate the preamble, break, or write a data character to the DATA register. Any of these actions will clear the TIDLE bit because the transmitter will then be busy.

11.8.1.3 Receiver Full Interrupt

This interrupt is enabled by setting the RFIE bit of the CTRL1 register. This interrupt indicates receive data is available in the DATA register. The interrupt service routine should read the STAT register, verifying the RDRF bit is set and then read the data from the STAT register, clearing the RDRF bit.

11.8.1.4 Receive Error Interrupt

This interrupt is enabled by setting the REIE bit of the CTRL1 register. This interrupt indicates any of the listed errors were detected by the receiver.

1. Noise Flag (NF) set
2. Parity Error Flag (PF) set
3. Framing Error Flag (FE) set
4. Overrun Flag (OR) set
5. LIN Synchronization Error Flag (LSE) set

The interrupt service routine should read the STAT register to determine which of the error flags was set. The error flag is cleared by writing anything to the STAT register. Then the appropriate action should be taken by the software to handle the error condition.

11.8.1.5 Recover From Wait Mode

Any enabled SCI interrupt request can bring the CPU out of Wait mode.

11.9 Resets

Any system reset completely resets the SCI.

Chapter 12
Serial Peripheral Interface (SPI)

Document Revision History for Chapter 12, Serial Peripheral Interface (SPI)

Version History	Description of Change
Rev. 0	Initial release

12.1 Introduction

This chapter describes the Serial Peripheral Interface (SPI) module. The module allows full-duplex, synchronous, serial communication between the 16-bit controller and peripheral devices, including other 16-bit controllers.

12.2 Features

Characteristics of the SPI module include:

- Full-duplex operation
- Master and Slave modes
- Double-buffered operation with separate transmit and receive registers
- Programmable length transmissions (two to 16 bits)
- Programmable transmit and receive shift order (MSB as first or last bit transmitted)
- Eight Master mode frequencies (maximum = module clock ÷ 2)
- Maximum Slave mode frequency = module clock ÷ 2
- Serial clock with programmable polarity and phase
- Two separately enabled interrupts
  - SPI Receiver Full/Error, supporting three interrupt flags:
    - Receiver Full
    - Mode Fault
    - Receiving overflow
  - SPI Transmitter Empty
- Wired OR mode functionality enabling connection to multiple SPIs

Note: Throughout this chapter, there are references to the SPI module clock. The SPI module clock is the IPBus clock, which is equal to the System Clock.

12.3 Block Diagram

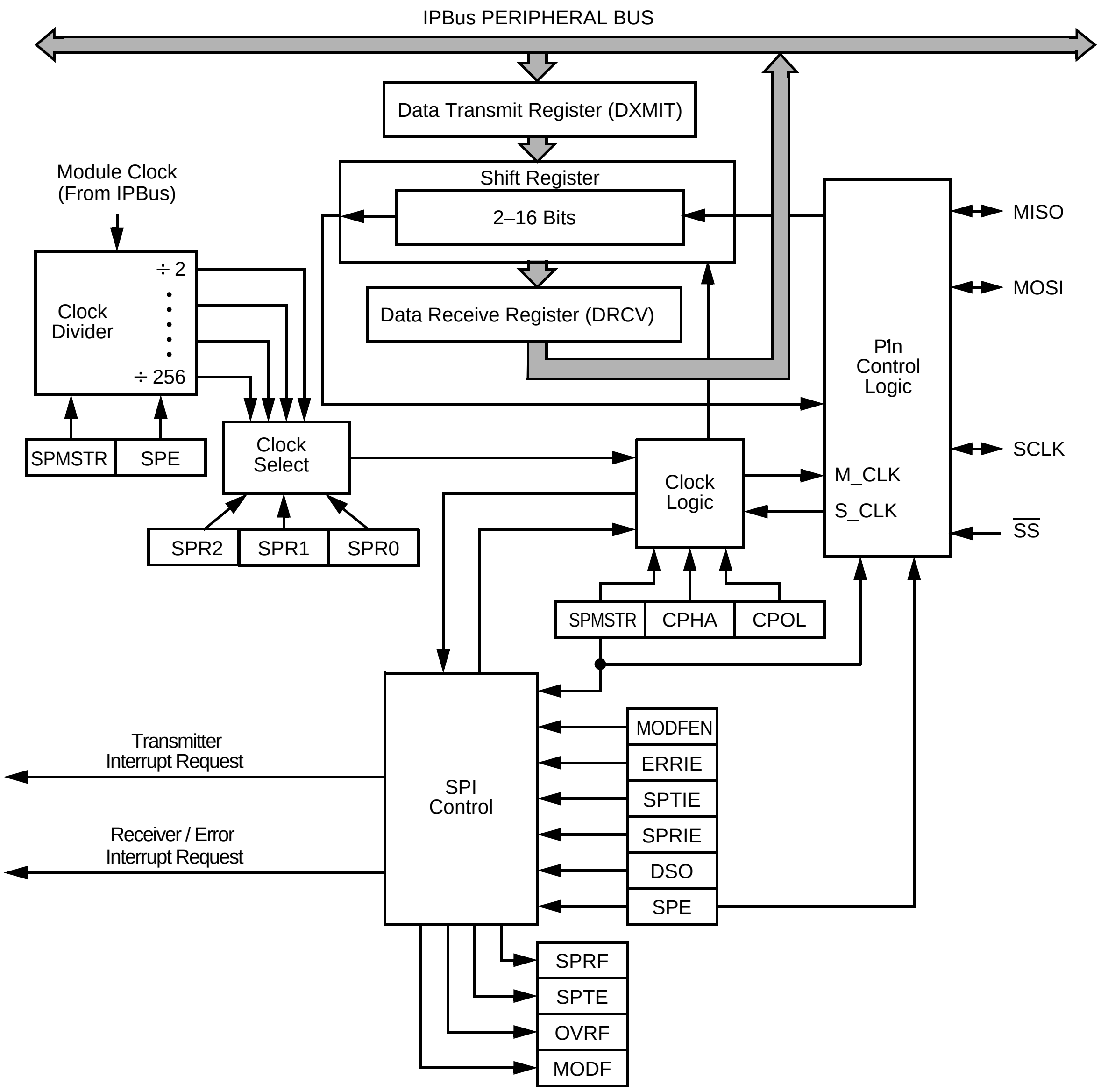


Figure 12-1. SPI Block Diagram

12.4 Operating Modes

The SPI has two operating modes:

- Master
- Slave

An operating mode is selected by the SPMSTR bit in the SPI Status and Control (CTRL) register as follows:

- SPMSTR = 0 (Slave mode)
- SPMSTR = 1 (Master mode)

Configuration steps for the typical SPI Slave mode:

- Set each of the SPI Status and Control (SCTRL) register bits as shown below:

  SPR=0, DSO=1, ERRIE=0, MODFEN=0, SPRIE=0, SPMSTR=0, CPOL=0, CPHA=1, SPE=0, SPTIE=0, SPRF=0, OVRF=0, MODF=1, and SPTE=0

- Set the Date Size and Control (DSCTRL) register:

  WOM=0, TDMAEN=0, RDMAEN=0, DS3=1, DS2=1, DS1=1, and DS0=1

- Enable the device.
- Store the empty character in the Data Transmit (DXMIT) register.
- Enable the Receive and Error Interrupt.
- The slaves should be initialized prior to the master’s operation.

Configuration steps for typical SPI Master mode:

- Set the SPI Status and Control (SCTRL) register bits as:

  SPR=1, DSO=1, ERRIE=0, MODFEN=0, SPRIE=0, SPMSTR=1, CPOL=0, CPHA=1, SPE=0, SPTIE=0, SPRF=0, OVRF=0, MODF=1, and SPTE=0

- Set the Data Size and Control (DSCTRL) register bits as:

  WOM=0, TDMAEN=0, RDMAEN=0, DS3=0, DS2=1, DS1=1, and DS0=1

- Initialize the software pointers and buffers.
- Enable the device and the Receive and Error Interrupts.

12.4.1 Master Mode

The SPI operates in Master mode when the SPI master bit, SPMSTR, is set. Only a master SPI module can initiate transmissions. With the SPI enabled, software begins the transmission from the master SPI module by writing to the SPI Data Transmit (DXMIT) register. If the Shift Register is empty, the data immediately transfers to the Shift Register, setting the SPI Transmitter Empty (SPTE) bit. The data begins shifting out on the Master Out/Slave In (MOSI) pin under the control of the SPI Serial Clock (SCLK).

The SPR[2:0] bits in the SPI Status and Control (SCTRL) register control the Baud Rate Generator, determining the speed of the Shift Register. The Baud Rate Generator of the master also controls the Shift Register of the slave peripheral via the SCLK pin.

As the data shifts out on the MOSI pin of the master, external data shifts in from the Slave on the Master In/Slave Out (MISO) pin. The transmission ends when the SPI Receiver Full (SPRF) bit in the SCTRL becomes set. At the same time the SPRF becomes set, the data from the slave transfers to the Data Receive (DRCV) register. In a normal operation SPRF signals the end of a transmission. Software clears the SPRF reading the DRCV. Writing to the DXMIT clears the SPTE bit.

Figure 12-2 is an example configuration for a Full-Duplex Master-Slave Configuration. Having the Slave Select ($\overline{SS}$) bit of the master 16-bit controller held high is only necessary if MODFEN=1. Tying the Slave 16-bit controller $\overline{SS}$ bit to ground should only be executed if CPHA=1.

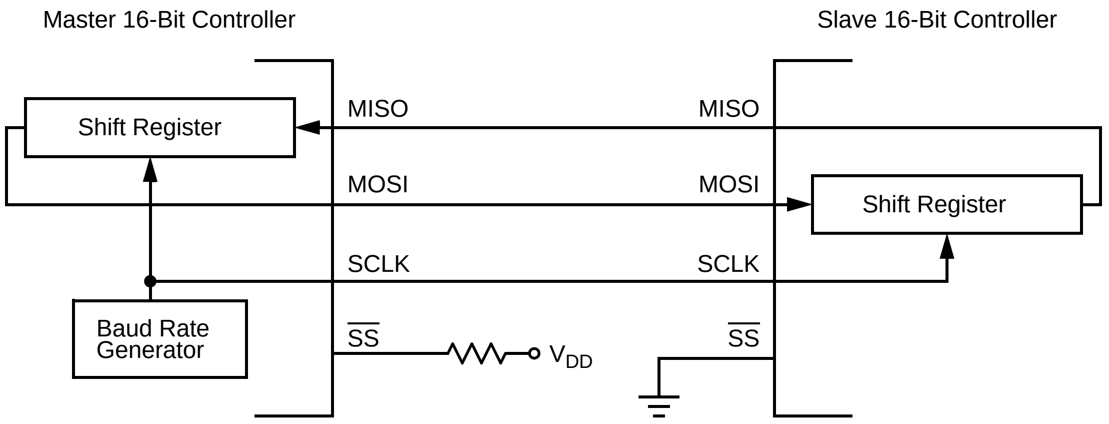


Figure 12-2. Full Duplex Master/Slave Connections

12.4.2 Slave Mode

The SPI operates in Slave mode when the SPMSTR bit is cleared. While in Slave mode, the SCLK pin acts as the input for the serial clock from the master 16-bit controller. Before a data transmission occurs, the $\overline{\text{SS}}$ pin of the slave SPI must be at Logic 0. $\overline{\text{SS}}$ must remain low until the transmission is complete or a Mode Fault Error occurs when MODFEN=1.

Note: The SPI must be enabled (SPE=1) for slave transmissions to be received.

Note: Data in the Slave’s transmitter Shift Register will be unaffected by SCLK transitions in the event the slave SPI is deselected ($\overline{\text{SS}}$=1).

In a slave SPI module, data enters the Shift Register under the control of the Serial Clock, (SCLK), from the master SPI module. After a full length data transmission enters the Shift Register of a slave SPI, it transfers to the DRCV register and the SPI Receiver Full (SPRF) bit in the SCTRL is set. If the SPI Receive Interrupt Enable (SPRIE) bit in the SCTRL register is set, a Receive Interrupt is also generated. To prevent an overflow condition, slave software must read the DRCV register before another full length data transmission enters the Shift Register.

Frequency of the SCLK for the SPI configured as a slave does not have to correspond to any particular SPI baud rate. The baud rate only controls the speed of the SCLK generated by the SPI configured as a master. Therefore, the frequency of the SCLK for a SPI configured as a slave can be any frequency less than or equal to half of the module clock.

When the master SPI starts a transmission, the data in the slave Shift Register begins shifting out on the MISO pin. The slave can load its Shift Register with new data for the next transmission by writing to its DXMIT. The slave must write to its DXMIT at least one bus cycle before the master starts the next transmission. Otherwise, the data already in the slave Shift Register shifts out on the MISO pin.

When the CPHA bit is set the first edge of SCLK starts a transmission. When CPHA is cleared the falling edge of $\overline{\text{SS}}$ starts a transmission.

Note: The Master SPI’s SCLK must be in the proper idle state (depends on setting of CPOL bit) before the slave is enabled to prevent SCLK from appearing as a clock edge.

12.4.3 Wired OR Mode

Wired-OR functionality is provided to permit the connection of multiple SPIs. **Figure 12-3** illustrates a single master controlling multiple slave SPIs. When the WOM bit is set, the outputs switch from conventional complementary CMOS output to open drain outputs. This lets the internal pull-up resistor bring the line high, and whichever SPI drives the line pulls it low as needed.

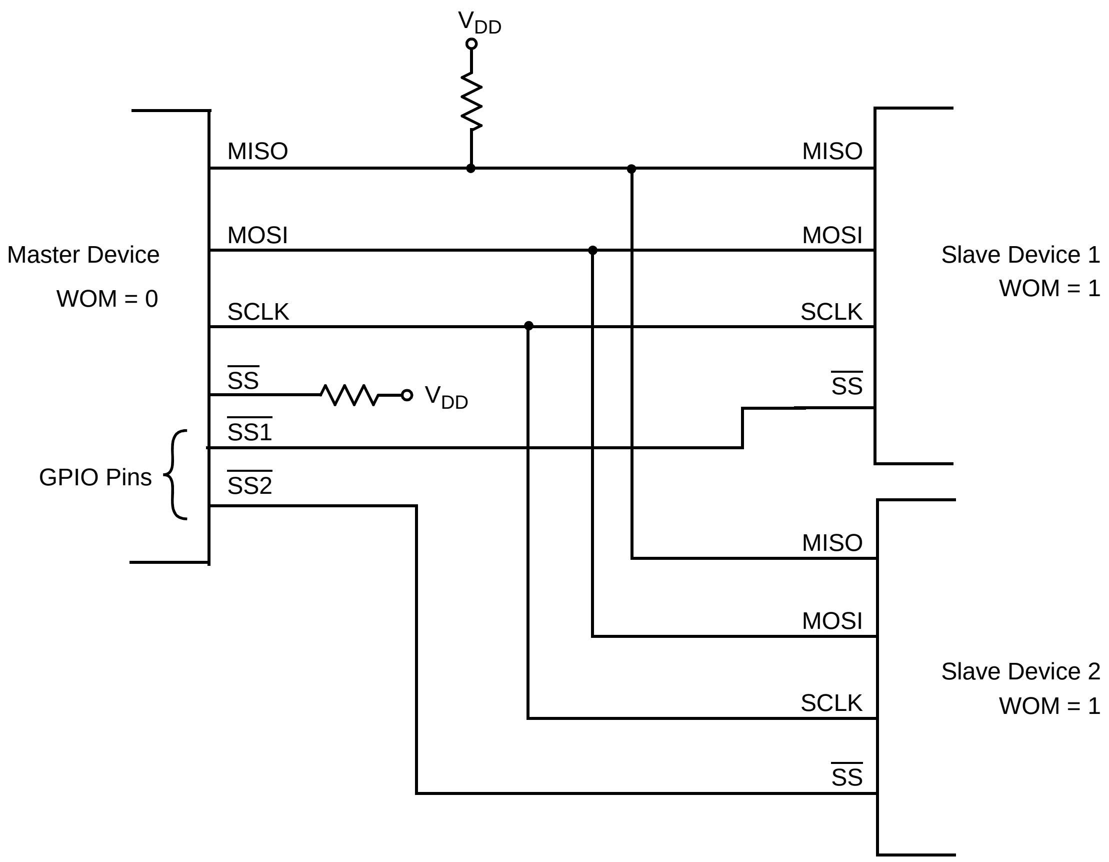


Figure 12-3. Master with Two Slaves

12.5 Pin Descriptions

There are four external SPI pins. Each is summarized in **Table 12-1.**

Table 12-1. Summary External I/O Signals

Signal Name	Description	Direction
MISO	Master-In Slave-Out Pad Pin	Bi-Directional
MOSI	Master-Out Slave-In Pad Pin	Bi-Directional
SCLK	Serial Clock Pad Pin	Bi-Directional
$\overline{\text{SS}}$	Slave Select Pad Pin (Active Low)	Input

12.5.1 Master In/Slave Out (MISO)

MISO is one of the two SPI module pins dedicated to transmit serial data. In full duplex operation, the MISO pin of the master SPI module is connected to the MISO pin of the slave SPI module. The master SPI simultaneously receives data on its MISO pin and transmits data from its MOSI pin. The slave SPI simultaneously transmits data on its MISO pin and receives data from its MOSI pin.

Slave output data on the MISO pin is enabled only when the SPI is configured as a slave. The SPI is configured as a slave when the SPMSTR bit, discussed in **Section 12.9.1,** is Logic 0 and its $\overline{\text{SS}}$ pin is Logic 0. To support a multiple slave system, a Logic 1 on the $\overline{\text{SS}}$ pin puts the MISO pin in a High Impedance state.

12.5.2 Master Out/Slave In (MOSI)

MOSI is the other SPI module pin dedicated to transmit serial data. In full duplex operation, the MOSI pin of the master SPI module is connected to the MOSI pin of the slave SPI module. The master SPI simultaneously transmits data from its MOSI pin and receives data on its MISO pin. The slave SPI simultaneously receives data from its MOSI pin and transmits data on its MISO pin.

12.5.3 Serial Clock (SCLK)

The serial clock synchronizes data transmission between master and slave devices. In a master 16-bit controller, the SCLK pin is the clock output. In a slave 16-bit controller, the SCLK pin is the clock input. In full duplex operation, the master and slave 16-bit controller exchange data in the same number of clock cycles as the number of bits of transmitted data.

12.5.4 Slave Select ($\overline{SS}$)

The $\overline{SS}$ pin has various functions depending on the current state of the SPI. For a SPI configured as a slave, the $\overline{SS}$ is used to select a slave. When the Clock Phase (CPHA) bit in the SCTRL is cleared, the $\overline{SS}$ is used to define the start of a transmission, so it must be toggled high and low between each full length data transmitted for the CPHA = 0 format. However, it can remain low between transmissions for the CPHA = 1 format as illustrated in **Figure 12-5**.

When a SPI is configured as a slave, the $\overline{SS}$ pin is always configured as an input. The MODFEN bit can prevent the state of the $\overline{SS}$ from creating a MODF error.

Note: A Logic 1 voltage on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high impedance state. The slave SPI ignores all incoming SCLK clocks, even if it was already in the middle of a transmission. A Mode Fault occurs if the $\overline{SS}$ pin changes state during a transmission and MODFEN=1.

When a SPI is configured as a master, the $\overline{SS}$ input can be used in conjunction with the MODF flag to prevent multiple masters from driving MOSI and SCLK. For the state of the $\overline{SS}$ pin to set the MODF flag, the MODFEN bit in the SCLK register must be set.

If the system has only one master, then the master's $\overline{SS}$, when configured as a GPIO, can be used to select a slave.

Table 12-2. SPI I/O Configuration

SPE	SPMSTR	MODFEN	SPI Configuration	State of $\overline{SS}$ Logic
0	x	x	Not Enabled	$\overline{SS}$ ignored by SPI
1	0	x	Slave	Input-only to SPI
1	1	0	Master without MODF	$\overline{SS}$ ignored by SPI
1	1	1	Master with MODF	Input-only to SPI

x = Don't care

12.6 Transmission Formats

During a SPI transmission, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). A serial clock synchronizes shifting and sampling on the two serial data lines. A slave select line allows selection of an individual slave SPI device; slave devices not selected do not interfere with SPI bus activities. On a master SPI device, the slave select line can optionally be used to indicate multiple-master bus contention.

12.6.1 Data Transmission Length

The SPI can support data lengths from two to 16 bits. This can be configured in the Data Size and Control (DSCTRL) register. When the data length is less than 16 bits, the DRCV register will pad the upper bits with zeros.

Note: Data can be lost if the data length is not the same for both master and slave devices.

12.6.2 Data Shift Ordering

The SPI can be configured to transmit or receive the MSB of the desired data first or last. This is controlled by the Data Shift Order (DSO) bit in the SCTRL register. Regardless which bit is transmitted or received first, the data shall always be written to the DXMIT register and read from the DRCV register with the LSB in bit zero and the MSB in the correct position, depending on the data transmission size.

12.6.3 Clock Phase and Polarity Controls

Software can select any of four combinations of Serial Clock (SCLK) phase and polarity using two bits in the SCTRL. The Clock Polarity (CPOL) is specified by the control bit. This selects an active high or low clock but has no other effect on the transmission format.

The Clock Phase (CPHA) control bit selects one of two fundamentally different transmission formats. The clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transmissions to allow a master device to communicate with peripheral slaves having different requirements.

Note: Before writing to either the CPOL or CPHA bits, disable the SPI by clearing the SPI Enable (SPE) bit.

12.6.4 Transmission Format When CPHA = 0

Figure 12-4 exhibits a SPI transmission with CPHA as Logic 0.

Note: The figure should not be used as a replacement for data sheet parametric information.

Two waveforms for the SCLK are shown:

1. CPOL = 0
2. CPOL = 1

The waveforms may be interpreted as a master or slave timing diagram since the Serial Clock (SCLK), Master In/Slave Out (MISO), and Master Out/Slave In (MOSI) pins are directly connected between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master.

The $\overline{\text{SS}}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its slave select input ($\overline{\text{SS}}$) is at Logic 0 because only the selected slave drives to the master. The $\overline{\text{SS}}$ pin of the master is not shown, but it is assumed to be inactive. When CPHA = 0, the first SCLK edge is the MSB capture strobe. Therefore, the slave must begin driving its data before the first SCLK edge and a falling edge on the $\overline{\text{SS}}$ pin is used to start the slave data transmission. The slave's $\overline{\text{SS}}$ pin must be toggled back to high and then low again between each full length data transmission as depicted in **Figure 12-5**.

Note: **Figure 12-4** assumes 16-bit data lengths and the MSB shifted out first.

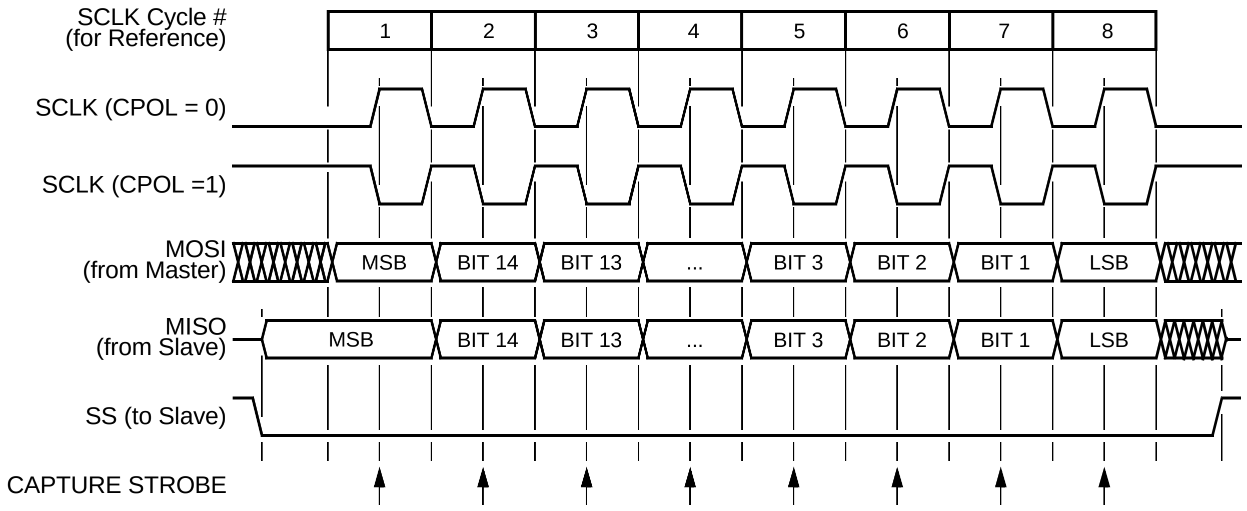


Figure 12-4. Transmission Format (CPHA = 0)

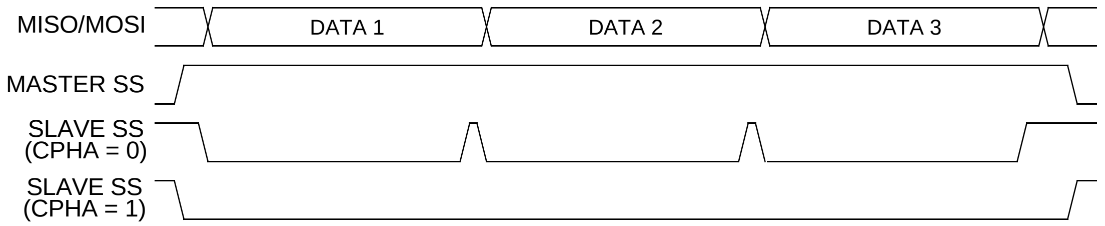


Figure 12-5. CPHA/$\overline{\text{SS}}$ Timing

When CPHA = 0 for a slave, the falling edge of $\overline{\text{SS}}$ indicates the beginning of the transmission. This causes the SPI to leave its idle state and begin driving the MISO pin with the first bit of its data. Once the transmission begins, no new data is allowed into the Shift Register from the DXMIT register. Therefore, the Data register of the slave must be loaded with transmit data before the falling edge of $\overline{SS}$. Any data written after the falling edge is stored in the DXMIT register and transferred to the Shift Register after the current transmission.

12.6.5 Transmission Format When CPHA = 1

A SPI transmission is shown in **Figure 12-6** where CPHA is Logic 1.

Note: The figure should not be used as a replacement for data sheet parametric information.

Two waveforms are shown for SCLK: 1 for CPOL=0 and another for CPOL=1. The waveforms may be interpreted as a master or slave timing diagram since the Serial Clock (SCLK), Master In/Slave Out (MISO), and Master Out/Slave In (MOSI) pins are directly connected between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The slave SPI drives its MISO output only when its Slave Select ($\overline{SS}$) input is at Logic 0, so only the selected slave drives to the master. When CPHA=1, the master begins driving its MOSI pin on the first SCLK edge. Therefore, the slave uses the first SCLK edge as a start transmission signal. The $\overline{SS}$ pin can remain low between transmissions. This format may be preferable in systems having only one master and slave driving the MISO data line.

Note: **Figure 12-6** assumes 16-bit data lengths and the MSB shifted out first.

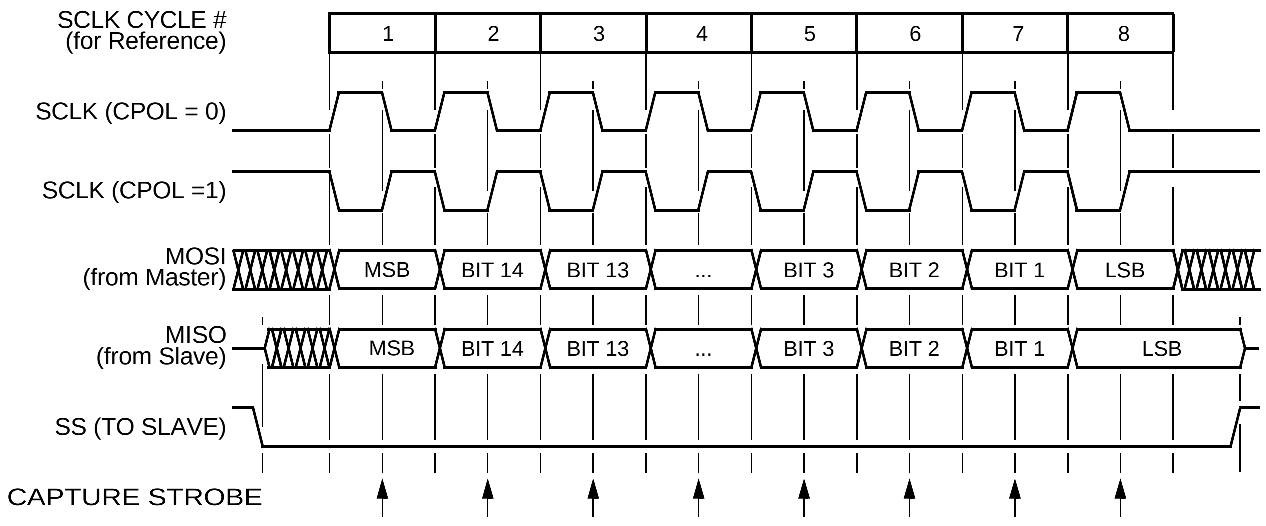


Figure 12-6. Transmission Format (CPHA = 1)

When CPHA=1 for a slave, the first edge of the SCLK indicates the beginning of the transmission. This causes the SPI to leave its idle state and begin driving the MISO pin with the first bit of its data. Once the transmission begins, no new data is allowed into the Shift Register from the DXMIT. Therefore, the Data register of the slave must be loaded with transmit data before the first edge of SCLK. Any data written after the first edge is stored in the DXMIT register and transferred to the Shift Register after the current transmission.

12.6.6 Transmission Initiation Latency

When the SPI is configured as a master (SPMSTR=1), writing to the DXMIT register begins a transmission. CPHA has no effect on the delay to the start of the transmission, but it does affect the initial state of the SCLK signal. When CPHA=0, the SCLK signal remains inactive for the first half of the first SCLK cycle. When CPHA=1, the first SCLK cycle begins with an edge on the SCLK line from its inactive to its active level. The SPI clock rate, selected by SPR[2:0], affects the delay from the write to DXMIT register and the start of the SPI transmission. The internal SPI clock in the master is a free-running derivative of the internal clock. To conserve power, it is enabled only when both the SPE and SPMSTR bits are set. Since the SPI clock is free-running, it is uncertain where the write to the DXMIT occurs relative to the slower SCLK. This uncertainty causes the variation in the initiation delay, demonstrated in **Figure 12-7.** This delay is no longer than a single SPI bit time. That is, the maximum delay is two bus cycles for DIV2, four IPBus cycles for DIV4, eight IPBus cycles for DIV8, and so on up to a maximum of 256 cycles for DIV256.

Note: **Figure 12-7** assumes 16-bit data lengths and the MSB shifted out first.

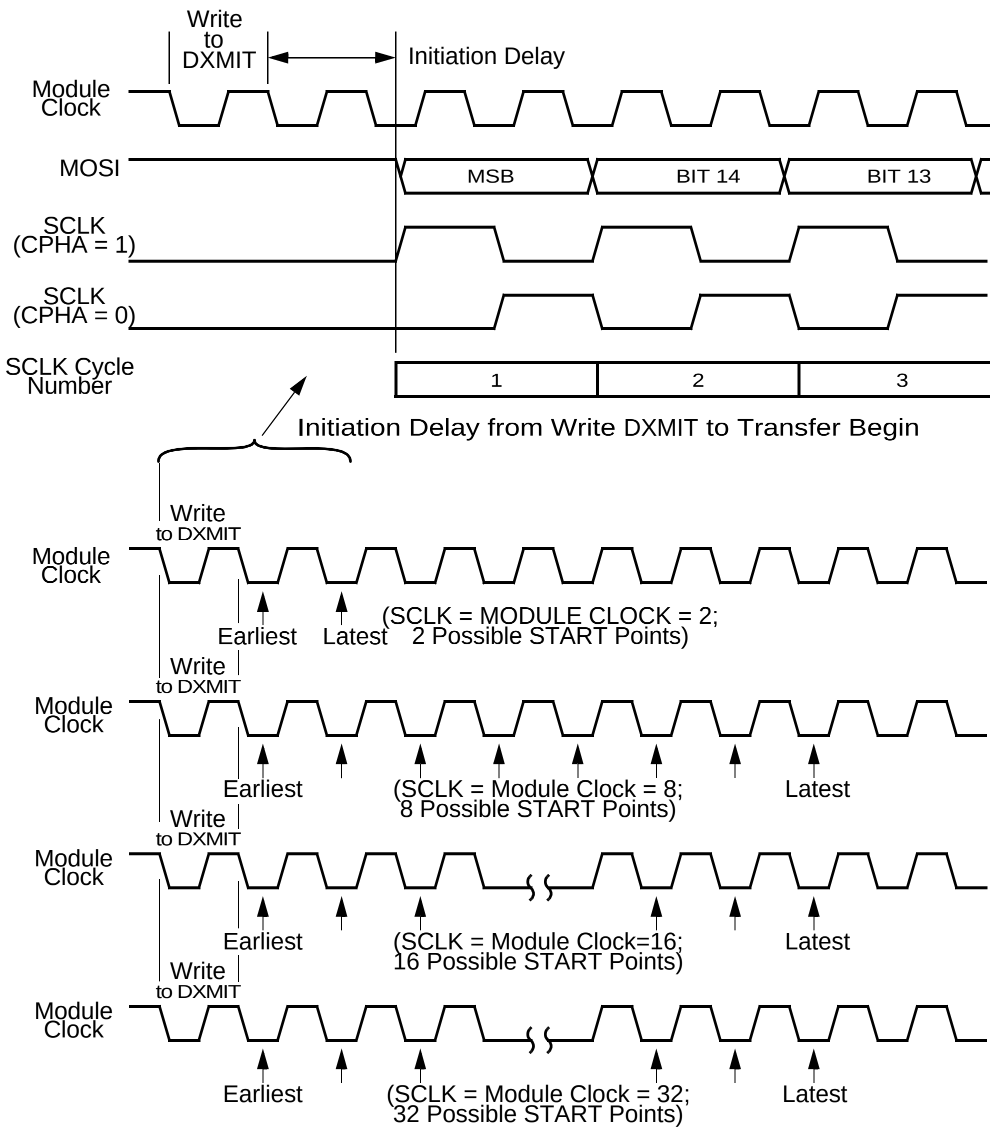
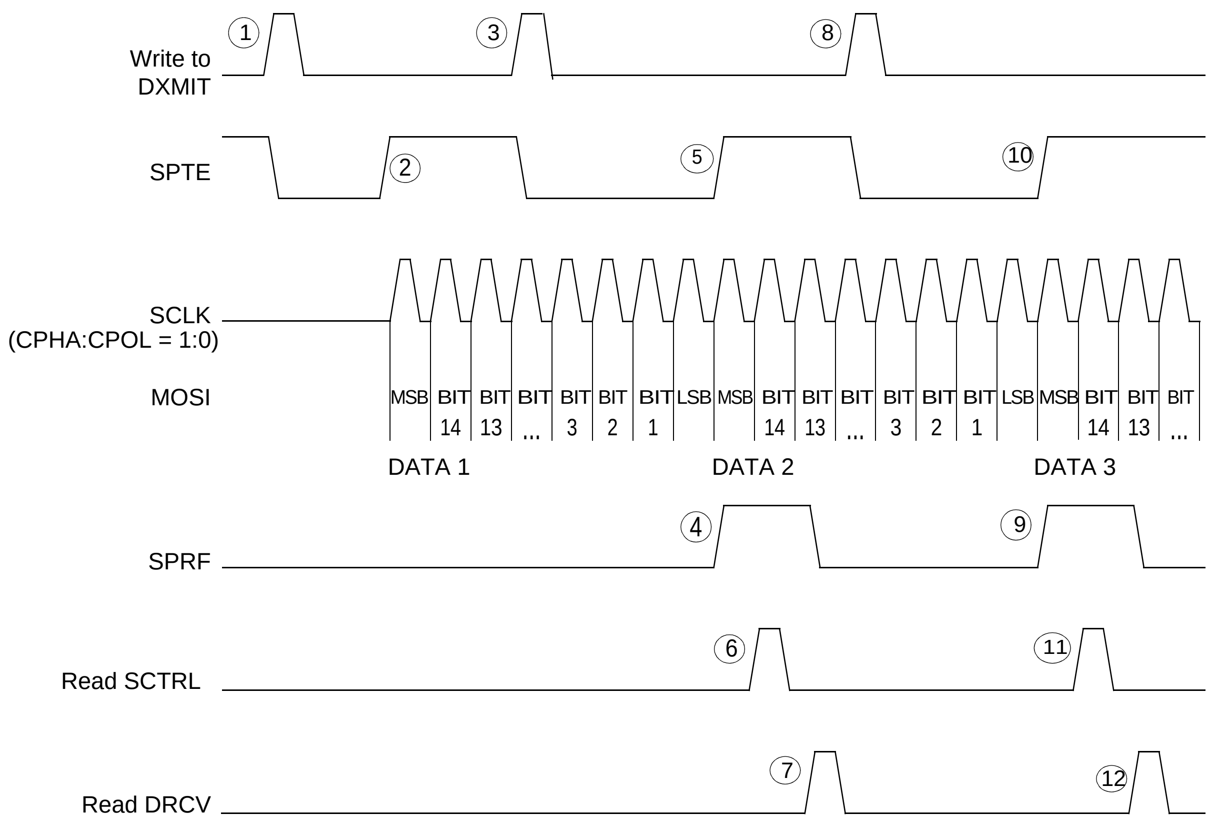


Figure 12-7. Transmission Start Delay (Master)

12.7 Transmission Data

The double-buffered DXMIT register allows data to be queued and transmitted. For a SPI configured as a master, the queued data is transmitted immediately after the previous transmission has completed. The SPTE flag indicates when the transmit data buffer is ready to accept new data. Write to the DXMIT register only when the SPTE bit is high. **Figure 12-8** illustrates the timing associated with doing back-to-back transmissions with the SPI (SCLK has CPHA: CPOL=1:0).

Note: **Figure 12-8** assumes 16-bit data lengths and the MSB shifted out first.


Write to DXMIT
SPTE
SCLK (CPHA:CPOL = 1:0)
MOSI
MSB BIT 14 BIT 13 BIT ... BIT 3 BIT 2 BIT 1 LSB MSB BIT 14 BIT 13 BIT ... BIT 3 BIT 2 BIT 1 LSB MSB BIT 14 BIT 13 BIT ...
DATA 1
DATA 2
DATA 3
SPRF
Read SCTRL
Read DRCV
1
2
3
4
5
6
7
8
9
10
11
12

1. 16-BIT CONTROLLER WRITES DATA 1 TO DXMIT, CLEARING THE SPTE BIT.
2. DATA 1 TRANSFERS FROM TRANSMIT DATA REGISTER TO SHIFT REGISTER, SETTING SPTE BIT.
3. 16-BIT CONTROLLER WRITES DATA 2 TO DXMIT, QUEUEING DATA2 AND CLEARING SPTE BIT.
4. FIRST INCOMING WORD TRANSFERS FROM THE SHIFT REGISTER TO THE DRCV REGISTER, SETTING THE SPRF BIT.
5. DATA 2 TRANSFERS FROM THE DXMIT REGISTER TO SHIFT REGISTER, SETTING THE SPTE BIT.
6. 16-BIT CONTROLLER READS SCTRL WITH THE SPRF BIT SET.
7. 16-BIT CONTROLLER READS DRCV, CLEARING SPRF BIT.
8. 16-BIT CONTROLLER WRITES DATA3 TO DXMIT, QUEUEING DATA3 AND CLEARING THE SPTE BIT.
9. SECOND INCOMING DATA TRANSFERS FROM SHIFT REGISTER TO DRCV REGISTER SETTING THE SPRF BIT.
10. DATA3 TRANSFERS FROM THE DXMIT REGISTER TO THE SHIFT REGISTER, SETTING THE SPTE BIT.
11. 16-BIT CONTROLLER READS SCTRL WITH THE SPRF BIT.
12. 16-BIT CONTROLLER READS DRCV, CLEARING THE SPRF BIT.

Figure 12-8. SPRF/SPTE Interrupt Timing

The transmit data buffer permits back-to-back transmissions without the slave precisely timing its writes between transmissions as is necessary in a system with a single data buffer. Also, if no new data is written to the data buffer, the last value contained in the Shift Register is the next data to be transmitted.

For an idle master without data loaded into its transmit buffer and no word currently being transmitted, the SPTE is set a gain no more than two bus cycles after DXMIT is written. This allows the user to queue up at most a 32-bit value to send. For a SPI operating in slave mode, the load of the Shift Register is controlled by the external master and back-to-back writes to the DXMIT register are not possible. The SPTE bit will indicate when the next write can occur.

12.8 Error Conditions

The following flags signal SPI error conditions:

- Overflow (OVRF)—Failing to read the Data register before the next full length data enters the Shift Register sets the OVRF bit. The new data will not transfer to the DRCV register, and the unread data can still be read. OVRF is in the SCTRL register.
- Mode Fault Error (MODF)—The MODF bit indicates the voltage on the Slave Select pin ($\overline{\text{SS}}$) is inconsistent with SPI mode. MODF is in the SCTRL register.

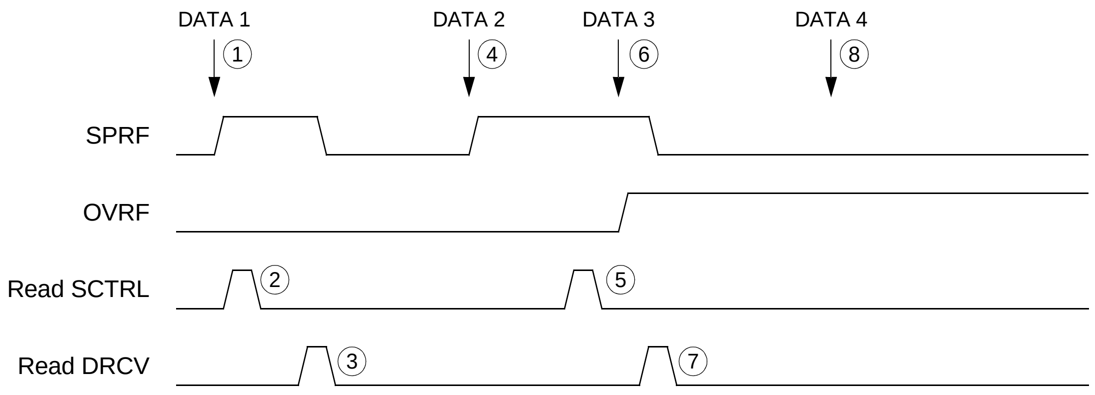
12.8.1 Overflow Error

The Overflow Flag (OVRF) becomes set if the last bit of a current transmission is received and the DRCV register still has unread data from a previous transmission. If an overflow occurs, all data received after the overflow, and before the OVRF bit is cleared, does not transfer to the DRCV register. It does not set the SPI Receiver Full (SPRF) bit. The unread data transferred to the DRCV register before the overflow occurred can still be read. Therefore, an overflow error always indicates the loss of data. Clear the overflow flag by reading the SCTRL register, and then read the DRCV register.

OVRF generates a receiver/error interrupt request if the Error Interrupt Enable (ERRIE) bit is also set. It is not possible to enable MODF to generate a receiver/error interrupt request while describing OVRF. However, leaving MODFEN low prevents MODF from being set.

If the SPI Receiver Full/Error (SPRF) bit interrupt is enabled and the Overflow Flag (OVRF) interrupt is not (ERRIE=0), watch for an overflow condition. **Figure 12-9** explains how it is possible to miss an overflow. The first element of the same figure illustrates how it is possible to read the SCTRL and DRCV registers to clear the SPI Receiver Full (SPRF) without problems. However, as illustrated by the second transmission example, the Overflow Flag (OVRF) bit can be set between the time SCTRL and DRCV registers are read.

In this case, an overflow can easily be missed. Since no more SPI Receiver Full (SPRF) bit interrupts can be generated until this Overflow Flag (OVRF) is serviced, it is not obvious data is being lost as more transmissions are completed. To prevent this loss, either enable the Overflow Flag (OVRF) interrupt (ERRIE=1) or take another read of the SCTRL register following the read of the DRCV register. This ensures the Overflow Flag (OVRF) was not set before the Receiver Full (SPRF) bit was cleared and future transmissions can set the SPRF bit.



1. DATA 1 SETS SPRF BIT.
2. 16-BIT CONTROLLER READS SCTRL WITH SPRF BIT SET AND OVRF BIT CLEAR.
3. 16-BIT CONTROLLER READS DATA 1 IN DRCV, CLEARING SPRF BIT.
4. DATA 2 SETS SPRF BIT.
5. 16-BIT CONTROLLER READS SCTRL WITH SPRF BIT SET AND OVRF BIT CLEAR.
6. DATA 3 SETS OVRF BIT. DATA 3 IS LOST.
7. 16-BIT CONTROLLER READS DATA 2 IN DRCV, CLEARING SPRF BIT BUT NOT THE OVRF BIT.
8. DATA 4 FAILS TO SET SPRF BIT BECAUSE OVRF BIT IS NOT CLEARED. DATA 4 IS LOST.

Figure 12-9. Missed Read of Overflow Condition

Figure 12-10 illustrates the described process. Generally, to avoid a second SCTRL register read, enable the Overflow Flag (OVRF) by setting the Error Interrupt Enable (ERRIE) bit.

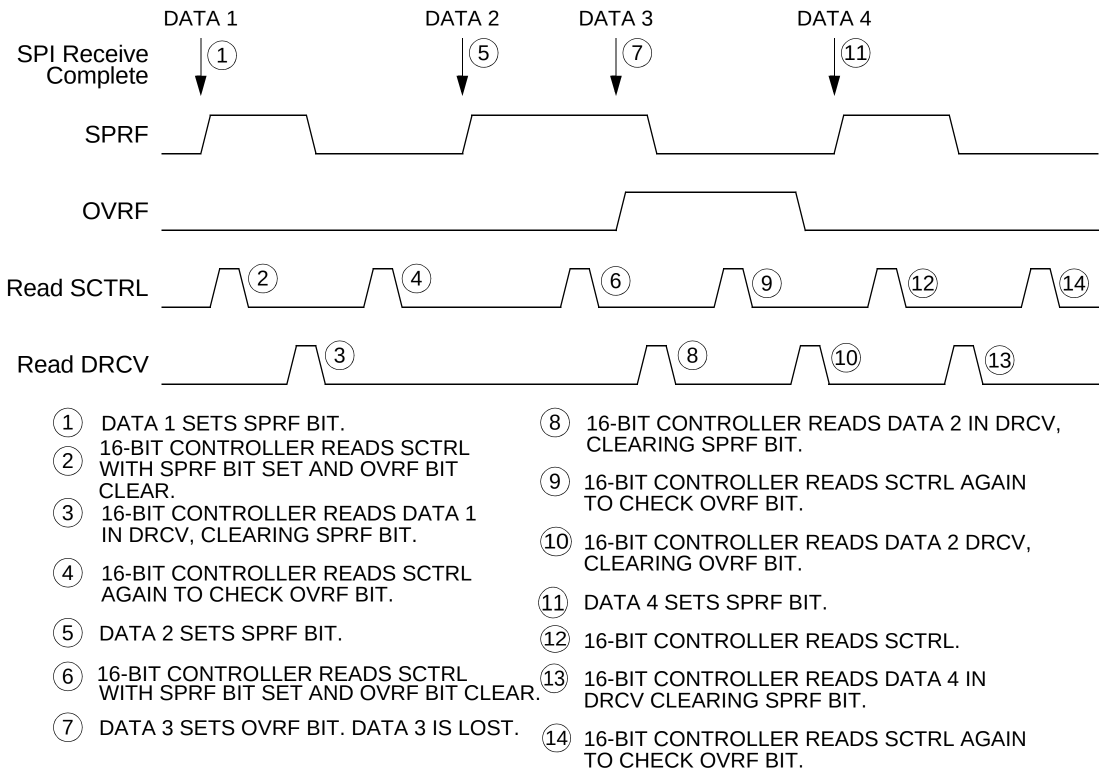

DATA 1
DATA 2
DATA 3
DATA 4
SPI Receive Complete
SPRF
OVRF
Read SCTRL
Read DRCV
1 DATA 1 SETS SPRF BIT.
2 16-BIT CONTROLLER READS SCTRL WITH SPRF BIT SET AND OVRF BIT CLEAR.
3 16-BIT CONTROLLER READS DATA 1 IN DRCV, CLEARING SPRF BIT.
4 16-BIT CONTROLLER READS SCTRL AGAIN TO CHECK OVRF BIT.
5 DATA 2 SETS SPRF BIT.
6 16-BIT CONTROLLER READS SCTRL WITH SPRF BIT SET AND OVRF BIT CLEAR.
7 DATA 3 SETS OVRF BIT. DATA 3 IS LOST.
8 16-BIT CONTROLLER READS DATA 2 IN DRCV, CLEARING SPRF BIT.
9 16-BIT CONTROLLER READS SCTRL AGAIN TO CHECK OVRF BIT.
10 16-BIT CONTROLLER READS DATA 2 DRCV, CLEARING OVRF BIT.
11 DATA 4 SETS SPRF BIT.
12 16-BIT CONTROLLER READS SCTRL.
13 16-BIT CONTROLLER READS DATA 4 IN DRCV CLEARING SPRF BIT.
14 16-BIT CONTROLLER READS SCTRL AGAIN TO CHECK OVRF BIT.

Figure 12-10. Clearing SPRF When OVRF Interrupt Is Not Enabled

12.8.2 Mode Fault Error

Setting the SPMSTR bit selects Master mode, configuring the SCLK and MOSI pins as outputs and the MISO pin as an input. Clearing SPMSTR selects Slave mode, configuring the SCLK and MOSI pins as inputs and the MISO pin as an output. The Mode Fault (MODF) bit becomes set any time the state of the $\overline{\text{SS}}$ pin is inconsistent with the mode selected by SPMSTR provided MODFEN=1. To prevent SPI pin contention and damage to the 16-bit controller, a Mode Fault Error occurs if:

- The $\overline{\text{SS}}$ pin of a slave SPI goes high during a transmission
- The $\overline{\text{SS}}$ pin of a master SPI goes low at any time

To set the MODF flag, Mode Fault Error Enable (MODFEN) bit must be set. Clearing the MODFEN bit does not clear the MODF flag but it does prevent the MODF from being set again after the MODF is cleared.

MODF generates a Receiver Full/Error interrupt request if the Error Interrupt Enable (ERRIE) bit is also set. It is not possible to enable MODF to generate a receiver/error interrupt request while disabling OVRF. However, leaving MODFEN low prevents MODF from being set.

12.8.2.1 Master SPI Mode Fault

In a master SPI with Mode Fault Enable (MODFEN) bit set, Mode Fault (MODF) flag is set if $\overline{SS}$ goes to Logic 0. A Mode Fault in a Master SPI causes the following events to occur:

- If ERRIE = 1, the SPI generates a SPI Receiver Full/Error interrupt request.
- The SPE bit is cleared
- The SPTE bit is set
- The SPI state counter is cleared

In a master SPI, the MODF flag will not be cleared until the $\overline{SS}$ pin is at Logic 1 or the SPI is configured as a slave.

Note: When CPHA=0, a MODF occurs if a slave is selected ($\overline{SS}$ is at Logic 0) and later unselected ($\overline{SS}$ is at Logic 1) even if no SCLK is sent to that slave. This happens because $\overline{SS}$ at Logic 0 indicates the start of the transmission (MISO driven out with the value of MSB) for CPHA=0. When CPHA=1, a slave can be selected and then later unselected with no transmission occurring. Therefore, MODF does not occur since a transmission was never begun.

12.8.2.2 Slave SPI Mode Fault

In a slave SPI (SPMSTR=0), the MODF bit generates a SPI Receiver/Error Interrupt request if the Error Interrupt Enable (ERRIE) bit is set. The MODF bit does not clear the SPI Enable (SPE) bit or reset the SPI in any way. Software can abort the SPI transmission by clearing the SPE bit of the slave.

Note: A Logic 1 voltage on the $\overline{SS}$ pin of a slave SPI puts the MISO pin in a high impedance state. Also, the slave SPI ignores all incoming SCLK clocks, even if it was already in the middle of a transmission.

When configured as a slave (SPMSTR=0), the MODF flag is set if the $\overline{SS}$ goes high during a transmission. When CPHA=0, a transmission begins when $\overline{SS}$ goes low and ends once the incoming SCLK goes back to its idle level, following the shift of the last data bit. When CPHA=1, the transmission begins when the SCLK leaves its idle level and $\overline{SS}$ is already low. The transmission continues until the SCLK returns to its idle level following the shift of the last data bit.

To clear the MODF flag, write a one to the bit field in the SCTRL register. The mode flag will not clear if the triggering condition continues.

In a slave SPI, if the MODF flag is not cleared by writing 1 to the MODF bit, the SPI Receiver Full/Error interrupt will continue to fire. In this case, the interrupt caused by the MODF flag can be cleared by disabling the EERIE or MODFEN bits (if set) or by disabling the SPI. Disabling the

SPI using the SPI Enable (SPE) bit will cause a partial reset of the SPI and may cause the loss of a message currently being received or transmitted.

12.9 Register Definitions

Table 12-3. SPI Memory Map

Device	Peripheral	Base Address
56F801X	SPI	$00F0C0

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 12-4 lists the SPI registers in ascending address order, including their acronyms and address offset of each register. The read/write registers should be accessed only with word accesses to avoid undefined results. The SPI peripheral has four registers.

Table 12-4. SPI Register Summary

Address Offset	Acronym	Register Name	Access Type	Location
Base + $0	SCTRL	Status and Control Register	Read/Write	**Section 12.9.1**
Base + $1	DSCTRL	Data Size and Control Register	Read/Write	**Section 12.9.2**
Base + $2	DRCV	Data Receive Register	*Read-Only*	**Section 12.9.3**
Base + $3	DXMIT	Data Transmit Register	*Write-Only*	**Section 12.9.4**

Bits of each of the four registers are summarized in **Figure 12-11.** Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0	SCTRL	R	SPR			DSO	ERRIE	MOD FEN	SPRIE	SPMSTR	CPOL	CPHA	SPE	SPTIE	SPRF	OVRF	MODF	SPTE
		W													▨	▨	▨	▨
$1	DSCTRL	R	WOM	0	0	0	0	0	0	0	0	0	0	0	DS3	DS2	DS1	DS0
		W		▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				
$2	DRCV	R	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
		W	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
$3	DXMIT	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		W	T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

R	0	Read as 0
W	▨	Reserved

Figure 12-11. SPI Register Map Summary

12.9.1 Status and Control Register (SCTRL)

The SPI Status and Control register:

- Selects master SPI baud rate
- Determines data shift order
- Enables SPI module interrupt requests
- Configures SPI module as master or slave
- Selects serial clock polarity and phase
- Indicates the SPI status

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	SPR			DSO	ERRIE	MODFEN	SPRIE	SPMSTR	CPOL	CPHA	SPE	SPTIE	SPRF	OVRF	MODF	SPTE
Write													▨	▨	▨	▨
Reset	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1

Figure 12-12. Status and Control (SCTRL) Register

Note: Using BFCLR or BFSET instructions on the SCTRL register can cause unintended side effects on the status bits. This is due to these instructions being a read/write modify instruction.

12.9.1.1 SPI Baud Rate Select (SPR)—Bits 15–13

While in the Master mode, these read/write bits select one of eight baud rates depicted in **Table 12-5**. SPR[2:0] have no effect in Slave mode. Use the formula below to calculate the SPI baud rate.

$$\text{Baud Rate} = \frac{\text{Module Clock}}{\text{Baud Rate Divisor}}$$

Table 12-5. SPI Master Baud Rate Selection

SPR[2:0]	Baud Rate Divisor (BD)
000	2
001	4
010	8
011	16
100	32
101	64
110	128
111	256

Note: The maximum data transmission rate for the SPI is typically limited by the bandwidth of the I/O drivers on the chip. Limits for the 56F801X are normal at 20MHz and 650kHz for Wired OR. These apply to both Master and Slave modes. The BD field needs to be set to keep the module within these ranges.

12.9.1.1.1 SPR Example

If you have a SPI device with a maximum baud rate of 1MHz and the SPI module clock is 32MHz, then the baud rate divisor must be greater than, or equal to, 32 (32MHz/1MHz). According to **Table 12-5,** the smallest baud rate divisor valid for this example would be 32. Thus, SPR[2:0] would be 4. Then the actual baud rate for this example would be 10MHz (32MHz/32).

12.9.1.2 Data Shift Order (DSO)—Bit 12

This read/write bit determines whether the MSB or LSB is transmitted or received first. Both Master and Slave SPI modules must transmit and receive the same length packets. Regardless how this bit is set, when reading from the Data Receive (DRCV) register or writing to the Data Transmit (DXMIT) register, the LSB will always be at bit location zero. If the data length is less than 16 bits, the data will be zero padded on the upper bits.

- 0 = MSB transmitted first
- 1 = LSB transmitted first

12.9.1.3 Error Interrupt Enable (ERRIE)—Bit 11

This read/write bit enables the MODF, if MODFEN is also set, and OVRF bits to generate interrupt requests.

- 0 = MODF and OVRF cannot generate interrupt requests
- 1 = MODF and OVRF can generate interrupt requests

12.9.1.4 Mode Fault Enable (MODFEN)—Bit 10

This read/write bit, when set to 1, allows the MODF flag to be set. If the MODF flag is set, clearing the MODFEN does not clear the MODF flag.

If the MODFEN bit is low, the level of the $\overline{\text{SS}}$ pin does not affect the operation of an enabled SPI configured as a master. For an enabled SPI configured as a slave, having MODFEN low only prevents the MODF flag from being set. It does not affect any other part of SPI operation.

12.9.1.5 SPI Receiver Interrupt Enable (SPRIE)—Bit 9

This read/write bit enables interrupt requests generated by the SPI Receiver Full (SPRF) bit. The SPRF bit is set when data transfers from the Shift Register to the DRCV register.

- 0 = SPRF interrupt requests disabled
- 1 = SPRF interrupt requests enabled

12.9.1.6 SPI Master (SPMSTR)—Bit 8

This read/write bit selects Master or Slave modes operation.

- 0 = Slave mode
- 1 = Master mode

12.9.1.7 Clock Polarity (CPOL)—Bit 7

This read/write bit determines the logic state of the SCLK pin between transmissions. To transmit data between SPI modules, the SPI modules must have identical CPOL values. Please see **Figure 12-4** and **Figure 12-6.**

12.9.1.8 Clock Phase (CPHA)—Bit 6

This read/write bit controls the timing relationship between the serial clock and SPI data. Please see **Figure 12-4** and **Figure 12-6.** To transmit data between SPI modules, the SPI modules must have identical CPHA values. When CPHA=0, the $\overline{\text{SS}}$ pin of the Slave SPI module must be set to Logic 1 between data transmissions, illustrated in **Figure 12-5.**

12.9.1.9 SPI Enable (SPE)—Bit 5

This read/write bit enables the SPI module. Clearing SPE causes a partial reset of the SPI. When setting/clearing this bit, *no* other bits in the SCTRL register should be changed. Failure to follow this statement may result in spurious clocks.

- 0 = SPI module disabled
- 1 = SPI module enabled

12.9.1.10 SPI Transmit Interrupt Enable (SPTIE)—Bit 4

This read/write bit enables interrupt requests generated by the Transmitter Empty (SPTE) bit. SPTE is set when data transfers from the DXMIT register to the Shift Register.

- 0 = SPTE interrupt requests disabled
- 1 = SPTE interrupt requests enabled

12.9.1.11 SPI Receiver Full (SPRF)—Bit 3

This *read-only* flag is set each time data transfers from the Shift Register to the DRCV register. SPRF generates an interrupt request if the Receiver Interrupt Enable (SPRIE) bit in the SCTRL register is set. This bit is cleared by reading the DRCV register.

- 0 = Data Receive (DRCV) register not full
- 1 = Data Receive (DRCV) register full

12.9.1.12 Overflow (OVRF)—Bit 2

This *read-only* flag is set if software does not read the data in the DRCV register before the next full data enters the Shift Register. In an overflow condition, the data already in the DRCV register is unaffected, and the data shifted in last is lost. Clear the Overflow (OVRF) bit by reading the SCTRL register and then reading the DRCV register. OVRF generates a Receiver/Error interrupt if the Error Interrupt Enable (EERIE) bit is set. See **Section 12.8.1** for more details.

- 0 = No overflow
- 1 = Overflow

12.9.1.13 Mode Fault (MODF)—Bit 1

This *read-only* flag is set in a slave SPI if the $\overline{\text{SS}}$ pin goes high during a transmission with the MODFEN bit set. In a master SPI, the MODF flag is set if the $\overline{\text{SS}}$ pin goes low at any time with the MODFEN bit set. Clear the MODF bit by writing a one to the MODF bit when it is set. MODF generates a Receive/Error interrupt if the EERIE bit is set. Please see **Section 12.8.2** for more details.

- 0 = $\overline{\text{SS}}$ pin at appropriate Logic level
- 1 = $\overline{\text{SS}}$ pin at inappropriate Logic level

12.9.1.14 SPI Transmitter Empty (SPTE)—Bit 0

This *read-only* flag is set each time the DXMIT register transfers data into the Shift Register. SPTE generates an interrupt request if the Transmit Interrupt Enable (SPTIE) bit in the SCTRL register is set. This bit is cleared by writing to the DXMIT.

- 0 = DXMIT register not empty
- 1 = DXMIT register empty

Note: Do not write to the DXMIT register unless the SPTE bit is high.

12.9.2 Data Size and Control Register (DSCTRL)

This read/write register determines the data length for each transmission. The master and slave must transfer the same size data on each transmission. A new value will only take effect at the time the SPI is enabled (SPE bit in SCTRL set from zero to one). In order to have a new value take effect, disable then re-enable the SPI with the new value in the register. The DSCTRL:

- Enables Wired OR mode on MISO and MOSI
- Configures the size of the transmission

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	WOM	0	0	0	0	0	0	0	0	0	0	0	DS3	DS2	DS1	DS0
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Figure 12-13. Data Size and Control (DSCTRL) Register

12.9.2.1 Wired OR Mode (WOM)—Bit 15

This control bit is used to select the nature of the SPI pins. When the WOM bit is set, the SPI pins are configured as open-drain drivers with the pull-ups disabled. When the WOM bit is cleared, the SPI pins are configured as push-pull drivers.

- 0 = Wired OR mode disabled
- 1 = Wired OR mode enabled

12.9.2.2 Reserved—Bits 14-4

These bits are reserved or not implemented. They are read as 0 and cannot be modified by writing.

12.9.2.3 Data Size (DS)—Bits 3-0

Please see **Table 12-6** for detailed transmission data.

Table 12-6. Data Size

DS[3:0]	Size of Transmission
$0	Not Allowed
$1	2 Bits
$2	3 Bits
$3	4 Bits
$4	5 Bits
$5	6 Bits
$6	7 Bits
$7	8 Bits
$8	9 Bits
$9	10 Bits
$A	11 Bits
$B	12 Bits
$C	13 Bits
$D	14 Bits
$E	15 Bits
$F	16 Bits

12.9.3 Data Receive Register (DRCV)

This *read-only* register will show the last data word received after a complete transmission. The SPRF bit is set when new data is transferred to this register.

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 12-14. Data Receive (DRCV) Register

12.9.4 Data Transmit Register (DXMIT)

This *write-only* register holds data to be transmitted. When the Transmitter Empty (SPTE) bit is set, new data should be written to this register. If new data is not written while in the Master mode, a new transaction will not be initiated until this register is written. When in Slave mode, the old data will be re-transmitted. All data should be written with the LSB at bit zero.

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Write	T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 12-15. Data Transmit (DXMIT) Register

12.10 Interrupts

Four SPI status flags can be enabled to generate interrupt requests.

Table 12-7. SPI Interrupts

Flag	Interrupt Enabled By	Description
SPTE (Transmitter Empty)	SPI Transmitter Interrupt Enable (SPTIE=1,SPE=1)	The SPI Transmitter Interrupt Enable (SPTIE) bit enables the SPTE flag to generate transmitter interrupt requests provided the SPI is enabled (SPE=1). The SPTE bit becomes set every time data transfers from the DXMIT to the Shift Register. The clearing mechanism for the SPTE flag is to write to the DXMIT register.
SPRF (Receiver Full)	SPI Receiver Interrupt Enable (SPRIE=1)	The SPI Receiver Interrupt Enable (SPRIE) bit enables the SPRF bit to generate receiver interrupt requests regardless of the state of the SPE bit. The SPRF is set every time data transfers from the Shift Register to the DRCV register. The clearing mechanism for the SPRF flag is to read the DRCV register.
OVRF (Overflow)	SPI Error Interrupt Enable (ERRIE=1)	The Error Interrupt Enable (ERRIE) bit enables both the MODF and OVRF bits to generate a receiver/error interrupt request.
MODF (Mode Fault)	SPI Error Interrupt Enable (ERRIE=1) Mode Fault Enable (MODFEN=1)	The Mode Fault Enable (MODFEN) bit can prevent the MODF flag from being set so only the OVRF bit is enabled by the ERRIE bit to generate receiver/error 16-bit controller interrupt requests.

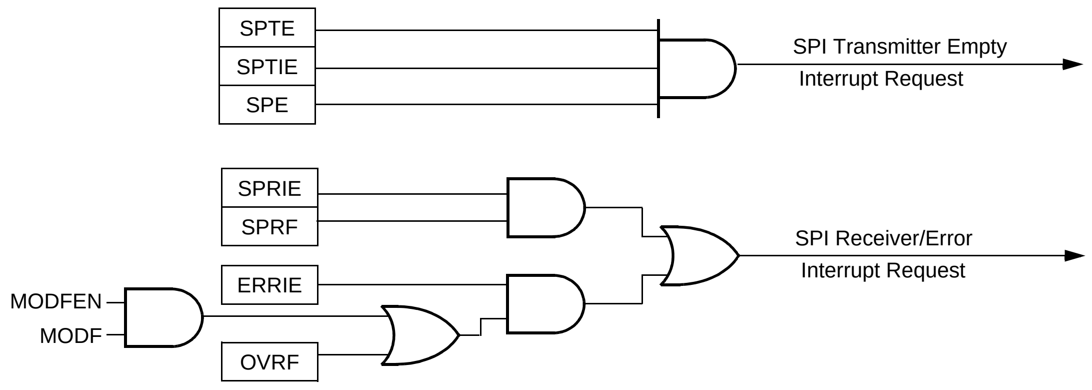


Figure 12-16. SPI Interrupt Request Generation

12.11 Resets

Any system reset completely resets the SPI. Partial resets occur whenever the SPI Enable (SPE) bit is low. Whenever SPE is low, the following will occur:

- SPTE flag is set
- Any Slave mode transmission currently in progress is aborted
- Any Master mode transmission currently in progress is continued to completion
- SPI state counter is cleared, making it ready for a new complete transmission
- All the SPI port Logic is disabled

The following items are reset only by a system reset:

- The DXMIT and DRCV registers
- All control bits in the SCTRL (MODFEN, ERRIE, SPR[2:0])
- The status flags SPRF, OVRF, and MODF

By not resetting the control bits when SPI Enable (SPE) is low, it is possible to clear SPE between transmissions without having to set all control bits again when SPE is set back high for the next transmission.

By not resetting the SPRF, OVRF, and MODF flags, it is possible to service the interrupts after the SPI has been disabled. Disable SPI by writing 0 to the SPI Enable (SPE) bit. SPI can also be disabled by a Mode Fault occurring in a SPI configured as a master.

Chapter 13
Quad Timer (TMR)

Document Revision History for Chapter 13, Quad Timer (TMR)

Version History	Description of Change
Rev. 0	Initial release
Rev.1	Clarification describing *PCS bits*, pg 23 deleting the word *input* after *Counter* in bullet desciptions Clarification describing *Count Once*, **Section 13.6.7.4** and *IPS bit* **Section 13.6.8.7**
Rev.2	Corrected typos on page 13-12
Rev. 4	In Section 13.6.7.4 (TMRCTRL[CM] bit description), replaced the text "*...the timer is stopped by changing the timer's Count Mode to Stop Mode (CM=0)*" with "*...the timer module changes its Count mode to Stop Mode (CM=0)*".

13.1 Introduction

The Quad Timer (TMR) module contains four identical counter/timer groups. Each 16-bit counter/timer group contains a:

- Prescaler
- Counter
- Load Register
- Hold Register
- Capture Register
- Two Compare Registers
- One Status and Control Register
- One Control Register

All of the registers are read/write capable.

Note: This document uses the terms *Timer* and *Counter* interchangeably because the counter/timers may perform either or both tasks.

The Load Register provides the initialization value to the counter when the counter's terminal value has been reached. The Hold Register captures the counter's value when other registers are being read. This feature supports the reading of cascaded counters. The Capture Register enables an external signal to take a snapshot of the counter's current value.

COMP1 and COMP2 registers provide values to which the counter is compared. If a match occurs, the OFLAG signal can be set, cleared, or toggled. At match time, an interrupt is generated if enabled, and the new compare value is loaded into COMP1 or COMP2 registers from CMPLD1 and CMPLD2 when enabled.

The Prescaler provides different time bases useful for clocking the counter/timer. The Counter provides the ability to count internal or external events. Input pins can be shared within a Timer module (set of four timer/counters).

13.2 Features

- Four 16-bit counters/timers
- Capable of counting up and down
- Counters will cascade
- Count modulo can be programmed
- Maximum count rate equals System Clock/2 for external clocks
- Maximum count rate equals System Clock, or 3× System Clock for internal clocks[1]
- Will count once or repeatedly
- Counters can be preloaded
- Counters can share available input pins
- Separate prescaler for each counter
- Each counter has capture and compare capability

13.3 Block Diagram

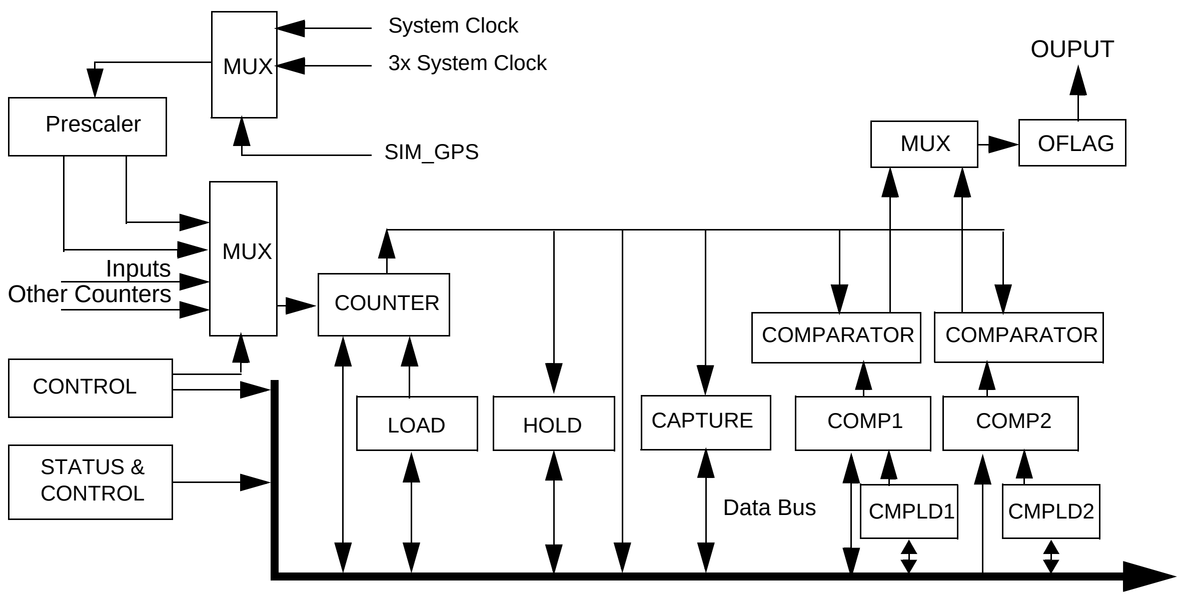


Figure 13-1. TMR Module Block Diagram

1. Tuner clock selection determined by TCR bit in SIM_GPS register.

13.4 Functional Description

The counter/timer has two basic modes of operation:

1. Count internal or external events
2. Count an internal clock source while an external input signal is asserted, or an external event has occurred, thereby timing the width of the external input signal, or the time between external events.

The counter can count the rising, falling, or both edges of the selected input pin. The counter can decode and count quadrature encoded input signals. The counter can also count up and down using dual inputs in a count with direction format. The counter's terminal count value (modulo) is programmable. The value loaded into the counter after reaching its terminal count is programmable. The counter can count repeatedly, or it can stop after completing one count cycle. The counter can be programmed to count to a programmed value before immediately re-initializing, or it can count through the compare value until the count rolls over to zero.

The external inputs to each counter/timer can be shared among each of the four counter/timers within the module. The external inputs can be used to:

- Generate count commands
- Generate timer commands
- Trigger current counter value to be captured
- Generate interrupt requests

The polarity of the external inputs can be selected. The primary output of each timer/counter is the output signal, OFLAG. The OFLAG output signal can be set, cleared, or toggled when the counter reaches the programmed value. The OFLAG output signal may be output to an external pin shared with an external input signal.

The OFLAG output signal enables each counter to generate square waves, PWM, or pulse stream outputs. The polarity of the OFLAG output signal is selectable.

Any counter/timer can be assigned as a master. A master's compare signal can be broadcast to the other counter/timers within the module. The other counters can be configured to reinitialize their counters and/or force their OFLAG output signals to predetermined values when a master's counter/timer compare event occurs.

13.4.1 Compare Registers Usage

The dual Compare registers (COMP1 and COMP2) provide a bidirectional modulo count capability.

- COMP1 register = used when the counter is *counting up*
- COMP2 register = used when the counter is *counting down*

Alternating Compare mode (OM = 100) is the only exception. (See below.)

The COMP1 register should be set to the desired maximum count value, or 0xFFFF, indicating the maximum unsigned value prior to roll-over. The COMP2 register should be set to the minimum count value, or 0x0000, indicating the minimum unsigned value prior to roll under.

If Output Mode is set to 100, the OFLAG toggles while using alternating Compare registers. In this variable frequency PWM mode, the COMP2 value defines the desired pulse width of the ON time. The COMP1 register defines the OFF time. The variable frequency PWM mode is defined for positive counting only.

Note: Use caution when changing COMP1 and COMP2 while the counter is active. If the counter has already passed the new value, it will count to 0xFFFF or 0x0000, roll-over, then begin counting toward the new value.

The check is: Count = COMP*n*, *not* Count > COMP1 or Count < COMP2.

The use of the CMPLD1 and CMPLD2 registers to preload compare values will help to minimize this problem.

13.4.2 Comparator Load Registers

The CMPLD1, CMPLD2, and CSCTRL offers a high degree of flexibility for loading Compare registers with user-defined values on different compare events. To ensure correct functionality while using these registers, it is recommended to use the method described in this section.

The purpose of the Comparator Load feature is to allow quicker updating of the compare registers. In prior families, a compare register could be updated using interrupts. However, because of the latency between an interrupt event occurring and the service of that interrupt, there is the possibility the counter may have already counted past the new compare value by the time the Compare register is updated by the interrupt service routine. The counter would then continue counting until it rolled over and reached the new compare value.

To address this, compare registers are updated in hardware in the same manner the Counter register is re-initialized to the value stored in the Load register. The Comparator Load feature allows calculation of new compare values to be stored in the Comparator Load registers. When a compare event occurs, the new compare values in the Comparator Load registers are written to the Compare registers, eliminating a software requirement to accomplish this.

The Compare Load feature is intended to be used in a variable frequency PWM mode. The COMP1 register determines the pulse width for the Logic Low part of OFLAG while COMP2 determines the pulse width for the Logic High part of OFLAG. The period of the waveform is determined by the sum of COMP1 and COMP2 values and the frequency of the primary clock source. Please see **Figure 13-2**.

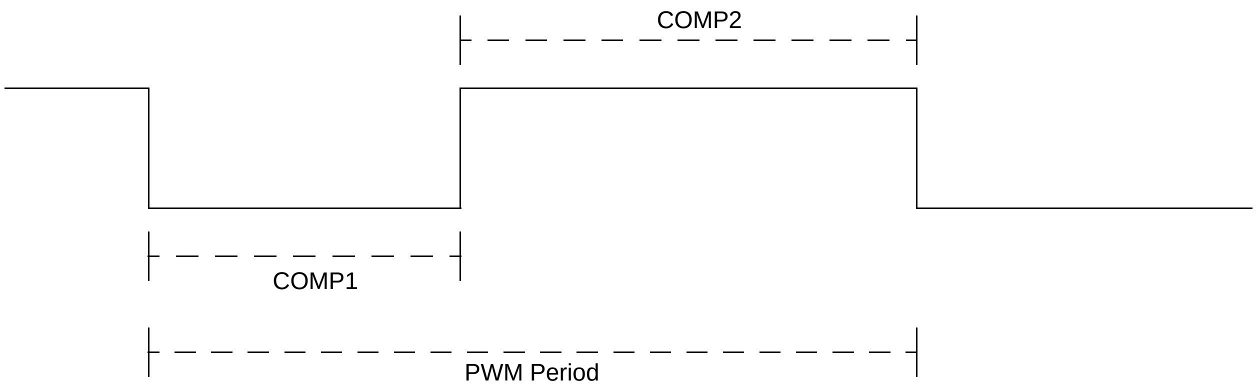


Figure 13-2. Variable PWM Waveform

If there is a desire to update the duty cycle or period of the above waveform, it is necessary to update the COMP1 and COMP2 values using the Compare Load feature.

13.4.3 Capture Register Usage

The Capture (CAPT) register stores a copy of the counter's value when an input edge (positive, negative, or both) is detected on the Secondary Count Source. Once a capture event occurs, no further updating of the Capture register will occur until the Input Edge Flag (IEF) is cleared by writing 0 to it.

13.5 Operating Modes

The various counting modes are detailed in the following subsections.

Selected external count signals are sampled at the TMR's base clock rate (peripheral clock), then run through a transition detector. The maximum count rate is one-half of the TMR's base clock rate. Internal clock sources can be used to clock the counters at the TMR's clock rate. These signals are all at the same clock rate.

If a counter is programmed to count to a specific value and then stop, Count Mode in the Timer Control (CTRL) register is cleared when the count terminates.

13.5.1 Stop Mode

When the Count Mode field is set to 000 in the control register (see **Section 13.6.7.1**), the counter is inert. No counting will occur.

13.5.2 Count Mode

When the Count mode field is set to 001 (see **Section 13.6.7.1**), the counter will count the rising edges of the selected clock source. This mode is useful for generating periodic interrupts for timing purposes, or counting external events such as *widgets* on a conveyor belt passing a sensor. If the selected input is inverted by setting the Input Polarity Select (IPS) bit (see **Section 13.6.8.7**), the negative edge of the selected external input signal is counted.

Example:

- Set the Control (CTRL) register PCS bits to desired input
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Control (CTRL) register CM bits to 001

In this example, the timer will count events from primary source and measure the counter register for every successful event detection.

Example:

- Set the Control (CTRL) register PCS bits to 1000
- Clear the Status and Control (SCTRL) register
- Set the Load (LOAD) register to 0x0000
- Set the Counter (CNTR) register to 0x0000
- Set the Compare 1 (COMP1) register to 25000
- Set the Comparator Load 1 (CMPLD1) register to 25000
- Set the Comparator Status and Control (CSCTRL) register TCF1EN bit to 1
- Set the Comparator Status and Control (CSCTRL) register CL1 bit to 1
- Set the Control (CTRL) register CM bits to 001

13.5.3 Edge Count Mode

When the Count Mode field is set to 010 (see **Section 13.6.7.1**), the counter will count both edges of the selected external clock source. This mode is useful for counting the changes in the external environment such as a simple encoder wheel.

Example:

- Set the Control (CTRL) register PCS bits to desired input
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Control (CTRL) register CM bits to 010

13.5.4 Gated Count Mode

When the Count Mode field is set to 011 (see **Section 13.6.7.1**), the counter will count while the selected secondary input signal is high. This mode is used to time the duration of external events. If the selected input is inverted by setting the Input Polarity Select (IPS) bit, the counter will count while the selected secondary input is low.

Example:

- Set the Control (CTRL) register PCS bits to the desired input
- Set the Control (CTRL) register SCS bits to the desired input
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Control (CTRL) register CM bits to 011

13.5.5 Quadrature Count Mode

When the Count Mode field is set to 100 (see **Section 13.6.7.1**), the counter will decode the primary and secondary external inputs as quadrature encoded signals. Quadrature signals are usually generated by rotary or linear sensors used to monitor movement of motor shafts or mechanical equipment. The quadrature signals are square waves, 90° out-of-phase. The decoding of quadrature signal provides both count and direction information. A timing diagram illustrating the basic operation of a quadrature incremental position encoder is provided in **Figure 13-3**.

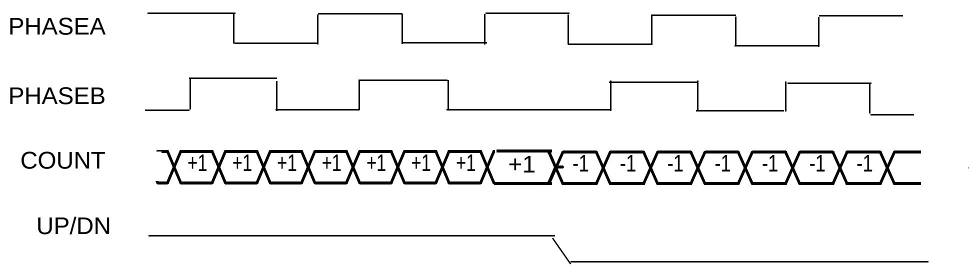


Figure 13-3. Quadrature Incremental Position Encoder

Example:

- Set the Control (CTRL) register PCS bits to 0000
- Set the Control (CTRL) register SCS bits to 01
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Compare 1 (COMP1) register to 0xFFFF
- Set the Compare 2 (COMP2) register to 0x0000
- Clear the Comparator Status and Control (CSCTRL) register
- Set the Control (CTRL) register CM bits to 100

13.5.6 Signed Count Mode

When the Count Mode field is set to 101 (see **Section 13.6.7.1**), the counter counts the primary clock source while the selected secondary source provides the selected count direction (up/down).

Example:

- Set the Control (CTRL) register PCS bits to the desired input
- Set the Control (CTRL) register SCS bits to the desired input
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Control (CTRL) register CM bits to 101

13.5.7 Triggered Count Mode

When the Count Mode field is set to 110 (see **Section 13.6.7.1**), the counter will begin counting the primary clock source after a positive transition (Negative Edge if IPS = 1) of the secondary input occurs. The counting will continue until a compare event occurs, or another positive input transition is detected. If a second input transition occurs before a terminal count is reached, counting will stop. Subsequent secondary input transitions will continue to cause the counting to restart and stop until a compare event occurs.

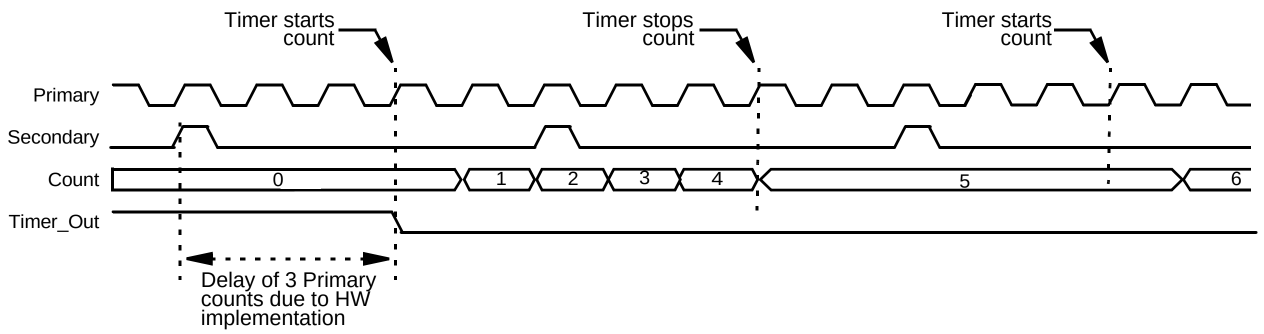


Figure 13-4. Triggered Count Mode (Length = 0)

Example:

- Set the Control (CTRL) register PCS bits to the desired input
- Set the Control (CTRL) register SCS bits to the desired input
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Compare 1 (COMP1) register to 0x0012
- Clear the Comparator Status and Control (CSCTRL) register
- Set the Control (CTRL) register CM bits to 110

13.5.8 One-Shot Mode

This is a sub-mode of Triggered Count Mode when Count Mode field is set to 110 (see **Section 13.6.7.1**) while:

- LENGTH is set (reinitialize on compare)
- The OFLAG Output Mode is set to 101 (Set OFLAG on compare, clear OFLAG on secondary input.)

In the above state, the counter works in a One-Shot mode. An external event causes the counter to count. When terminal count is reached, the OFLAG output is asserted. This delayed output assertion can be used to provide timing delays. When used with timer C2 or C3, this one-shot mode may be used to delay the ADC acquisition of new samples until a specified period of time has passed since a Pulse Width Modulated (PWM) module reload.

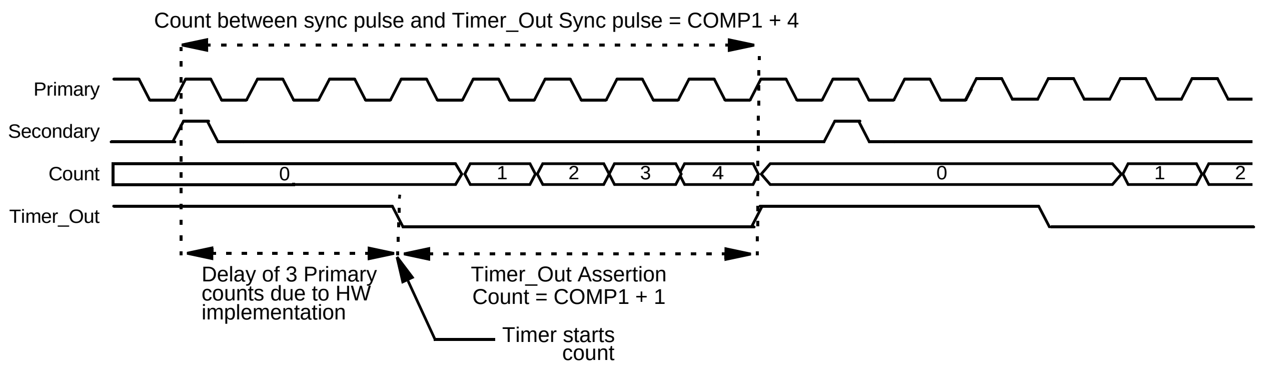

Count between sync pulse and Timer_Out Sync pulse = COMP1 + 4
Primary
Secondary
Count
Timer_Out
0
1
2
3
4
0
1
2
Delay of 3 Primary counts due to HW implementation
Timer_Out Assertion Count = COMP1 + 1
Timer starts count

Figure 13-5. One-Shot Mode (LOAD = 0, COMP1 = 4)

Example:

- Set the Control (CTRL) register PCS bits to the desired input
- Set the Control (CTRL) register SCS bits to the desired input
- Set the Control (CTRL) register LENGTH bit to 1
- Set the Control (CTRL) register OM bits to 101
- Clear the Status and Control (SCTRL) register
- Set the Counter (CNTR) register to 0x0000
- Set the Load (LOAD) register to 0x0000
- Set the Compare 1 (COMP1) register to the desired delay
- Clear the Comparator Status and Control (CSCTRL) register
- Set the Control (CTRL) register CM bits to 110

13.5.9 Cascaded Count Mode

When the Count Mode field is set to 111 (see **Section 13.6.7.1**), the counter's input is connected to the output of another selected counter. The counter will count up and down as compare events occur in the selected source counter. This cascaded, or daisy-chained mode, enables multiple counters to be cascaded in order to yield longer counter lengths. When operating in Cascaded Count Mode, a special high speed signal path is used between modules rather than the OFLAG output signal. If the selected source counter is counting up, and it experiences a compare event, the counter will be incremented. If the selected source counter is counting down and it experiences a compare event, the counter will be decremented. Up to four counters may be cascaded to create a 64-bit wide synchronous counter. Whenever any counter is read within a timer module, all of the timers' values within the module are captured in their respective Hold registers. This action supports the reading of a cascaded counter chain. First, read any counter of a cascaded counter chain, then read the Hold registers of the other counters in the chain. Cascaded count mode is synchronous.

Note: It is possible to connect counters together by using the other (non-cascade) counter modes and selecting the outputs of other counters as a clock source. In this case, the counters are operating in a *ripple* mode, where higher order counters will transition a clock later than a purely synchronous design.

Example:

- Set Timer2 Control (CTRL) register PCS bits to the desired input
- Set Timer2 Control (CTRL) register LENGTH bits to 01
- Set Timer3 Control (CTRL) register PCS bits to 110
- Set Timer3 Control (CTRL) register CM bits to 111
- Set Timer3 Control (CTRL) register LENGTH bits to 01
- Clear Status and Control (SCTRL) register for both timers
- Clear the Counter (CNTR) and Load (LOAD) registers from both timers
- Set Timer3 Comparator 1 (COMP1) register to 30000
- Set Timer3 Comparator Load 1 (CMPLD1) register to 30000
- Set Timer2 Comparator 1 (COMP1) register to 32000
- Set Timer2 Comparator Load 1 (CMPLD1) register to 32000
- Set Timer3 Comparator Status and Control (SCTRL) register to 0x0041
- Set Timer2 Comparator Status and Control (SCTRL) register to 0x0001
- Set Timer2 Control (CTRL) register CM bits to 001

13.5.10 Pulse Output Mode

The Counter will output a stream of pulses with the same frequency of the selected clock source (cannot be IPBus clock divided by 1) if the counter is setup for:

- Count Mode (CM = 001) (Count rising edges of primary source)
- The OFLAG Output Mode is set to 111 (Gated Clock Output)
- The Count Once bit is set (Count until compare and then stop)

The number of output pulses is equal to the compare value minus the initial value. This mode is useful for driving step motor systems.

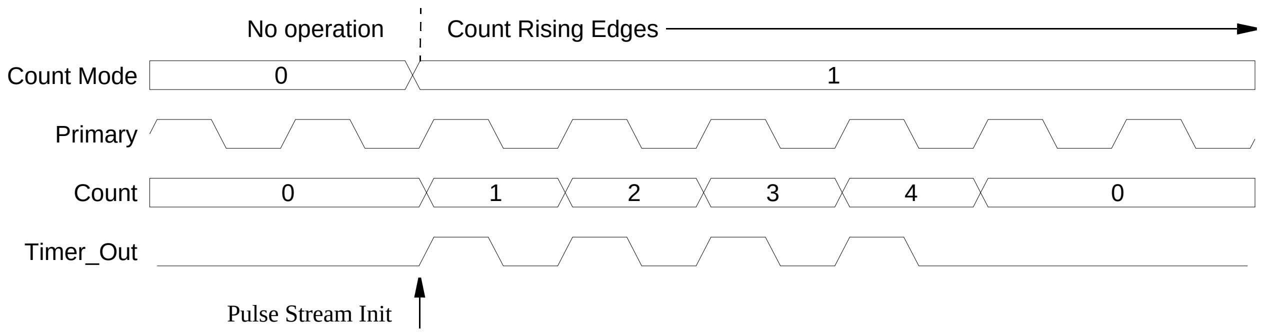


Figure 13-6. Pulse Output Mode (LOAD = 0, COMP1 = 4)

13.5.11 Fixed Frequency PWM Mode

The Counter output yields a Pulse Width Modulated (PWM) signal with a frequency equal to the count clock frequency, divided by 65,536. It has a pulse width duty cycle equal to the compare value divided by 65,536, if the counter is setup for:

- Count Mode (CM = 001) (Count rising edges of primary source)
- Count through roll-over (LENGTH = 0)
- Continuous count (ONCE = 0)
- OFLAG Output Mode is 110 (set on compare, cleared on counter roll-over)
- Output polarity inverted (OPS = 1)

This mode of operation is often used to drive PWM amplifiers used to power motors and inverters.

13.5.12 Variable Frequency PWM Mode

If the counter is setup for:

- Continuous count (ONCE = 0)
- Count Mode (CM = 001), count until compare (LENGTH = 1)
- OFLAG Output Mode is 100 (toggle OFLAG and alternate Compare registers)

The counter output then yields a Pulse Width Modulated (PWM) signal whose frequency and pulse width is determined by the values programmed into the COMP1 and COMP2 registers and the input clock frequency. This method of PWM generation has the advantage of allowing almost any desired PWM frequency and/or constant on or off periods. This mode of operation is often used to drive PWM amplifiers used to power motors and inverters. The CMPLD1 and CMPLD2 registers are especially useful for this mode because they permit time to calculate values for the next PWM cycle while the PWM current cycle is underway.

To setup the TMR to run in Variable Frequency PWM mode with compare preload please use the following setup for the specific counter you would like to use. When performing the setup, the CTRL register is suggested to be updated *last* because the counter will start counting if the Count mode is changed to any other value other than three bits with a value of 000, assuming the Primary Count Source is already active.

The following sections detail the timer settings required to implement variable frequency PWM operation.

13.5.12.1 Timer Control Register (CTRL)

For complete register details, please refer to **Section 13.6.7**.

- CM = 001 (count rising edges of primary source)
- PCS = 1000 (IPBus clock for best granularity for waveform timing)
- SCS = Any (ignored in this mode)
- ONCE = 0 (want to count repeatedly)
- LENGTH = 1 (want to count until compare value is reached and re-initialize counter register)
- DIR = Any (make a choice. The compare register values need to be chosen carefully to account for things like roll-under, etc.)
- CoINIT = 0 (can be set if this function is required)
- OM = 00 (toggle OFLAG output using alternating compare registers)

13.5.12.2 Timer Status and Control Register (SCTRL)

For complete register details, please refer to **Section 13.6.8**.

- OEN = 1 (output enable to allow OFLAG output to be put on an external pin. Set this bit as required)
- OPS = Any (make a choice)
- Make certain the rest of the bits are cleared for this register. Interrupts in the Comparator Status and Control (CSCTRL) register are enabled instead of this register.

13.5.12.3 Comparator Status and Control Register (CSCTRL)

For complete register details, please refer to **Section 13.6.11**.

- TCF2EN = 1 (allows interrupt to be issued when TCF2 is set)
- TCF1EN = 0 (does not allow interrupt to be issued when TCF1 is set)
- TCF1 = 0 (clear Timer Compare Flag 1. This is set when CNTR register equals COMP1 register value and OFLAG is low)
- TCF2 = 0 (clear Timer Compare Flag 2. This is set when CNTR register equals COMP2 register value and OFLAG is high)
- CL1[1:0] = 10 (Load Compare1 register with CMPLD1 register when TCF2 is asserted)
- CL2[1:0] = 01 (Load Compare2 register with CMPLD2 register when TCF1 is asserted)

13.5.12.4 Interrupt Service Routines

To service the TCF2 interrupts generated by the timer, the Interrupt Controller must be configured, enabling the interrupts for the particular timer being used. Additionally, it is necessary to write an interrupt service routine to do the following at a minimum:

- Clear the TCF2 and TCF1 flags
- Calculate and write new values for both CMPLD1 and CMPLD2

13.5.12.5 Timing

Figure 13-7 contains the timing to use the Compare Preload feature. The Compare Preload cycle begins with a compare event on COMP2, causing TCF2 to be asserted. COMP1 is loaded with the value in the CMPLD1, one IPBus clock later. Additionally, an interrupt is asserted by the timer and the interrupt service routine is executed. During this time both Comparator Load registers are updated with new values. When TCF1 is asserted, COMP2 is loaded with the value in CMPLD2. On the subsequent TCF2 event, COMP1 is loaded with the value in CMPLD1. The cycle starts over again as an interrupt is asserted and the interrupt service routine clears TCF1 and TCF2, calculating new values for CMPLD1 and CMPLD2.

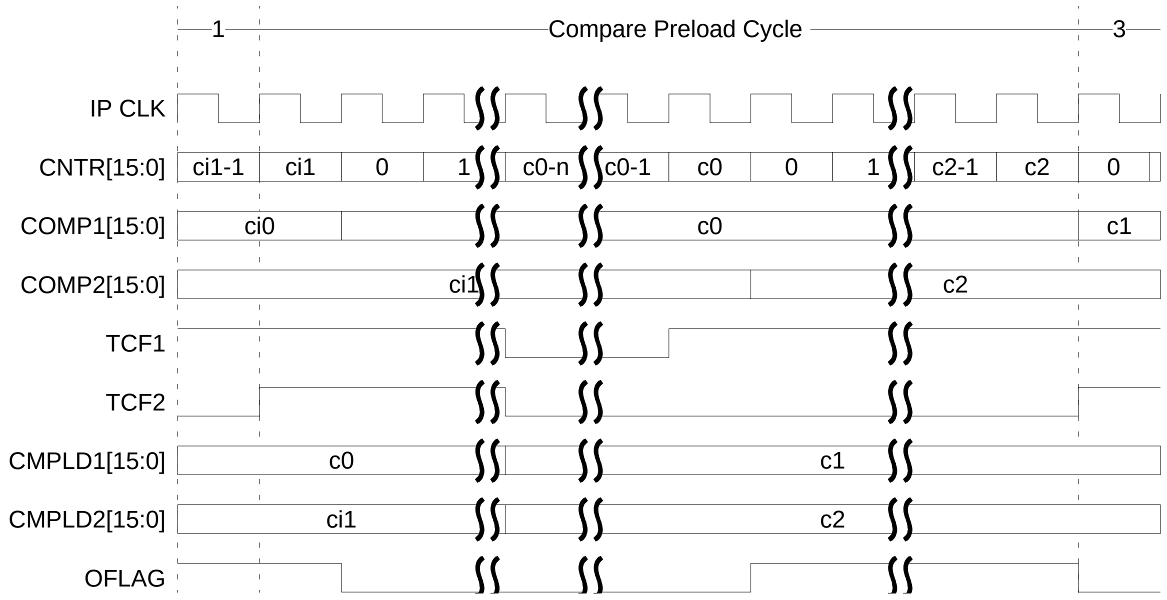


Figure 13-7. Compare Preload Timing

13.6 Register Definitions

Table 13-1. TMR Memory Map

Device	Peripheral	Base Address
56F801X	TMR0	$00F000
	TMR1	$00F010
	TMR2	$00F020
	TMR3	$00F030

A register address is the sum of a base address and an address offset. The base address is defined at the device level and the address offset is defined at the module level.

Table 13-2 lists the TMR registers in ascending address, including their acronyms and address offset of each of the 44 registers. Make certain to check which timer channels have external I/O.

Table 13-2. TMR Register Summary

Register Address Offsets	Register Acronym[1]	Register Name	Access Type	Chapter Location
Base + $0, $10, $20, $30	COMP1	Compare 1 Registers	Read/Write	Section 13.6.1
Base + $1, $11, $21, $31	COMP2	Compare 2 Registers	Read/Write	Section 13.6.2
Base + $2, $12, $22, $32	CAPT	Capture Registers	Read/Write	Section 13.6.3
Base + $3, $13, $23, $33	LOAD	Load Registers	Read/Write	Section 13.6.4
Base + $4, $14, $24, $34	HOLD	Hold Registers	Read/Write	Section 13.6.5
Base + $5, $15, $25, $35	CNTR	Counter Registers	Read/Write	Section 13.6.6
Base + $6, $16, $26, $36	CTRL	Control Registers	Read/Write	Section 13.6.7
Base + $7, $17, $27, $37	SCTRL	Status and Control Registers	Read/Write	Section 13.6.8
Base + $8, $18, $28, $38	CMPLD1	Comparator Load 1 Registers	Read/Write	Section 13.6.9
Base + $9, $19, $29, $39	CMPLD2	Comparator Load 2 Registers	Read/Write	Section 13.6.10
Base + $A, $1A, $2A, $3A	CSCTRL	Comparator Status and Control Register	Read/Write	Section 13.6.11

1. In the device's Data Sheet Timer acronyms are made globally unique with the addition of a prefix of the form TMR*x n,* where *x* = A, B, C . . . to identify multiple Quad Timers and *n* = 0, 1, 2, 3 . . . to identify individual timers within a Quad Timer.

Bits of each of the 44 registers are illustrated in **Figure 13-8**. Details of each follow.

Add. Offset	Register Acronym		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$0,$10, $20,$30	COMP1	R	COMPARISON 1															
		W																
$1,$11, $21,$31	COMP2	R	COMPARISON 2															
		W																
$2,$12, $22,$32	CAPT	R	CAPTURE															
		W																
$3,$13, $23,$33	LOAD	R	LOAD															
		W																
$4,$14, $24,$34	HOLD	R	HOLD															
		W																
$5,$15, $25,$35	CNTR	R	COUNTER															
		W																
$6,$16, $26,$36	CTRL	R	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
		W																
$7,$17, $27,$37	SCTRL	R	TCF	TCFIE	TOF	TOFIE	IEF	IEFIE	IPS	INPUT	CAPTURE MODE		MSTR	EEOF	VAL	0 ▨	OPS	OEN
		W								▨						FORCE		
$8,$18, $28,$38	CMPLD1	R	COMPARATOR LOAD 1															
		W																
$9,$19, $29,$39	CMPLD2	R	COMPARATOR LOAD 2															
		W																
$A,$1A, $2A,$3A	CSCTRL	R	0 ▨	0 ▨	0 ▨	0 ▨	0 ▨	0 ▨	0 ▨	0 ▨	TCF2 EN	TCF1 EN	TCF2	TCF1	CL2		CL1	
		W	▨	▨	▨	▨	▨	▨	▨	▨								

R	0 ▨	Read as 0
W	▨	Reserved

Figure 13-8. TMR Register Map Summary

13.6.1 Compare 1 Registers (COMP1)

These four read/write Timer Compare 1 (COMP1) registers store the value used for comparison with counter value. Their addresses are:

TMR0_COMP1 (Channel 0 Compare 1)—Address: TMR_BASE + $0
TMR1_COMP1 (Channel 1 Compare 1)—Address: TMR_BASE + $10
TMR2_COMP1 (Channel 2 Compare 1)—Address: TMR_BASE + $20
TMR3_COMP1 (Channel 3 Compare 1)—Address: TMR_BASE + $30

Base + $0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	COMPARISON 1															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-9. Compare 1 (COMP1) Register

13.6.2 Compare 2 Registers (COMP2)

These four read/write Timer Compare 2 (COMP2) registers store the value used for comparison with counter value. Their addresses are:

TMR0_COMP2 (Channel 0 Compare 2)—Address: TMR_BASE + $1
TMR1_COMP2 (Channel 1 Compare 2)—Address: TMR_BASE + $11
TMR2_COMP2 (Channel 2 Compare 2)—Address: TMR_BASE + $21
TMR3_COMP2 (Channel 3 Compare 2)—Address: TMR_BASE + $31

Base + $1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	COMPARISON 2															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-10. Compare 2 (COMP2) Registers

13.6.3 Capture Registers (CAPT)

These four read/write Timer Capture (CAPT) registers store the values captured from the counters. Their addresses are:

TMR0_CAPT (Channel 0 Capture)—Address: TMR_BASE + $2
TMR1_CAPT (Channel 1 Capture)—Address: TMR_BASE + $12
TMR2_CAPT (Channel 2 Capture)—Address: TMR_BASE + $22
TMR3_CAPT (Channel 3 Capture)—Address: TMR_BASE + $32

Please see **Section 13.6.8.9** for information on which inputs will generate a capture of the counter’s value.

Base + $2	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTURE															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-11. Capture (CAPT) Registers

13.6.4 Load Registers (LOAD)

These four read/write Timer Load Registers (LOAD) store the value used to load the counter. Their addresses are:

TMR0_LOAD (Channel 0 Load)—Address: TMR_BASE + $3
TMR1_LOAD (Channel 1 Load)—Address: TMR_BASE + $13
TMR2_LOAD (Channel 2 Load)—Address: TMR_BASE + $23
TMR3_LOAD (Channel 3 Load)—Address: TMR_BASE + $33

Base + $3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	LOAD															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-12. Load (LOAD) Registers

13.6.5 Hold Registers (HOLD)

These four read/write registers store the channel's value whenever any Timer Counter (CNTR) register is read. Their addresses are:

TMR0_HOLD (Channel 0 Hold)—Address: TMR_BASE + $4
TMR1_HOLD (Channel 1 Hold)—Address: TMR_BASE + $14
TMR2_HOLD (Channel 2 Hold)—Address: TMR_BASE + $24
TMR3_HOLD (Channel 3 Hold)—Address: TMR_BASE + $34

Base + $4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	HOLD															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-13. Hold (HOLD) Registers

13.6.6 Counter Registers (CNTR)

There are four read/write Timer Counter (CNTR) registers. Their addresses are:

TMR0_CNTR (Channel 0 Counter)—Address: TMR_BASE + $5
TMR1_CNTR (Channel 1 Counter)—Address: TMR_BASE + $15
TMR2_CNTR (Channel 2 Counter)—Address: TMR_BASE + $25
TMR3_CNTR (Channel 3 Counter)—Address: TMR_BASE + $35

Base +$5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	COUNTER															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-14. Counter (CNTR) Registers

13.6.7 Control Registers (CTRL)

There are four Timer Control (CTRL) registers. For details regarding the timer settings required to implement variable frequency PWM operation, please refer to **Section 13.5.12.1**. Their addresses are:

TMR0_CTRL (Channel 0 Control)—Address: TMR_BASE + $6
TMR1_CTRL (Channel 1 Control)—Address: TMR_BASE + $16
TMR2_CTRL (Channel 2 Control)—Address: TMR_BASE + $26
TMR3_CTRL (Channel 3 Control)—Address: TMR_BASE + $36

Base + $6	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-15. Control (CTRL) Registers

13.6.7.1 Count Mode (CM)—Bits 15–13

These bits control the basic counting behavior of the counter.

- 000 = Stop mode, no operation
- 001 = Counting mode, count Rising/Falling edges of Primary Source[1]
  — Rising edges if IPS = 0 (see **Section 13.6.8.7**)
  — Falling edges if IPS = 1(see **Section 13.6.8.7**)
- 010 = Edge Count mode, count rising and falling edges of Primary Source
- 011 = Gated Count mode, count rising edges of Primary Source while secondary input high active

1. If Primary Count Source is IPBus clock, divide by one, then only rising edges are counted regardless of IPS value.

- 100 = Quadrature Count mode, uses Primary and Secondary Sources
- 101 = Signed Count mode, count edges of Primary Source
  - IPS = 0 then Secondary Input = 0; Count up on rising edges of Primary Input
  - IPS = 0 then Secondary Input = 1; Count down on rising edges of Primary Input
  - IPS = 1 then Secondary Input = 0; Count up on falling edges of Primary Input
  - IPS = 1 then Secondary Input = 1; Count down on falling edges of Primary Input
- 110 = Triggered Count mode, edge of secondary source triggers primary count until compare
- 111 = Cascaded Counter mode (up/down)[1]

13.6.7.2 Primary Count Source (PCS)—Bits 12–9

These bits select the Primary Count Source.

- 0000 = Counter 0 pin (T0)
- 0001 = Counter 1 pin (T1)
- 0010 = Counter 2 pin (T2)
- 0011 = Counter 3 pin (T3)
- 0100 = Counter 0 OFLAG
- 0101 = Counter 1 OFLAG
- 0110 = Counter 2 OFLAG
- 0111 = Counter 3 OFLAG3
- 1000 = Prescaler (System Clock or 3× System Clock[2] divide by 1)
- 1001 = Prescaler (System Clock or 3× System Clock[2] divide by 2)
- 1010 = Prescaler (System Clock or 3× System Clock[2] divide by 4)
- 1011 = Prescaler (System Clock or 3× System Clock[2] divide by 8)
- 1100 = Prescaler (System Clock or 3× System Clock[2] divide by 16)
- 1101 = Prescaler (System Clock or 3× System Clock[2] divide by 32)
- 1110 = Prescaler (System Clock or 3× System Clock[2] divide by 64)
- 1111 = Prescaler (System Clock or 3× System Clock[2] divide by 128)

Note: A timer selecting its own output for input is not a legal choice. The result is no counting.

1. Primary Count Source must be set to one of the counter outputs.
2. 3× System Clock must be selected via SIM_GPS register.

13.6.7.3 Secondary Count Source (SCS)—Bits 8–7

These bits identify the external input pin to be used as a count command. The selected input can trigger the timer to capture the current value of the CNTR register. The selected input can also be used to specify the count direction. The polarity of the signal can be inverted by the IPS bit of the SCTRL register.

- 00 = Counter 0 pin
- 01 = Counter 1 pin
- 10 = Counter 2 pin
- 11 = Counter 3 pin

13.6.7.4 Count Once (ONCE)—Bit 6

This bit selects Continuous or One-Shot Counting modes.

- 0 = Count repeatedly
- 1 = Count until compare and then stop. If *counting up*, successful compare occurs when the counter reaches a COMP1 value. If *counting down*, successful compare occurs when the counter reaches a COMP2 value. When the compare occurs, the timer module changes its Count Mode to *Stop Mode* (CM=0).

13.6.7.5 Count Length (LENGTH)—Bit 5

This bit determines whether the counter counts to the compare value and then reinitializes itself to the value specified in the Load register, or the counter continues counting past the compare value, to the binary roll-over.

- 0 = Roll-over
- 1 = Count till compare, then reinitialize. If *counting up*, successful compare occurs when the counter reaches a COMP1 value. If *counting down*, successful compare occurs when the counter reaches a COMP2 value[1]

1. When using Output Mode 100, alternating values of COMP1 and COMP2 are used to generate successful comparisons. For example, when the Output Mode is 100, the counter counts until COMP1 value is reached, reinitializes, then counts until COMP2 value is reached, reinitializes, then counts until COMP1 value is reached, and so on.

13.6.7.6 Count Direction (DIR)—Bit 4

This bit selects either the normal count direction *up*, or the reverse *down* direction.

- 0 = Count up
- 1 = Count down

13.6.7.7 Co-Channel Initialization (CoINIT)—Bit 3

This bit enables another counter/timer within the same module to force the reinitialization of this counter/timer when it has an active compare event. The Master channel forcing reinitialization has MSTR bit set. Please see **Section 13.6.8.10.**

- 0 = Co-Channel counter/timers cannot force a reinitialization of this counter/timer
- 1 = Co-Channel counter/timers can force a reinitialization of this counter/timer

13.6.7.8 Output Mode (OM)—Bits 2–0

This bit field determines the mode of operation for the OFLAG output signal.

- 000 = Assert OFLAG while counter is active
- 001 = Clear OFLAG output on successful compare
- 010 = Set OFLAG output on successful compare
- 011 = Toggle OFLAG output on successful compare
- 100 = Toggle OFLAG output using alternating Compare registers[1]
- 101 = Set OFLAG on compare, cleared on secondary source input edge
- 110 = Set OFLAG on compare, cleared on counter roll-over
- 111 = Enable Gated Clock output while counter is active

Note: Unexpected results may occur if OM field is set to use alternating Compare registers (Mode 100) and the ONCE bit are set.

1. When using Output Mode 100, alternating values of COMP1 and COMP2 are used to generate successful comparisons. For example, when the Output Mode is 100, the counter calculates until COMP1 value is reached., reinitializes, then counts until COMP2 value is reached, reinitializes, then counts until COMP1 value is reached, and so on.

13.6.8 Status and Control Registers (SCTRL)

There are four Timer Status/Control (SCTRL) registers. For details regarding the timer settings required to implement variable frequency PWM operation, please refer to **Section 13.5.12.2**. Their addresses are:

TMR0_SCTRL (Channel 0 Status and Control)—Address: TMR_BASE + $7
TMR1_SCTRL (Channel 1 Status and Control)—Address: TMR_BASE + $17
TMR2_SCTRL (Channel 2 Status and Control)—Address: TMR_BASE + $27
TMR3_SCTRL (Channel 3 Status and Control)—Address: TMR_BASE + $37

Base + $7	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	TCF	TCFIE	TOF	TOFIE	IEF	IEFIE	IPS	INPUT	CAPTURE MODE		MSTR	EEOF	VAL	0	OPS	OEN
Write														FORCE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-16. Status and Control (SCTRL) Registers

13.6.8.1 Timer Compare Flag (TCF)—Bit 15

This bit is set when a successful compare occurs. Clear the bit by writing 0 to it.

TCF asserts every time there is a compare event (either when counter = COMP1 or counter = COMP2).

13.6.8.2 Timer Compare Flag Interrupt Enable (TCFIE)—Bit 14

When set, the timer compare interrupt is enabled.

13.6.8.3 Timer Overflow Flag (TOF)—Bit 13

This bit is set when the counter rolls over its maximum or minimum value $FFFF or $0000, depending on count direction. Clear the bit by writing 0 to it.

13.6.8.4 Timer Overflow Flag Interrupt Enable (TOFIE)—Bit 12

When set, this bit enables interrupts when the TOF bit is set.

13.6.8.5 Input Edge Flag (IEF)—Bit 11

This bit is set when a transition occurs on an input selected as a Secondary Count Source while the counter is enabled. Clear the bit by writing 0 to it.

Note: The Control register's Secondary Count Source determines which external input pin is monitored by the detection circuitry. Please see Input Capture Mode, **Section 13.6.8.9**, below for how IEF is set.

13.6.8.6 Input Edge Flag Interrupt Enable (IEFIE)—Bit 10

When set, this bit enables interrupts if the IEF bit is set.

13.6.8.7 Input Polarity Select (IPS)—Bit 9

When set, this bit inverts the polarity of both the primary and secondary inputs.

13.6.8.8 External Input Signal (INPUT)—Bit 8

This *read-only* bit reflects the current state of the external input pin selected via the Secondary Count Source after application of the IPS bit.

13.6.8.9 Input Capture Mode (CAPTURE MODE)—Bits 7–6

These bits specify the operation of the CAPT register as well as the operation of the Input Edge Flag (IEF). The input source is the Secondary Count Source.

Table 13-3. Setting Combination of Capture Mode and IPS State

Capture Mode	IPS	Action
00	x	Capture Disabled
01	0	Load on Rising Edge
01	1	Load on Falling Edge
10	0	Load on Falling Edge
10	1	Load on Rising Edge
11	x	Load on Both Edges

13.6.8.10 Master Mode (MSTR)—Bit 5

When set, this bit enables the Compare register function output to be broadcasted to the other counter/timers in the module. This signal then can be used to reinitialize the other counters and/or force their OFLAG signal outputs. Other timer channels within the Quad Timer accept the reinitialization signal when their CoINIT bit is set. Please see **Section 13.6.7.7.**

13.6.8.11 Enable External OFLAG Force (EEOF)—Bit 4

When set, this bit enables the Compare register from another counter/timer within the same module to force the state of this counter's OFLAG output signal.

13.6.8.12 Forced OFLAG Value (VAL)—Bit 3

This bit determines the value of the OFLAG output signal when a software triggered FORCE command occurs, i.e. when FORCE = 1, discussed in **Section 13.6.8.13.**

13.6.8.13 Force OFLAG Output (FORCE)—Bit 2

This *write-only* bit forces the current value of the VAL bit to be written to the OFLAG output. This bit is always read as 0. The VAL and FORCE bits can be written simultaneously in a single write operation. Write to the FORCE bit only if the counter is disabled.

Note: Setting this bit while the counter is enabled may yield unpredictable results.

13.6.8.14 Output Polarity Select (OPS)—Bit 1

This bit determines the polarity of the OFLAG output signal.

- 0 = True polarity
- 1 = Inverted polarity

13.6.8.15 Output Enable (OEN)—Bit 0

When set, this bit determines the direction of the external pin.

- 0 = The external pin is configured as an input.
- 1 = OFLAG output signal will be driven on the external pin. The polarity of the signal will be determined by the OPS bit.

13.6.9 Comparator Load 1 Registers (CMPLD1)

There are four read/write registers making up the Comparator 1 preload value for the CMPLD1 Register of the corresponding channel in a timer module. Please see **Section 13.4.2** for additional information. Their addresses are:

TMR0_CMPLD1 (Channel 0 Comparator Load 1)—Address: TMR_BASE + $8
TMR1_CMPLD1 (Channel 1 Comparator Load 1)—Address: TMR_BASE + $18
TMR2_CMPLD1 (Channel 2 Comparator Load 1)—Address: TMR_BASE + $28
TMR3_CMPLD1 (Channel 3 Comparator Load 1)—Address: TMR_BASE + $38

Please see **Section 13.6.11.7** for information on how to control comparator loading.

Base +$8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	COMPARATOR LOAD 1															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-17. Comparator Load 1 (CMPLD1) Register

For details regarding the timer settings required to implement variable frequency PWM operation, please refer to **Section 13.5.12.1**.

13.6.10 Comparator Load 2 Registers (CMPLD2)

There are four read/write registers making up the Comparator 2 preload value for the CMPLD2 register of the corresponding channel in a timer module. Please see **Section 13.4.2** for additional information. Their addresses are:

TMR0_CMPLD2 (Channel 0 Comparator Load 2)—Address: TMR_BASE + $9
TMR1_CMPLD2 (Channel 1 Comparator Load 2)—Address: TMR_BASE + $19
TMR2_CMPLD2 (Channel 2 Comparator Load 2)—Address: TMR_BASE + $29
TMR3_CMPLD2 (Channel 3 Comparator Load 2)—Address: TMR_BASE + $39

Please see **Section 13.6.11.6** for information on how to control the loading of Comparator 2.

Base +$9	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	COMPARATOR LOAD 2															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-18. Comparator Load 2 (CMPLD2) Register

13.6.11 Comparator Status/Control Registers (CSCTRL)

There are four read/write registers making up the Timer Comparator Status and Control register (CSCTRL). For details regarding the timer settings required to implement variable frequency PWM operation, please refer to **Section 13.5.12.3.** Their addresses are:

TMR0_CSCTRL (Channel 0 Comparator Status/Control)—Address: TMR_BASE + $A
TMR1_CSCTRL (Channel 1 Comparator Status/Control)—Address: TMR_BASE + $1A
TMR2_CSCTRL (Channel 2 Comparator Status/Control)—Address: TMR_BASE + $2A
TMR3_CSCTRL (Channel 3 Comparator Status/Control)—Address: TMR_BASE + $3A

Base + $A	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Read	0	0	0	0	0	0	0	0	TCF2EN	TCF1EN	TCF2	TCF1	CL2		CL1	
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-19. Comparator Status/Control (CSCTRL) Registers

13.6.11.1 Reserved—Bits 15–8

This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.

13.6.11.2 Timer Compare 2 Interrupt Enable (TCF2EN)—Bit 7

An interrupt is issued when both this bit and the TCF2 bit are set.

13.6.11.3 Timer Compare 1 Interrupt Enable (TCF1EN)—Bit 6

An interrupt is issued when both this bit and the TCF1 bit are set.

13.6.11.4 Timer Compare Flag 2 (TCF2)—Bit 5

When set, this bit indicates a successful comparison of the timer and COMP2 register has occurred. This bit is sticky, and will remain set until explicitly cleared by writing 0 to this bit location.

13.6.11.5 Timer Compare Flag 1 (TCF1)—Bit 4

When set, this bit indicates a successful comparison of the timer and COMP1 register has occurred. This bit is sticky, and will remain set until explicitly cleared by writing 0 to this bit location.

13.6.11.6 Compare Load Control 2 (CL2)—Bit 3–2

These bits control when COMP2 is preloaded with the value from CMPLD2.

Table 13-4. Values for Compare Preload Control 2

Value	Meaning
00	Never Preload
01	Load Upon Successful Compare with the Value in COMP1, i.e. when TCF1 is set
10	Load Upon Successful Compare with the Value in COMP2, i.e. when TCF2 is set
11	Reserved

13.6.11.7 Compare Load Control 1 (CL1)—Bit 1–0

These bits control when COMP1 is preloaded with the value from CMPLD1.

Table 13-5. Values for Compare Preload Control 1

Value	Meaning
00	Never preload
01	Load Upon Successful Compare with the Value in COMP1, i.e. when TCF1 is set
10	Load Upon Successful Compare with the Value in COMP2, i.e. when TCF2 is set
11	Reserved

13.7 Clocks

The Timer operates from the IPBus Clock or high speed IPBus clock.

13.8 Interrupts

Each of the four timers in a timer module can generate an interrupt, serviced by the Interrupt Service Routine (ISR) identified by the Interrupt Vector Table. Please see the chip's data sheet for more information on the Vector Table.The ISR will have to check the SCTRL register and the CSCTRL register for which of the five interrupt flag bits are high.

Table 13-6 describes each of these flag bits.

Table 13-6. Timer Interrupt Flags

Acronym	Name	Located In		Location
		SCTRL	CSCTRL	
TCF	Timer Compare Flag	x	—	Section 13.6.8.1
TOF	Timer Overflow Flag	x	—	Section 13.6.8.3
IEF	Input Edge Flag	x	—	Section 13.6.8.5
TCF1	Timer Compare Flag 1	—	x	Section 13.6.11.5
TCF2	Timer Compare Flag 2	—	x	Section 13.6.11.4

The ISR should reset each set flag bit by writing 0 to the bit.

13.9 Resets

The TMR module can only be reset b y the chip wide $\overline{\text{RST}}$ signal. This forces all registers to their reset state and clears the OFLAG signal. The counter will be turned off until the settings in the CTRL register are changed.

Chapter 14
Voltage Regulator (VREG)

Document Revision History for Chapter 14, Voltage Regulator (VREG)

Version History	Description of Change
Rev 0	Initial release

14.1 Introduction

This module provides an on-chip mechanism to regulate an external 3.3V supply down to 2.5V levels for use with the internal core logic, or *large regulator*. It also provides a precision voltage to the on-chip Relaxation Oscillator and PLL using a *small regulator*. The PLL enjoys additional noise immunity because of the small regulator. On-board regulators are stable and have sufficient capacity and dynamic response allowing the host chip to operate correctly at all times and under all combinations of specified loads, frequencies, voltages, temperatures, and process variations.

14.2 Features

Qualities of the Linear Voltage Regulator contain:

- Provide a 2.5V ±10% accuracy
- Provide an average current of at least 125mA for the large regulator
- Provide an average current of at least 1mA for the small regulator

14.3 Block Diagram

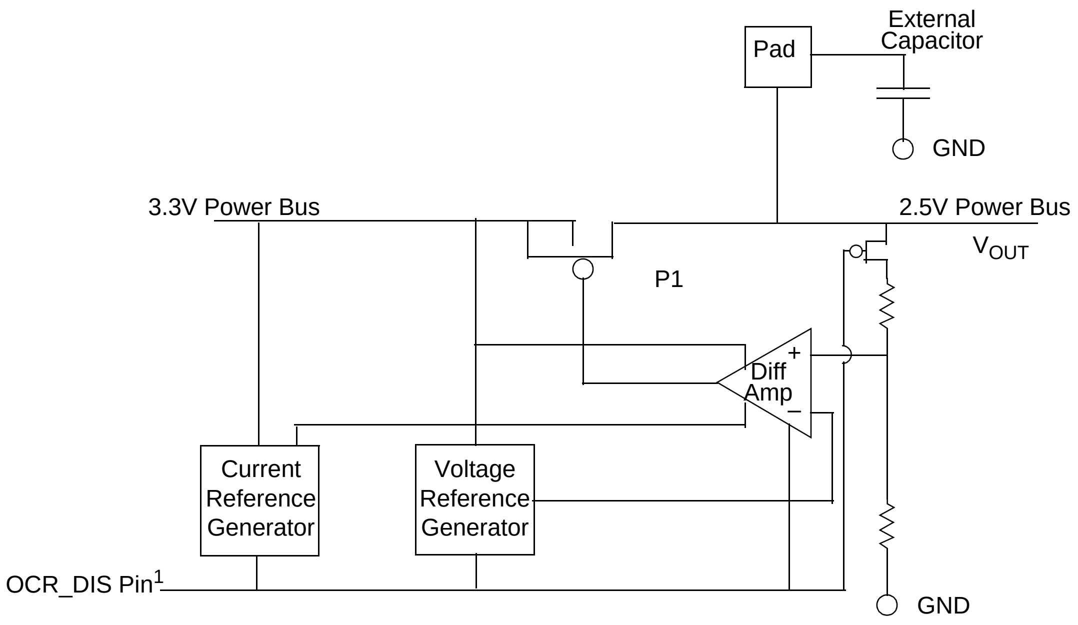


Figure 14-1. Large Voltage Regulator Block Diagram

1. Not available on all devices. Tied to V_{SS} when not available as a pin.

14.4 Functional Description

Internal regulators (typically one for logic, and one for analog) are added to regulate down from 3.3 to 2.5V. The large core voltage regulator look like the circuit illustrated in **Figure 14-1.** Small voltage regulator do not have a pad.

The large regulator, illustrated in **Figure 14-1,** is a linear series-pass low drop-out regulator. It uses a very large p-channel MOSFET as the pass device, and it uses an equally sized overload protection device. It takes an input of 3.3V ±10 percent and provides a regulated 2.5V output at a maximum current level of 125mA. It has a power down feature to facilitate the use of an external regulator on devices with an OCR_DIS pin. When the internal regulator is disabled through OCR_DIS, the V_{CAP} pin(s) will be used to provide the 2.5V V_{DD_CORE} core voltage. For devices with no OCR_DIS pin, the internal regulator must be used.

The large regulator, illustrated in **Figure 14-1,** requires external 4.4µF capacitors to be tied to the V_{CAP} pin(s) to help maintain stability. For optimal transient performance, the 4.4µF filter cap should be low Equivalent Series Resistance (ESR) Multi-Layer Ceramic Chip (MLCC) caps and should be placed as close as possible to the chip's V_{CAP} pin(s). Using electrolytic capacitors is discouraged because of their high ESR. Using high ESR capacitors causes the regulator to have poor transient load regulation performance. Filter capacitors can be used, since they improve the transient load regulation performance. However, any filter capacitor used must be at least 4.4µF.

14.5 Operating Modes

This is an analog part excited by the introduction of the input voltage. Its modes of operation are ON and standby capability.

On some devices, the large regulator for the on-chip digital logic can be disabled, or turned OFF via the OCR_DIS pin. This can be useful if an external regulated 2.5V supply is provided by the system design. When the on-chip regulator is disabled the V_{CAP} pin(s) are used to provide the regulated 2.5V V_{DD_CORE} to the core logic.

14.6 Memory Map

This device has no memory mapped registers except the SIM Power Control (SIM_PWR) register provided below.

14.6.1 SIM Power Control Register (SIM_PWR)

This register controls the Standby mode of the large regulator. The large regulator derives the core digital logic power supply from the I/O power supply. In some circumstances the large regulator may be placed in a reduced-power Standby mode without interfering with part operation. In this mode voltage will be regulated down to 2.5V, but the current output will be reduced. Please refer to the overview of Power-Down modes and the overview of clock generation for more information on the use of large regulator standby.

Base + $8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LRSTDBY	
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14-2. SIM Power Control (SIM_PWR) Register

14.6.1.1 Large Regulator Standby Mode (LRSTDBY)—Bits 1–0

- 00 = Large regulator is in Normal mode
- 01 = Large regulator is in Standby (reduced power) mode
- 10 = Large regulator is in Normal mode and the LRSTDBY bit field is write- protected until the next reset
- 11 = Large regulator is in Standby mode and the LRSTDBY bit field is write-protected until the next reset

14.7 Pin Descriptions

Table 14-1. Signal Properties

Name	I/O Type	Function	Reset State	Notes
V_{SS}	DC Input	Ground	N/A	—
V_{DD}	DC Source	Input Voltage Supply	N/A	—
V_{CAP}	—	Capacitor	N/A	Not included on the smaller regulators
OCR_DIS	Input	On-Chip Regulator Disable	N/A	Tie to V_{SS} or V_{DD}

14.7.1 Input Voltage (V_{DD})

V_{DD} is the input voltage of 3.3V typically required by the regulator to convert to 2.5V.

14.7.2 Capacitor Pin(s) (V_{CAP})

4.4µF capacitor must be connected to the V_{CAP} pin(s) on the larger regulator for proper operation. If OCR_DIS=1, then V_{CAP} provides 2.5V input to the core logic, oscillator, and PLL.

14.7.3 On-Chip Regulator Disable (OCR_DIS)

OCR_DIS is not available on all devices. If available, this pin should be tied to either V_{SS} or V_{DD} at power-up.

- To *enable* the on-chip regulator, tie this pin to V_{SS}
- To *disable* the on-chip regulator, tie this pin to V_{DD}

Note: This pin is intended to be a static DC signal from power-up to shut down. Do not toggled this pin for power savings during operation.

14.8 Clocks

There are no clocks used by this module.

14.9 Resets

There are no resets for this device.

14.10 Interrupts

There are no interrupts generated by this module.

Appendix A
Glossary

Document Revision History for Appendix A, Glossary

Version History	Description of Change
Rev. 0	Initial release

A.1 Glossary

This glossary is intended to reduce potential confusion caused by the use of many acronyms and abbreviations throughout this manual.

ACIM	A/C Induction Motors
A/D	Analog-to-Digital
ADC	Analog-to-Digital Converter
ADC Clock	The ADC blocks run at a slower speed than the rest of the system. A separate ADC clock is derived from the system IPBus clock based upon the divisor specified in the ADC Control Register 2 (CTRL2).
ADDR	I^2C Address Register
BDC	Brush DC Motor
BLDC	Brushless DC Motor
BOTNEG	Bottom-side PWM Polarity Bit
BSDL	Boundary Scan Description Language
BSR	JTAG Boundary Scan Register
CAN	Controller Area Network
CAL	ADC Calibration Register
CAPT	TMR Capture Register
CCTRL	PWM Channel Control Register
CLIST1	ADC Channel List 1 Register
CLIST2	ADC Channel List 2 Register
CID	JTAG Chip Identification Register
CLKO	Clock Output pin
CLKDIV	FM Clock Divider Register
CMD	FM Command Register
CMOS	Complementary Metal Oxide Semiconductor, (a form of digital logic characterized by low power consumption, wide power supply range, and high noise immunity.)
CMOD	PWM Counter Modulo Register
CNFG	FM and PWM Configuration Registers
CNTR	COP, PWM, and TMR Counter Registers
Codec	Coder/Decoder
Command Sequence	A three-step Digital Signal Controller (DSC) instruction sequence to program or erase the Flash.

CMPLD1	TMR Comparator Load 1 Register
CMPLD2	TMR Comparator Load 2 Register
COMP1	TMR Compare 1 Register
COMP2	TMR Compare 2 Register
COP	Computer Operating Properly
CPHA	Clock Phase
CPOL	Clock Polarity
CRC	Cyclic Redundancy Code
CSCTRL	TMR Comparator Status and Control Register
CTRL	COP, I^2C, OCCS, PS, PWM, and TMR Control Registers
CTRL1	ADC and SCI Control 1 Registers
CTRL2	ADC and SCI Control 2 Register
DAC	Digital-to-Analog Converter
DATA	I^2C, FM, GPIO and SCI Data Registers
DDIR	GPIO Data Direction Register
DIVBY	OCCS Divide-By Register
DMAP1-2	PWM Disable Mapping 1-2 Registers
DRCV	SPI Data Receive Register
DRIVE	GPIO Drive Strength Control Register
DSCTRL	SPI Data Size and Control Register
DTIM0-1	PWM Deadtime 0-1 Registers
DXMIT	SPI Data Transmit Register
EN	Enable
EOnCE	Enhanced On-Chip Emulation (unit)
ESR	Equivalent Series Resistance
EXTBOOT	External Boot
FCTRL	PWM Fault Control Register
FAULT	Fault Input to PWM
Flash Module	Includes Bus Interface, Command Control, and the Flash physical block
FLTACK	PWM Fault Status/Acknowledge Register
FOSC	Oscillator Frequency
FREF	Reference Frequency

FDIV	I^{2}C Frequency Divider Register
GPIO	General Purpose Input/Output
Harvard Architecture	This is a microprocessor architecture using separate buses for program and data. This architecture is typically used on Digital Signal Controller (DSC) to optimize the data throughput.
HILIM	ADC High Limit 0-7 Registers
HOLD	TMR Hold Register
IASSRT	GPIO Interrupt Assert Register
ICCTRL	PWM Internal Correction Control Register
IEDGE	GPIO Interrupt Edge Sensitive Register
IEN	GPIO Interrupt Enable Register
IPEND	GPIO Interrupt Pending Register
IPOL	GPIO Interrupt Polarity Register
ITCN	Interrupt Controller (See device Data Sheet for complete information)
I/O	Input/Output
IPBus	Intellectual Properties Bus. This Motorola proprietary bus architecture accommodates various intellectual properties operating at slower speed than the system bus.
IPBus Clock	System peripheral clock
IRQ	Interrupt Request
JTAG	Joint Test Action Group
LIMSTAT	ADC Limit Status Register
LOAD	TMR Load Register
LOLIM	ADC Low Limit 0-7 Registers
LSB	Least Significant Bit
LSH_ID	Least Significant Half of JTAG_ID
LVI	Low Voltage Interrupt
MHz	Megahertz
MISO	Master In/Slave Out
MOSI	Master Out/Slave In
MSB	Most Significant Bit
MSH_ID	Most Significant Half of JTAG ID
MISR	A Multiple Input Signature Analyzer Register is an output response analyzer implemented using a linear feedback shift register.
MUX	Multiplexer

Multiple Input (MISR). **Analyzer**	An output response analyzer implemented using a linear feedback Shift register; **Signature**
NFILT	I^2C Noise Filter Register
OCCS	On-Chip Clock Synthesis
OCTRL	OCCS Oscillator Control Register
OFFST	ADC Offset 0-7 Registers
OnCE	On-Chip Emulation (unit)
OPT1	FM Optional Data 1 Register
OUT	PWM Output Control Register
PAB	Program Address Bus
PEREN	GPIO Peripheral Enable
PFLASH	Program Flash
PLL	Phase Locked Loop
POL	Polarity
POR	Power-On Reset
PORT	PWM Port Register
PPOUTM	GPIO Push-Pull Output Mode Control Register
PRAM	Program RAM
PROT	FM Protection Register
PUPEN	GPIO Pull-Up Enable Register
PWM	Pulse Width Modulator
PWR	ADC Power Control Register
RDATA	GPIO Raw Data Register
RAM	Random Access Memory
RATE	SCI Baud Rate Register
RSLT0-7	ADC Result 0-7 Registers
ROM	Read Only Memory
SCI	Serial Communications Interface
SCTRL	PWM Source Control Register
SCTRL	SPI and TMR Status and Control Registers
SDIS	ADC Sample Disable Register
SECHI	FM Security High Half Register

SECLO	FM Security Low Half Register
S/H	Sample and Hold
SHUTDN	OCCS Shutdown Register
SIM	System Integration Module
SOC	System-On-Chip
SPI	Serial Peripheral Interface
SRM	Switched Reluctance Motor
STAT	ADC, I^2C, OCCS, PS, and SCI Status Registers
System Clock	A free running clock generated by the PLL and oscillators. In reality, each system component will receive a unique version of this clock generated by the Clock Generation Module and controlled by the System Integration Module (SIM).
TAP	Test Access Port
TCK	TAP Clock
TDI	TAP Data In
TDO	TAP Data Out
TLM	TAP Linking Module
TOUT	COP Time-Out Register
TSMC IP	Taiwan Semiconductor Manufacturing Company Intellectual Property. In the Flash module, the TSMC IPs are the 128k, 64k, 16k, and 8k byte Flash hard blocks.
TSTSIG	FM Test Array Signature Register
USTAT	FM User Status Register
VAL0-7	PWM Value 0-7 Registers
VCO	Voltage Controlled Oscillator
V_{DD}	Power
V_{DDA}	Analog Power
V_{REF}	Voltage Reference
VRM	Variable Reluctance Motor
V_{SS}	Ground
V_{SSA}	Analog Ground
XRAM	Data RAM
ZXCTRL	ADC Zero Crossing Control Register

Appendix B
Programmer Sheets

Document Revision History for Appendix B, Programmer Sheets

Version History	Description of Change
Rev. 0	Initial release
Rev. 1	COP Programmer Sheet, Control Register, pg 41: Remove bit 4 BYPS. It is a reserved bit. PWM Programmer Sheet, Source Control Register, pg 101: SC0 is two bits rather than three. PWM Programmer Sheet, Source Control Register, pg 101: SC1 bit description should not be OUTCTL2 to OUTCTL2, rather it should be OUTCTL 2 to OUTCTL3

B.1 Introduction

The following pages provide a set of reference tables and programming sheets intended to simplify programming the 56F8000Family. The programming sheets provide room to add the value of each bit and the hexadecimal value for each register. These pages may be photocopied.

For complete instruction set details, refer to Chapter 4 of the *DSP56800 Reference Manual*.

B.2 Legacy and New Acronym Cross Reference

Register acronyms are revised from previous device peripheral manuals to provide a cleaner register description. A cross reference to legacy and revised acronyms are provided in the following table.

Module	Register Name	Peripheral Reference Manual		Data Sheet		Processor Expert Acronym	Memory Address	
		New Acronym	Legacy Acronym	New Acronym	Legacy Acronym		Start	End
ADC	Control Register 1	CTRL1	ADCR1	ADC_CTRL1	ADC_ADCR1	ADC_ADCR1	0xF080	
	Control Register 2	CTRL2	ADCR2	ADC_CTRL2	ADC_ADCR2	ADC_ADCR2	0xF081	
	Zero Crossing Control Register	ZXCTRL	ADZCC	ADC_ZXCTRL	ADC_ADZCC	ADC_ADZCC	0xF082	
	Channel List Register 1	CLIST1	ADLST1	ADC_CLIST1	ADC_ADLST1	ADC_ADLST1	0xF083	
	Channel List Register 2	CLIST2	ADLST2	ADC_CLIST2	ADC_ADLST2	ADC_ADLST2	0xF084	
	Sample Disable Register	SDIS	ADSDIS	ADC_SDIS	ADC_ADSDIS	ADC_ADSDIS	0xF085	
	Status Register	STAT	ADSTAT	ADC_STAT	ADC_ADSTAT	ADC_ADSTAT	0xF086	
	Limit Status Register	LIMSTAT	ADLSTAT	ADC_LIMSTAT	ADC_ADLSTAT	ADC_ADLSTAT	0xF087	
	Zero Crossing Status Register	ZXSTAT	ADZCSTAT	ADC_ZXSTAT	ADC_ADZCSTAT	ADC_ADZCSTAT	0xF088	
	Result Registers 0-7	RSLT0-7	ADRSLT0-7	ADC_RSLT0-7	ADC_ADRSLT0-7	ADC_ADRSLT0-7	0xF089	0XF090
	Low Limit Registers 0-7	LOLIM0-7	ADLLMT0-7	ADC_LOLIM0-7	ADC_ADLLMT0-7	ADC_ADLLMT0-7	0XF091	0XF098
	High Limit Registers 0-7	HILIM0-7	ADHLMT0-7	ADC_HILIM0-7	ADC_ADHLMT0-7	ADC_ADHLMT0-7	0XF099	0XF0A0
	Offset Registers 0-7	OFFST0-7	ADOFS0-7	ADC_OFFST0-7	ADC_ADOFS0-7	ADC_ADOFS0-7	0XF0A1	0XF0A8
	Power Control Register	PWR	ADPOWER	ADC_PWR	ADC_ADPOWER	ADC_ADPOWER	0XF0A9	
	Voltage Reference Register	VREF	ADVREF	ADC_VREF	ADC_ADVREF	ADC_ADVREF	0XF0AA	
COP	Control Register	CTRL	COPCTL	COP_CTRL	COPCTL	COPCTL	0XF0E0	
	Time-Out Register	TOUT	COPTO	COP_TOUT	COPTO	COPTO	0XF0E1	
	Counter Register	CNTR	COPCTR	COP_CNTR	COPCTR	COPCTR	0XF0E2	
I^{2}C	Address Register	ADDR	IBAD	I2C_ADDR	I2C_IBAD	IBAD	0xF0D0	
	Frequency Divider Register	FDIV	IBFD	I2C_FDIV	I2C_IBFD	IBFD	0xF0D1	
	Control Register	CTRL	IBCR	I2C_CTRL	I2C_IBCR	IBCR	0xF0D2	
	Status Register	STAT	IBSR	I2C_STAT	I2C_IBSR	IBSR	0xF0D3	
	Data I./O Register	DATA	IBDR	I2C_DATA	I2C_IBDR	IBDR	0xF0D4	
	Noise Filter Register	NFILT	IBNR	I2C_NFILT	I2C_IBNR	IBNR	0xF0D5	

Module	Register Name	Peripheral Reference Manual		Data Sheet		Processor Expert Acronym	Memory Address	
		New Acronym	Legacy Acronym	New Acronym	Legacy Acronym		Start	End
OCCS	Control Register	CTRL	PLLCR	OCCS_CTRL	PLLCR	PLLCR	0xF0F0	
	Divide-By Register	DIVBY	PLLDB	OCCS_DIVBY	PLLDB	PLLDB	0xF0F1	
	Status Register	STAT	PLLSR	OCCS_STAT	PLLSR	PLLSR	0xF0F2	
	Shutdown Register	SHUTDN	SHUTDOWN	OCCS_SHUTDN	SHUTDOWN	SHUTDOWN	0xF0F4	
	Oscillator Control Register	OCTRL	OSCTL	OCCS_OCTRL	OSCTL	OSCTL	0xF0F5	
FM	Clock Divider Register	CLKDIV	FMCLKD	FM_CLKDIV	FMCLKD	FMCLKD	0xF400	
	Configuration Register	CNFG	FMCR	FM_CNFG	FMCR	FMCR	0xF400	
	Security High Half Register	SECHI	FMSECH	FM_SECHI	FMSECH	FMSECH	0xF400	
	Security Low Half Register	SECLO	FMSECL	FM_SECLO	FMSECL	FMSECL	0xF400	
	Protection Register	PROT	FMPROT	FM_PROT	FMPROT	FMPROT	0xF410	
	User Status Register	USTAT	FMUSTAT	FM_USTAT	FMUSTAT	FMUSTAT	0xF413	
	Command Register	CMD	FMCMD	FM_CMD	FMCMD	FMCMD	0xF414	
	Address Register	ADDR	FMADDR	FM_ADDR	FMADDR		0xF416	
	Data Buffer Register	DATA	FMDATA	FM_DATA	FMDATA		0xF418	
	Optional Data 1 Register	OPT1	FMOPT1	FM_OPT1	FMOPT1	FMOPT1	0xF41B	
	Test Array Signature Register	TSTSIG	FMTST_SIG	FM_TSTSIG	FMTST_SIG		0xF41D	
						x = A (***n***=0) B (***n***=1) C (***n***=2) D (***n***=3)		
GPIO	Pull-Up Enable Register	PUPEN	PUR	GPIO*x*_PUPEN	GPIO*x*_PUR	GPIO_***x***_PUR	0xF1***n***0	
	Data Register	DATA	DR	GPIO*x*_DATA	GPIO*x*_DR	GPIO_***x***_DR	0xF1***n***1	
	Data Direction Register	DDIR	DDR	GPIO*x*_DDIR	GPIO*x*_DDR	GPIO_***x***_DDR	0xF1***n***2	
	Peripheral Enable Register	PEREN	PER	GPIO*x*_PEREN	GPIO*x*_PER	GPIO_***x***_PER	0xF1***n***3	
	Interrupt Assert Register	IASSRT	IAR	GPIO*x*_IASSRT	GPIO*x*_IAR	GPIO_***x***_IAR	0xF1***n***4	
	Interrupt Enable Register	IEN	IENR	GPIO*x*_IEN	GPIO*x*_IENR	GPIO_***x***_IENR	0xF1***n***5	
	Interrupt Polarity Register	IPOL	IPOLR	GPIO*x*_IPOL	GPIO*x*_IPOLR	GPIO_***x***_IPOLR	0xF1***n***6	
	Interrupt Pending Register	IPEND	IPR	GPIO*x*_IPEND	GPIO*x*_IPR	GPIO_***x***_IPR	0xF1***n***7	
	Interrupt Edge Sensitive Register	IEDGE	IESR	GPIO*x*_IEDGE	GPIO*x*_IESR	GPIO_***x***_IESR	0xF1***n***8	
	Push-Pull Output Mode Control Regis.	PPOUTM	PPMODE	GPIO*x*_PPOUTM	GPIO*x*_PPMODE	GPIO_***x***_PPMODE	0xF1***n***9	
	Raw Data Register	RDATA	RAWDATA	GPIO*x*_RDATA	GPIO*x*_RAWDATA	GPIO_***x***_RAWDATA	0xF1***n***A	
	Drive Strength Control Register	DRIVE	DRIVE	GPIO*x*_DRIVE	GPIO*x*_DRIVE	GPIO_***x***_DRIVE	0xF1***n***B	
PS	Control Register	CTRL	LVICONTROL	PS_CTRL	LVICONTROL	LVICTRL	0xF160	
	Status Register	STAT	LVISTATUS	PS_STAT	LVISTATUS	LVISR	0xF161	

PWM	Control Register	CTRL	PMCTL	PWM_CTRL	PWM_PMCTL	PWM_PMCTL	0xF040	
	Fault Control Register	FCTRL	PMFCTL	PWM_FCTRL	PWM_PMFCTL	PWM_PMFCTL	0xF041	
	Fault Status/Acknowledge Regis.	FLTACK	PMFSA	PWM_FLTACK	PWM_PMFSA	PWM_PMFSA	0xF042	
	Output Control Register	OUT	PMOUT	PWM_OUT	PWM_PMOUT	PWM_PMOUT	0xF043	
	Counter Register	CNTR	PMCNT	PWM_CNTR	PWM_PMCNT	PWM_PMCNT	0xF044	
	Counter Modulo Register	CMOD	MCM	PWM_CMOD	PWM_MCM	PWM_MCM	0xF045	
	Value Register 0-5	VAL0-5	PMVAL0-5	PWM_VAL0-5	PWM_PMVAL0-5	PWM_PMVAL0-5	0xF046	0xF04B
	Deadtime Register 0-1	DTIM0-1	PMDEADTM0-1	PWM_DTIM0-1	PWM_PMDEADTM0-1	PWM_PMDEADTM0-1	0xF04C	0xF04D
	Disable Mapping Register 1-2	DMAP1-2	PMDISMAP1-2	PWM_DMAP1-2	PWM_PMDISMAP1-2	PWM_PMDISMAP1-2	0xF04E	0xF04F
	Configure Register	CNFG	PMCFG	PWM_CNFG	PWM_PMCFG	PWM_PMCFG	0xF050	
	Channel Control Register	CCTRL	PMCCR	PWM_CCTRL	PWM_PMCCR	PWM_PMCCR	0xF051	
	Port Register	PORT	PMPORT	PWM_PORT	PWM_PMPORT	PWM_PMPORT	0xF052	
	Internal Correction Control Regis.	ICCTRL	PMICCR	PWM_ICCTRL	PWM_PMICCR	PWM_PMICCR	0xF053	
	Source Control Register	SCTRL	PMSRC	PWM_SCTRL	PWM_PMSRC	PWM_PMSRC	0xF054	
Module	**Register Name**	**Peripheral Reference Manual**		**Data Sheet**		**Processor Expert Acronym**	**Memory Address**	
		New Acronym	**Legacy Acronym**	**New Acronym**	**Legacy Acronym**		**Start**	**End**

SCI	Baud Rate Register	RATE	SCIBR	SCI_RATE	SCI_SCIBR	SCI_SCIBR	0xF0B0
	Control Register 1	CTRL1	SCICR	SCI_CTRL1	SCI_SCICR	SCI_SCICR	0xF0B1
	Control Register 2	CTRL2	SCICR2	SCI_CTRL2	SCI_SCICR2	SCI_SCICR2	0xF0B2
	Status Register	STAT	SCISR	SCI_STAT	SCI_SCISR	SCI_SCISR	0xF0B3
	Data Register	DATA	SCIDR	SCI_DATA	SCI_SCIDR	SCI_SCIDR	0xF0B4

SPI	Status and Control Register	SCTRL	SPSCR	SPI_SCTRL	SPI_SPSCR	SPI_SPSCR	0xF0C0
	Data Size and Control Register	DSCTRL	SPDSR	SPI_DSCTRL	SPI_SPDSR	SPI_SPDSR	0xF0C1
	Data Receive Register	DRCV	SPDRR	SPI_DRCV	SPI_SPDRR	SPI_SPDRR	0xF0C2
	Data Transmit Register	DXMIT	SPDTR	SPI_DXMIT	SPI_SPDTR	SPI_SPDTR	0xF0C3

						n = 0, 1, 2, 3	
TMR	Compare Register 1	COMP1	TMRCMP1	TMR***n***_COMP1	TMR***n***_CMP1	TMR***n***_CMP1	0xF0***n***0
	Compare Register 2	COMP2	TMRCMP2	TMR***n***_COMP2	TMR***n***_CMP2	TMR***n***_CMP2	0xF0***n***1
	Capture Register	CAPT	TMRCAP	TMR***n***_CAPT	TMR***n***_CAP	TMR***n***_CAP	0xF0***n***2
	Load Register	LOAD	TMRLOAD	TMR***n***_LOAD	TMR***n***_LOAD	TMR***n***_LOAD	0xF0***n***3
	Hold Register	HOLD	TMRHOLD	TMR***n***_HOLD	TMR***n***_HOLD	TMR***n***_HOLD	0xF0***n***4
	Counter Register	CNTR	TMRCNTR	TMR***n***_CNTR	TMR***n***_CNTR	TMR***n***_CNTR	0xF0***n***5
	Control Register	CTRL	TMRCTRL	TMR***n***_CTRL	TMR***n***_CTRL	TMR***n***_CTRL	0xF0***n***6
	Status and Control Register	SCTRL	TMRSCR	TMR***n***_SCTRL	TMR***n***_SCR	TMR***n***_SCR	0xF0***n***7
	Comparator Load Register 1	CMPLD1	TMRCMPLD1	TMR***n***_CMPLD1	TMR***n***_CMPLD1	TMR***n***_CMPLD1	0xF0***n***8
	Comparator Load Register 2	CMPLD2	TMRCMPLD2	TMR***n***_CMPLD2	TMR***n***_CMPLD2	TMR***n***_CMPLD2	0xF0***n***9
	Comparator Status/Control Regis.	CSCTRL	TMRCOMSCR	TMR***n***_CSCTRL	TMR***n***_COMSCR	TMR***n***_COMSCR	0xF0***n***A

B.3 Programmer Sheets

The following pages provide programmer sheets summarizing functions of the bits in various registers in the 56F8000family found in this manual. The programmer sheets provide room to write in the value of each bit and the hexadecimal value for each register. Programmers may photocopy these sheets.

The programmer sheets are arranged in the same order as the sections in this document. Table B-1 lists the programmer sheets by module, the registers in each module, and the pages in this appendix where the programmer sheets are located.

Note: Reserved bits in all registers should only be set to 0 unless otherwise stated.

Note: Please see the applicable *Device Data Sheet* or *Core Manual* for register information about references to *Data Sht* and *Core Man.* in the following table. Interrupt Controller (ITCN) and System Integration Module (SIM) register are, however, included in this addendum for convenience.

Table B-1. List of Programmer Sheets

Register Type/Name	Base Address/ New Acronym	Page/ Figure
Interrupt Control (ITCN)	**ITCN_BASE: 56F801x = $00F060**	
	ITCN_	
Interrupt Priority 0-4 Register	IPR0-4	Data Sht
Vector Base Address Register	VBA	Data Sht
Fast Interrupt Match 0 Register	FIM0	Data Sht
Fast Interrupt Vector Address Low 0 Register	FIVAL0	Data Sht
Fast Interrupt Vector Address High 0 Register	FIVAH0	Data Sht
Fast Interrupt Match 1 Register	FIM1	Data Sht
Fast Interrupt Vector Address Low 1 Register	FIVAL1	Data Sht
Fast Interrupt Vector Address High 1 Register	FIVAH1	Data Sht
Interrupt Pending 0-2 Register	IRQP0-2	Data Sht
Interrupt Control Register	ICTRL	Data Sht
Enhanced On Chip Emulation (EOnCE)	**EOnCE_BASE: 56F801x = $FFF8A-$FFFFFF**	
	EOnCE_	
External Signal Control Register	OESCR	Core Man
Breakpoint [0] Unit Counter	OBCNTR	Core Man
Breakpoint 1 Unit [0] Mask Register	OBMSK (32 Bits)	Core Man
Breakpoint 2 Unit [0] Address Register	OBAR2 (32 Bits)	Core Man
Breakpoint 1 Unit [0] Address Register	OBAR1 (24 Bits)	Core Man
Breakpoint Unit [0] Control Register	OBCR (24 Bits)	Core Man
Trace Buffer Register Stages	OTB (21-24 Bits/stage)	Core Man
Trace Buffer Pointer Register	OTBPR (8 Bits)	Core Man
Trace Buffer Control Register	OTBCR	Core Man
Peripheral Base Address Register	OBASE (8 Bits)	Core Man

Table B-1. List of Programmer Sheets (Continued)

Register Type/Name	Base Address/ New Acronym	Page/ Figure
Status Register	OSR	Core Man
Instruction Step Counter	OXCNTR (24 Bits)	Core Man
Control Register	OCR	Core Man
Core Lock/Unlock Status Register	OCLSR (8 Bits)	Core Man
Transmit and Receive Status and Control Register	OTXRXSR (8 Bits)	Core Man
Transmit Register/Receive Register	OTX/ORX (32 Bits)	Core Man
Transmit Register Upper Word/Receive Register Upper Word	OTX1/ORX1	Core Man

System Integration Module (SIM)	**SIM_BASE: 56F801x = $00F140**	
	SIM_	
Control Register	CTRL	Data Sht
Reset Status Register	RSTAT	Data Sht
Software Control 0-3 Registers	SWC0-3	Data Sht
Most Significant Half of JTAG ID Register	MSHID	Data Sht
Least Significant Half of JTAG ID Register	LSHID	Data Sht
Power Control Register	PWR	Data Sht
Clock Out Select Register	CLKOUT	Data Sht
Quad Decoder 1/Timer B/SPI1 Select Register	GPS	Data Sht
Peripheral Clock Enable Register	PCE	Data Sht
I/O Short Address Location High Register	IOSAHI	Data Sht
I/O Short Address Location Low Register	IOSALO	Data Sht

Analog to Digital Converter (ADC)	**ADC_BASE: 56F801x = $00F080**	
	ADC_	
Control 1 Register	CTRL1	B-13
Control 2 Register	CTRL2	B-16
Zero Crossing Control Register	ZXCTRL	B-18
Channel List 1 Register	CLST1	B-19
Channel List 2 Register	CLST2	B-20
Sample Disable Register	SDIS	B-21
Status Register	STAT	B-22
Limit Status Register	LIMSTAT	B-24
Zero Crossing Status Register	ZXSTAT	B-25
Result 0-7 Registers	RSLT0–7	B-26
Low Limit 0–7 Register s	LOLIM0–7	B-27
High Limit 0–7 Registers	HILIM0–7	B-27
Offset 0–7 Registers	OFFST0–7	B-28
Power Control Register	PWR	B-29

Table B-1. List of Programmer Sheets (Continued)

Register Type/Name	Base Address/ New Acronym	Page/ Figure
Voltage Reference Register	VREF	B-32
Computer Operating Properly (COP)	**COP_BASE: 56F801x = $00F0E0**	
	COP_	
Control Register	CTRL	B-33
Time-Out Register	TOUT	B-34
Counter Register	CTNR	B-35
Inter-Integrated Circuit (I^2C)	**I^2C_BASE: 56F801x = $00F0D0**	
	I^2C_	
Address Register	ADDR	B-36
Frequency Divider Register	FDIV	B-37
Control Register	CTRL	B-38
Status Register	STAT	B-40
Data I/O Register	DATA	B-42
Noise Filter Register	NFILT	B-43
On Chip Clock Synthesis (OCCS)	**OCCS_BASE: 56F801x = $00F0F0**	
	OCCS_	
Control Register	CTRL	B-44
Divide-By Register	DIVBY	B-46
Status Register	STAT	B-47
Shutdown Register	SHUTDN	B-49
Oscillator Control Register	OCTRL	B-50
Flash Module (FM)	**FM_BASE: 56F801x = $00F400**	
	FM_	
Clock Divider Register	CLKDIV	B-51
Configuration Register	CNFG	B-52
Security High Register	SECHI	B-53
Security Low Register	SECLO	B-53
Protection Register	PROT	B-54
User Status Register	USTAT	B-55
Command Register	CMD	B-57
Data Register	DATA	B-58
Optional Data 1 Register	OPT1	B-59
Test Array Signature Register	TSTSIG	B-60
General Purpose Input/Output (GPIO)	**GPIOA_BASE: 56F801x = See Data Sheet for Address**	

Table B-1. List of Programmer Sheets (Continued)

Register Type/Name	Base Address/ New Acronym	Page/ Figure
	GPIO_	
Pull-Up Enable Register	PUPEN	B-61
Data Register	DATA	B-62
Data Direction Register	DDIR	B-63
Peripheral Enable Register	PEREN	B-64
Interrupt Assert Register	IASSRT	B-65
Interrupt Enable Register	IEN	B-66
Interrupt Polarity Register	IPOR	B-67
Interrupt Pending Register	IPEND	B-68
Interrupt Edge Sensitive Register	IEDGE	B-69
Push/Pull Output Mode Control Register	PPOUTM	B-70
Raw Data Register	RDATA	B-71
Drive Strength Control Register	DRIVE	B-72

Power Supervisor (PS)	**PS_BASE: 56F801x = $00F160**	
	PS_	
Control Register	CTRL	B-73
Status Register	STAT	B-74

Pulse Width Module (PWM)	**PWM_BASE: 56F801x = $00F040**	
	PWM_	
Control Register	CTRL	B-75
Fault Control Register	FCTRL	B-77
Fault Status and Acknowledge Register	FLTACK	B-78
Output Control Register	OUT	B-79
Counter Register	CNTR	B-80
Counter Modulo Register	CMOD	B-81
Value Registers	VAL0-5	B-82
Deadtime Register	DTIM0-1	B-83
Disable Mapping 1 Register	DMAP1	B-84
Disable Mapping 2 Register	DMAP2	B-84
Configuration Register	CNFG	B-85
Channel Control Register	CCTRL	B-87
Port Register	PORT	B-89
Internal Correction Control Register	ICCTRL	B-90
Source Control Register	SCTRL	B-91

Serial Communication Interface (SCI)	**SCI_BASE: 56F801x = $00F0B0**	
	SCI_	

Table B-1. List of Programmer Sheets (Continued)

Register Type/Name	Base Address/ New Acronym	Page/ Figure
Baud Rate Register	RATE	B-94
Control 1 Register	CTRL1	B-95
Control 2 Register	CTRL2	B-99
Status Register	STAT	B-100
Data Register	DATA	B-103
Serial Peripheral Interface (SPI)	**SPI _BASE: 56F801x = $00F0C0**	
SPI_		
Status and Control Register	SCTRL	B-104
Data Size and Control Register	DSCTRL	B-107
Data Receive Register	DRCV	B-108
Data Transmit Register	DXMIT	B-109

Quad Timer (TMR)	**TMR0_BASE: 56F801x = $00F000**	
	TMR1_BASE: 56F801x = $00F010	
	TMR2_BASE: 56F801x = $00F020	
	TMR3_BASE: 56F801x = $00F030	
TMR_		
Compare 1 Registers	COMP1	B-110
Compare 2 Registers	COMP2	B-111
Capture Registers	CAPT	B-112
Load Registers	LOAD	B-113
Hold Registers	HOLD	B-114
Counter Registers	CNTR	B-115
Control Registers	CTRL	B-116
Status and Control Registers	SCTRL	B-121
Comparator Load 1 Registers	CMPLD1	B-123
Comparator Load 2 Registers	CMPLD2	B-124
Comparator Status/Control Registers	CSCTRL	B-125

Voltage Regulator (VREG)	**VREG_BASE: 56F801x = $00F140**	
VREG_		
Power Control Register	PWR	B-126

Interrupt Controller (ITCN)	**ITCN_BASE: 56F801x = $00F060**	
ITCN_		
Interrupt Priority 0 Register	IPR0	B-127
Interrupt Priority 1 Register	IPR1	B-129
Interrupt Priority 2 Register	IPR2	B-131
Interrupt Priority 3 Register	IPR3	B-133

Table B-1.   List of Programmer Sheets (Continued)

Register Type/Name	Base Address/ New Acronym	Page/ Figure
Interrupt Priority 4 Register	IPR4	B-135
Vector Base Address Register	VBA	B-136
Fast Interrupt Match 0 Register	FIM0	B-137
Fast Interrupt 0 Vector Address Low Register	FIVAL0	B-138
Fast Interrupt 0 Vector Address High Register	FIVAH0	B-139
Fast Interrupt 1 Match Register	FIM1	B-140
Fast Interrupt 1 Vector Address Low Register	FIVAL1	B-141
Fast Interrupt 1 Vector Address High Register	FIVAH1	B-142
IRQ Pending 0 Register	IRQP0	B-143
IRQ Pending 1 Register	IRQP1	B-144
IRQ Pending 2 Register	IRQP2	B-145
Interrupt Control Register	ICTRL	B-146

System Integrated Module (SIM)	**SIM_BASE: 56F801x = $00F140**	
SIM_		
Control Register	CTRL	B-147
Reset Status Register	RSTAT	B-149
Software Control 0-3 Registers	SWC0-3	B-150
Most Significant Half of JTAG ID Register	MSHID	B-151
Least Significant Half of JTAG ID Register	LSHID	B-152
Power Control Register	PWR	B-153
GPIO Peripheral Select Register	GPS	B-154
Peripheral Clock Enable Register	PCE	B-157
I/O Short Address Location High Register	IOSAHI	B-158
I/O Short Address Location Low Register	IOSALO	B-159

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 1 of 20

ADC

Control 1 Register (CTRL1)

Please see the following page for continuation of this register

Bits	Name	Description	
14	**STOP0**	**Stop**	
		When selected, the current conversion process is stopped. Any further SYNC0 input pulses or modifications to the START0 bit are ignored until the STOP0 bit is cleared. After the ADC is in Stop mode, the results registers can be modified by the processor. **Note** This is not the same as Stop mode for the whole chip.	
		0	Normal operation
		1	Stop mode
13	**START0**	**Start**	
		A scan is started by writing 1 to the START0 bit. This is a *write-only* bit. Writing 1 to the START0 bit again will be ignored until the end of the current scan. The ADC must be in a stable power configuration prior to writing the START bit. Refer to the functional description of power modes for further details.	
		0	No action
		1	Start command is issued
12	**SYNC0**	**Synchronization**	
		A conversion can be initiated by asserting a positive edge on the SYNC0 input. Any subsequent SYNC0 input pulses while the scan remains in process are ignored. Refer to the functional description of power modes for further details.	
		0	Scan is initiated by a write to START0 bit only
		1	Use the SYNC0 input pulse bit to initiate a scan
11	**EOSIE0**	**End of Scan Interrupt Enable**	
		This bit enables an EOSI0interrupt to be generated upon completion of the scan. For looping scan modes, the interrupt will trigger after the completion of each iteration of the loop.	
		0	Interrupt disabled
		1	Interrupt enabled

Control 1 Register (CTRL1) Base+$0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	STOP0	0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				0	SMODE		
	Write	▨	STOP0	START0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				▨	SMODE		
	Reset	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 2 of 20

ADC

Control 1 Register (CTRL1) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
10	ZCIE	**Zero Crossing Interrupt Enable**	
		This bit enables the zero crossing interrupt if the current result value has a sign change from the previous result, configured by the ZXCTRL register.	
		0	Interrupt disabled
		1	Interrupt enabled
9	LLMTIE	**Low Limit Interrupt Enable**	
		This bit enables the Low Limit exceeded interrupt when the current result value is less than the Low Limit register value. The raw result value is compared to the Low Limit (LOLIM*n*) register, bits LLMT[11:0], before the Offset register value is subtracted.	
		0	Interrupt disabled
		1	Interrupt enabled
8	HLMTIE	**High Limit Interrupt Enable**	
		This bit enables the High Limit exceeded interrupt if the current result value is greater than the High Limit register value. The raw result value is compared to the High Limit (HILIM*n*) register, bits HLMT[11:0], before the Offset register value is subtracted.	
		0	Interrupt disabled
		1	Interrupt enabled
7 - 4	CHNCFG	**Channel Configure**	
		The inputs can be configured for either single-ended or differential conversions.	
		xxx1	Inputs AN0–AN1 configured as differential inputs (AN0 is + and AN1 is -)
		xxx0	Inputs AN0–AN1 configured as single ended inputs
		xx1x	Inputs AN2–AN3 configured as differential inputs (AN2 is + and AN3 is -)
		xx0x	Inputs AN2–AN3 configured as singled ended inputs
		x1xx	Inputs AN4–AN5 configured as differential inputs (AN4 is + and AN5 is -)
		x0xx	Inputs AN4–AN5 configured as singled ended inputs
		1xxx	Inputs AN6–AN7 configured as differential inputs (AN6 is + and AN7 is -)
		0xxx	Inputs AN6–AN7 configured as singled ended inputs

	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Control 1 Register (CTRL1) Base+$0	**Read**	0	STOP	0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				0	SMODE		
	Write	▨	STOP	START	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				▨	SMODE		
	Reset	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1

▨ Reserved Bits

Application:	Date:
	Programmer:
	Sheet 3 of 20

ADC

Control 1 Register (CTRL1) Continued

Bits	**Name**	**Description**	
2 - 0	**SMODE**	**Scan Mode Control**	
		SMODE controls the Scan mode of the ADC module. All scan modes make use of the eight sample slots defined by the CLST1 and CLST2 registers. A scan is the process of stepping through these sample slots, converting the analog input indicated by that slot, and storing the result. Un-required slots may be disabled by writing 1 to the appropriate bits of the SDIS register.	
		000	Once Sequential
		001	Once Parallel
		010	Loop Sequential
		011	Loop Parallel
		100	Triggered Sequential
		101	Triggered Parallel (default)
		110	Reserved use
		111	Reserved use

Control 1 Register (CTRL1) Base+$0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	STOP	0	SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE	CHNCFG				0	SMODE		
	Write	▨		START										▨			
	Reset	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	4 of 20

ADC

Control 2 Register (CTRL2) Under Sequential Scan Modes

Bits	Name	Description
4 - 0	DIV	**Clock Divisor Select**
		Operating mode dependencies occur when the ADC's scan mode (SMODE in the CTRL1 register) is set to Once Sequential, Loop Sequential, or Triggered Sequential bits 15–5 are reserved. Only the DIV field is available.

Control 2 Register (CTRL2) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	DIV				
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨					
	Reset	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 5 of 20

ADC

Control 2 Register (CTRL2) Under Parallel Scan Modes

Bits	Name	Description	
14	STOP1	**Stop**	
		During parallel scan modes when SIMULT=0, setting STOP1 stops parallel scans in the B Converter and prevents new ones from starting. Any further SYNC1 input pulses (please see SYNC1 bit) or writes to the START1 bit are ignored until the STOP1 bit is cleared.	
		0	Normal operation
		1	Stop command issued
13	START1	**Start**	
		During parallel scan modes when SIMULT=0, a B converter parallel scan is started by writing 1 to the START1 bit. This is a *write-only* bit. Writing 1 to the START1 bit again will be ignored until the end of the current scan.	
		0	No action
		1	Start a B Converter parallel scan
12	SYNC1	**Synchronization Enable**	
		During parallel scan modes when SIMULT=0, setting SYNC1 to 1 permits a B Converter parallel scan to be start by asserting the SYNC1 input for at least one ADC clock cycle.	
		0	B Converter parallel scan is initiated by a write to START1 bit only
		1	Use a SYNC1 input pulse or START1 bit to initiate a B Converter parallel scan
11	EOSIE1	**End of Scan Interrupt Enable 1**	
		During parallel scan modes when SIMULT=0, this bit enables an EOSI1 interrupt to be generated upon completion of a B Converter parallel scan. For looping Scan mode, the interrupt will trigger upon the completion of each iteration of the loop.	
		0	Interrupt disabled
		1	Interrupt enabled
5	SIMULT	**Simultaneous Mode**	
		This bit only affects parallel scan modes.	
		0	Parallel scans achieved independently
		1	Parallel scans achieved simultaneously (default)
4 - 0	DIV	**Clock Divisor Select**	
		The divider circuit generates the ADC clock by dividing the system clock by 2× (DIV[4:0]+1). A DIV value must be chosen so the ADC clock does not exceed 5.33MHz. Table 2-7 shows ADC clock frequency based on the value of DIV for these various OCCS configurations.	

Control 2 Register (CTRL2) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	STOP1	0	SYNC1	EOSIE1	0	0	0	0	0	SIMULT	DIV				
	Write			START1													
	Reset	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	6 of 20

ADC

Zero Crossing Control Register (ZXCTRL)

Bits	Name	Description	
15-0	ZCE*n*	**Zero Crossing Enable *n***	
		The ADC Zero Crossing Control (ZXCTRL) register provides the ability to monitor the selected channels and determine the direction of zero crossing triggering the optional interrupt. Zero crossing logic monitors only the sign change between current and previous sample. ZCE0 bit monitors the sample stored in RSLT0, ZCE1 bit monitors RSLT1, ZCE7 bit monitors RSLT7. When the Zero Crossing is disabled for a selected result register, sign changes are not monitored or updated in the ZXSTAT register.	
		00	Zero crossing disabled
		01	Zero crossing enabled for positive to negative sign change
		10	Zero crossing enabled for negative to positive sign change
		11	Zero crossing enabled for any sign change

Zero Crossing Control Register (ZXCTRL) Base+$2	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ZCE7		ZCE6		ZCE5		ZCE4		ZCE3		ZCE2		ZCE1		ZCE0	
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 20

ADC

Channel List Registers (CLST1 and CLST2)

Please see the following page for continuation of this register

Bits	Name	Description
14-12	SAMPLE3	**Sample3**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.
10-8	SAMPLE2	**Sample2**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.
6-4	SAMPLE1	**Sample1**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.
2-0	SAMPLE0	**Sample0**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.

Channel List 1 Register (CLST1) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	SAMPLE3			0	SAMPLE2			0	SAMPLE1			0	SAMPLE0		
	Write	▨				▨				▨				▨			
	Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 20

ADC

Channel List Registers (CLST1 and CLST2) Continued

Bits	Name	Description
14-12	SAMPLE7	**Sample7**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.
10-8	SAMPLE6	**Sample6**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.
6-4	SAMPLE5	**Sample5**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.
2-0	SAMPLE4	**Sample4**
		The Channel List Register contains an ordered list of the analog input channels to be converted when the next scan is initiated. If all samples are enabled in the ADSDIS Register, a sequential scan of inputs proceeds in order of SAMPLE0 through SAMPLE7. If one of the Parallel Sampling modes is selected instead, the Converter A sampling order is SAMPLE0-3 and the Converter B sampling order is SAMPLE4-7.

Channel List 2 Register (CLST2) Base+$4	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	SAMPLE7			0	SAMPLE6			0	SAMPLE5			0	SAMPLE4		
	Write	▨				▨				▨				▨			
	Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

▨ Reserved Bits

Application: ____ Date: ____________________

____ Programmer: ____________________

Sheet 9 of 20

ADC

Sample Disable Register (SDIS)

Bits	**Name**	**Description**	
7-0	**DS*n***	**Disable Sample7-0**	
		The respective SAMPLE*n* can be enabled or disabled where *n* = 0–7. **Note:** When TEST is configured for Test mode, VREF is applied to AN0 and AN4 between the analog muxing and the ADC core. Only AN0 and AN4 will have valid results so only AN0 and AN45 should be sampled.	
		0	Enable SAMPLE*n*
		1	Disable SAMPLE*n* and all subsequent samples. Which samples are actually disabled depends on the conversion mode, sequential/parallel, and the value of SIMULT.

Sample Disable Register (SDIS) Base+$5	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	DS7	DS6	DS5	DS4	DS3	DS2	DS1	DS0
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 10 of 20

ADC

Status Register (STAT)

Please see the following page for continuation of this register

Bits	Name	Description	
15	**CIP0**	**Conversion in Progress 0**	
		This bit indicates when a scan is in progress.	
		0	Idle state
		1	A scan cycle is in progress. The ADC will ignore all sync pulses or start commands.
14	**CIP1**	**Conversion in Progress 1**	
		This bit indicates when a scan is in progress.	
		0	Idle state
		1	A scan cycle is in progress. The ADC will ignore all sync pulses or start commands.
12	**EOSI1**	**End of Scan Interrupt 1**	
		This bit indicates if a scan of analog inputs has been completed since the last read of the STAT register, or since a reset. The EOSI1 bit is cleared by writing 1 to it. This bit cannot be set by software.	
		0	Scan is not completed; no end of scan IRQ pending
		1	Scan cycle is completed; end of scan IRQ pending
11	**EOSI0**	**End of Scan Interrupt 0**	
		This bit indicates if a scan of analog inputs has been completed since the last read of the STAT register, or since a reset. The EOSI0 bit is cleared by writing 1 to it. This bit cannot be set by software. EOSI0 is the preferred bit to poll for scan completion if interrupts are not enabled.	
		0	Scan is not completed; no end of scan IRQ pending
		1	Scan cycle is completed; end of scan IRQ pending
10	**ZCI**	**Zero Crossing Interrupt**	
		This bit is asserted at the completion of an individual conversion experiencing a zero crossing enabled in ADC Zero Crossing Control (ZXCTRL) register. The bit is set as soon as an enabled zero crossing event occurs rather than at the end of the ADC scan.	
		0	No ZCI interrupt request
		1	Zero crossing encountered; IRQ pending if ZCIE is set.

Status Register (STAT) Base+$6	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	CIP0	CIP1	0	EOSI1	EOSI0	ZCI	LLMTI	HLMT	RDY7	RDY6	RDY5	RDY4	RDY3	RDY2	RDY1	RDY0
	Write	▨	▨	▨			▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ____________________ Date: __________

____________________ Programmer: __________

Sheet 11 of 20

ADC

Status Register (STAT) Continued

Bits	Name	Description	
9	LLMTI	**Low Limit Interrupt**	
		If any Low Limit (LOLIM*n*) register is enabled by having a value other than $0000, low limit checking is enabled. This bit is set at the completion of an individual conversion which may or may not be the end of a scan.	
		0	No low limit interrupt request
		1	Low limit exceeded; IRQ pending if LLMTIE is set
8	HLMTI	**High Limit Interrupt**	
		If any High Limit (HILIM*n*) register is enabled by having a value other than 0x7FF8, high limit checking is enabled. This bit is set at the completion of an individual conversion which may or may not be the end of a scan. The HLMTI bit is cleared by writing 1 to all active HLS[7:0] bits in the LIMSTAT register.	
		0	No high limit interrupt request
		1	High limit exceeded; IRQ pending if HLMTIE is set
7-0	RDY*n*	**Ready Sample7-0**	
		These bits indicate samples seven through zero are ready to be read. The RDY*n* bits are set as the individual channel conversions are completed and stored in a RSLTn register. These bits are cleared after a read from the corresponding ADC Results (RSLT*n*) Register. If polling the RDY*n* bits to determine if a particular sample is executed, care should be taken not to start a new scan until all enabled samples are completed. **Note:** RDY*n* bits can be cleared when the debugger reads the corresponding Results register during a debug session.	
		0	Sample not ready or has been read
		1	Sample ready to be read

Status Register (STAT) Base+$6	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	CIP0	CIP1	0	EOSI1	EOSI0	ZCI	LLMTI	HLMT	RDY7	RDY6	RDY5	RDY4	RDY3	RDY2	RDY1	RDY0
	Write	▨	▨	▨			▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 12 of 20

ADC

Limit Status Register (LIMSTAT)

Bits	Name	Description
15-8	HLS*n*	**High Limit Status*n***
		The Limit Status register latches in the result of the comparison between the result of the sample in the RSLT*n* register and the respective Limit register, HILIM*n* and LOLIM*n*.
7-0	LLS*n*	**Low Limit Status*n***
		The Limit Status register latches in the result of the comparison between the result of the sample and the respective limit register, HILIM*n* and LOLIM*n*.

Limit Status Register (LIMSTAT) Base+$7	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	HLS7	HLS6	HLS5	HLS4	HLS3	HLS2	HLS1	HLS0	LLS7	LLS6	LLS5	LLS4	LLS3	LLS2	LLS1	LLS0
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 13 of 20

ADC

Zero Crossing Status Register (ZXSTAT)

Bits	**Name**	**Description**	
7-0	***ZCSn***	**Zero Crossing Status**	
		The zero crossing condition is determined by examining the ADC value after it is adjusted by the offset for the RSLT register. Each bit of the register is cleared by writing 1 to the register bit.	
		0	A sign change did not occur in a comparing the current RSLT*n* value and the previous RSLT*n* value, --OR-- Zero crossing control is disabled for sample n in the ZXCTRL register.
		1	In a comparison between the current channel*n* result and the previous channel*n* result, a sign change occurred as defined in the ZXCTRL register.

Zero Crossing Status Register (ZXSTAT) Base+$8	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	ZCS7	ZCS6	ZCS5	ZCS4	ZCS3	ZCS2	ZCS1	ZCS0
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	__________
	______________________________	Programmer:	__________
		Sheet	14 of 20

ADC

Result 0-7 Registers (RSLT0–7)

Bits	Name	Description
15	SEXT	**Sign Extend**
		SEXT is the sign-extend bit of the result. When the SEXT bit is set to one it implies a negative result. Set to zero, the implication is a positive result. If only positive results are required, the respective OFFST*n* register must be set to a value of zero.
14-3	RSLT	**Digital Result of the Conversion**
		RSLT can be interpreted as either a signed integer or a signed fixed point fractional number. As a fixed point number, the RSLT can be used directly. As a signed integer, one has the option to right shift with sign extend (ASR) three places to fit it into the range [0,4095], Or one can accept the number as presented in the register, knowing there are missing codes because the lower three LSBs are always zero. Negative results (SEXT = 1) are always presented in twos complement format. If it is a requirement of an application, the Result registers always be positive, the Offset register must always be set to zero.
14-3	TEST_DATA	**Test Data**
		When the ADC is stopped or in Power-Down mode this field can be written by accessing the register in the memory map. Please see **Section 2.4.3** more information.

Result Registers (RSLT) Base+$9-$10	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SEXT	RSLT												0	0	0
	Write	▨	TEST_DATA												▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 15 of 20

ADC

Low and High Limit Registers (LOLIM0–7 & HILIM0-7)

Bits	Name	Description
14-3	LLMT	**Low Limit**
		Each ADC sample is compared against the values in the Limit Registers. The comparison is based upon the raw conversion value before the offset correction is applied. Refer to **Figure 2-7**. ADC Limit Registers (LOLIM*n* and HILIM*n)* correspond to Results (RSLT*n*) registers. The High Limit register is used for the comparison of *Result* > *High Limit*. The Low Limit register is used for the comparison of *Result* < *Low Limit*. The limit checking can be disabled by programming the respective limit register with 0x7FF8 for the high limit and 0x0000 for the low limit. At reset, limit checking is disabled.
14-3	HLMT	**High Limit**
		Each ADC sample is compared against the values in the Limit Registers. The comparison is based upon the raw conversion value before the offset correction is applied. Refer to **Figure 2-7**. ADC Limit Registers (LOLIM*n* and HILIM*n)* correspond to Results (RSLT*n*) registers. The High Limit register is used for the comparison of *Result* > *High Limit*. The Low Limit register is used for the comparison of *Result* < *Low Limit*. The limit checking can be disabled by programming the respective limit register with 0x7FF8 for the high limit and 0x0000 for the low limit. At reset, limit checking is disabled.

Low Limit 0-7 Registers (LOLIM0-7) Base+$11-$18	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	LLMT												0	0	0
	Write	▨													▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

High Limit 0-7 Registers (HILIM0-7) Base+$19-$20	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	HLMT												0	0	0
	Write	▨													▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 16 of 20

ADC

Offset Registers (OFFST0–7)

Bits	Name	Description
14-3	OFFSET	**Offset**
		Value of the Offset (OFFST*n*) register is used to correct the ADC result before it is stored in the RSLT*n* registers.

Offset 0-7 Registers (OFFST0-7) Base+$21-$28	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	OFFSET												0	0	0
	Write	▨													▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 17 of 20

ADC

Power Control Register (PWR)

Please see the following page for continuation of this register

Bits	Name	Description	
15	ASB	**Auto Standby**	
		The ASB bit selects Auto Standby mode. ASB is ignored if APD is 1. When the ADC is idle, Auto Standby mode selects the standby clock as the ADC clock source and puts the converters into Standby Current mode. At the start of any scan, the conversion clock is selected as the ADC clock and a delay of PUDELAY ADC clock cycles is imposed for current levels to stabilize. After this delay, the ADC will initiate the scan. When the ADC returns to the idle state, the standby clock is again selected and the converters revert to the standby current state.	
		0	Auto standby mode disabled
		1	Auto standby mode enabled
12	PSTS2	**Voltage Reference Power Status 2**	
		PSTS2 is a *read-only* bit. It simply reflects whether the voltage reference circuit is currently enabled.	
		0	Voltage reference circuit is currently powered up
		1	Voltage reference circuit is currently powered down
11	PSTS1	**Converter B Power Status 1**	
		PSTS1 is a *read-only* bit. It is asserted immediately following a write of 1 to PD1. It is deasserted PUDELAY ADC clock cycles after writing 0 to PD1 if APD is 0. This bit can be read as a status bit to determine when the ADC is ready for operation. During Auto Power-Down mode, this bit indicates the current powered state of Converter B.	
		0	Converter B is currently powered up
		1	Converter B is currently powered down

Power Control Register (PWR) Base+$29	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ASB	0	0	PSTS2	PSTS1	PSTS0	PUDELAY						APD	PD2	PD1	PD0
	Write		▨	▨	▨	▨	▨										
	Reset	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	1

▨ Reserved Bits

Application:	Date:
	Programmer:

Sheet 18 of 20

ADC

Power Control Register (PWR) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
10	PSTS0	**Converter A Power Status 0**	
		PSTS0 is a *read-only* bit. It is asserted immediately following a write of 1 to PD0. It is deasserted PUDELAY ADC clock cycles after writing 0 to PD0 if APD is 0. This bit can be read as a status bit to determine when the ADC is ready for operation. During Auto Power-Down mode, this bit indicates the current powered state of Converter A.	
		0	Converter A is currently powered up
		1	Converter A is currently powered down
9-4	PUDELAY	**Power-Up Delay**	
		This 6-bit field determines the number of ADC clocks provided to power-up an ADC converter (after setting PD0 or PD1 to 0) before allowing a scan to start. It also determines the number of ADC clocks of delay provided in Auto Power-Down (APD) and Auto Standby (ASB) modes between when the ADC goes from the idle to active state and when the scan is allowed to start. The default value is 13 ADC clocks. Accuracy of the initial conversions in a scan will be degraded if PUDELAY is set to too small a value. **Note:** PUDELAY defaults to a value typically sufficient for any power mode. The latency of a scan can be reduced by reducing PUDELAY to the lowest value for which accuracy is not degraded. Please refer to the Device Data Sheet for further details.	

Power Control Register (PWR) Base+$29	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ASB	0	0	PSTS2	PSTS1	PSTS0	PUDELAY						APD	PD2	PD1	PD0
	Write		▨	▨	▨	▨	▨										
	Reset	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 19 of 20

ADC

Power Control Register (PWR) Continued

Bits	Name	Description	
3	APD	**Auto Power-Down**	
		Auto Power-Down mode powers down converters when not in use for a scan. APD takes precedence over ASB. When a scan is started in APD mode, a delay of PUDELAY ADC clock cycles is imposed during which the needed converter(s), if idle, are powered up. The ADC will then initiate a scan equivalent to when APD is not active. When the scan is completed, the converter(s) are powered down again.	
		0	Auto Power-Down mode is not active
		1	Auto Power-Down mode is active
2	PD2	**Power-Down Control for Voltage Reference Circuit 2**	
		This bit controls the power-down of the ADC's voltage reference current. The voltage reference circuit is shared by both converters. When PD2=1 the voltage reference will be activated whenever PD1 or PD0 are powered up. It is not usually necessary to modify this bit, since powering down both Converter A and Converter B will automatically power-down the voltage reference.	
		0	Manually Power-Up voltage reference circuit
		1	Power-Down voltage reference circuit is controlled by PD0 and PD1 (default)
1	PD1	**Manual Power-Down for Converter B**	
		This bit forces ADC Converter B to power-down.	
		0	Power-Up ADC Converter B
		1	Power-Down ADC Converter B
0	PD0	**Manual Power-Down for Converter A**	
		This bit forces ADC Converter A to power-down	
		0	Power-Up ADC Converter A
		1	Power-Down ADC Converter A

Power Control Register (PWR) Base+$29	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ASB	0	0	PSTS2	PSTS1	PSTS0	PUDELAY						APD	PD2	PD1	PD0
	Write		▨	▨	▨	▨	▨										
	Reset	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	1

▨ Reserved Bits

Application: ____ Date: ______________

____ Programmer: ______________

Sheet 20 of 20

ADC

Voltage Reference Register (VREF)

Bits	Name	Description	
15	**SEL_VREFH**	**Select V_{REFH} Source**	
		This bit selects the source of the V_{REFH} reference for all conversions (calibration and regular conversions).	
		0	Internal V_{DDA}
		1	ANA2
14	**SEL_VREFLO**	**Select V_{REFLO} Source**	
		This bit selects the source of the V_{REFLO} reference for all conversions (calibration and regular conversions).	
		0	Internal V_{SSA}
		1	ANB2

Voltage Reference Register (VREF) Base+$2A	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SEL_VREFH	SEL_VREFLO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Write			▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet **1 of 3**

COP

Control Register (CTRL)

Bits	Name	Description	
3	**CSEN**	**COP Stop Mode Enable**	
		This bit controls the operation of the COP counter in Stop mode. This bit can only be changed when CWP is set to zero.	
		0	CNTR will stop in Stop mode (default)
		1	CNTR will run in Stop mode when CEN is set to one
2	**CWEN**	**COP Wait Mode Enable**	
		This bit controls the operation of the COP counter in Wait mode. This bit can only be changed when CWP is set to zero.	
		0	CNTR will stop in Stop mode (default)
		1	CNTR will run in Stop mode when CEN is set to one
1	**CEN**	COP Enable	
		This bit controls the operation of the Counter (CTNR) register. It can only be changed when CWP is set to zero. This bit always reads as 0 when the chip is in Debug mode.	
		0	CNTR is disabled (default)
		1	CNTR is enabled
0	**CWP**	**COP Write Protect**	
		This bit controls the write protection feature of both the COPCTL and Time-Out (COPTO) registers. Once set, this bit can only be cleared by resetting the module.	
		0	CTRL and TOUT can be read and modified by writing (default)
		1	CTRL and TOUT are *read-only*

Control Register (CTRL) Base+$0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	0	0	0	0	CSEN	CWEN	CEN	CWP
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	____________
	______________________________	Programmer:	____________
		Sheet	**2 of 3**

COP

Time-Out Register (TOUT)

Bits	Name	Description
15-0	TIMEOUT	**Time-Out Period**
		The value in this register determines the time-out period of the COP counter. TIMEOUT should be written before enabling COP. Once COP is enabled, the recommended procedure for changing TIMEOUT is to disable the COP, write to the TOUT register, then re-enable the COP. This procedure ensures that the new TIMEOUT is loaded into the counter. Alternatively, the 16-bit controller can write to the TOUT register prior to writing the proper service patterns to the CNTR register, thereby causing the counter to reload with the new TIMEOUT value. The COP Counter is not reset by a write to the TOUT register. Changing TIMEOUT while the COP is enabled results in a time-out period differing from the expected value. These bits can be changed only when CWP is set to zero.

Time-Out Register (TOUT) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TIMEOUT															
	Write																
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 3 of 3

COP

Counter Register (CNTR)

Bits	Name	Description
15-0	COUNT	**Count *(Read-Only)***
		This is the current value of the COP counter as it counts down from the time-out value to 0. A reset is issued when this count reaches zero.
15-0	SERVICE	**Service (*Write-Only)***
		When enabled, the COP requires a service sequence be performed periodically in order to clear its counter and prevent a reset from being issued. This routine consists of writing $5555 to the CNTR register followed by writing $AAAA before the time-out period expires. The writes to the CNTR register must be performed in the correct order, but any number of other instructions (and writes to other registers) may be executed between the two writes.

Counter Register (CNTR) Base+$2	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	COUNT															
	Write	SERVICE															
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 1 of 8

I^2C

Bus Address Register (ADDR)

Bits	Name	Description
7-1	ADR7-1	**Slave Address**
		These bits contain the specific slave address to be used by the I^2C Bus module. The Default mode of the I^2C Bus is Slave mode for an address match on the bus.

Address Register (ADDR) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨								▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	____________________	Date:	__________
	____________________	Programmer:	__________
		Sheet	2 of 8

I^{2}C

Frequency Divider Register (FDIV)

Bits	Name	Description
7-0	IBC7-0	**Bus Clock Rate**
		This bit field is used to prescale the clock for bit rate selection. The bit clock generator is implemented as a prescale divider (IBC7–6), prescaled Shift register (IBC5–3), select the prescaler divider and IBC1–0 select register tap point. The IBC bits are decoded to give the tap and prescale values provided in **Table 4-3** in this manual.

Frequency Divider Register (FDIV) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 3 of 8

I^{2}C

Control Register (CTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
7	**IBEN**	**I-Bus Enable**	
		This bit controls the software reset of the entire I^{2}C Bus module.	
		0	The module is reset and disabled. This is the power-on reset situation. When low, the interface is held in reset but registers can still be accessed.
		1	The I ^{2}C Bus module is enabled. This must be set before any other CTRL register bits have any effect.
6	**IBIE**	**I-Bus Interrupt Enable**	
		0	Interrupts from the I ^{2}C Bus module are disabled. This does not clear any currently pending interrupt condition.
		1	Interrupts from the I ^{2}C Bus module are enabled. An I ^{2}C Bus interrupt occurs provided the IBIF bit in the Status Register is also set.
5	**MS/$\overline{\text{SL}}$**	**I-Bus Master/Slave Mode Select**	
		This bit is cleared upon reset. When this bit is changed from 0 to 1, a Start signal is generated on the bus, and the Master mode is selected. When this bit is changed from 1 to 0, a Stop signal is generated and the Operation mode changes from Master to Slave. A Stop signal should only be generated if the IBIF flag is set. MS/$\overline{\text{SL}}$ is cleared without generating a Stop signal when the master loses arbitration.	
		0	Slave mode
		1	Master mode

	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Control Register (CTRL) Base+$2	**Read**	0	0	0	0	0	0	0	0	IBEN	IBIE	MS/$\overline{\text{SL}}$	TX/$\overline{\text{RX}}$	TXAK	0	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨						RSTA	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______	Date:	______
	______	Programmer:	______
		Sheet	4 of 8

I^2C

Control Register (CTRL) Continued

Bits	Name	Description	
4	$TX/\overline{RX}$	**Transmit Acknowledge Enable**	
		This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the STAT register. In Master mode, this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit will always be high.	
		0	Receive
		1	Transmit
3	TXAK	**Transmit Acknowledge Enable**	
		This bit specifies the value driven onto SDA during acknowledge cycles for both master and slave receivers. The I^2C module will always acknowledge address matches, provided it is enabled, regardless of the value of TXAK. Values written to this bit are only used when the I^2C Bus is a receiver, not a transmitter.	
		0	An acknowledge signal is sent to the bus at the ninth clock bit after receiving one byte data
		1	No acknowledge signal response is sent (i.e., acknowledge bit = 1)
2	RSTA	**Repeat Start**	
		Writing 1 to this bit generates a repeated Start condition on the bus, provided it is the current bus master. This bit will always be read as a low. Attempting a repeated start at the wrong time, if the bus is owned by another master, will result in loss of arbitration.	
		0	No action
		1	Generate repeat Start cycle

Control Register (CTRL) Base+$2	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	IBEN	IBIE	$MS/\overline{SL}$	$TX/\overline{RX}$	TXAK	0	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	IBEN	IBIE	$MS/\overline{SL}$	$TX/\overline{RX}$	TXAK	RSTA	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________	Date:	________
	______________________	Programmer:	________
		Sheet	5 of 8

I^{2}C

Status Register (STAT)

Please see the following page for continuation of this register

Bits	Name	Description	
7	TCF	**Transfer Complete Flag**	
		While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the ninth clock of a byte transfer. This bit is only valid during or immediately following a transfer to/from the I^{2}C module.	
		0	Transfer in progress
		1	Transfer complete
6	IAAS	**I-Bus Addressed As Slave**	
		When its own specific address (I^{2}C Bus Address register) is matched with the calling address, this bit is set before the ACK bit. The Core is interrupted provided the IBIE is set. At this point the CPU needs to check the SRW bit and set its Tx/Rx mode accordingly. Writing to the I^{2}C Bus Control register clears this bit.	
		0	Not addressed or cleared by a write to the Control register
		1	Addressed as a slave
5	IBB	**I-Bus Busy**	
		This bit indicates the status of the bus. When a Start signal is detected, the IBB is set. If a Stop signal is detected, IBB is cleared and the bus enters an idle state.	
		0	Bus is idle
		1	Bus is busy

Status Register (STAT) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	IBAL	▨	▨	IBIF	▨
	Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 6 of 8

I^{2}C

Status Register (STAT) Continued

Bits	Name	Description	
4	IBAL	**I-Bus Arbitration Lost**	
		The I-Bus Arbitration Lost (IBAL) bit is set by hardware when the arbitration procedure is lost. Arbitration is lost in the circumstances listed in **Section 4.6.4.5** of this manual.	
2	SRW	**Slave Read/Write**	
		When IAAS is set this *read-only* bit indicates the value of the R/W command bit of the calling address sent from the Master. This bit is only valid when the I-Bus is in Slave mode, a complete address transfer has occurred with an address match and no other transfers are initiated. Checking this bit, the CPU can select Slave Transmit/Receive mode according to the command of the Master.	
		0	Slave receive, Master writing to Slave
		1	Slave transmit, Master reading from Slave
1	IBIF	**I-Bus Interrupt**	
		This bit is set when one of the following conditions occurs:	
		– Arbitration Lost (IBAL bit set)	
		– Byte transfer complete (TCF bit set)	
		– Addressed as Slave (IAAS bit set)	
0	RXAK	Received Acknowledge	
		The value of SDA during the acknowledge bit of a bus cycle. If the received acknowledge bit (RXAK) is low, it indicates an acknowledge signal was received after the completion of eight bits data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the ninth clock.	
		0	Acknowledge received
		1	No acknowledge received

Status Register (STAT) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	IBAL	▨	▨	IBIF	▨
	Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 8

I^2C

Data I/O Register (DATA)

Bits	Name	Description
7-0	D7-D0	**I-Bus Written Data**
		In Master Transmit mode, when data is written to the I^2C Bus Data I/O (DATA) register a data transfer is initiated. The Most Significant Bit (MSB) is sent first. In Master receive mode, reading this register initiates next byte data receiving. In Slave mode, the same functions are available after an address match occurred. Reading the DATA register returns the last byte received while the I^2C is configured in either master receive or slave receive modes. The DATA register does not reflect every byte transmitted on the I^2C Bus, nor can software verify a byte was written to the DATA register correctly by reading it back. In Master Transmit mode, the first byte of data written to DATA register following assertion of MS/SL is used for the address transfer and should comprise of the calling address (in position D7-D1) concatenated with the required R/W bit (in position D0). **Note:** The Tx/Rx bit in the CTRL register must correctly reflect the desired direction of transfer in Master and Slave modes for the transmission to begin. For instance, if the I^2C is configured for master transmit, but a master received is desired, reading the DATA register will not initiate the receive.

Data I/O Register (DATA) Base+$4	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	D7	D6	D5	D4	D3	D2	D1	D0
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 8

I²C

Noise Filter Register (NFILT)

Bits	Name	Description
3-0	NF3-NF0	**Noise Filter Count**
		The Filter Clock frequency and NF counts dictate the size of glitch suppression. The size of glitch suppressed is less than NF/(Filter Clock frequency). The filter can be completely bypassed by setting the Noise Filter to Hex 0.

Noise Filter Register (NFILT) Base+$5	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	0	NF3	NF2	NF1	NF0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 1 of 7

OCCS

Control Register (CTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
15-14	PLLIE1	**PLL Interrupt Enable 1**	
		An optional interrupt can be generated when the PLL Lock (LCK1) status bit in the PLL Status (STAT) register changes:	
		00	Disable interrupt
		01	Enable interrupt on any rising edge of LCK1
		10	Enable interrupt on falling edge of LCK1
		11	Enable interrupt on any edge change of LCK1
13-12	PLLIE0	**PLL Interrupt Enable 0**	
		An optional interrupt can be generated if the PLL Lock (LCK0) status bit in the PLL Status (STAT) register changes:	
		00	Disable interrupt
		01	Enable interrupt on any rising edge of LCK0
		10	Enable interrupt on falling edge of LCK0
		11	Enable interrupt on any edge change of LCK0
11	LOCIE	**Loss of Reference clock Interrupt Enable**	
		Loss of the reference clock circuit monitors the output of the selected clock source. In the event of reference clock loss, an interrupt can be generated.	
		0	Interrupt disabled
		1	Interrupt enabled
7	LCKON	**Lock Detector On**	
		0	Lock detector disabled
		1	Lock detector enabled

Control Register (CTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	PLLIE1		PLLIE0		LOCIE	0	0	0	LCKON	CHPMPTRI	0	PLLPD	0	PRECS	ZSRC	
	Write						▨	▨	▨			▨		▨			
	Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 2 of 7

OCCS

Control Register (CTRL) Continued

Bits	Name	Description	
6	CHPMPTRI	**Charge Pump Tri-State**	
		During normal chip operation the CHPMPTRI bit should be set to a value of zero. In the event of loss of reference clock, the CHPMPTRI bit must be set to a value of one.	
		0	Normal operation
		1	Isolates the charge pump from the loop filter allowing the PLL output to slowly drift, thereby providing enough time to shutdown the chip. Activating this bit will render the PLL inoperable and should not be executed during standard operation of the chip.
4	PLLPD	**PLL Power-Down**	
		The PLL can be turned off by setting the PLLPD bit. There is a four IPBus clock delay from changing the bit to signaling the PLL. When the PLL is powered down, the gear shifting logic automatically switches to ZSRC=1, preventing loss of reference clock to the core.	
		0	PLL enabled
		1	PLL powered down
2	PRECS	**Prescaler Clock Select**	
		This bit is used to select between the external clock source or the internal relaxation oscillator. **Note:** This bit should not be set unless the external reference (CLKIN) is enabled.	
		0	Relaxation Oscillator selected (reset value)
		1	External reference selected
1-0	ZSRC	**Clock Source**	
		The Clock Source (ZSRC) determines the SYS_CLK_x2 source to the SIM module, generating divided down versions of this signal for use by memories and the IPBus. ZSRC is automatically set to one during Stop mode, or when PLLPD is set, preventing loss of the reference clock to the core. For the 160 family parts, ZSRC may have the following values.	
		00	Reserved
		01	MSTR_OSC
		10	Postscaler output
		11	Reserved

Control Register (CTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	PLLIE1		PLLIE0		LOCIE	0	0	0	LCKON	CHPMPTRI	0	PLLPD	0	PRECS	ZSRC	
	Write						▨	▨	▨			▨		▨			
	Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 3 of 7

OCCS

Divide-By Register (DIVBY)

Bits	Name	Description	
15-12	LORTP	**Loss of Reference Clock Timer Period**	
		These bits control the amount of time required for the loss of reference clock interrupt to be generated. This failure detection time is LORTP × 10 × PLL-output-clock-time-period. The Loss of Reference Clock Detector block counts FOUT/2 clocks continuously, illustrated in **Figure 5-1** of this manual. The MSTR_OSC clock input resets this counter. If the counter ever exceeds LORTP × 10 the loss of reference clock interrupt is generated.	
10-8	PLLCOD	**PLL Clock Out Divide or Postscaler**	
		The PLL output clock can be divided down by a 2-bit postscaler, defined in **Figure 5-1** in this manual. The output of the postscaler is a selectable clock source for the core as determined by the ZSRC bit in the CTRL register. Legal combinations of PLL settings may be found in **Table 5-2** also in this manual.	
		000	Divide-by one
		001	Divide-by two
		010	Divide-by four
		011	Divide-by eight
		100	Divide-by sixteen
		101	Divide-by thirty-two
		11x	Divide-by thirty-two

Divide-By Register (DIVBY) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	LORTP				0	PLLCOD			0	0	0	0	0	0	0	0
	Write					▨				▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ____________________ Date: __________

____________________ Programmer: __________

Sheet 4 of 7

OCCS

Status Register (STAT)

Please see the following page for continuation of this register

Bits	**Name**	**Description**	
15	**LOLI1**	**Loss of Lock Interrupt**	
		LOLI1 displays the status of the lock detector state from LCK1 circuit. The interrupt is cleared by writing 1 to LOLI1. **Note:** This bit will not be set by the hardware if the corresponding CTRL register PLLIE1 bit is cleared or set to 0.	
		0	PLL locked
		1	PLL unlocked
14	**LOLI0**	**Loss of Lock Interrupt 0**	
		LOCI shows the status of the reference clock detection circuit. **Note:** This bit will not set by the hardware if the CTRL register PLLIE0 bit is cleared or set to 0.	
		0	Oscillator clock normal
		1	Lost oscillator clock
13	**LOCI**	**Loss of Reference Clock Interrupt**	
		LOCI shows the status of the reference clock detection circuit.	
		0	Oscillator clock normal
		1	Lost oscillator clock

Status Register (STAT) Base+$2	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	LOLI1	LOLI0	LOCI	0	0	0	0	0	0	LCK1	LCK0	PLLPDN	0	0	ZSRCS	
	Write				▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

▨ Reserved Bits

Application:	Date:
	Programmer:
Sheet	5 of 7

OCCS

Status Register (STAT) Continued

Bits	Name	Description	
6	**LCK1**	**Loss of Lock 1**	
		0	PLL is unlocked
		1	PLL is locked (fine)
5	**LCK0**	**Loss of Lock 0**	
		0	PLL is unlocked
		1	PLL is locked (coarse)
4	**PLLPDN**	**PLL Power Down**	
		PLL power-down status is delayed by four IPBus clocks from the PLLPD bit in the CTRL register.	
		0	PLL not powered down
		1	PLL powered down
1-0	**ZSRCS**	**Clock Source Status**	
		ZSRCS indicates the current SYS_CLK_x2 clock source. Since the synchronizing circuit switches the system clock source, ZSRCS takes more than one IPBus clock to indicate the new selection.	
		00	Synchronizing in progress
		01	Prescaler output
		10	Postscaler output
		11	Synchronizing in progress

Status Register (STAT) Base+$2	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	LOLI1	LOLI0	LOCI	0	0	0	0	0	0	LCK1	LCK0	PLLPDN	0	0	ZSRCS	
	Write				▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

▨ Reserved Bits

Application:	______________________________	Date: ________
	______________________________	Programmer: ________
		Sheet 6 of 7

OCCS

Shutdown Register (SHUTDN)

Bits	Name	Description
16-0	SHUTDOWN	**Shutdown**
		This register can be used to shutdown all system clocks, including SYS_CLK_×2, SYS_CLK. **CAUTION:** *This is a terminal condition.* The *only* recovery is to assert the RST. This function is intended to place the device in a known state in the following specific circumstance:
		– The Loss of Reference (LOR) interrupt is, and continues to be, asserted AND
		– The CHPMPTRI bit in the CTRL register is set, AND
		– The value 0xDEAD is written to this register
		This procedure occurs when the chip clock source or crystal input dies, an LOR interrupt is issued, and the PLL is placed in a mode where it ignores the reference clock input and continues to run open loop. The PLL is designed to run at the target frequency for at least 1000 cycles. During this time, the LOR routine can shutdown the system in a controlled manner, terminating in actually shutting down the system clocks as well. A write of 0×DEAD, or any other value, to this register will have no effect unless the first two conditions above are met. If the LOR is clear, or CHPMPTRI is not set, no action is taken and the register is not written. Clock shutdown may take several cycles, so the write to this register should be followed by an idle loop, simply branching to itself.

Shutdown Register (SHUTDN) Base+$4	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SHUTDOWN															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 7

OCCS

Oscillator Control Register (OCTRL)

Bits	Name	Description	
15	**ROPD**	**Relaxation Oscillator Power-Down**	
		This bit operates the power-down usage of the crystal oscillator. It may be power-down the relaxation oscillator if the external reference is being used. In order to prevent loss of clock to the core or the PLL, this bit should only be asserted if the clock source is changed to the external source by setting the PRECS bit in the CTRL register.	
		0	Relaxation Oscillator enabled
		1	Relaxation Oscillator powered down
14	**ROSB**	**Relaxation Oscillator Standby**	
		This bit controls the power usage and gross frequency of the relaxation oscillator. It is reset to the more accurate but higher power state.	
		0	Normal mode. The relaxation oscillator output frequency is 8MHz.
		1	Standby mode. The relaxation oscillator output frequency is reduced to 400KHz (±50%). The PLL should be disabled in this mode and MSTR_OSC should be selected as the output clock.
9-0	**TRIM**	**Internal Relaxation Oscillator Trim**	
		These bits change the size of the internal capacitor used by the internal relaxation oscillator. By testing the frequency of the internal clock and incrementing or decrementing this factor accordingly, the accuracy of the internal clock can be improved by 40%. Incrementing these bits by one increases the clock period by 0.078% of the unadjusted value. Decrementing this register by one decreases the clock period by 0.078%. Reset sets these bits to $200, centering the range of possible adjustment.	

Oscillator Control Register (OCTRL) Base+$5	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	ROPD	ROSB	0	0	0	0	TRIM									
	Write			▨	▨	▨	▨										
	Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	1 of 10

FLASH

Clock Divider Register (CLKDIV)

Bits	Name	Description	
7	DIVLD	**Clock Divider Loaded**	
		This is a status-only bit and may not receive writing in any mode. Writing has no effect.	
		0	Register has not been written
		1	Register has received writing since the last reset
6	PRDIV8	**Enable Prescaler By 8**	
		This is a read and write bit.	
		0	Prescaler divides oscillator clock by 1
		1	Prescaler divides oscillator clock by 8
5-0	DIV	**Clock Divider**	
		The Clock Divider register bits PRDIV8 and DIV must be set with appropriate values before programming or erasing the FM array. Because FCLK is re-timed into the system clock domain, the values of PRDIV8 and DIV are affected by the system bus frequency as well. Refer to the Functional Description of Writing the CLKDIV register for the detailed algorithm for determining the settings for DIV and PRDIV8. Bits 6-0 are not write-protected once written; and they can be modified even after being initialized.	

Clock Divider Register (CLKDIV) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	DIVLD	PRDIV8	DIV					
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨							
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 2 of 10

FLASH

Configuration Register (CNFG)

Bits	Name	Description	
10	LOCK	**Write Lock Control**	
		This bit can always be read. In user mode, it is a set-once field. Once set, it can't be cleared except by reset. In Test mode, the LOCK bit will always receive writing. This bit provides additional security for the Flash array by disabling writes to the protection register.	
		0	The PROT register can be modified by writing
		1	The PROT register is write-locked
8	AEIE	**Access Error Interrupt Enable**	
		This read/write bit enables an interrupt in case of an ACCERR flag being set.	
		0	ACCERR interrupt s disabled
		1	An interrupt is requested whenever the ACCERR flag is set
7	CBEIE	**Command Buffer Empty Interrupt Enable**	
		This read/write bit enables an interrupt in case of an empty command buffer in the Flash.	
		0	Command Buffer Empty interrupts disabled
		1	An interrupt is requested whenever the CBEIF flag is set
6	CCIE	**Command Complete Interrupt Enable**	
		This read/write bit enables an interrupt in case of all commands being completed in the Flash.	
		0	Command complete interrupts disabled
		1	An interrupt is requested whenever the CCIF flag is set
5	KEYACC	**Enable Security Key Writing**	
		This bit can be read; however, it can receive writing if the KEYEN bit in the SECHI register is set.	
		0	Flash writes are interpreted as the start of a program or erase sequence
		1	Writes to Flash are interpreted as keys to open the back door

	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Configuration Register (CNFG) Base+$1	Read	0	0	0	0	0	LOCK	0	AEIE	CBEIE	CCIE	KEYACC	0	0	0	0	0
	Write	▨	▨	▨	▨	▨		▨					▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______	Date:	______
	______	Programmer:	______
		Sheet	3 of 10

FLASH

Security Registers (SECHI and SECLO)

Bits	Name	Description	
15	KEYEN	**Enable Back Door Key to Security**	
		0	Back door to Flash is disabled
		1	Back door to Flash is enabled
14	SECSTAT	**Security Status**	
		0	Flash Security is disabled
		1	Flash Security is enabled

Security High Register (SECHI) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	KEYEN	SECSTAT	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	F[1]	F[2]	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. Reset state loaded from Flash array during reset.
2. Reset state determined by security state of module.

Bits	Name	Description
15	SEC	**Security**
		Value loaded into The Security (SEC) bits from the configuration field at reset in turn determines the state of Flash Security at reset (SECSTAT). **Table 6-6** in this manual outlines the single code enabling the security feature in the Flash Memory.

Security Low Register (SECLO) Base+$4	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SEC															
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

▨ Reserved Bits

Application:	______________________________	Date:	__________
	______________________________	Programmer:	__________
		Sheet	4 of 10

FLASH

Protection Register (PROT)

Bits	Name	Description
15-0	PROTECT	**Protection Register**
		This register may always be read, but may be modified by writing only when LOCK=0. This register value is reset to the PROT_VALUE in the PROT register; however, this is only possible if the LOCK bit in CNFG is zero. To change the Flash protection loaded on reset, the sector [15] of program Flash must first be unprotected as just described above, then the protection word in the Configuration Field at addresses defined in **Table 6-1** in this manual must be programmed with the desired value.

Protection Register (PROT) Base+$10	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	PROTECT															
	Write																
	Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

Application: ____________________ Date: __________

____________________ Programmer: __________

Sheet 5 of 10

FLASH

User Status Register (USTAT)

Please see the following page for continuation of this register

Bits	Name	Description	
7	CBEIF	**Command Buffer Empty Interrupt Flag**	
		The CBEIF flag indicates the address, data, and command buffers are empty allowing a new command sequence to be started. The CBEIF flag is cleared by writing 1. Clearing the flag results in the CMD, DATA, and ADDR registers being transferred to the internal state machine for launch of a command. Writing 0 has no effect on CBEIF, but it can be used to abort a command sequence. The CBEIF bit can generate an interrupt if the CBEIE bit in the CNFG register is set. While the CBEIF flag is clear the CMD, DATA, and ADDR registers cannot be modified by writing.	
		0	Buffers are full
		1	Buffers are ready to accept a new command
6	CCIF	**Command Complete Interrupt Flag**	
		The CCIF flag indicates there are no other pending commands. The CCIF flag is set and cleared automatically upon start and completion of a command. Writing to CCIF has no effect. The CCIF bit can generate an interrupt if the CCIE bit in the CNFG register is set.	
		0	Command in progress
		1	All commands are completed
5	PVIOL	**Protection Violation**	
		The PVIOL flag indicates an attempt was made to program or erase an address in a protected Flash memory area. The PVIOL flag is cleared by writing 1. Writing 0 has no effect on PVIOL. While the PVIOL flag is set, it is not possible to launch another command.	
		0	No Failure
		1	A protection violation has occurred

User Status Register (USTAT) Base+$13	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	CBEIF	▨	PVIOL	ACCERR	▨	BLANK	▨	▨
	Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 6 of 10

FLASH

User Status Register (USTAT) Continued

Bits	Name	Description	
4	ACCERR	**Access Error**	
		The ACCERR flag indicates an illegal access to the FM array or registers caused by a bad program or erase sequence. The ACCERR flag is cleared by writing 1. Writing 0 to ACCERR bit has no effect. While the ACCERR flag is set, it is not possible to launch another command. The ACCERR relates to FM array writes from the 56F801XE core buses and will not be set by writing directly to the data and address registers from the IPBuses. Please see **Section 6.5.5.3** in this manual to set ACCERR flag details.	
		0	No failure
		1	Access error has occurred
2	BLANK	**Flash Blank Verified Erased**	
		The BLANK flag indicates an Erase Verify command (RDARY1) checked the Flash block and found it to be blank. The BLANK flag is cleared by writing 1. Writing 0 has no effect.	
		0	If an Erase Verify command is requested, and the CCIF flag is set, a zero in BLANK indicates the block is not erased.
		1	Flash block verifies as erased

User Status Register (USTAT) Base+$13	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	CBEIF	▨	PVIOL	ACCERR	▨	BLANK	▨	▨
	Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

▨ Reserved Bits

Application:	____________________	Date:	__________
	____________________	Programmer:	__________
		Sheet	7 of 10

FLASH

Command Register (CMD)

Bits	Name	Description
6-0	COMMAND	**Command**
		Valid User mode commands are shown in **Table 6-7** in this manual. Writing a command in User mode other than those listed in **Table 6-7** will cause the ACCERR flag in the USTAT register to set

Command Register (CMD) Base+$14	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	COMMAND						
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨							
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨	Reserved Bits

Application:	________________	Date:	________
	________________	Programmer:	________
		Sheet	8 of 10

FLASH

Data Register (DATA)

Bits	Name	Description
15-0	DATA	**Data Buffer**
		The results of the Calculate Data Signature and Calculate IFR Block Signature commands are available in this register after execution of these commands.

Data Register (DATA) Base+$18	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	DATA															
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 9 of 10

FLASH

Option Data 1 Register (OPT1)

Bits	Name	Description
9-0	ROTV	**Relaxation Oscillator Trim Value**
		This bit field contains a factory-measured trim value for the relaxation oscillator. Copy this value into the Trim Field of the OCCS OCTRL register.

Optional Data 1 Register (OPT1) Base+$1B	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	RELAXATION OSCILLACTOR TRIM VALUE (ROTV)									
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	10 of 10

FLASH

Test Array Signature Register (TSTSIG)

Bits	Name	Description
15-0	TSTSIG	**Test Array Signature**
		The TSTSIG register stores the Flash Information Block Signature, generated by the Calculate IFR Block Signature command during factory test. The value in the TSTSIG register is compared to the result of the Calculate IFR Block Signature throughout the life of the part to confirm data used by the user commands and internal module adjustments has not been compromised.

Test Array Signature Register (TSTSIG) Base+$1D	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TEST SIGNATURE															
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]	F[1]

1. Reset state loaded from Flash array during reset.

▨ Reserved Bits

Application:	____________________	Date:	________
	____________________	Programmer:	________
		Sheet	1 of 12

GPIO

Pull-Up Enable Register (PUPEN)

Bits	Name	Description	
7-0	PU	**Pull-Up Enable**	
		Table 7-2 provides the state of the pin and Pull-Up register as a function of current peripheral output state and control register values.	
		0	Pull-Up is disabled
		1	Pull-Up is enabled (when DDIR=0 and PEREN=0)

Pull-Up Enable Register (PUPEN) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	PU							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 2 of 12

GPIO

Data Register (DATA)

Bits	Name	Description
7-0	D	**Data**
		This read/write register holds data coming either from the pin or the IPBus. Please see **Table 7-3** in this manual for possible data transfers between the I/O pin and the IPBus.

Data Register (DATA) Base+$1	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	D							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	3 of 12

GPIO

Data Direction Register (DDIR)

Bits	Name	Description
7-0	DD	**Data Direction**
		This read/write register configures the state of the pin as either an input or output when PEREN is set to 0. When DDIR is set to 0, the pin is an input with the pull-up device controlled by the corresponding GPIOn PUPEN. When DDIR is set to 1, the pin is an output. Please see **Table 7-2** in this manual for additional information.

Data Direction Register (DDIR) Base+$2	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	DD							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 4 of 12

GPIO

Peripheral Enable Register (PEREN)

Bits	Name	Description	
7-0	PE	**Peripheral Enable**	
		This read/write register determines the GPIO configuration. When PEREN value is one, the peripheral masters the GPIO pin. This mastership includes configuring the GPIO pin as a required input, with or without pull-up, or output depending on the status of the peripheral output disable and includes data transfers from the pin to the peripheral. Please see **Table 7-2** and **Table 7-3** in this manual for additional information. When the PEREN value is zero the DDIR determines the direction of data flow. The GPIO module is configured for GPIO mode when the PEREN value is zero. In this mode, the GPIO module controls the output enable to the pin and the supplying of any data to be output. Also in GPIO mode, any input data can be read from a GPIO memory mapped register.	
		0	Pin is for GPIO (GPIO mode)
		1	Pin is for peripheral (Normal mode)

Peripheral Enable Register (PEREN) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	PE							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 5 of 12

GPIO

Interrupt Assert Register (IASSRT)

Bits	Name	Description	
7-0	IA	**Interrupt Assert**	
		This read/write register is used for software testing only. An interrupt is asserted when the IASSRT is set to 1. The register is cleared by writing 0s.	
		0	Pin is for GPIO (GPIO mode)
		1	Pin is for peripheral (Normal mode)

Interrupt Assert Register (ASSRT) Base+$4	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	IA							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	6 of 12

GPIO

Interrupt Enable Register (IEN)

Bits	Name	Description	
7-0	IEN	**Interrupt Enable**	
		This read/write register enables or disables the edge detection for any incoming interrupt from the pin. Set this register to 1 for interrupt detection. The interrupts are recorded in the IPEND.	
		0	Interrupt is disabled
		1	Interrupt is enabled

Interrupt Enable Register (IEN) Base+$5	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	IEN							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 12

GPIO

Interrupt Polarity Register (IPOL)

Bits	Name	Description	
7-0	IPOL	**Interrupt Polarity**	
		This read/write register controls the polarity of any external interrupts. The interrupt at the pin is active low when this register is set to 1, the falling edge causing the interrupt. The interrupt at the pin is active high when this register is set to 0, the rising edge causing the interrupt. This is true only when the IEN is set at 1. There is no effect on the interrupt if the IEN is set to 0.	
		0	Interrupt is active high
		1	Interrupt is active low

Interrupt Polarity Register (IPOL) Base+$6	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	IPOL							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 12

GPIO

Interrupt Pending Register (IPEND)

Bits	Name	Description
7-0	IPR	**Interrupt Pending**
		This *read-only* register records any incoming interrupts. This register is read to determine which pin caused an interrupt. Writing 0s into this register will clear it if the interrupt is caused by software. For external interrupts, the IPEND is cleared by writing 1s into the IEDGE register.

Interrupt Pending Register (IPEND) Base+$7	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	IPR							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 9 of 12

GPIO

Interrupt Edge Sensitive Register (IEDGE)

Bits	Name	Description
7-0	IES	**Interrupt Edge**
		This *read-only* register records any incoming interrupts. This register is read to determine which pin caused an interrupt. Writing 0s into this register will clear it if the interrupt is caused by software. For external interrupts, the IPEND is cleared by writing 1s into the IEDGE register.

Interrupt Edge Sensitive Register (IEDGE) Base+$8	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	IES							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 10 of 12

GPIO

Push/Pull Output Mode Control Register (PPOUTM)

Bits	Name	Description	
7-0	OEN	**Push/Pull Output Mode**	
		This read/write register explicitly sets each output driver to either push/pull or Open Drain mode. This is an optional feature of the GPIO design. **Note:** The PPOUTM register can be used to tri-state any pin on the GPIO port without switching that pin to input mode. This is useful for some functions, including the keypad interface. These bits default to one upon reset, giving all GPIO ports the same default functionality in this regard.	
		0	Push/Pull mode
		1	Open drain mode

Push/Pull Output Mode Control Register (PPOUTM) Base+$9	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	OEN							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	____________________	Date:	__________
	____________________	Programmer:	__________
		Sheet	11 of 12

GPIO

Raw Data Register (RDATA)

Bits	Name	Description	
7-0	RAWDATA	**Raw Data**	
		This *read-only* register allows the controller direct access to the logic values on each GPIO pin even when pins are not in the GPIO mode. Values are not clocked and are subject to change at any time. The reset state is unknown. Read several times to assure stable value.	
		0	Logic 0 present on GPIO pin
		1	Logic 1 present on GPIO pin

Raw Data Register (RDATA) Base+$A	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	RAWDATA							
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application:	____________________	Date:	________
	____________________	Programmer:	________
		Sheet	12 of 12

GPIO

Drive Strength Control Register (DRIVE)

Bits	Name	Description	
7-0	DRIVE	**Drive**	
		This register can be used to explicitly set the drive strength of each output driver. **Note:** The DRIVE register can be used to enable high drive strength on certain pins. This is useful for some functions including the PWM outputs. These bits default to zero upon reset providing all GPIO ports with the same default functionality.	
		0	4mA drive strength outputs
		1	8mA drive strength outputs

Drive Strength Control Register (DRIVE) Base+$B	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	DRIVE							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 1 of 2

PS

Control Register (CTRL)

Bits	Name	Description	
1	LVIE27	**2.7V Low Voltage Interrupt Enable**	
		This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.	
		0	Interrupt is disabled
		1	Interrupt is enabled
0	LVIE22	**22V Low Voltage Interrupt Enable**	
		This bit field is reserved or not implemented. It is read as 0 and cannot be modified by writing.	
		0	Interrupt is disabled
		1	Interrupt is enabled

Control Register (CTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LVIE 27	LVIE 22
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 2 of 2

PS

Status Register (STAT)

Bits	Name	Description	
4	**LVI**	**Low Voltage Interrupt**	
		LVI is a sticky status bit derived from the combination of LVIS27 and LVIE27, or LVIS22 and LVIE22. Write 1 to this bit to clear it. It may be set again immediately if voltages are still low. Writing 0 has no effect.	
		0	Interrupt not asserted
		1	Interrupt asserted
3	**LVIS27S**	**Sticky 2.7V Low Voltage Interrupt Source**	
		When this bit is set it must be cleared explicitly by writing 1 to it. Writing 0 has no effect.	
		0	2.7V interrupt threshold has not been passed
		1	3.3V supply has dropped below the 2.7V interrupt threshold
2	**LVIS22S**	**Sticky 2.2V Low Voltage Interrupt Source**	
		When this bit is set, it must be cleared explicitly by writing 1 to it. Writing 0 has no effect.	
		0	2.2V interrupt threshold has not been passed
		1	2.5V supply dropped below the 2.2V interrupt threshold
1	**LVIS27**	**Non-Sticky 2.7V Low Voltage Interrupt Source**	
		This bit may reset itself if the supply voltage raises above the comparator threshold point. This bit cannot be modified.	
		0	2.7V interrupt threshold has not been passed
		1	3.3V supply has dropped below the 2.7V interrupt threshold
0	**LVIS22**	**Non-Sticky 2.2V Low Voltage Interrupt Source**	
		This *read-only* bit will reset itself if the supply voltage raises above the comparator threshold point. This bit cannot be modified.	
		0	2.2V interrupt threshold has not been passed
		1	2.5V supply is below the 2.2V interrupt threshold

Status Register (STAT) Base+$1	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	0	0	0	LVI	LVIS27S	LVIS22S	LVIS27	LVIS22
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨				▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 1 of 19

PWM

Control Register (CTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
15-12	LDFQ	**Load Frequency**	
		These buffered read/write bits select the PWM load frequency according to **Table 10-10**. Reset clears the LDFQ bits, selecting loading every PWM opportunity. The occurrence of a PWM opportunity is determined by the half bit.	
11	HALF	**Half Cycle Reload**	
		This read/write bit enables half cycle reloads in Center-Aligned PWM mode. This bit has no effect on Edge-Aligned PWMs.	
		0	Half cycle reloads disabled
		1	Half cycle reloads enabled
10	IPOL2	**Current Polarity 2**	
		This buffered read/write bit selects the PWM value register for the PWM4 and PWM5 pins in top/bottom manual deadtime correction.	
		0	VAL4 register in next PWM cycle
		1	VAL5 register in next PWM cycle
9	IPOL1	**Current Polarity 1**	
		This buffered read/write bit selects the PWM value register for the PWM2 and PWM3 pins in top/bottom manual deadtime correction.	
		0	VAL2 register in next PWM cycle
		1	VAL3 register in next PWM cycle
8	IPOL0	**Current Polarity 0**	
		This buffered read/write bit selects the PWM value register for the PWM0 and PWM1 pins in top/bottom manual deadtime correction.	
		0	VAL0 register in next PWM cycle
		1	VAL1 register in next PWM cycle

Control Register (CTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	LDFQ				HALF	IPOL2	IPOL1	IPOL0	PRSC		PWMRIE	PWMF	ISENS		LDOK	PWMEN
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 2 of 19

PWM

Control Register (CTRL) Continued

Bits	Name	Description	
7-6	PRSC	**Prescaler**	
		These buffered read/write bits select the PWM clock frequency illustrated in **Figure 10-10** in this manual.	
5	PWMRIE	**PWM Reload Interrupt Enable**	
		This read/write bit enables the PWMF flag to generate interrupt requests.	
		0	PWMF bit interrupt request disabled
		1	PWMF bit interrupt request enabled
4	PWMF	**PWM Reload Flag**	
		This read/write flag is set at the beginning of every reload cycle regardless of the state of the LDOK bit. Clear PWMF by reading it, then write a Logic 0 to it. If another reload occurs before the clearing sequence is complete, writing Logic 0 to PWMF has no effect.	
		0	No new reload cycle since last PWMF clearing
		1	New reload cycle since last PWMF clearing
3-2	ISENS	**Current STatus**	
		These read/write bits select the top/bottom deadtime correction scheme, illustrated in **Figure 10-12** in this manual. Reset clears the ISENS*n* bits. This selects manual correction or no correction or automatic deadtime correction.	
1	LDOK	**Load Okay**	
		This read/write bit loads the prescaler bits of CTRL and the entire CMOD registers and VALn registers into a set of buffers. The buffered prescaler divisor, PWM counter modulus value, and PWM pulse width take effect at the next PWM reload. Set LDOK by reading it, then write a Logic 1 to it. LDOK is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a Logic 0 to it.	
		0	No action is taken
		1	Load prescaler, PWM modulus and PWM values into buffers
0	PWMEN	**PWM Enable**	
		This read/write bit enables the PWM generator. When PWMEN=0, the PWM outputs are in their inactive status unless OUT*n*=1.	
		0	PWM generator and PWM outputs disabled unless OUT*n*=1
		1	PWM generator and PWM outputs enabled

Control Register (CTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	LDFQ				HALF	IPOL2	IPOL1	IPOL0	PRSC		PWMRIE	PWMF	ISENS		LDOK	PWMEN
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 3 of 19

PWM

Fault Control Register (FCTRL)

Bits	**Name**	**Description**	
7,5,3,1	**FIE*n***	**FAULT*n* Pin Interrupt Enable**	
		This read/write bits enable the interrupt request generated by the FAULT*n* pin.	
		0	FAULT*n* interrupt request disabled
		1	FAULT*n* interrupt request enabled
6,4,2,0	**FMODE*n***	**FAULT*n* Pin Clearing Mode**	
		These read/write bits select automatic or manual clearing of FAULT*n* pin faults.	
		0	Manual fault clearing of FAULT*n* pin faults
		1	Automatic fault clearing of FAULT*n* pin fault

Fault Control Register (FCTRL) Base+$1	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	FIE3	FMODE3	FIE2	FMODE2	FIE1	FMODE1	FIE0	FMODE0
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________	Date:	________
	______________________	Programmer:	________
		Sheet	4 of 19

PWM

Fault Status and Acknowledge Register (FLTACK)

Bits	Name	Description	
15,13, 11, 9	**FPIN*n***	**FAULT*n* Pin**	
		These *read-only* bits reflect the current state of the filtered FAULT*n* bit. A reset has no effect on FPIN*n*.	
		0	Logic 0 on the FAULT*n* bit
		1	Logic 1 on the FAULT*n* bit
14, 12, 10, 8	**FFLAG*n***	**FAULT*n* Pin Flag**	
		These *read-only* flags are set within two IPBus cycles after a rising edge on the FAULT*n* pin. Clear FFLAG*n* by writing a Logic 1 to the FTACK*n* bits in the FLTACK register. A reset clears FFLAG*n*.	
		0	No fault on the FAULT*n* bit
		1	Fault on the FAULT*n* bit
6, 4, 2, 0	**FTACK*n***	**FAULT*n* Pin Acknowledge**	
		Writing a Logic 1 to FTACKn clears FFLAG*n*. Writing a Logic 0 has no effect. Reading these bits reads the appropriate DT*n* bit. Please see**Section 10.10.3.4** in this manual. Reset clears FTACK*n*. The fault protection is enabled even when the PWM is not enabled; therefore if a fault is latched, it must be cleared prior to enabling the PWM to prevent an unexpected interrupt.	
5-0	**DTn**	**Deadtime*n***	
		These are *read-only* bits. The DT*n* bits are grouped in pairs, DT0 and DT1, DT2 and DT3, DT4 and DT5. Each pair reflects the corresponding IS*n* pin value as sampled during deadtime period. A reset clears these bits.	

Fault Status & Acknowledge Register (FLTACK) Base+$2	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1	FPIN0	FFLAG0	0	0	DT5	DT4	DT3	DT2	DT1	DT0
	Write	▨	▨	▨	▨	▨	▨	▨	▨		FTACK3	▨	FTACK2	▨	FTACK1	▨	FTACK0
	Reset	U	0	U	0	U	0	U	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 5 of 19

PWM

Output Control Register (OUT)

Bits	Name	Description	
15	PAD_EN	**Output Pad Enable**	
		The PWM output pads can be enabled or disabled by setting the PAD_EN bit. The power-up default has the pads disabled. This bit does not affect the functionality of the PWM, so the PWM module can be energized with the output pads disabled. This enable is to power-up with a safe default value for the PWM drivers.	
		0	Output pads disabled (tri-stated)
		1	Output pads enabled (not tri-stated)
13-8	OUTCTL5-0	**Output Control Enables**	
		These read/write bits enable software control of their corresponding PWM pin. When OUTCTL*n* is set, the OUT*n* bit activates and deactivates the PWM*n* output. A reset clears the OUTCTL bits. When operating the PWM in Complementary mode, these bits must be switched in pairs for proper operation. Please see**Section 10.5** in this manual for details.	
		0	Software control disabled
		1	Software control enabled
5-0	OUT5-0	**Output Control**	
		When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins, illustrated in **Table 10.12** in this manual.	

Output Control Register (OUT) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	PAD_EN	0	OUT CTL5	OUT CTL4	OUT CTL3	OUT CTL2	OUT CTL1	OUT CTL0	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
	Write		▨							▨	▨						
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 6 of 19

PWM

Counter Register (CNTR)

Bits	Name	Description
14-0	**CNT**	**Counter**
		This *read-only* bit field displays the state of the 15-bit PWM counter.

Counter Register (CNTR) Base+$4	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	CNT														
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 19

PWM

Counter Modulo Register (CMOD)

Bits	Name	Description
14-0	CM	**Counter Modulo**
		The 15-bit unsigned value written to this buffered read/write register defines the number of PWM clocks to use in determining the PWM period. Do not write a modulus value of zeros into these bits. **Note:** The PWM counter modulo register is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading CMOD register reads the value in a buffer. It is not necessarily the value the PWM generator is currently using.

Counter Modulo Register (CMOD) Base+$5	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	CM														
	Write	▨															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 8 of 19

PWM

Value Registers (VAL0-5)

Bits	Name	Description
15-0	VAL	**Value**
		The PWM Value registers are buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading VAL*n* register reads the value in a buffer and not necessarily the value the PWM generator is currently using. A PWM value less than, or equal to zero, deactivates the PWM output for the entire PWM period. A PWM value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. Please see **Table 10.1** in this manual. **Note:** The terms activate and deactivate refer to the high and low logic states of the PWM outputs.

Value Registers (VAL0-5) Base+$6-$B	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	VAL															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 9 of 19

PWM

Deadtime 0-1 Registers (DTIM0-1)

Bits	Name	Description
11-0	PWMDT0-1	**Deadtime 0-1**
		The 12-bit value written to this write-protectable register is the number of PWM clock cycles in complementary channel operation. A reset sets the Deadtime (DTIM) register to a default value of 0×0FFF, selecting a deadtime of 4096-PWM clock cycles minus one IPBus clock cycle. This register is write protected after the Write Protect (WP) bit in the PWM configuration register is set. Please refer to **Section 10.10.10** in this manual. **Note:** Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT=P × PWMDT - 1 IPBus clocks, where DT is deadtime, P is the prescaler value, PWMDT is the programmed value of deadtime. For example: if the prescaler is programmed for a divide-by-two and PWMDT is set to five, then P=2 and the deadtime value is equal to: DT = 2 × 5 - 1 = 9 IPBus clock cycles

Deadtime Register (DTIM0) Base+$C	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	PWMDT0											
	Write	▨	▨	▨	▨												
	Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Deadtime Register (DTIM1) Base+$D	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	PWMDT1											
	Write	▨	▨	▨	▨												
	Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application:	Date:
	Programmer:
	Sheet 10 of 19

PWM

Disable Mapping 1-2 Registers (DMAP1-2)

Bits	Name	Description
15-0	DISMAP1	**Disable Mapping 1**
23-16	DISMAP2	**Disable Mapping 2**
		These *write-protectable* registers determine which PWM pins are disabled by the fault protection inputs, provided in **Table 10-6** in this manual. Reset sets all of the bits used in the DMAP1-2 registers. These registers are write-protected after the WP bit in the CNFG register is set. Reserved bits 15-8 in the DMAP2 register cannot be modified. The bits are read as 0.

Disable Mapping Register (DMAP1) Base+$E	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	DISMAP1[15:0]															
	Write																
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Disable Mapping Register (DMAP2) Base+$F	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	DISMAP2[23:16]							
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application: ____________________ Date: __________

____________________ Programmer: __________

Sheet 11 of 19

PWM

Configure Register (CNFG)

Please see the following page for continuation of this register

Bits	Name	Description	
14	DBG_EN	**Debug Enable**	
		When this *write-protectable* bit is set to 1, the PWM will continue to run while the chip is in EOnCE Debug mode. If the device enters the EOnCE mode and this bit is 0, the PWM outputs will be switched to their inactive state until the EOnCE mode is exited. At that point the PWM pins will resume operation as programmed in the PWM registers. **Note:** PWM parameter updates will not occur in the EOncE Debug mode.	
13	WAIT_EN	**Wait Enable**	
		When this *write-protectable* bit is set to 1, the PWM will continue to run while the chip is in the Wait mode. In this mode, the peripheral clock continues to run; however, the core clock does not. If the device enters the Wait mode and this bit is 0, the PWM outputs will be switched to their inactive state until the Wait mode is exited. At the point of exit the PWM pins will resume operation as programmed in the PWM registers. **Note:** PWM parameter updates will not occur in Wait mode.	
12	EDG	**Edge-Aligned or Center-Aligned PWMs**	
		This *write-protectable* bit determines whether all PWM channels will use Edge-Aligned or Center-Aligned wave forms.	
		0	Center-Aligned PWMs
		1	Edge-Aligned PWMs
10-8	TOPNEG	**Top-Side PWM Polarity**	
		This *write-protectable* bit determines the polarity for the top-side PWMs.	
		0	Positive top-side polarity
		1	Negative top-side polarity

Configure Register (CNFG) Base+$10	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	DBG_EN	WAIT_EN	EDG	0	TOP NEG45	TOP NEG23	TOP NEG01	0	BOT NEG45	BOT NEG 23	BOT NEG01	INDEP 45	INDEP 23	INDEP 01	WP
	Write	▨				▨				▨							
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	__________
	______________________________	Programmer:	__________
		Sheet	12 of 19

PWM

Configure Register (CNFG) Continued

Bits	Name	Description	
6-4	BOTNEG	**Bottom-Side PWM Polarity**	
		This *write-protectable* bit determines the polarity for the bottom-side PWMs.	
		0	Positive bottom-side polarity
		1	Negative bottom-side polarity
3-1	INDEP	**Independent or Complement Pair Operation**	
		This *write-protectable* bit determines if the PWM channels will be independent PWMs or complementary PWM pairs. **Note:** Each pair of PWM channels can be configured: Channel 0-1, Channel 2-3, and Channel 4-5.	
		0	Complementary PWM pair
		1	Independent PWMs
0	WP	**Write Protect**	
		This bit enables write-protection for all write-protectable registers. While clear, WP allows write-protected registers and bits to be written. When set, WP prevents writes to write-protectable registers and bits. Once set, WP can be cleared only by a reset. Write-protectable registers and bits include: DMAP1–2, DTIM, CNFG, and the ENHA bit in the CCTRL register. The VLMODE, SWP45, SWP23, and SWP01 bits in the CCTRL are protected when the Enable Hardware Acceleration (ENHA) bit is set to zero in the CCTRL. ENHA is in turn, protected by setting the WP bit in the CNFG. **Note:** The write to CNFG setting the WP bit is the last write accepted to the register until reset.	
		0	Write-protectable registers may be written
		1	Write-protectable register are write protected

Configure Register (CNFG) Base+$10	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	DBG_EN	WAIT_EN	EDG	0	TOPNEG45	TOPNEG23	TOPNEG01	0	BOTNEG45	BOTNEG23	BOTNEG01	INDEP45	INDEP23	INDEP01	WP
	Write	▨				▨				▨							
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 13 of 19

PWM

Channel Control Register (CCTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
15	**ENHA**	**Enable Hardware Acceleration**	
		This bit enables writing to the nBX, VLMODE, SWP45, SWP23, and SWP01 bits. The bit is write-protectable by the WP bit in the CNFG register.	
		0	Disable writing to nBX, VLMODE, SWP45, SWP23, and SWP01 bits
		1	Enable writing to nBX, VLMODE, SWP45, SWP23, and SWP01 bits
14	**nBX**	**56F80x Compatibility**	
		This bit is used to enable/disable improved SWAP and MASK operations. If it is cleared, SWAP/MASK operates identical to the 56F80x version of this module. See 56F80x User's Manual for details. If it is set, SWAP and MASK operations are described in this manual which are not compatible features with 56F80 and are not supported. This bit is write-protected when ENHA is zero.	
		0	SWAP and MASK provide 56F80x compatible operation
		1	SWAP*n* and MASK*n* provide new functionality described in this manual
13-8	**MSK5-0**	**Mask**	
		These six bits determine the mask for each of the PWM logical channels.	
		0	Unmasked
		1	Masked, channel deactivated
5-4	**VLMODE**	**Value Register Load Mode**	
		These two bits determine the way the Value registers are being loaded. These bits are write-protected when ENHA is zero.	
		00	Each VAL*n* register is accessed independently
		01	Writing to VAL0 register also writes to registers VAL1-5
		10	Writing to VAL0 register also writes to registers VAL1-3
		11	Reserved

Channel Control Register (CCTRL) Base+$11	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ENHA	nBX	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	0	0	VLMODE		0	SWP 45	SWP 23	SWP 01
	Write									▨	▨			▨			
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date: ____________
	______________________________	Programmer: ____________
		Sheet 14 of 19

PWM

Channel Control Register (CCTRL) Continued

Bits	Name	Description	
2	**SWP45**	**Swap45**	
		This bit is write-protected when ENHA is zero.	
		0	No swap
		1	Channels four and five are swapped
1	**SWP23**	**Swap23**	
		This bit is write-protected when ENHA is zero.	
		0	No swap
		1	Channels two and three are swapped
0	**SWP01**	**Swap01**	
		This bit is write-protected when ENHA is zero.	
		0	No swap
		1	Channels zero and one are swapped

Channel Control Register (CCTRL) Base+$11	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ENHA	nBX	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	0	0	VLMODE		0	SWP 45	SWP 23	SWP 01
	Write									▨	▨			▨			
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	__________
	______________________________	Programmer:	__________
		Sheet	15 of 19

PWM

Port Register (PORT)

Bits	Name	Description
6-0	PORT	**Port**
		This bit field is *read-only*; therefore any writes to them are not affected.

Port Register (PORT) Base+$12	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	IS2	IS1	IS0	FAULT3	FAULT2	FAULT1	FAULT0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	U	U	U	U	U	U	U

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 16 of 19

PWM

Internal Correction Control Register (ICCTRL)

Bits	Name	Description	
2	**ICC2**	**Internal Current Control 2**	
		This bit controls the PWM4/PWM5 pair.	
		0	VAL*n* register use is determined by the ISENS[0:1] setting. The current direction sensed is captured in the DT4 and DT5 bits of the FLTACK register.
		1	Use VAL4 register when the PWM counter is counting up. Use VAL5 register when counting down.
1	**ICC1**	**Internal Current Control 1**	
		This bit controls the PWM2/PWM3 pair.	
		0	VAL*n* register use is determined by the ISENS[0:1] setting. The current direction sensed is captured in the DT2 and DT3 bits of the FLTACK register.
		1	Use VAL2 register when the PWM counter is counting up. Use VAL3 register when counting down.
0	**ICC0**	**Internal Current Control 0**	
		This bit controls the PWM0/PWM1 pair.	
		0	VAL*n* register use is determined by the ISENS[0:1] setting. The current direction sensed is captured in the DT0 and DT1 bits of the FLTACK register.
		1	Use VAL0 register when the PWM counter is counting up. Use VAL1 register when counting down.

Internal Correction Control Register (ICCTRL) Base+$13	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	0	0	0	0	0	ICC2	ICC1	ICC0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨			
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________ Date: __________

______________________ Programmer: __________

Sheet 17 of 19

PWM

Source Control Register (SCTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
13	**CINV5**	**Compare Invert 5**	
		This bit control when the PWM waveform is high. When this bit is zero, the output operates as shown in **Figure 10-7** and **Figure 10-8** in this manual. When the bit is set to 1, the output is inverted from what is shown in the above figures.	
		0	PWM output five is high when the CNTR register is less than VAL5
		1	PWM output five is high when the CNTR register is greater than VAL5
12	**CINV4**	**Compare Invert 4**	
		This bit control when the PWM waveform is high. When this bit is zero, the output operates as shown in **Figure 10-7** and **Figure 10-8** in this manual. When the bit is set to 1, the output is inverted from what is shown in the above figures.	
		0	PWM output four is high when the CNTR register is less than VAL4
		1	PWM output four is high when the CNTR register is greater than VAL4
11	**CINV3**	**Compare Invert 3**	
		This bit control when the PWM waveform is high. When this bit is zero, the output operates as shown in **Figure 10-7** and **Figure 10-8** in this manual. When the bit is set to 1, the output is inverted from what is shown in the above figures.	
		0	PWM output three is high when the CNTR register is less than VAL3
		1	PWM output three is high when the CNTR register is greater than VAL3
10	**CINV2**	**Compare Invert 2**	
		This bit control when the PWM waveform is high. When this bit is zero, the output operates as shown in **Figure 10-7** and **Figure 10-8** in this manual. When the bit is set to 1, the output is inverted from what is shown in the above figures.	
		0	PWM output two is high when the CNTR register is less than VAL2
		1	PWM output two is high when the CNTR register is greater than VAL2

Source Control Register (SCTRL) Base+$14	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
	Write	▨	▨														
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 18 of 19

PWM

Source Control Register (SCTRL) Continued

Please see the following page for continuation of this register

Bits	**Name**	**Description**	
9	**CINV1**	**Compare Invert 1**	
		This bit control when the PWM waveform is high. When this bit is zero, the output operates as shown in **Figure 10-7** and **Figure 10-8** in this manual. When the bit is set to 1, the output is inverted from what is shown in the above figures.	
		0	PWM output one is high when the CNTR register is less than VAL1
		1	PWM output one is high when the CNTR register is greater than VAL1
8	**CINV0**	**Compare Invert 0**	
		This bit control when the PWM waveform is high. When this bit is zero, the output operates as shown in **Figure 10-7** and **Figure 10-8** in this manual. When the bit is set to 1, the output is inverted from what is shown in the above figures.	
		0	PWM output zero is high when the CNTR register is less than VAL0
		1	PWM output zero is high when the CNTR register is greater than VAL0
7-5	**SRC2**	**Source 2**	
		This field controls the PWM5/PWM4 pair. Be certain OUTCTL4 and OUTCTL5 bits of the PWM OUT register are set when using these bits.	
		000	Use OUT4 bit as the PWM source
		001	Use ADC input as the PWM source
		010	Use GPIOB4 input as the PWM source
		011	Use TMR3 as the PWM source
		1xx	Use the value selected in SCR0 as the PWM source

Source Control Register (SCTRL) Base+$14	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
	Write	▨	▨														
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 19 of 19

PWM

Source Control Register (SCTRL) Continued

Bits	Name	Description	
4-2	**SRC1**	**Source 1**	
		This field controls the PWM2/PWM3 pair. Be certain OUTCTL2 and OUTCTL3 bits of the OUT register are set when using these bits.	
		000	Use OUT2 bit as the PWM source
		001	Use ADC1 input as the PWM source
		010	Use GPIOB3 input as the PWM source
		011	Use TMR2 as the PWM source
		1xx	Use the value selected in SRC0 as the PWM source
1-0	**SRC0**	**Source 0**	
		This field controls the PWM0/PWM1 pair. Be certain OUTCTL0 and OUTCTL1 bits of the OUT register are set when using these bits.	
		00	Use OUT0 bit as the PWM source
		01	Use ADC0 input as the PWM source
		10	Use GPIOB2 input as the PWM source
		11	Use TMR0 as the PWM source

Source Control Register (SCTRL) Base+$14	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
	Write	▨	▨														
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 1 of 10

SCI

Baud Rate Register (RATE)

Bits	Name	Description
12-0	SBR	**SCI Baud Rate**
		The count in this register determines the baud rate of the SCI. The formula for calculating the baud rate is found on in **Section 11.4.2** of this manual. Contents of the Baud Rate register has a value of 1 to 8191. **Note:** The baud rate generator is disabled until the TE or the RE bits are set for the first time after reset. The baud rate generator is disabled when SBR=0.

Baud Rate Register (RATE) Base+$0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	SBR												
	Write	▨	▨	▨													
	Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	__________
	______________________________	Programmer:	__________
		Sheet	2 of 10

SCI

Control 1 Register (CTRL1)

Please see the following page for continuation of this register

Bits	Name	Description	
15	LOOP	**Loop Select**	
		This bit enables loop operation. Please see **Section 11.5.2** in this manual. The loop operation disconnects the RXD pin from the SCI and the transmitter output goes into the receiver input.	
		0	Normal operation enabled
		1	Loop operation enabled
14	SWAI	**Stop in Wait Mode**	
		The SWAI bit disables the SCI in the Wait mode. Further information is available in **Section 11.5.4.2** of this manual.	
		0	SCI enabled in Wait mode
		1	SCI disabled in Wait mode
13	RSRC	**Receiver Source**	
		When LOOP=1, the RSRC bit determines the internal feedback path for the receiver. See **Section 11.5.2** and **Table 11-7** in this manual for more details.	
		0	Receiver input internally connected to transmitter output
		1	Receiver input connected to TXD pin
12	M	**Data Format Mode**	
		This bit determines whether data characters are eight or nine bits long.	
		0	One START bit, eight data bits, one STOP bit
		1	One START bit, nine data bits, one STOP bit

Control 1 Register (CTRL1) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read / Write	LOOP	SWAI	RSRC	M	WAKE	POL	PE	PT	TEIE	TIIE	RFIE	REIE	TE	RE	RWU	SBK
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 3 of 10

SCI

Control 1 Register (CTRL1) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
11	**WAKE**	**Wake-Up Condition**	
		This bit determines which condition wakes up the SCI. **Note:** Address Mark Wake-Up is not a valid option when the parity function is enabled (PE=1) because the ADDRESS bit and the PARITY bit both occupy the MSB.	
		0	Idle Line Wake-Up
		1	Address Mark Wake-Up (a Logic 1 in the MSB position of a receive data character)
10	**POL**	**Polarity**	
		This bit determines whether to invert the data as it goes from the transmitter to the TXD pin and from the RXD pin to the receiver. All bits, START, DATA, and STOP, are inverted as they leave the Transmit Shift register and before they enter the Receive Shift register. **Note:** It is recommended the POL bit be toggled only when both TE and RE=0.	
		0	Doesn't invert transmit and receive data bits (Normal mode)
		1	Invert transmit and receive data bits (Inverted mode)
9	**PE**	**Parity Enable**	
		This bit enables the parity function. When enabled, the parity function replaces the MSB of the data character with a parity bit. **Note:** Address Mark Wake-Up (WAKE=1) is not a valid option when the parity function is enabled because the ADDRESS bit and the PARITY bit both occupy the MSB.	
		0	Parity function disabled
		1	Parity function enabled

Control 1 Register (CTRL1) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read / Write	LOOP	SWAI	RSRC	M	WAKE	POL	PE	PT	TEIE	TIIE	RFIE	REIE	TE	RE	RWU	SBK
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 4 of 10

SCI

Control 1 Register (CTRL1) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
8	**PT**	**Parity Type**	
		This bit determines whether the SCI generates and checks for even or odd parity of the data bits. With *even parity*, an *even* number of *ones clear* the PARITY bit while an *odd* number of *ones*, *sets* the PARITY bit. However, with *odd parity*, an *odd* number of *ones clear* the PARITY bit while an *even* number of *ones*, *sets* the PARITY bit.	
		0	Even parity
		1	Odd parity
7	**TEIE**	**Transmitter Empty Interrupt Enable**	
		This bit enables the TDRE flag to generate interrupt requests.	
		0	TDRE interrupt requests disabled
		1	TDRE interrupt requests enabled
6	**TIIE**	**Transmitter Idle Interrupt Enable**	
		This bit enables the TIDLE flag to generate interrupt requests.	
		0	TIDLE interrupt requests disabled
		1	TIDLE interrupt requests enabled
5	**RFIE**	**Receiver Full Interrupt Enable**	
		This bit enables the RDRF flag or the OR flag to generate interrupt requests.	
		0	RDRF and OR interrupt requests disabled
		1	RDRF and OR interrupt requests enabled

Control 1 Register (CTRL1) Base+$1	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read / **Write**	LOOP	SWAI	RSRC	M	WAKE	POL	PE	PT	TEIE	TIIE	RFIE	REIE	TE	RE	RWU	SBK
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	____________
	______________________________	Programmer:	____________
		Sheet	5 of 10

SCI

Control 1 Register (CTRL1) Continued

Bits	Name	Description	
4	REIE	**Receive Error Interrupt Enable**	
		This bit enables the Receive Error (RE) flags (NF, PF, FE, and OR) to generate interrupt requests. The status bits can be checked during the error interrupt process.	
		0	Error interrupt requests disabled
		1	Error interrupt requests enabled
3	TE	**Transmitter Enable**	
		This bit enables the SCI transmitter and configures the TXD pin as the SCI transmitter output. The TE bit can be used to queue an idle preamble.	
		0	Transmitter disabled
		1	Transmitter enabled
2	RE	**Receiver Enabled**	
		This bit enables the SCI Receiver.	
		0	Receiver disabled
		1	Receiver enabled
1	RWU	**Receiver Wake-Up**	
		This bit enables the wake-up function, inhibiting further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU. Please refer to **Section 11.4.4.6** in this manual for a description of Receive Wake-Up.	
		0	Normal Operation
		1	Standby state
0	SBK	**Send Break**	
		Setting SBK sends one break character (all Logic 0s, include START, DATA, and STOP). As long as SBK is set, transmitter sends uninterrupted break characters.	
		0	No break characters
		1	Transmit break characters

Control 1 Register (CTRL1) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	LOOP	SWAI	RSRC	M	WAKE	POL	PE	PT	TEIE	TIIE	RFIE	REIE	TE	RE	RWU	SBK
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 6 of 10

SCI

Control 2 Register (CTRL2)

Bits	Name	Description	
3	LIN MODE	**Local Interconnect Network Mode**	
		This bit should only be used in Local Interconnect Network (LIN) applications. During initialization the SBR register should be loaded to a value that is within 15% of the actual master data rate, otherwise 0x00 data may be misinterpreted as a break. If the first character following a break is not the LIN sync character (0x55) the SBR will NOT be adjusted.	
		0	The LIN auto baud feature is disabled and the SBR register will maintain whatever value the processor writes to it.
		1	Enable a search for the break followed by sync char (0x55) from the master LIN device. When the break is detected the following sync character will be used to measure the baud rate of the transmitting master and the SBR register will be automatically reloaded with the value needed to match that baud rate.

Control 2 Register (CTRL2) Base+$2	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	0	LIN MODE	0	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 10

SCI

Status Register (STAT)

Please see the following page for continuation of this register

Bits	Name	Description	
15	TDRE	**Transmit Data Register Empty Flag**	
		This bit is set when the Transmit Shift register receives a character from the DATA register. Clear TDRE by reading the STAT register, then write to the DATA register.	
		0	No character transferred to Transmit Shift register
		1	Character transferred to Transmit Shift register
14	TIDLE	**Transmitter Idle Flag**	
		This bit is set when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TIDLE is set, the TXD pin becomes idle (Logic 1). Clear TIDLE by reading the STAT register, then write to the DATA register.	
		0	Transmission in progress
		1	No transmission in progress
13	RDRF	**Receive Data Register Full Flag**	
		This bit is set when the data in the Receive Shift register transfers to the DATA register. Clear RDRF by reading the STAT register, then read the DATA register.	
		0	Data not available in the DATA register
		1	Received data available in the DATA register
12	RIDLE	**Receiver Idle Line Flag**	
		This bit is set when ten consecutive Logic 1s (if M=0) or eleven consecutive Logic 1s (if M=1) appear on the receiver input. Once the RIDLE flag is cleared by the receiver detecting a Logic 0, a valid frame must again set the RDRF flag before an idle condition can set the RIDLE flag.When the Receiver Wake-Up (RWU) bit is set, an idle line condition does not set the RIDLE flag.	
		0	Receiver input is either active now or has never become active since the RIDLE flag was last cleared by reset
		1	Receiver input has become idle (after receiving a valid frame)

Status Register (STAT) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TDRE	TIDLE	RDRF	RIDLE	OR	NF	FE	PF	0	0	0	0	LSE	0	0	RAF
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 10

SCI

Status Register (STAT) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
11	**OR**	**Overrun Flag**	
		This bit is set when software fails to read the DATA register before the Receive Shift register receives the next frame. The data in the Shift register is lost, but the data already in the DATA register is not affected. Clear OR by reading the STAT register, then write the STAT register with any value.	
		0	No overrun
		1	Overrun
10	**NF**	**Noise Flag**	
		This bit is set when the SCI detects noise on the receiver input. The NF bit is set during the same cycle as the RDRF flag, but it is not set in the case of an overrun. Clear NF by reading the STAT register, then write the STAT register with any value.	
		0	No noise
		1	Noise
9	**FE**	**Framing Error Flag**	
		This bit is set when a Logic 0 is accepted as the STOP bit. The FE bit is set during the same cycle as the RDRF flag but it is not set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading the STAT register, then write the STAT register with any value.	
		0	No framing error
		1	Framing error
8	**PF**	**Parity Error**	
		This bit is set when the Parity Enable (PE) bit is set and the parity of the received data does not match its parity bit. Clear PF by reading the STAT register, then write the STAT register with any value.	
		0	No parity error
		1	Parity error

Status Register (STAT) Base+$3	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	TDRE	TIDLE	RDRF	RIDLE	OR	NF	FE	PF	0	0	0	0	LSE	0	0	RAF
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 9 of 10

SCI

Status Register (STAT) Continued

Bits	Name	Description	
3	**LSE**	**Local Interconnect Network Sync Error**	
		This bit is only active when LIN MODE is set. When LSE is set, an RERR interrupt will occur if REIE is set. LSE is set when a LIN sync search detects a non-sync character (anything other than 0x55). Having this bit set indicates either a protocol error was detected from the Master LIN device or there is a gross mis-match in data rates. This bit is cleared by reading the STAT register with LSE set and then writing the STAT register with any value.	
		0	No error occurred since the SCI LIN MODE was enabled or the bit was last cleared.
		1	A sync error prevented loading of the SBR with a revised value after the break was detected.
0	**RAF**	**Receiver Active Flag**	
		This bit is set when the receiver detects a Logic 0 during the RT1 time period of the START bit search. RAF is cleared when the receiver detects false start bits (usually from noise or baud rate mismatch) or when the receiver detects a preamble.	
		0	No reception in progress
		1	Reception in progress

Status Register (STAT) Base+$3	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	TDRE	TIDLE	RDRF	RIDLE	OR	NF	FE	PF	0	0	0	0	LSE	0	0	RAF
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________ Date: __________

______________________ Programmer: __________

Sheet 10 of 10

SCI

Data Register (DATA)

Bits	Name	Description
8-0	RECEIVE	**Receive Data**
		Writing to these bits loads the transmit data. Reading these bits accesses the receive data. When configured for 8-bit data, bits 7-0 contain the data received.
8-0	TRANSMIT	**Transmit Data**
		Writing to these bits loads the transmit data. Reading these bits accesses the receive data. When configured for 8-bit data, bits 7-0 contain the data received.

Data Register (DATA) Base+$4	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	RECEIVE DATA								
	Write	▨	▨	▨	▨	▨	▨	▨	TRANSMIT DATA								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 1 of 6

SPI

Status and Control Register (SCTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
15-13	**SPR**	**SPI Baud Rate Select**	
		While in the Master mode, these read/write bits select one of eight baud rates depicted in **Table 12-5** in this manual. SPR[2:0] have no effect in Slave mode. Use the formula located in **Section 12.9.1.1** in this manual to calculate the SPI baud rate.	
12	**ERRIE**	**Error Interrupt Enable**	
		This read/write bit enables the MODF, if MODFEN is also set, and OVRF bits to generate interrupt requests.	
		0	MODF and OVRF cannot generate interrupt requests
		1	MODF and OVRF can generate interrupt requests
10	**MODFEN**	**Mode Fault Enable**	
		This read/write bit, when set to 1, allows the MODF flag to be set. If the MODF flag is set, clearing the MODFEN does not clear the MODF flag. If the MODFEN bit is low, the level of the SS pin does not affect the operation of an enabled SPI configured as a master. For an enabled SPI configured as a slave, having MODFEN low only prevents the MODF flag from being set. It does not affect any other part of SPI operation.	
9	**SPRIE**	**SPI Receiver Interrupt Enable**	
		This read/write bit enables interrupt requests generated by the SPI Receiver Full (SPRF) bit. The SPRF bit is set when data transfers from the Shift register to the DRCV register.	
		0	SPRF interrupt requests disabled
		1	SPRF interrupt requests enabled
8	**SPMSTR**	**SPI Master**	
		This read/write bit selects Master or Slave modes operation.	
		0	Slave mode
		1	Master mode

Status and Control Register (SCTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SPR			DSO	ERRIE	MODFEN	SPRIE	SPMSTR	CPOL	CPHA	SPE	SPTIE	SPRF	OVRF	MODF	SPTE
	Write													▨	▨	▨	▨
	Reset	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1

▨ Reserved Bits

Application: ______________________ Date: __________

______________________ Programmer: __________

Sheet 2 of 6

SPI

Status and Control Register (SCTRL) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
7	CPOL	**Clock Polarity**	
		This read/write bit determines the logic state of the SCLK pin between transmissions. To transmit data between SPI modules, the SPI modules must have identical CPOL values.	
		0	Falling edge of SCLK starts transmission
		1	Rising edge of SCLK starts transmission
6	CPHA	**Clock Phase**	
		This read/write bit controls the timing relationship between the serial clock and SPI data. Please see **Figure 12-4** and **Figure 12-6**, both in this manual. To transmit data between SPI modules, the SPI modules must have identical CPHA values. When CPHA=0, the SS pin of the Slave SPI module must be set to Logic 1 between data transmissions, illustrated in **Figure 12-5** in this manual.	
5	SPE	**SPI Enable**	
		This read/write bit enables the SPI module. Clearing SPE causes a partial reset of the SPI. When setting/clearing this bit, no other bits in the SCTRL register should be changed. Failure to follow this statement may result in spurious clocks.	
		0	SPI module disabled
		1	SPI module enabled
4	SPTIE	**SPI Transmit Interrupt Enable**	
		This read/write bit enables interrupt requests generated by the Transmitter Empty (SPTE) bit. SPTE is set when data transfers from the DXMIT register to the Shift register.	
		0	SPTE interrupt requests disabled
		1	SPTE interrupt requests enabled
3	SPRF	**SPI Receiver Full**	
		This *read-only* flag is set each time data transfers from the Shift register to the DRCV register. SPRF generates an interrupt request if the Receiver Interrupt Enable (SPRIE) bit in the SCTRL register is set. This bit is cleared by reading the DRCV register.	
		0	Data Receive (DRCV) register not full
		1	Data Receive (DRCV) register full

Status and Control Register (SCTRL) Base+$0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SPR			DSO	ERRIE	MODFEN	SPRIE	SPMSTR	CPOL	CPHA	SPE	SPTIE	SPRF	OVRF	MODF	SPTE
	Write													▨	▨	▨	▨
	Reset	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 3 of 6

SPI

Status and Control Register (SCTRL) Continued

Bits	Name	Description	
2	**OVRF**	**Overflow**	
		This *read-only* flag is set if software does not read the data in the DRCV register before the next full data enters the Shift register. In an overflow condition, the data already in the DRCV register is unaffected, and the data shifted in last is lost. Clear the Overflow (OVRF) bit by reading the SCTRL register and then reading the DRCV register. OVRF generates a Receiver/Error interrupt if the Error Interrupt Enable (EERIE) bit is set. See **Section 12.8.1** in this manual for more details.	
		0	No overflow
		1	Overflow
1	**MODF**	**Mode Fault**	
		This *read-only* flag is set in a slave SPI if the SS pin goes high during a transmission with the MODFEN bit set. In a master SPI, the MODF flag is set if the SS pin goes low at any time with the MODFEN bit set. Clear the MODF bit by writing a one to the MODF bit when it is set. MODF generates a Receive/Error interrupt if the EERIE bit is set. Please see **Section 12.8.2** in this manual for more details.	
		0	$\overline{\text{SS}}$ pin at appropriate Logic level
		1	$\overline{\text{SS}}$ pin at inappropriate Logic level
0	**SPTE**	**SPI Transmitter Empty**	
		This *read-only* flag is set each time the DXMIT register transfers data into the Shift register. SPTE generates an interrupt request if the Transmit Interrupt Enable (SPTIE) bit in the SCTRL register is set. This bit is cleared by writing to the SPDTR.	
		0	DXMIT register not empty
		1	DXMIT register empty

Status and Control Register (SCTRL) Base+$0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	SPR			DSO	ERRIE	MODFEN	SPRIE	SPMSTR	CPOL	CPHA	SPE	SPTIE	SPRF	OVRF	MODF	SPTE
	Write													▨	▨	▨	▨
	Reset	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 4 of 6

SPI

Data Size and Control Register (DSCTRL)

Bits	Name	Description	
15	WOM	**Wired OR Mode**	
		This control bit is used to select the nature of the SPI pins. When the WOM bit is set, the SPI pins are configured as open-drain drivers with the pull-ups disabled. When the WOM bit is cleared, the SPI pins are configured as push-pull drivers.	
		0	Wired OR mode disabled
		1	Wired OR mode enabled
3-0	DS	**Data Size**	
		Detailed transmission data provided below	

DS [3:0]	Size of Transmission
$0	Not Allowed
$1	2 Bits
$2	3 Bits
$3	4 Bits
$4	5 Bits
$5	6 Bits
$6	7 Bits
$7	8 Bits
$8	9 Bits
$9	10 Bits
$A	11 Bits
$B	12 Bits
$C	13 Bits
$D	14 Bits
$E	15 Bits
$F	16 Bits

Data Size and Control Register (DSCTRL) Base+$1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	WOM	0	0	0	0	0	0	0	0	0	0	0	DS3	DS2	DS1	DS0
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

▨ Reserved Bits

Application:	____________________	Date:	________
	____________________	Programmer:	________
		Sheet	5 of 6

SPI

Data Receive Register (DRCV)

Bits	Name	Description
15-0	**R*n***	**Receive *n***
		This *read-only* register will show the last data word received after a complete transmission. The SPRF bit is set when new data is transferred to this register.

Data Receive Register (DRCV) Base+$2	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 6 of 6

SPI

Data Transmit Register (DXMIT)

Bits	Name	Description
15-0	T*n*	**Transmit *n***
		This *write-only* register holds data to be transmitted. When the Transmitter Empty (SPTE) bit is set, new data should be written to this register. If new data is not written while in the Master mode, a new transaction will not be initiated until this register is written. When in Slave mode, the old data will be re-transmitted. All data should be written with the LSB at bit zero.

Data Transmit Register (DXMIT) Base+$3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Write	T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:		Date:	
		Programmer:	
		Sheet	1 of 16

TMR

Compare 1 Registers (COMP1)

Bits	Name	Description
15-0	COMPARISON1	**Comparison 1**
		These four read/write Timer Compare 1 (COMP1) registers store the value used for comparison with counter value. Their addresses are: Base Address + $0, $10, $20, and $30.

Compare 1 Registers (COMP1)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	COMPARISON 1															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $0, $10, $20, and $30

▨ Reserved Bits

Application: ________________________________ Date: ____________

________________________________ Programmer: ____________

Sheet 2 of 16

TMR

Compare 2 Registers (COMP2)

Bits	Name	Description
15-0	COMPARISON 2	**Comparison 2**
		These four read/write Timer Compare 2 (COMP2) registers store the value used for comparison with counter value. Their addresses are: Base Address + $1, $11, $21, and $31.

Compare 2 Registers (COMP2)	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	COMPARISON 2															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $1, $11, $21, and $31

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 3 of 16

TMR

Capture Registers (CAPT)

Bits	Name	Description
15-0	CAPTURE	**Capture**
		These four read/write Timer Capture (CAPT) registers store the values captured from the counters. Their addresses are: Base Address + $2, $12, $22, and $32.

Capture Registers (CAPT)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	CAPTURE															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $2, $12, $22, and $32

▨ Reserved Bits

Application:	______	Date:	______
	______	Programmer:	______
		Sheet	4 of 16

TMR

Load Registers (LOAD)

Bits	Name	Description
15-0	LOAD	**Load**
		These four read/write Timer Load (LOAD) registers store the values used to load the counter. Their addresses are: Base Address + $3, $13, $23, and $33.

Load Registers (LOAD)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	LOAD															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $3, $13, $23, and $33

▨ Reserved Bits

Application:	Date:
	Programmer:

Sheet 5 of 16

TMR

Hold Registers (HOLD)

Bits	Name	Description
15-0	HOLD	**Hold**
		These four read/write registers store the channel’s value whenever any Timer Counter (CNTR) register is read. Their addresses are: Base Address + $4, $14, $24, and $34.

Hold Registers (HOLD)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	HOLD															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $4, $14, $24, and $34

▨ Reserved Bits

Application:	______________________________	Date:	______________
	______________________________	Programmer:	______________
		Sheet	6 of 16

TMR

Counter Registers (CNTR)

Bits	Name	Description
15-0	COUNTER	**Counter**
		These four read/write Timer Counter (CNTR) registers. Their addresses are: Base Address + $5, $15, $25, and $35.

Counter Registers (CNTR)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	COUNTER															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $5, $15, $25, and $35

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 16

TMR

Control Registers (CTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
15-13	**CM**	**Count Mode**	
		These bits control the basic counting behavior of the counter. Their addresses are: $6, $16, $26, and $36	
		000	Stop mode, no operation
		001	Counting mode, count rising edges of Primary Source
			–Rising edges if IPS=0
			–Falling edges if IPS=1
		010	Edge Count mode, count rising and falling edges of Primary Source
		011	Gated Count mode, count rising edges of Primary Source while secondary input high active
		100	Quadrature Count mode, uses Primary and Secondary Sources
		101	Signed Count mode, count edge of Primary Source
			–IPS=0 then Secondary Input=0; Count up on rising edges of Primary Input
			–IPS=0 then Secondary Input=1; Count down on rising edges of Primary Input
			–IPS=1 then Secondary Input=0; Count up on falling edges of Primary Input
			–IPS=1 then Secondary Input=1; Count down on falling edges of Primary Input
		110	Triggered Count mode, edge of secondary source triggers primary count until compare
		111	Cascaded Counter mode (up/down)

Control Registers (CTRL)	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read / **Write**	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $6, $16, $26, and $36

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 16

TMR

Control Registers (CTRL) Continued

Please see the following page for continuation of this register

Bits	**Name**	**Description**	
12-9	**PCS**	**Primary Count Source**	
		These bits select the primary count source.	
		0000	Counter 0 input pin (T0)
		0001	Counter 1 input pin (T1)
		0010	Counter 2 input pin (T2)
		0011	Counter 3 input pin (T3)
		0100	Counter 0 output (OFLAG0)
		0101	Counter 1 output (OFLAG1)
		0110	Counter 2 output (OFLAG2)
		0111	Counter 3 output (OFLAG3)
		1000	Prescaler (System Clock or 3x System Clock divide by 1)
		1001	Prescaler (System Clock or 3x System Clock divide by 2)
		1010	Prescaler (System Clock or 3x System Clock divide by 4)
		1011	Prescaler (System Clock or 3x System Clock divide by 8)
		1100	Prescaler (System Clock or 3x System Clock divide by 16)
		1101	Prescaler (System Clock or 3x System Clock divide by 32)
		1110	Prescaler (System Clock or 3x System Clock divide by 64)
		1111	Prescaler (System Clock or 3x System Clock divide by 128)

Control Registers (CTRL)	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read / **Write**	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $6, $16, $26, and $36

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application:	________________________________	Date:	____________
	________________________________	Programmer:	____________
		Sheet	9 of 16

TMR

Control Registers (CTRL) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
8-7	SCS	**Secondary Count Source**	
		These bits identify the external input pin to be used as a count command. The selected input can trigger the timer to capture the current value of the CNTR register. The selected input can also be used to specify the count direction. The polarity of the signal can be inverted by the IPS bit of the SCTRL register.	
		00	Counter 0 input pin
		01	Counter 1 input pin
		10	Counter 2 input pin
		11	Counter 3 input pin
6	ONCE	**Count Once**	
		This bit selects Continuous or One-Shot Counting modes.	
		0	Count repeatedly
		1	Count until compare and then stop. If *counting up*, successful compare occurs when the counter reaches a COMP1 value. If *counting down*, successful compare occurs when the counter reaches a COMP2 value.

Control Registers (CTRL)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $6, $16, $26, and $36

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 10 of 16

TMR

Control Registers (CTRL) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
5	**LENGTH**	**Count Length**	
		This bit determines whether the counter counts to the compare value and then re initializes itself to the value specified in the Load register, or the counter continues counting past the compare value, to the binary roll-over.	
		0	Roll-over
		1	Count till compare, then re initialize. If counting up, successful compare occurs when the counter reaches a COMP1 value. If counting down, successful compare occurs when the counter reaches a COMP2 value.
4	**DIR**	**Count Direction**	
		This bit selects either the normal count direction *up*, or the reverse *down* direction.	
		0	Count up
		1	Count down
3	**CO INIT**	**Co-Channel Initialization**	
		This bit enables another counter/timer within the same module to force the re initialization of this counter/timer when it has an active compare event. The Master channel forcing re initialization has MSTR bit set. Please see **Section 13.6.8.10** in this manual.	
		0	Co-Channel counter/timers cannot force a re initialization of this counter/timer
		1	Co-Channel counter/timers can force a re initialization of this counter/timer

Control Registers (CTRL)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $6, $16, $26, and $36

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 11 of 16

TMR

Control Registers (CTRL) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
2-0	**OM**	**Output Mode**	
		This bit field determines the mode of operation for the OFLAG output signal. Unexpected results may occur if OM field is set to use alternating Compare registers (Mode 100) and the ONCE bit are set.	
		000	Assert OFLAG while counter is active
		001	Clear OFLAG output on successful compare
		010	Set OFLAG output on successful compare
		011	Toggle OFLAG output on successful compare
		100	Toggle OFLAG output using alternating Compare registers
		101	Set OFLAG on compare, cleared on secondary source input edge
		110	Set OFLAG on compare, cleared on counter roll-over
		111	Enable Gated Clock output while counter is active

Control Registers (CTRL)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read / Write	CM			PCS				SCS		ONCE	LENGTH	DIR	Co INIT	OM		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $6, $16, $26, and $36

▨ Reserved Bits

56F801X Peripheral Reference Manual, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 12 of 16

TMR

Status and Control Registers (SCTRL)

Please see the following page for continuation of this register

Bits	Name	Description
15	TCF	**Timer Compare Flag**
		This bit is set when a successful compare occurs. Clear the bit by writing 0 to it. TCF asserts every time there is a compare event (either when counter=COMP1 or counter=COMP2).
14	TCFIE	**Timer Compare Flag Interrupt Enable**
		When set, the timer compare interrupt is enabled.
13	TOF	**Timer Overflow Flag**
		This bit is set when the counter rolls over its maximum or minimum value $FFFF or $0000, depending on count direction. Clear the bit by writing 0 to it.
12	TOFIE	**Timer Overflow Flag Interrupt Enable**
		When set, this bit enables interrupts when the TOF bit is set.
11	IEF	**Input Edge Flag**
		This bit is set when a transition occurs on an input selected as a secondary count source while the counter is enabled. Clear the bit by writing 0 to it. **Note:** The Control register's secondary count source determines which external input pin is monitored by the detection circuitry. Please see Input Capture Mode, Section 13.6.8.9, below for how IEF is set.
10	IEFIE	**Input Edge Flag Interrupt Enable**
		When set, this bit enables interrupts if the IEF bit is set.
9	IPS	**Input Polarity Select**
		When set, this bit inverts the input signal polarity.
8	INPUT	**External Input Signal**
		The *read-only* bit reflects the current state of the external input pin selected via the secondary count source after application of the IPS bit.

Status and Control Registers (SCTRL)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TCF	TCFIE	TOF	TOFIE	IEF	IEFIE	IPS	INPUT	CAPTURE MODE		MSTR	EEOF	VAL	0	OPS	OEN
	Write								▨						FORCE		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $7, $17, $27, and $37

▨ Reserved Bits

Application:	________________________	Date:	____________
	________________________	Programmer:	____________
		Sheet	13 of 16

TMR

Status and Control Registers (SCTRL) Continued

Bits	Name	Description	
7-6	CAPTURE MODE	**Input Capture Mode**	
		These bits specify the operation of the CAPT register as well as the operation of the Input Edge Flag (IEF). The input source is the secondary count source.	
5	MSTR	**Master Mode**	
		When set, this bit enables the Compare register function output to be broadcasted to the other counter/timers in the module. This signal then can be used to re initialize the other counters and/or force their OFLAG signal outputs. Other timer channels within the Quad Timer accept the re initialization signal when their COINIT bit is set. Please see **Section 13.6.7.7** in this manual.	
4	EEOF	**Enable External OFLAG Force**	
		When set, this bit enables the Compare register from another counter/timer within the same module to force the state of this counter's OFLAG output signal.	
3	VAL	**Forced OFLAG Value**	
		This bit determines the value of the OFLAG output signal when a software triggered FORCE command occurs, i.e. when FORCE=1, discussed in **Section 13.6.8.13** in this manual.	
2	FORCE	**Force OFLAG Output**	
		This *write-only* bit forces the current value of the VAL bit to be written to the OFLAG output. This bit is always read as 0. The VAL and FORCE bits can be written simultaneously in a single write operation. Write to the FORCE bit only if the counter is disabled.	
1	OPS	**Output Polarity Select**	
		This bit determines the polarity of the OFLAG output signal.	
		0	True polarity
		1	Inverted polarity
0	OEN	**Output Enable**	
		When set, this bit determines the direction of the external pin.	
		0	The external pin is configured as an input
		1	OFLAG output signal will be driven on the external pin. Other timer groups using this external pin as their input will see the OFLAG value even if this bit is clear. The polarity of the signal will be determined by the OPS bit.

Status and Control Registers (SCTRL)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TCF	TCFIE	TOF	TOFIE	IEF	IEFIE	IPS	INPUT	CAPTURE MODE		MSTR	EEOF	VAL	0	OPS	OEN
	Write								▨						FORCE		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $7, $17, $27, and $37

▨ Reserved Bits

Application: ____________________ Date: ____________

____________________ Programmer: ____________

Sheet 14 of 16

TMR

Comparator Load 1 Registers (CMPLD1)

Bits	Name	Description
15-0	COMPARATOR LOAD 1	**Comparator Load 1**
		There are four read/write registers making up the Comparator 1 preload value for the CMPLD1 Register of the corresponding channel in a timer module. Please see **Section 13.4.2** in this manual for additional information. For details regarding the timer settings required to implement variable frequency PWM operation, please refer to **Section 13.5.12.1** in this manual.

Comparator Load 1 Registers (CMPLD1)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	COMPARATOR LOAD 1															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $8, $18, $28, and $38

▨ Reserved Bits

Application:	____________________	Date: __________
	____________________	Programmer: __________
		Sheet 15 of 16

TMR

Comparator Load 2 Registers (CMPLD2)

Bits	Name	Description
15-0	COMPARATOR LOAD 2	**Comparator Load 2**
		There are four read/write registers making up the Comparator 2 preload value for the CMPLD2 register of the corresponding channel in a timer module. Please see **Section 13.4.2** in this manual for additional information. For details regarding how to control the loading of Comparator 2, please refer to **Section 13.6.11.6** in this manual.

Comparator Load 2 Registers (CMPLD2)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	COMPARATOR LOAD 2															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $9, $19, $29, and $39

▨ Reserved Bits

Application: ________________________ Date: ____________

________________________ Programmer: ____________

Sheet 16 of 16

TMR

Comparator Status and Control Registers (SCTRL)

Bits	Name	Description
15-14	DBG_EN	**Debug Actions Enable**
		This bit field allows the TMR module to perform certain actions in response to the chip entering Debug mode.
7	TCF2EN	**Timer Compare 2 Interrupt Enable**
		An interrupt is issued when both this bit and the TCF2 bit are set.
6	TDF1EN	**Timer Compare 1 Interrupt Enable**
		An interrupt is issued when both this bit and the TDF1 bit are set.
5	TCF2	**Timer Compare Flag 2**
		When set, this bit indicates a successful comparison of the timer and COMP2 register has occurred. This bit is sticky, and will remain set until explicitly cleared by writing 0 to this bit location.
4	TCF1	**Timer Compare Flag 1**
		When set, this bit indicates a successful comparison of the timer and COMP1 register has occurred. This bit is sticky, and will remain set until explicitly cleared by writing 0 to this bit location.
3-2	CL2	**Compare Load Control 2**
		This bit field controls when CMP2 is preloaded with the value from CMPLD2.
1-0	CL1	**Compare Load Control 1**
		This bit field control when COMP1 is preloaded with the value from CMPLD1.

Comparator Status and Control Registers (SCTRL)	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	DBG_EN		0	0	0	0	0	0	TCF2EN	TCF1EN	TCF2	TCF1	CL2		CL1	
	Write			▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Base Address + $A, $1A, $2A, and $3A

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 1 of 1

VREG

SIM Power Control Register (SIM_PWR)

Bits	**Name**	**Description**	
1-0	**LRSTDBY**	**Large Regulator Standby Mode**	
		00	Large regulator is in Normal mode
		01	Large regulator is in Standby (reduced power) mode
		10	Large regulator is in Normal mode and the LRSTDBY field is write-protected until the next reset
		11	Large regulator is in Standby mode and the LRSTDBY field is write-protected until the next reset

Power Control Register (SIM_PWR) Base + $8	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LRSTDBY	
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ____________________ Date: ____________

____________________ Programmer: ____________

Sheet 1 of 20

ITCN

Interrupt Priority 0 Register (IPR0)

Please see the following page for continuation of this register

Bits	Name	Description	
15-14	**LVI IPL**	**Low Voltage Interrupt Interrupt Priority Levels**	
		This field is used to set the interrupt priority levels for a peripheral IRQ. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
9-8	**RX_REG IPL**	**EOnCE Receive Register Full Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 1–3 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 1
		10	IRQ is priority level 2
		11	IRQ is priority level 3
7-6	**TX_REG IPL**	**EOnCE Transmit Register Empty Interrupt Priority Level**	
		This field is used to set the interrupt priority levvels for IRQs. This IRQ is limited to priorities 1–3 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 1
		10	IRQ is priority level 2
		11	IRQ is priority level 3

Interrupt Priority 0 Register (IPR0) Base + $0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	LVI IPL		0	0	0	0	RX_REG IPL		TX_REG IPL		TRBUF IPL		BKPT_U IPL		STPCENT IPL	
	Write			▨	▨	▨	▨										
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ____________________ Date: __________

____________________ Programmer: __________

Sheet 2 of 20

ITCN

Interrupt Priority 0 Register (IPR0) Continued

Bits	Name	Description	
5-4	**TRBUF IPL**	**EOnCE Trace Buffer Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 1–3 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
3-2	**BKPT_U IPL**	**EOnCE Breakpoint Unit Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 1–3 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 1
		10	IRQ is priority level 2
		11	IRQ is priority level 3
1-0	**STPCNT IPL**	**EOnCE Step Counter Interrupt Priority Level**	
		This field is used to set the interrupt priority levvels for IRQs. This IRQ is limited to priorities 1–3 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 1
		10	IRQ is priority level 2
		11	IRQ is priority level 3

Interrupt Priority 0 Register (IPR0) Base + $0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	LVI IPL		0	0	0	0	RX_REG IPL		TX_REG IPL		TRBUF IPL		BKPT_U IPL		STPCNT IPL	
	Write			▨	▨	▨	▨										
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 3 of 20

ITCN

Interrupt Priority 1 Register (IPR1)

Please see the following page for continuation of this register

Bits	Name	Description	
15-14	**GPIOB IPL**	**GPIOB Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
13-12	**GPIOC IPL**	**GPIOC Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
11-10	**GPIOD IPL**	**GPIOD Interrupt Priority Level**	
		This field is used to set the interrupt priority levvels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 1 Register (IPR1) Base + $1	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	GPIOB IPL		GPIOC IPL		GPIOD IPL		0	0	FM_CBE IPL		FM_CC IPL		FM_ERR IPL		PLL IPL	
	Write							▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 4 of 20

ITCN

Interrupt Priority 1 Register (IPR1) Continued

Bits	Name	Description	
7-6	**FM_CBE IPL**	**FM Command, Data, Address Buffers Empty Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
5-4	**FM_CC IPL**	**FM Command Complete Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
3-2	**FM_ERR IPL**	**FM Error Interrupt Priority Level**	
		This field is used to set the interrupt priority levvels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
1-0	**PLL IPL**	**FM Error Interrupt Priority Level**	
		This field is used to set the interrupt priority levvels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 1 Register (IPR1) Base + $1	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	GPIOB IPL		GPIOC IPL		GPIOD IPL		0	0	FM_CBE IPL		FM_CC IPL		FM_ERR IPL		PLL IPL	
	Write							▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 5 of 20

ITCN

Interrupt Priority 2 Register (IPR2)

Please see the following page for continuation of this register

Bits	Name	Description	
15-14	SCI_RCV IPL	**SCI Receiver Full Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
13-12	SCI_RERR IPL	**SCI Receiver Error Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
9-8	SCI_TIDL IPL	**SCI Transmitter Idle Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
7-6	SCI_XMIT IPL	**SCI Transmitter Empty Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 2 Register (IPR2) Base + $2	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SCI_PCV IPL		SCI_RERR IPL		0	0	SCI_TIDL IPL		SCI_XMIT IPL		SPI_XMIT IPL		SPI_RCV IPL		GPIOA IPL	
	Write					▨	▨										
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	____________________	Date:	__________
	____________________	Programmer:	__________
		Sheet	6 of 20

ITCN

Interrupt Priority 2 Register (IPR2) Continued

Bits	Name	Description	
5-4	**SPI_XMIT IPL**	**SPI Transmitter Empty Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
3-2	**SPI_RCV IPL**	**SPI Receiver Full Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
1-0	**GPIOA IPL**	**GPIOA Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 2 Register (IPR2) Base + $2	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	SCI_PCV IPL		SCI_RERR IPL		0	0	SCI_TIDL IPL		SCI_XMIT IPL		SPI_XMIT IPL		SPI_RCV IPL		GPIOA IPL	
	Write					▨	▨										
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 20

ITCN

Interrupt Priority 3 Register (IPR3)

Please see the following page for continuation of this register

Bits	Name	Description	
15-14	**ADCA_CC IPL**	**ADCA Conversion Complete Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
13-12	**TMR_3 IPL**	**Timer Channel 3 Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
11-10	**TMR_2 IPL**	**Timer Channel 2 Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 3 Register (IPR3) Base + $3	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	ADCA_CCIPL		TMR_3 IPL		TMR_2 IPL		TMR_1 IPL		TMR_0 IPL		I2C_ADDR IPL		0	0	0	0
	Write													▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 20

ITCN

Interrupt Priority 3 Register (IPR3) Continued

Bits	Name	Description	
9-8	**TMR_1 IPL**	**Timer Channel 1 Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
7-6	**TMR_0 IPL**	**Timer Channel 0 Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
5-4	**I^{2}C_ADDR IPL**	**I^{2}C Address Detect Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 3 Register (IPR3) Base + $3	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ADCA_CCIPL		TMR_3 IPL		TMR_2 IPL		TMR_1 IPL		TMR_0 IPL		I2C_ADDR IPL		0	0	0	0
	Write													▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 9 of 20

ITCN

Interrupt Priority 4 Register (IPR4)

Bits	Name	Description	
7-6	**PWM_F IPL**	**PWM Fault Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
5-4	**PWM_RL IPL**	**PWM Reload Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
3-2	**ADC_ZC_LE IPL**	**ADC Zero Crossing or Limit Error Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2
1-0	**ADCB_CC IPL**	**ADC Conversion Complete Interrupt Priority Level**	
		This field is used to set the interrupt priority levels for IRQs. This IRQ is limited to priorities 0–2 and is disabled by default.	
		00	IRQ disabled (default)
		01	IRQ is priority level 0
		10	IRQ is priority level 1
		11	IRQ is priority level 2

Interrupt Priority 4 Register (IPR4) Base + $4	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	PWM_F IPL		PWM_RL IPL		ADC_ZC_LE IPL		ADCB_CC IPL	
	Write	▨	▨	▨	▨	▨	▨	▨	▨								
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	____________
	______________________________	Programmer:	____________
		Sheet	10 of 20

ITCN

Vector Base Address Register (VBA)

Bits	Name	Description
13-0	VECTOR_BASE _ADDRESS	**Vector Base Address**
		The valaue in this register is used as the upper 14 bits of the interrupt vector VAB[20:0]. The lower 7 bits are determined based on the highest priority interrupt and are then appended onto VBA before presenting the full VAB to the Core.

Vector Base Address Register (VBA) Base + $5	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0 ▨	0 ▨	VECTOR_BASE_ADDRESS													
	Write	▨	▨														
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 11 of 20

ITCN

Fast Interrupt Match 0 Register (FIM0)

Bits	Name	Description
5-0	FAST INTERRUPT 0	**Fast Interrupt 0 Vector Number**
		These values determine which IRQ will be Fast Interrupt 0. Fast Interrupts vector directly to a service routine based on values in the Fast Interrupt Vector Address registers without having to go to a jump table first. IRQs used as Fast Interrupts *must* be set to priority level 2. Unexpected results will occur if a Fast Interrupt vector is set to any other priority. A Fast Interrupt automatically becomes the highest-priority level 2 interrupt regardless of its location in the interrupt table prior to being declared as Fast Interrupt. Fast Interrupt 0 has priority over fast Interrupt 1. To determine the vector number of each IRQ, refer to the vector table.

Fast Interrupt Match 0 Register (FIM0) Base + $6	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	FAST INTERRUPT 0					
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨						
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	____________________	Date:	________
	____________________	Programmer:	________
		Sheet	12 of 20

ITCN

Fast Interrupt 0 Vector Address Low Register (FIVAL0)

Bits	Name	Description
15-0	FAST INTERRUPT 0 VECTOR ADDRESS LOW	**Fast Interrupt 0 Vector Address Low**
		The lower 16 bits of the vector address used for Fast Interrupt 0. This register is combined with FIVAH0 to form the 21-bit vector address for Fast Interrupt 0 defined in the FIM0 register.

Fast Interrupt 0 Vector Address Low Register (FIVAL0) Base + $7	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	FAST INTERRUPT 0 VECTOR ADDRESS LOW															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 13 of 20

ITCN

Fast Interrupt 0 Vector Address High Register (FIVAH0)

Bits	Name	Description
4-0	FAST INTERRUPT 0 VECTOR ADDRESS HIGH	**Fast Interrupt 1 Vector Address High**
		The upper five bits of the vector address used for Fast Interrupt 0. This register is combined with FIVAL0 to form the 21-bit vector address for Fast Interrupt 0 defined in the FIM0 register.

Fast Interrupt 0 Vector Address High Register (FIVAH0) Base + $B	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	FAST INTERRUPT 0 VECTOR ADDRESS HIGH				
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨					
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	14 of 20

ITCN

Fast Interrupt 1 Match Register (FIM1)

Bits	Name	Description
5-0	FAST INTERRUPT 1	**Fast Interrupt 1 Match**
		These values determine which IRQ will be Fast Interrupt 1. Fast Interrupts vector directly to a service routine based on values in the Fast Interrupt Vector Address registers without having to go to a jump table first. IRQs used as Fast Interrupts must be set to priority level 2. Unexpected results will occur if a Fast Interrupt vector is set to any other priority. A Fast Interrupt automatically becomes the highest-priority level 2 interrupt regardless of its location in the interrupt table prior to being declared as Fast Interrupt. Fast Interrupt 0 has priority over Fast Interrupt 1. To determine the vector number of each IRQ, refer to the vector table.

Fast Interrupt1 Match Register (FIM1) Base + $9	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	FAST INTERRUPT 1					
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨						
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	________________	Date:	________
	________________	Programmer:	________
		Sheet	15 of 20

ITCN

Fast Interrupt 1 Vector Address Low Register (FIVAL1)

Bits	Name	Description
15-0	FAST INTERRUPT 1 VECTOR ADDRESS LOW	**Fast Interrupt 1 Vector Address Low**
		The lower 16 bits of the vector are address used for Fast Interrupt 1. This register is combined with FIVAH1 to form the 21-bit vector address for Fast Interrupt 1 defined in the FIM1 register.

Fast Interrupt1 Vector Address Low Register (FIVAL1) Base + $A	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	FAST INTERRUPT 1 VECTOR ADDRESS LOW															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date: ______________
	______________________________	Programmer: __________
		Sheet 16 of 20

ITCN

Fast Interrupt 1 Vector Address High Register (FIVAH1)

Bits	Name	Description
15-0	FAST INTERRUPT 1 VECTOR ADDRESS HIGH	**Fast Interrupt 1 Vector Address High**
		The upper five bits of the vector are address used for Fast Interrupt 1. This register is combined with FIVAL1 to form the 21-bit vector address for Fast Interrupt 1 defined in the FIM1 register.

Fast Interrupt1 Vector Address High Register (FIVAH1) Base + $B	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	FAST INTERRUPT 1 VECTOR ADDRESS HIGH				
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨					
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	______________________________	Date:	__________
	______________________________	Programmer:	__________
		Sheet	17 of 20

ITCN

IRQ Pending 0 Register (IRQP0)

Bits	Name	Description	
15-1	PENDING[16:1]	**Pending**	
		This register combines with the other two to represent the pending IRQs for interrupt vector numbers 2 through 45.	
		0	IRQ pending for this vector number
		1	No IRQ pending for this vector number

IRQ Pending 0 Register (IRQP0) Base + $C	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	PENDING [16:1]															▨ 1
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	18 of 20

ITCN

IRQ Pending 1 Register (IRQP1)

Bits	Name	Description	
15-0	**PENDING [32:17]**	**Pending**	
		This register combines with the other two to represent the pending IRQs for interrupt vector numbers 2 through 45.	
		0	IRQ pending for this vector number
		1	No IRQ pending for this vector number

IRQ Pending 1 Register (IRQP1) Base + $D	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	PENDING [32:17]															
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 19 of 20

ITCN

IRQ Pending 2 Register (IRQP2)

Bits	Name	Description	
12-0	PENDING [45:33]	Pending	
		This register combines with the other two to represent the pending IRQs for interrupt vector numbers 2 through 45.	
		0	IRQ pending for this vector number
		1	No IRQ pending for this vector number

IRQ Pending 2 Register (IRQP2) Base + $E	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	1	1	1	PENDING [45:33]												
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application:	______________________	Date:	__________
	______________________	Programmer:	__________
		Sheet	20 of 20

ITCN

Interrupt Control Register (ICTRL)

Bits	Name	Description	
15	**INT**	**Interrupt**	
		This read-only bit reflects the state of the interrupt to the 56800E core.	
		0	No interrupt is being sent to the 56800E core
		1	An interrupt is being sent to the 56800E core
14-13	**IPIC**	**Interrupt Priority Level IRQ**	
		These *read-only* bits reflect the state of the new interrupt priority level bits being presented to the 56800E core. These bits indicate the priority level needed for a new IRQ to interrupt the current interrupt being sent to the 56800E core. This field is only updated when the 56800E core jumps to a new interrupt service routine.	
		00	Required nested exception priority levels are 0, 1, 2, or 3
		01	Required nested exception priority levels are 1, 2, or 3
		10	Required nested exception priority levels are 2 or 3
		11	Required nested exception priority level is 3
12-6	**VAB**	Vector Address Bus	
		This *read-only* field shows the vector number (VAB[7:1]) used at the time the last IRQ was taken. In the case of a Fast Interrupt, it shows the lower address bits of the jump address. This field is only updated when the 56800E core jumps to a new interrupt service routine.	
5	**INT_DIS**	**Interrupt Disable**	
		This bit allows all interrupts to be disabled.	

Interrupt Control Register (ICTRL) Base + $12	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	INT	IPIC		VAB							INT_DIS	1	1	1	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

▨ Reserved Bits

Application:	Date:
	Programmer:

Sheet 1 of 13

SIM

Control Register (SIM_CTRL)

Please see the following page for continuation of this register

Bits	Name	Description	
15	**TC3_SD**	**Timer Channel 3 Stop Disable**	
		This bit enables the operation of the Timer Channel 3 peripheral clock in Stop mode.	
		0	Timer Channel 3 disabled in Stop mode
		1	Timer channel 3 enabled in Stop mode
14	**TC2_SD**	**Timer Channel 2 Stop Disable**	
		This bit enables the operation of the Timer Channel 2 peripheral clock in Stop mode.	
		0	Timer Channel 2 disabled in Stop mode
		1	Timer channel 2 enabled in Stop mode
13	**TC1_SD**	**Timer Channel 1 Stop Disable**	
		This bit enables the operation of the Timer Channel 1 peripheral clock in Stop mode.	
		0	Timer Channel 1 disabled in Stop mode
		1	Timer channel 1 enabled in Stop mode
12	**TC0_SD**	**Timer Channel 0 Stop Disable**	
		This bit enables the operation of the Timer Channel 0 peripheral clock in Stop mode.	
		0	Timer Channel 0 disabled in Stop mode
		1	Timer channel 0 enabled in Stop mode
11	**SCI_SD**	**SCI Stop Disable**	
		This bit enables the operation of the SCI peripheral clock in Stop mode. This is recommended for use in LIN mode so the SCI can generate interrupts and recover from Stop mode while the LIN interfacre is in Sleep mode and using Stop mode to reduce power consumption.	
		0	SCI disabled in Stop mode
		1	SCI ensabled in Stop mode

Control Register (SIM_CTRL) Base + $0	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TC3_SD	TC2_SD	TC1_SD	TC0_SD	SCI_SD	0	TC3_INP	0	0	0	ONCE EBL	SW RST	STOP_DISABLE		WAIT_DISABLE	
	Write						▨		▨	▨	▨						
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	____________________	Date:	________
	____________________	Programmer:	________
		Sheet	2 of 13

SIM

Control Register (SIM_CTRL) Continued

Bits	Name	Description	
9	**TC3_INP**	**Timer Channel 3 Input**	
		This bit selects the input of Timer Channel 3 to be from the PWM or GPIO.	
		0	Time Channel 3 Input controlled by SIM_GPS register, CFG_B3 and CFG_A5 fields
		1	Timer Channel 3 Input from PWM reload_sync signal
5	**ONCEEBL**	**OnCE Enable**	
		0	OnCE clock to 56800E core enabled when core TAP is enabled
		1	OnCE clock to 56800E core is alwasy enabled
4	**SW RST**	**Software Reset**	
		Writing 1 to this field will cause the part to reset	
3-2	**STOP_DISABLE**	**Stop Disable**	
		00	Stop mode will be entered when the 56800E core executes a STOP instruction
		01	The 56800E STOP instruction will not cause entry into Stop mode
		10	Stop mode will be entered when the 56800E core executes a STOP instruction and the STOP_DISABLE field is write-protected until the next reset
		11	The 56800E STOP instruction will not cause entry into Stop mode and the STOP_DISABLE field is write-protected until the next reset
1-0	**WAIT_DISABLE**	**Wait Disable**	
		00	Wait mode will be entered when the 56800E core executes a WAIT instruction
		01	The 56800E WAIT instruction will not cause entry into Wait mode
		10	Wait mode will be entered when the 56800E core executes a WAIT instruction and the WAIT_DISABLE field is write-protected until the next reset
		11	The 56800E WAIT instruction will not cause entry into Wait mode and the WAIT_DISABLE field is write-protected until the next reset

Control Register (SIM_CTRL) Base + $0	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	TC3_ SD	TC2_ SD	TC1_ SD	TC0_ SD	SCI_ SD	0	TC3_ INP	0	0	0	ONCE EBL	SW RST	STOP_ DISABLE		WAIT_ DISABLE	
	Write						▨		▨	▨	▨						
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ____________

______________________________ Programmer: ____________

Sheet 3 of 13

SIM

Reset Status Register (SIM_RSTAT)

Bits	Name	Description
5	SWR	**Software Reset**
		When set, this bit indicates the previous system reset occurred as a result of a software reset (written 1 to SW RST bit in the SIM_CTRL register). It will not be set if a COP, external, or POR reset also occurred.
4	COPR	**COP Reset**
		When set, this bit indicates the previous system reset was caused by the Computer Operating Properly (COP) timer. It will not be set if an external or POR reset also occurred. If COPR is set as code starts executing, the COP reset vector in the vector table will be used. Otherwise, the normal reset vector is used.
3	EXTR	**External Reset**
		When set, this bit indicates that the previous system reset was caused by an external reset. It will only be set if the external reset pin was asserted or remained asserted after the power-on reset deasserted.
2	POR	**Power-On Reset**
		This bit is set during a power-on reset.

Reset Status Register (SIM_RSTAT) Base + $1	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	SWR	COPR	EXTR	POR	0	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨					▨	▨
	Reset	0	0	0	0	0	0	0	0	0	0					0	0

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ____________________ Date: __________

____________________ Programmer: __________

Sheet 4 of 13

SIM

Software Control Registers (SIM_SWC0-3)

Bits	Name	Description
15-0	SOFTWARE CONTROL DATA 0-3	**Softare Control Data 0-3**
		Only SIM_SWC0 is shown. The other three registers are identical in functionality.
		This register is reset only by the Power-On Reset (POR). It has no part-specific functionality and is intended for use by a software developer to contain data that will be unaffected by the other reset sources (RESET pin, software reset, and COP reset).

Software Control Data0-3 Registers (SIM_SWC0-3) Base + $2-$5	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	SOFTWARE CONTROL DATA 0-3															
	Write																
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	________________	Date:	________
	________________	Programmer:	________
		Sheet	5 of 13

SIM

Most Significant Half of JTAG ID Register (SIM_MSHID)

Bits	Name	Description
15-0	n/a	***Read-Only*** **Bits**
		This *read-only* register displays the most significant half of the JTAG ID for the chip. This register reads $01F2.

Most Significant Half/JTAG ID Register (SIM_MSHID) Base + $6	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	1	1	1	1	1	0	0	1	0
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	0	0	0	0	0	0	1	1	1	1	1	0	0	1	0

▨ Reserved Bits

Application:	______________________	Date:	________
	______________________	Programmer:	________
		Sheet	6 of 13

SIM

Least Significant Half of JTAG ID Register (SIM_LSHID)

Bits	Name	Description
15-0	n/a	***Read-Only*** **Bits**
		This *read-only* register displays the least significant half of the JTAG ID for the chip. This register reads $401D.

Least Significant Half/JTAG ID Register (SIM_LSHID) Base + $7	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0	1
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Reset	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0	1

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 7 of 13

SIM

Power Control Register (SIM_PWR)

Bits	Name	Description	
1-0	**LRSTDBY**	**Large Regulator Standby Mode**	
		00	Large regulator is in Normal mode
		01	Large regulator is in Standby (reduced power) mode
		10	Large regulator is in Normal mode and the LRSTDBY field is write-protected until the next reset
		11	Large regulator is in Standby mode and the LRSTDBY field is write-protected until the next reset

Power Control Register (SIM_PWR) Base + $8	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LRSTDBY	
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 8 of 13

SIM

GPIO Peripheral Select Register (SIM_GPS)

Please see the following page for continuation of this register

Bits	Name	Description	
15	TCR	**TMR Clock Rate**	
		This bit selects the clock speed for the TMR module.	
		0	TMR module clock rate equals core clock rate, typically 32MHz (default)
		1	TMR module clock rate equals three times core clock rate
14	PCR	**PWM Clock Rate**	
		This bit selects the clock speed for the PWM module.	
		0	PWM module clock rate equals core clock rate, typically 32MHz (default)
		1	PWM module clock rate equals three times core clock rate
11	CFG_B7	**Configure GPIOB7**	
		This bit selects the alternate function for GPIOB7.	
		0	TXD (default)
		1	SCL
10	CFG_B6	**Configure GPIOB6**	
		This bit selects the alternate function for GPIOB6. The CLKMODE bit in the OCCS Oscillator Control register can enable this pin as the source clock to the chip. In this mode, make sure that no on-chip peripheral (including the GPIO) is driving this pin.	
		0	RXD (default)
		1	SDA
9	CFG_B5	**Configure GPIOB5**	
		This bit selects the alternate function for GPIOB5.	
		0	TI (default)
		1	FAULT3

GPIO Peripheral Select Register (SIM_GPS) Base + $B	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	TCR	PCR	0	0	CFG_B7	CFG_B6	CFG_B5	CFG_B4	CFG_B3	CFG_B2	CFG_B1	CFG_B0	CFG_A5		CFG_A4	
	Write			▨	▨												
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 9 of 13

SIM

GPIO Peripheral Select Register (SIM_GPS) Continued

Please see the following page for continuation of this register

Bits	Name	Description	
8	CFG_B4	**Configure GPIOB4**	
		This bit selects the alternater function for GPIOB4.	
		0	T0 (default)
		1	CLKO
7	CFG_B3	**Configure GPIOB3**	
		This bit selects the alternater function for GPIOB3.	
		0	MOSI (default)
		1	T3
6	CFG_B2	**Configure GPIOB2**	
		This bit selects the alternater function for GPIOB2.	
		0	MISO (default)
		1	T2
5	CFG_B1	**Configure GPIOB1**	
		This bit selects the alternater function for GPIOB1.	
		0	$\overline{\text{SS}}$ (default)
		1	SDA
4	CFG_B0	**Configure GPIOB0**	
		This bit selects the alternater function for GPIOB0.	
		0	SCLK (default)
		1	SCL

GPIO Peripheral Select Register (SIM_GPS) Base + $B	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	TCR	PCR	0	0	CFG_B7	CFG_B6	CFG_B5	CFG_B4	CFG_B3	CFG_B2	CFG_B1	CFG_B0	CFG_A5		CFG_A4	
	Write			▨	▨												
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Appendix B - Programmer Sheets, Rev. 5

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 10 of 13

SIM

GPIO Peripheral Select Register (SIM_GPS) Continued

Bits	Name	Description	
3-2	**CFG_A5**	**Configure GPIOA5**	
		This bit selects the alternater function for GPIOA5.	
		00	Select PWM5 when peripheral mode is enabled in GPIOA5 (default)
		01	Select PWM5 when peripheral mode is enabled in GPIOA5
		10	Select FAULT2 wehn peripheral mode is enabled in GPIO5
		11	Select T3 when peripheral mode is enabled in GPIOA5
1-0	**CFG_A4**	**Configure GPIOA4**	
		This bit selects the alternater function for GPIOA4.	
		00	Select PWM4 when peripheral mode is enabled in GPIOA4 (default)
		01	Select PWM4 when peripheral mode is enabled in GPIOA4
		10	Select FAULT1 wehn peripheral mode is enabled in GPIOA4
		11	Select T2 when peripheral mode is enabled in GPIOA4

GPIO Peripheral Select Register (SIM_GPS) Base + $B	**Bits**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Read	TCR	PCR	0	0	CFG_B7	CFG_B6	CFG_B5	CFG_B4	CFG_B3	CFG_B2	CFG_B1	CFG_B0	CFG_A5		CFG_A4	
	Write			▨	▨												
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application: ______________________________ Date: ______________

______________________________ Programmer: ______________

Sheet 11 of 13

SIM

Peripheral Clock Enable Register (SIM_PCE)

Bits	Name	Description	
15	**I2C**	**I^{2}C IPBus Clock Enable**	
		Each bit control clocks to the indicated peripheral.	
		0	The clock is not provided to the peripheral (the peripheral is disabled)
		1	Clocks are enabled
13	**ADC**	**Analog-to-Digital Converter IPBus Clock Enable**	
		Each bit controls clocks to the indicated peripheral.	
		0	The clock is not provided to the peripheral (the peripheral is disabled)
		1	Clocks are enabled
6	**TMR**	**Timer IPBus clock Enable**	
		Each bit controls clocks to the indicated peripheral.	
		0	The clock is not provided to the peripheral (the peripheral is disabled)
		1	Clocks are enabled
4	**SCI**	**SCI IPBus Clock Enable**	
		Each bit controls clocks to the indicated peripheral.	
		0	The clock is not provided to the peripheral (the peripheral is disabled)
		1	Clocks are enabled
2	**SPI**	**SPI IPBus Clock Enable**	
		Each bit controls clocks to the indicated peripheral.	
		0	The clock is not provided to the peripheral (the peripheral is disabled)
		1	Clocks are enabled
1	**PWM**	**PWM IPBus Clock Enable**	
		Each bit controls clocks to the indicated peripheral.	
		0	The clock is not provided to the peripheral (the peripheral is disabled)
		1	Clocks are enabled

Peripheral Clock Enable Register (SIM_PCE) Base + $C	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	I2C	0	ADC	0	0	0	0	0	0	TMR	0	SCI	0	SPI	0	PWM
	Write		▨		▨	▨	▨	▨	▨	▨		▨		▨		▨	
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ Reserved Bits

Application:	_______________	Date:	________
	_______________	Programmer:	________
		Sheet	12 of 13

SIM

I/O Short Address High Location Register (SIM_IOSAHI)

Bits	Name	Description
1-0	ISAL[23:22]	**Input/Output Short Address High**
		This field represents the upper two address bits of the *hard coded* I/O short address.

I/O Short Address Register (SIM_IOSAHI) Base + $D	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ISAL[23:22]	
	Write	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

▨ Reserved Bits

Application:	______	Date:	______
	______	Programmer:	______
		Sheet	13 of 13

SIM

I/O Short Address Low Location Register (SIM_IOSALO)

Bits	Name	Description
15-0	ISAL[21:6]	**Input/Output Short Address Location Low**
		This field represents the lower 16 address bits of the *hard coded* I/O short address.

I/O Short Address Register (SIM_IOSALO) Base + $E	Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Read	ISAL[21:6]															
	Write																
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

▨ Reserved Bits

Application:		Date:	
		Programmer:	
			Sheet

INDEX

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

MC56F8000RM
Rev. 5
08/2007