Freescale Semiconductor
Application Note

AN2984
Rev. 0, 07/2005

Programming the Analog-to-Digital Converter on M68HC08 Microcontrollers

**by: Luis Reynoso Covarrubias
RTAC Americas**

Overview

This document is a quick reference for an embedded engineer to get the analog-to-digital converter (ADC) module up and running for any M68HC08 MCU. Basic knowledge about the functional description and configuration options will give the user a better understanding of how the ADC module works. This application note provides examples that illustrate one use of the ADC module within the M68HC08 Family of microcontrollers. The examples are intended to be modified to suit the specific needs of any application.

Description

The ADC in the M68HC08 MCUs is a unipolar, successive-approximation converter available for 8-bit or 8-/10-bit resolution, with monoticity guaranteed for both options. Freescale's M68HC08 microcontrollers include ADC modules with a wide range of options for the user, covering in this way many different applications.

Some of the features of the ADC module include:

- From 4 to 24 ADC channels with a wide range of options in between.
- Two options for resolution: 8-bit or 8-/10-bit ADC modules are available, depending on the microcontroller family.
- Conversion type adaptable to each application: allows single or continuous conversion, with the option of auto-scan conversion in some microcontrollers.
- Supports both programming approaches: includes a conversion complete flag and a conversion complete interrupt, allowing the user to choose polling or an interrupt-based approach.
- Selectable ADC clock frequency: includes a clock prescaler, and some microcontrollers include the option to choose between the bus clock or the crystal as input clock.
- Flexibility: The 8-/10-bit ADC contains four options for conversion result justification (10-bit left justified, 10-bit right justified, 10-bit signed left justified, and 8-bit truncation mode).

For more information on the ADC differences among the families of microcontrollers, refer to Appendix A: Differences Among Derivatives.

For further information regarding analog-to-digital converters on Freescale microcontrollers, consult the *HC08 ADC Reference Manual* (ADCRM). For specific information about each microcontroller's ADC, refer to the corresponding datasheet in sections "Analog-to-Digital Converter (ADC) Module" and "Electrical Specifications Analog-to-Digital Converter (ADC) Characteristics".

The main registers found within the ADC module are:

1. The ADC status and control register (ADSCR)
   - Flags an ADC conversion complete (bit 7, COCO)
   - Enables ADC interrupts (bit 6, AIEN)
   - Selects continuous ADC conversions or one ADC sample (bit 5, ADCO)
   - Selects one of the ADC channels to be scanned (bit 5:0, CH4:CH0)
2. The ADC clock register (ADCLK or ADICLK)
   - Selects the clock prescaler (bit 7:5, ADIV2:ADIV0)
   - Selects the input clock for the internal ADC clock (bit 4, ADICLK) [1]
   - Selects among four modes of result justification (bit 3:2, MODE1:MODE0)[1]
3. The ADC data register (ADR)
   - Is updated with the result each time an ADC conversion completes, and can be either:

   a. One byte long for 8-bit ADCs.

   b. Two bytes long for 8-/10-bit ADCs containing the data in the selected format (ADRH-ADRL or ADRH0:ADRL0)[1]
4. The ADC auto-scan mode data registers (ADRL1-ADRL3), which are one-byte-long registers for channels ADC1 to ADC3 when the ADC is operating in auto-scan mode.[1]

[1] Optional Feature: Please see the appendix for the different M68HC08 Families and their ADC capabilities.

5. The ADC auto-scan control register (ADASCR)
   - Defines the number of auto-scan channels to be used when in auto-scan mode (bit 2:1, AUTO1:AUTO0)[1]
   - Enables the auto-scan mode (bit 0, ASCAN)[1]

Code and Explanation

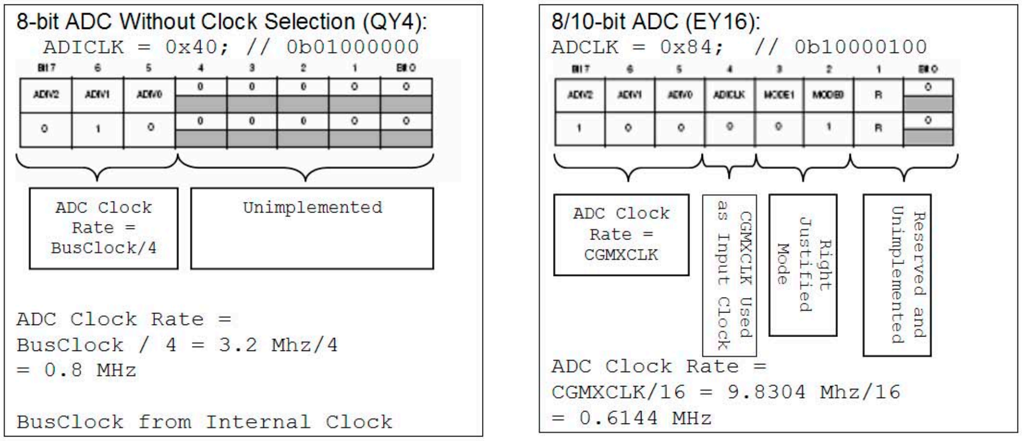
The following example describes the initialization code for both the 8-bit and 8-/10-bit ADC modules, using the interrupt-based approach, in single-sample mode. Following these three simple steps, the user can use the ADC module:

1. Configure the ADCLK/ADICLK register for the ADC module.

The ADCLK will configure the clock for the ADC module. It's important that the ADC clock must operate between a specific range of frequencies for correct operation. If the selected clock source is not fast enough, the ADC will generate incorrect conversions. The usual internal clock for the M68HC08 ADC module works between 500 kHz and 1.048 MHz; however, some families offer a wide range of options starting from 32 kHz to a maximum of 6 MHz.

It's also important to consider that not all M68HC08 Families have an ADC module with the ability to choose between the internal bus clock and the oscillator clock as an input to generate the internal ADC clock.

The second function of this register is choosing the conversion result justification mode, which is available only for 8-/10-bit converters.



```
ADICLK = 0x40; // 0b01000000
```

ADC Clock Rate =
BusClock / 4 = 3.2 Mhz/4
= 0.8 MHz

BusClock from Internal Clock

```
ADCLK = 0x84;  // 0b10000100
```

ADC Clock Rate =
CGMXCLK/16 = 9.8304 Mhz/16
= 0.6144 MHz

Figure 1. Configuring ADC Clock for the MC68HC908QY4 and MC68HC908EY16 Derivatives

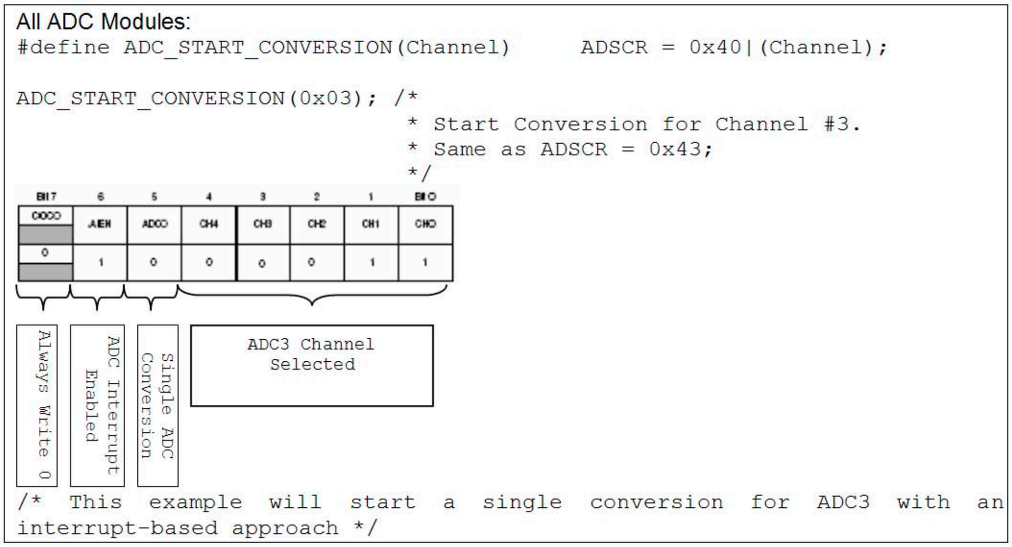
[1] Optional Feature: Please see the appendix for the different M68HC08 Families and their ADC capabilities.

2. Write the ADSCR to start ADC conversions.

The ADSCR has the capability of enabling the ADC interrupt and starting either a single conversion or the continuous conversion mode for a specific channel.

Note the following:

- This register is the same for all ADC modules, but the options for CH4:CH0 will depend on the available channels for each microcontroller.
- In continuous conversion mode, the COCO bit is set only after the first conversion is completed.
- The write functionality for COCO is reserved, so it's recommended to have a 0 in the COCO position for every write to ADSCR.
- If interrupts are enabled, COCO always reads as 0.
- Interrupt signal is cleared when ADR is read or ADSCR is written.



All ADC Modules:

```
#define ADC_START_CONVERSION(Channel)     ADSCR = 0x40|(Channel);

ADC_START_CONVERSION(0x03); /*
                             * Start Conversion for Channel #3.
                             * Same as ADSCR = 0x43;
                             */
```

BIT 7	6	5	4	3	2	1	BIT 0
COCO	AIEN	ADCO	CH4	CH3	CH2	CH1	CH0
0	1	0	0	0	0	1	1

```
/* This example will start a single conversion for ADC3 with an
interrupt-based approach */
```

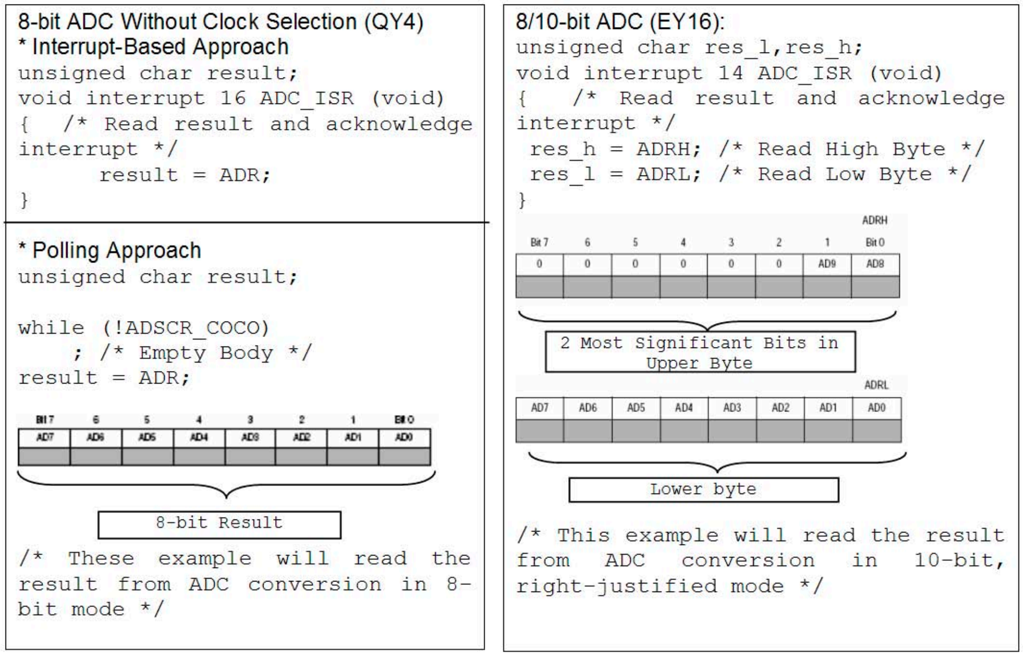
Figure 2. Starting a Single ADC Conversion

3. Read result from ADR.

Using a polling method, the user can read the result of the ADC conversion from the ADR register after COCO bit is set. After reading the ADR register, the COCO bit is cleared.

Using an interrupt-based approach, the user can read the result in the interrupt service routine and this will acknowledge the interrupt.

If the microcontroller has an 8-bit ADC, the result is placed in the one-byte ADR register. If it has an 8-/10-bit ADC, the result is placed in the ADRH:ADRL registers, according to the settings in the MODE1:MODE0 bits. It's important to remember that ADRH must be accessed first, followed by an access to ADRL, to avoid locking the ADC. In this way, 16-bit accesses in C must be avoided.



8-bit ADC Without Clock Selection (QY4)
* Interrupt-Based Approach

```
unsigned char result;
void interrupt 16 ADC_ISR (void)
{   /* Read result and acknowledge
interrupt */
      result = ADR;
}
```

* Polling Approach

```
unsigned char result;

while (!ADSCR_COCO)
    ; /* Empty Body */
result = ADR;
```

```
/* These example will read the
result from ADC conversion in 8-
bit mode */
```

8/10-bit ADC (EY16):

```
unsigned char res_l,res_h;
void interrupt 14 ADC_ISR (void)
{   /* Read result and acknowledge
interrupt */
 res_h = ADRH; /* Read High Byte */
 res_l = ADRL; /* Read Low Byte */
}
```

```
/* This example will read the result
from ADC conversion in 10-bit,
right-justified mode */
```

Figure 3. Reading ADC Result for the MC68HC908QY4 and MC68HC908EY16 Derivatives

Considerations

This example was developed using Metrowerks CodeWarrior IDE version 3.0 for M68HC08, and was expressly made for the MC68HC908QY4 and MC68HC908EY16. There may be changes needed in code to initialize another MCU. Every microcontroller needs an initialization code that depends on the application and the microcontroller itself.

References

- Analog-to-Digital Converter Reference Manual:
  – http://e-www.freescale.com/files/microcontrollers/doc/ref_manual/ADCRM.pdf
- AN2438/D – ADC Definitions and Specifications
  – http://e-www.freesacle.com/files/microcontrollers/doc/app_note/AN2438.pdf

Appendix A: Differences Among Derivatives

Table 1. Differences in ADC between HC08 Derivatives

MCU	ADC	Channels	Bits	ADC Clock Register	AutoScan
AB	x	8	8	x	—
AP	x	8	8/10	x	x
AS/AZ	x	15	8	x	—
BD	x	6	8	—	—
EY	x	8	8/10	x	—
GP	x	8	8	x	—
GR4/8	x	4,6	8	x	—
GR16	x	8	8/10	x	—
GR32/48/60	x	24	8/10	x	—
GT	x	8	8	x	—
GZ8/16	x	8	8/10	x	—
GZ32/48/60	x	24	8/10	x	—
JG	x	8	8	—	—
JK/JL	x	10,12,13	8	—	—
KX	x	4	8	x	—
LD	x	6	8	—	—
LJ	x	4,6	8/10	x	—
LK	x	6	8/10	x	—
MR	x	4,7,9,10	8/10	x	—
QT/QY2/4	x	4	8	—	—
SR	x	11,14	8/10	x	x

Notes:

- The name of the ADC clock register can be ADCLK or ADICLK.
- All microcontrollers with 8-/10-bit ADC have MODE1:MODE0 bits in the ADC clock register to select justification mode.
- All microcontrollers with 8-bit ADC have one 8-bit ADC result register.

- All microcontrollers with 8-/10-bit ADC have two 8-bit ADC result registers that have the conversion result in the selected format.
- All microcontrollers with auto-scan mode available have an additional functionality described in Appendix B: Auto-Scan.

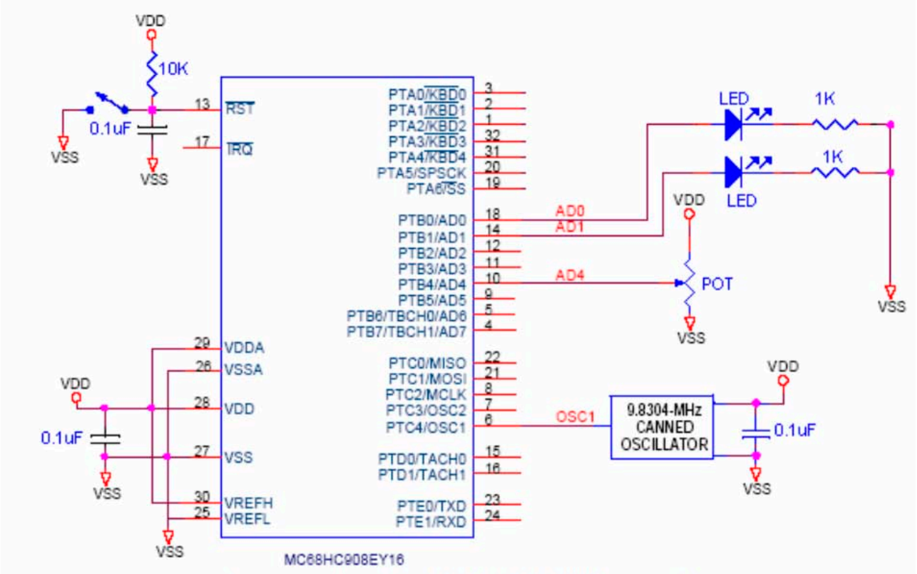
Appendix B: Auto-Scan

Microcontrollers with auto-scan mode are capable of starting an automatic conversion starting from channel 0 up to the channel defined in AUTO1:AUTO0 of the auto-scan control register (ADASCR). The process is started by writing ADSCR, then each conversion data is stored in ADRLx, and the COCO-bit is set. After that, a counter is incremented, and the process repeats until the counter reaches the programmed value.

Microcontrollers with this feature available have three additional 8-bit result registers (ADC auto-scan mode data registers, ADRL1–ADRL3), each containing the result for the corresponding channel (ADC1 to ADC3). They also have an ADC scan control register (ADASCR) to configure and enable the auto-scan mode.

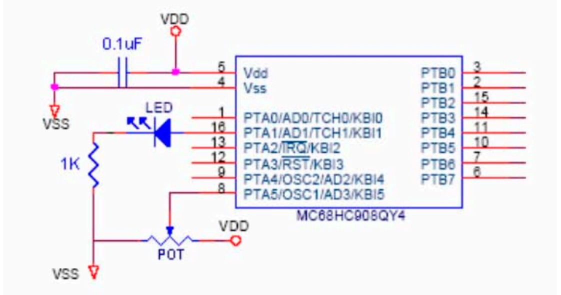
Because the auto-scan only uses 8-bit registers, the user must select 8-bit truncation mode (in MODE1:MODE0) for proper functionality.

Appendix C: Schematics



Schematic Used for MC68HC908EY16 Example

Figure 4. Schematic Used for MC68HC908EY16 Example



Schematic Used for MC68HC908QY4 Example

Figure 5. Schematic Used for MC68HC908QY4 Example

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

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.