Document Number: PEXMC36XSDPUG
Rev. 1.0, 2/2016

36V eXtreme Switch Programming Guide

Contents

1 General Info

This documentation introduces Processor Expert component named 36VeXtremeSwitch. This component supports and provides flexible software solution for these analog parts:

NXP MC36XSD: 36V Industrial Low RDSon eXtreme Switch

NXP offers Tower board solution based on these chips, namely TWR-MC36XSDEVB board. Detailed description can be found in related hardware user guides and datasheets.

2 Embedded Component Description

2.1 Component API

36VeXtremeSwitch component provides API, which can be used for dynamic real-time configuration of device in user code. Available methods and events are listed under component selection Some of those methods/events are marked with ticks and other ones with crosses, it distinguishes which methods/events are supposed to be generated or not. You can change this setting in Processor Expert Inspector. Note that methods with grey text are always generated because they are needed for proper functionality. This forced behavior depends on various combinations of settings of component properties. For summarization of available API methods and events and their descriptions, see Table 1 36VeXtremeSwitch Component API

Table 1

Method	Description
Init	Initializes the device(s) according to the set properties. This method writes the data gathered from the component properties into registers via SPI. When auto initialization is enabled, this method will be called automatically within PE initialization function PE_low_level_init().
ReadRegister	Reads value of the given register via SPI. This method allows the user to read content from a register of the device(s).
WriteRegister	Writes value to the given register via SPI. This method allows the user to set a custom value to a register of the device(s).
GetStatus	Returns current general status information. It can be used for quick check of devices. Current status of all devices on daisy chain will be stored in an array.
GetFaultInfo	Gets fault status information. It can be used to check failure details when fault happens. The fault information will be stored to an array.
SetPWMDuty	Sets PWM duty for specified channel. Calling this method will set PWM output immediately (if PWM is enabled).
SetPWMState	Enables or disables the PWM module. Activates the internal PWM module of both channels simultaneously according to the values of duty cycle and turnon delays set previously. When the device works in PARALLEL mode, the linked channel cannot be set differently.
FeedWatchdog	Feeds the watchdog to avoid watchdog timeout in case the watchdog is enabled. When automatic watchdog toggling in disabled or no SPI data is sent (when watchdog toggling is enabled), please make sure this method is called at least twice during the watchdog timeout period (typ. 310 ms). In case the watchdog timeout happens, the device will enter failsafe mode. Calling this method again will turn the device from failsafe mode back to normal mode (all registers are reset).

ConfigureWatchdog	Allows to configure (enable or disable) watchdog. After calling this method to enable watchdog, be sure SPI data are sent (when automatic watchdog toggling is enabled) or FeedWatchdog() is called (when watchdog toggling is disabled) periodically to avoid the watchdog timeout.
Diagnosis	Runs a diagnosis routine. It will get channel configuration, product identification, current logic status of direct inputs, report of external clock failure (if external clock is enabled) and report of calibration failure. These information will be stored to an array.
ConfigureMonitoring	Sets current/temperature monitoring option of CSNS pin. If several eSwitch devices share the same CSNS pin and MCU ADC input pin, please make sure no more than 1 channel monitor is enabled at one time.

2.2 Events

There are no Events in this component

2.3 Methods

Init - Initializes the device(s) according to the set properties. This method writes the data gathered from the component properties into registers via SPI. When auto initialization is enabled, this method will be called automatically within PE initialization function - PE_low_level_init().

ANSIC prototype: bool Init(void)
Return value:bool - Return true if the operation was successful

ReadRegister - Reads value of the given register via SPI. This method allows the user to read content from a register of the device(s).

ANSIC prototype: bool ReadRegister(uint8_t regAddr,uint16_t *regVal)
regAddr:uint8_t - Address of the register, these address already defined in MC06XSD200.h file in project.

regVal: Pointer to uint16_t - Pointer to the variable for resulting data. If there are more devices in a daisy chain, all devices are read at once. The size of the array must match the length of the daisy-chain configured in component. In case there is only single device, the size is 1.

Return value:bool - Return ES_ERR_OK if the operation was successful

WriteRegister - Writes value to the given register via SPI. This method allows the user to set a custom value to a register of the device(s).

ANSIC prototype: bool WriteRegister(uint8_t regAddr,uint16_t *regVal)
regAddr:uint8_t - Register address

regVal: Pointer to uint16_t - Pointer to the value(s) to be written. If there are more devices in a daisy chain, all devices are written. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1.

Return value:bool - Return ES_ERR_OK if the operation was successful

GetStatus - Returns current general status information. It can be used for quick check of devices. Current status of all devices on daisy chain will be stored in an array.

ANSIC prototype: result GetStatus(uint16_t *statusData)
statusData: Pointer to uint16_t - Pointer to array that will be filled with the status information. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1. Each item of the array will be a combination of these values:
ES_STATUS_OVERVOLTAGE - overvoltage fault
ES_STATUS_UNDERVOLTAGE - undervoltage fault
ES_STATUS_POR - power-on reset (POR) has occurred

ES_STATUS_FAULT0 - faults are detected on channel 0
ES_STATUS_FAULT1 - faults are detected on channel 1
ES_STATUS_AUTORETRY0 - auto-retry counter is full on channel 0
ES_STATUS_AUTORETRY1 - auto-retry counter is full on channel 1
ES_STATUS_OUT0 - indicate the status of channel 0 is ON:1 OFF:0
ES_STATUS_OUT1 - indicate the status of channel 1 is ON:1 OFF:0

Return value:result - Returns ES_ERR_OK if operation was successful.

GetFaultInfo - Gets fault status information. It can be used to check failure details when fault happens. The fault information will be stored to an array.

ANSIC prototype: result GetFaultInfo(uint16_t channel,uint16_t *faultInfo)
channel:uint16_t - The number of the channel that we request information for.

faultInfo: Pointer to uint16_t - Pointer to array of uint16_t that will be filled with the fault information. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1. Each item of the array will be a combination of these values:
ES_ERR_OC - overcurrent fault on channel
ES_ERR_SC - severe short-circuit
ES_ERR_OS - output shorted to VPWR
ES_ERR_OLOFF - open load in OFF state
ES_ERR_OLON - open load in ON state
ES_ERR_OTW - overtemperature

Return value:result - Returns ES_ERR_OK if operation was successful.

SetPWMDuty - Sets PWM duty for specified channel. Calling this method will set PWM output immediately (if PWM is enabled).

ANSIC prototype: result SetPWMDuty(uint8_t channel,uint16_t *dutyValues)
channel:uint8_t - Output channel number

dutyValues: Pointer to uint16_t - Pointer to array that contains the PWM duty values. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1. Input value must be in range 0 - 256 for the duty cycle 0 - 100 percent.

Return value:result - Returns ES_ERR_OK if operation was successful.

SetPWMState - Enables or disables the PWM module. Activates the internal PWM module of both channels simultaneously according to the values of duty cycle and turn-on delays set previously. When the device works in PARALLEL mode, the linked channel cannot be set differently.

ANSIC prototype: result SetPWMState(uint8_t *channelStates)
channelStates: Pointer to uint8_t - Pointer to array containing the PWM configuration values. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1. When a channel is set PWM disabled, direct input pins are applied. The values in array can be:
ES_PWM_DISABLE_ALL - PWM module is disabled for both channels
ES_PWM_ENABLE_CH0_ONLY - only set channel 0 enabled
ES_PWM_ENABLE_CH1_ONLY - only set channel 1 enabled
ES_PWM_ENABLE_ALL - PWM module is enabled for both channels

Return value:result - Returns ES_ERR_OK if operation was successful.

FeedWatchdog - Feeds the watchdog to avoid watchdog timeout in case the watchdog is enabled. When automatic watchdog toggling in disabled or no SPI data is sent (when watchdog toggling is enabled), please make sure this method is called at least twice during the watchdog timeout period (typ. 310 ms). In case the watchdog timeout happens, the device will enter fail-safe mode. Calling this method again will turn the device from fail-safe mode back to normal mode (all registers are reset).

ANSIC prototype: result FeedWatchdog(void)
Return value:result - Returns ES_ERR_OK if operation was successful.

ConfigureWatchdog - Allows to configure (enable or disable) watchdog. After calling this method to enable watchdog, be sure SPI data are sent (when automatic watchdog toggling is enabled) or FeedWatchdog() is called (when watchdog toggling is disabled) periodically to avoid the watchdog timeout.

ANSIC prototype: result ConfigureWatchdog(bool state,uint32_t deviceMask)

state:bool - State of the watchdog. TRUE = watchdog enabled

deviceMask:uint32_t - Each bit in this parameter represents one device on the daisy chain. If a bit is '1' then the configuration is applied to the device. '0' means no change for the device with the index. To apply value to all device, use the ES_ALL_DEVICES constant.

Return value:result - Returns ES_ERR_OK if operation was successful.

Diagnosis - Runs a diagnosis routine. It will get channel configuration, product identification, current logic status of direct inputs, report of external clock failure (if external clock is enabled) and report of calibration failure. These information will be stored to an array.

ANSIC prototype: result Diagnosis(uint16_t *diagData)

diagData: Pointer to uint16_t - Pointer to array of uint16_t elements that will be filled with the diagnosis information. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1. Each element of the array will contain a combination of these values:

ES_DIAG_CH1_DC_MOTOR - load is configured as a DC motor type, if not present it's bulb type

ES_DIAG_CH0_DC_MOTOR - load is configured as a DC motor type, if not present it's bulb type

ES_DIAG_PRODUCT_ID_BIT1 - product identification higher bit

ES_DIAG_PRODUCT_ID_BIT0 - product identification lower bit

ES_DIAG_IN1_ON - current logic state of the direct input IN1

ES_DIAG_IN0_ON - current logic state of the direct input IN0

ES_DIAG_CLOCKFAIL - external clock error occurred

ES_DIAG_CALIBFAIL - calibration failure occurred during calibration of channel's internal clock period

Return value:result - Returns ES_ERR_OK if operation was successful.

ConfigureMonitoring - Sets current/temperature monitoring option of CSNS pin. If several eSwitch devices share the same CSNS pin and MCU ADC input pin, please make sure no more than 1 channel monitor is enabled at one time.

ANSIC prototype: result ConfigureMonitoring(uint16_t *selection)

selection: Pointer to uint16_t - Pointer to array containing the monitoring settings. The size of the array must match the length of the daisy chain configured in component. In case there is only single device, the size is 1. Each item of the array contains one of the following values:

ES_SENSE_DISABLE - sensing disabled

ES_CURRENTSENSE0 - current sensing for channel 0 (will not apply if highest overcurrent range is not selected and the device works in Parallel mode)

ES_CURRENTSENSE1 - current sensing for channel 1

ES_TEMPERATURESENSE - temperature sensing

ES_CURRENTSENSE_SUM - current sensing for summed channels (this option will apply only if highest overcurrent range is not selected and the device works in Parallel mode)

Return value:result - Returns ES_ERR_OK if operation was successful.

2.4 Properties

Component Name - Name of the component.

SPI_Device - Linked component handling SPI communication (for details about settings see Component Inheritance & Component Sharing).

Use 8-bit SPI communitation - If this property is set, all SPI communitation is performed using 8-bit access, instead of 16-bit. It allows to use the component on the MCUs, where the SPI interface provides only 8-bit mode.

There are 2 options:

yes
no

Automatic watchdog toggling - If set, the WDIN bit is automaticaly toggled in every SPI read and write to feed watchdog. Otherwise it's necessary to call FeedWatchdog method in a timely manner or disable the watchdog functionality within the device configuration.

There are 2 options:

yes
no

Reset Pin control - Enable this property to insert a BitIO_LDD component to current project and provide interface to control ResetB pin of ESwitch device.

The following items are available only if the group is enabled (the value is "Enabled"):

RSTB pin - Select a component to interface to device.

Devices On Daisy Chain - Configuration assignments. Each item in this list assigns a specific configuration (from the Configurations group) to one device in the daisy chain. The number and order of items must match the hardware for correct function of SPI communication. If you are not using daisy chaining, set only 1 configuration.

One Item of the list looks like:

Configuration Index - Configuration for each device There are 17 options:

Configuration_0
Configuration_1
Configuration_2
Configuration_3
Configuration_4
Configuration_5
Configuration_6
Configuration_7
Configuration_8
Configuration_9
Configuration_10
Configuration_11
Configuration_12
Configuration_13
Configuration_14
Configuration_15
Configuration_16

Configurations - Defined configurations that can be used or shared by devices in daisy chain

One Item of the list looks like:

Device - Device linked through one SPI daisy chain, every device has independent configuration.

Global Configuration - Global configuration the device.

PWM channel 0 - Enable this item to activate the internal PWM function for channel 0 simultaneously according to the values of duty cycle and turn-on delays set.

There are 2 options:

Enabled: Enable PWM control mode on Channel 0

Disabled: Disable PWM control mode on Channel 0

PWM channel 1 - Enable this item to activate the internal PWM function for channel 1 simultaneously according to the values of duty cycle and turn-on delays set.

There are 2 options:

Enabled: Enable PWM control mode on Channel 1

Disabled: Disable PWM control mode on Channel 1

Parallel Mode - Enable this item to set device to parallel mode (improved switching synchronization between both channels).

There are 2 options:

Enabled: Improved switching synchronization between both channels

Disabled: No switching synchronization between both channels

Track & Hold current sensing - Enable this item to activate Track & Hold current sensing mode. When T& H is activated, the value of the channels load current is kept available after turn-off.

There are 2 options:

Enabled: Enable Track& Hold mode,channels load current is kept available after turn-off

Disabled: Disable Track& Hold mode,channels load current is not kept after turn-off

Watchdog - When watchdog is enabled, watchdog feeding should be sent to device to avoid watchdog timeout. Once watchdog timeout occurs, the device enters fail-safe mode to indicate loss communication with controller.

There are 2 options:

Enabled: Enable internal watchdog

Disabled: Disable internal watchdog

VDD failure detection - Enable or disable VDD failure detection, the device will enter fail safe mode after VDD<VDD(fail)

There are 2 options:

Enabled: Enable VDD fault detection

Disabled: Disable VDD fault detection

Overvoltage protection - Disable overvoltage detection if needed, default value is enable.

There are 2 options:

Enabled: Enable overvoltage protectction on VPWR pin

Disabled: Disable overvoltage protectction on VPWR pin

CSNS Pin Function - CSNS pin can output analog signal to indicate current on channel 0, current on channel 1, summed current for both channels or temperature. Select function to set which signal the device will output on this pin.

There are 5 options:

Disabled: Disable any sensing on CSNS pin

Current Sensing on Channel 0: Channel 0 Current Sensing on CSNS pin

Current Sensing on Channel 1: Channel 1 Current Sensing on CSNS pin

Temperature Sensing: Temperature Sensing on CSNS pin

Current Sensing Summed Currents: Current of Channel 0 and Channel 1 will be summed and output on CSNS pin

Output - Number of High Side output channels in one ESwitch device.

One Item of the list looks like:

HS - Settings related to eSwitch channel

Direct control - Enable or disable direct IO control to turn on/off the eSwitch channel. When enabled, the device allows MCU use a GPIO to turn on/off the device channel, otherwise the direct control won't work.

There are 2 options:

Enabled: enable directin control on INx pin

Disabled: input voltage on INx pin is invalid

PWM duty - Duty cycle set for PWM control for each channel, it can be set from 0 to 256, that reflects 0 to 100% duty cycle. The PWM channel should be enabled in global configuration to make it work.

PWM Switch-on Delay - The number of internal/external clock periods that determine device's turn on delay time.

There are 8 options:

No Delay: No delay when switch on

32 PWM Clocks: 32 PWM clocks delay when switch on

64 PWM Clocks: 64 PWM clocks delay when switch on

96 PWM Clocks: 96 PWM clocks delay when switch on

128 PWM Clocks: 128 PWM clocks delay when switch on

160 PWM Clocks: 160 PWM clocks delay when switch on

192 PWM Clocks: 192 PWM clocks delay when switch on

224 PWM Clocks: 224 PWM clocks delay when switch on

PWM clocksource - This item chooses the internal/external clock of the selected channel.

There are 2 options:

External: Configures the PWM module to use an external clock signal.

Internal: Activates the internal clock of the selected channel.

External PWM Clock Divider - This item controls which of two divider values are used to create the PWM frequency from the external clock.

There are 2 options:

256: 256 divider for external PWM clock

512: 512 divider for external PWM clock

Overcurrent profile - The item here has no actural work, it's just for indicate to config the CONF0 and CONF1 pin voltage level. ESwitch support two overcurrent profiles: DC Motor andBulb. If DC Motor load is selected then a pull-down resistor should be connected on CONFx pin, if Bulb load selected then pull-up resistor should be connected on CONFx pin.

There are 2 options:

DC Motor

Bulb

Short circuit detection - Enables/disables detection of short-circuits between the channels output pin and the VPWR pin.

There are 2 options:

Enabled: report fault when short circuit happen

Disabled: no detecton when short circuit happen

OpenLoad Detection in ON state - Enables/disables detection of OpenLoad in the On state for the selected channel.

There are 2 options:

Enabled: enable openload detection when eswitch channel is ON status

Disabled: disable openload detection when eswitch channel is ON status

OpenLoad Detection in OFF state - Enables/disables detection of OpenLoad in the OFF state for the selected channel.

There are 2 options:

Enabled: enable openload detection when eswitch channel is OFF status

Disabled: disable openload detection when eswitch channel is OFF status

Slew rate - Control the slew rate at turn on and turn off. Rising and falling edge slew rates are identical.

There are 4 options:

Low SR: Low slew rate configuration

Medium SR: Medium slew rate configuration

Medium SR <SR <High SR: High slew rate configuration

High SR: Max slew rate configuration

Random Current Offset - Set the random offset current to add or substract to CSNS pin.
There are 2 options:

Add random offset: random offset current to be added to the sensed current (pin CSNS).

Subtract random offset: Subtract random offset current to be added to the sensed current (pin CSNS).

Max Auto-Retry Count - Set the amount of auto-retries to 16 or unlimited auto-retry.
There are 2 options:

Limited to 16 retries: Try 16 times to switch on if failed.

Infinite retries: always retry to switch on if failed.

Auto-Retry Period - Selection of the value of the auto-retry period among four predefined values.
There are 4 options:

tAUTO_00: paremeters on datasheet
tAUTO_01: paremeters on datasheet
tAUTO_10: paremeters on datasheet
tAUTO_11: paremeters on datasheet

Auto-Retry Function - Enables/disables auto-retry, accordingly to setting of the CONF pin.
There are 2 options:

Enabled
Disabled

Low Current Threshold - Selection of the defined value of the lowest overcurrent threshold.
There are 3 options:

I_OCL1: HOCR low, OCL_s high
I_OCL2: HOCR high, OCL_s low
I_OCL3: HOCR high, OCL_s high

Medium Current Threshold - Selection of the defined value of the medium overcurrent threshold.
There are 2 options:

I_OCM1: I_OCM1_s low
I_OCM2: I_OCM1_s high

High Current Threshold - Selection of the defined value of the upper overcurrent threshold.
There are 2 options:

I_OCH1: I_OCH1_s low
I_OCH2: I_OCH1_s high

Threshold Activation Times - Dynamic Overcurrent Threshold Activation Times for Bulb -and DC motor profiles.
There are 4 options:

tOCH1 and tOCM1_L: tOCH1 and tOCM1_L for Bulb overcurrent detection profile
tOCH1 and tOCM2_L: tOCH1 and tOCM2_L for Bulb overcurrent detection profile
tOCH2 and tOCM1_L: tOCH2 and tOCM1_L for Bulb overcurrent detection profile
tOCH2 and tOCM2_L: tOCH2 and tOCM2_L for Bulb overcurrent detection profile

Current Sense Ratio - This item is used to raise the accuracy of current sensing. The 'low-current sense ratio' is optimal for measuring currents in the lower range. The 'high-current sensing ratio' is for measuring in the higher range.
There are 2 options:

high-current: raise accuracy on CSNS pin when sensing high current
low-current: raise accuracy on CSNS pin when sensing low current

Auto Initialization - Automatically generates initialization code if enabled.

There are 2 options:

yes
no

3 Typical Usage

The following examples show basic device control.

1. PWM control on daisy chain devices.

SetPWMDuty method is used to set to PWM to ON status and give a duty cycle to devices linked. It can only operate one channel of each devices at one time, if the other channel also need to be controlled, it should be called twice unless the device is running in PARALLEL mode.

Listing 1: Source code

```
/* declare a buffer array, the XSD1_DAISY_CHAIN_LENGTH is defined
   automatically by component according to it's settings */
uint16_t chain_data[XSD1_DAISY_CHAIN_LENGTH];
chain_data[0] = 0xFF; /* data for duty cycle of device 1 = 100%(fully on)
   0~100% = 0~0xFF */
chain_data[1] = 0x80; /* data for duty cycle of device 2 = 50% */
chain_data[2] = 0x40; /* data for duty cycle of device 3 = 25% */
chain_data[3] = 0x00; /* data for duty cycle of device 4 = 0%(fully off) */
/* set PWM duty cycle for channel 0 of all devices in daisy chain */
XSD1_SetPWMDuty(0, chain_data);
/* set PWM duty cycle for channel 1 of all devices in daisy chain */
XSD1_SetPWMDuty(1, chain_data);
```

2. Monitoring current/temperature from devices - [optional]Enable current/temperature monitoring function on CSNS pin in Configuration_x setting and link this configuration to target device.

- Add ADC component into project that provides interface to convert analog signal to digital.

- [optional]Switch monitoring channels that CSNS pin can only have one of the four monitoring functions at one time. SYNC is used to sync with CSNS pin. When the analog signal is ready, SYNC is set LOW.

1) current of channel 0

2) current of channel 1

3) summed current of channels 0 and 1

4) temperature

The method ConfigureMonitoring can be used to perform this monitoring channel switching:

Listing 2: Source code

```
#define XSD1_DAISY_CHAIN_LENGTH 4 /* this definition is generated
   automatically by component, defined by "Devices On Daisy Chain" property
   */
uint16_t chain_data[XSD1_DAISY_CHAIN_LENGTH]; /* declare a buffer array */
chain_data[0] = ES_CURRENTSENSE0; /* data for enable device 1 monitoring
   current on channel 0 */
chain_data[1] = ES_SENSE_DISABLE; /* data for disable device 2 monitoring */
chain_data[2] = ES_SENSE_DISABLE; /* data for disable device 3 monitoring */
chain_data[3] = ES_SENSE_DISABLE; /* data for disable device 4 monitoring */
XSD1_ConfigureMonitoring(chain_data); /* send the configuration data to
   daisy chain devices */
```

3. Direct Access to Registers

The interface generated by this component can be easily used to communicate with devices linked in daisy chain after configuration in Component Inspector view base on upper hardware connection.

Two low level methods can be called:

Listing 3: Source code

```
bool XSD1_ReadRegister(uint8_t regAddr, uint16_t *regVal);
bool XSD1_WriteRegister(uint8_t regAddr, uint16_t *regVal);
```

The two methods are used for basic SPI communication with devices, user can send/read custom data to/from defined register of devices. Register names are defined in MC06XSD200.h file.

Listing 4: Source code

```
uint16_t values[4]; /* declare array as buffer to store data */
XSD1_ReadRegister(XS_SO_STATR, values); /* read the register content from
    devices */
```

4 User Types

***ComponentName*_result** = *return type of methods*

How to Reach Us:

Home Page:
NXP.com

Web Support:
http://www.nxp.com/support

Document Number: PEXMC36XSDPUG
Rev. 1.0
2/2016