TB3211

Getting Started with Analog Comparator (AC)

Introduction

Author: Cristian Pop, Microchip Technology Inc.

Microchip tinyAVR® 0- and 1-series, megaAVR® 0-series and AVR® Dx devices feature an Analog Comparator (AC) with flexible input selection, selectable hysteresis and configurable output (interrupts on rising/falling or both edges, event generation, and output inversion).

This technical brief explains the AC concepts and its implementation in tinyAVR® 0- and 1-series, megaAVR® 0-series and AVR® Dx devices with the following use cases:

- **Level Crossing Detector:**
  This example shows how to use the AC to detect when a critical value of an analog signal is reached (for example, the battery level).
- **Preventing False Spike Detection:**
  This example demonstrates how to use the hysteresis feature to minimize the number of false transitions in a noisy environment.
- **Analog Signal Pulse Duration/Frequency Measurement:**
  The example describes how to use the AC together with a timer to measure the pulse duration and/or the period of analog signals with minimal intervention of the AVR core.

Note: For each use case described in this document, there are two code examples: One bare metal developed on ATmega4809, and one generated with MPLAB® Code Configurator (MCC) developed on AVR128DA48.

View the ATmega4809 Code Examples on GitHub
Click to browse repository

View the AVR128DA48 Code Examples on GitHub
Click to browse repository

Table of Contents

1. Relevant Devices

This section lists the relevant devices for this document. The following figures show the different family devices, laying out pin count variants and memory sizes:

- Vertical migration upwards is possible without code modification, as these devices are pin-compatible and provide the same or more features. Downward migration on tinyAVR® 1-series devices may require code modification due to fewer available instances of some peripherals
- Horizontal migration to the left reduces the pin count and, therefore, the available features
- Devices with different Flash memory sizes typically also have different SRAM and EEPROM

Figure 1-1. tinyAVR® 0-series Overview

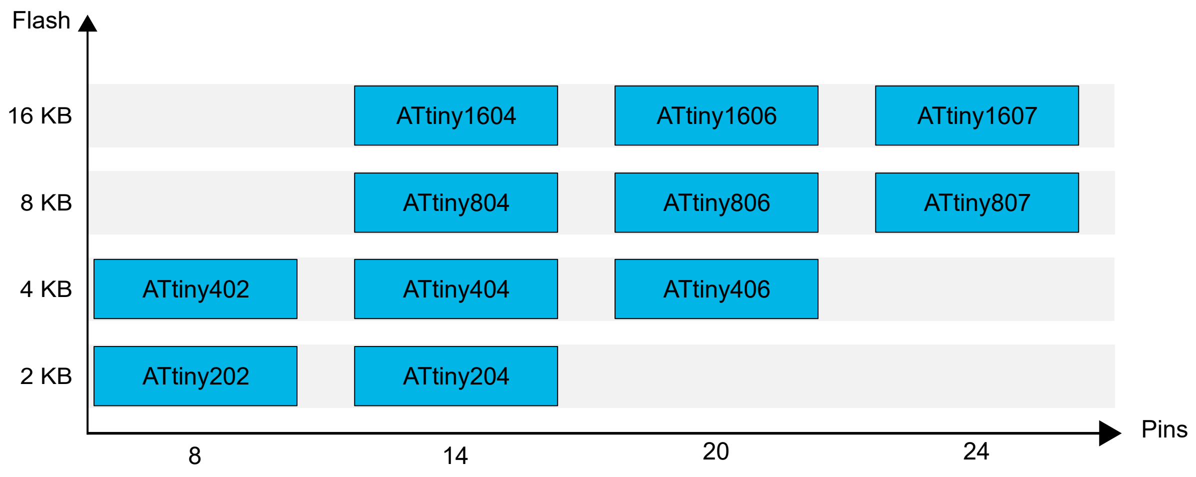


Figure 1-2. tinyAVR® 1-series Overview

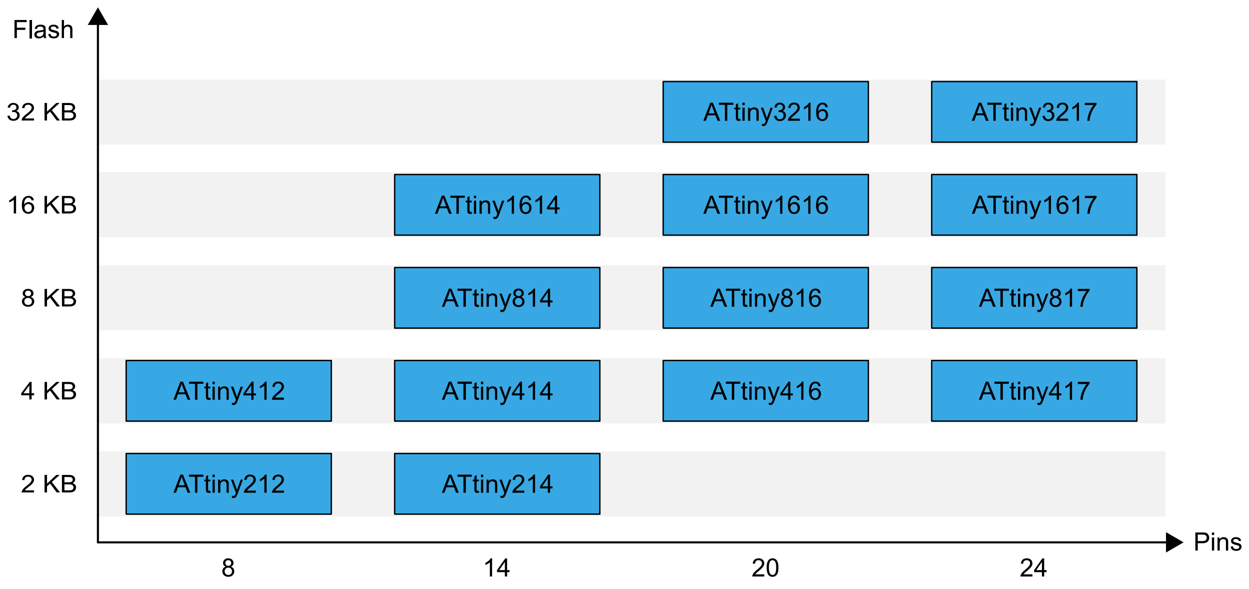


Figure 1-3. megaAVR® 0-series Overview

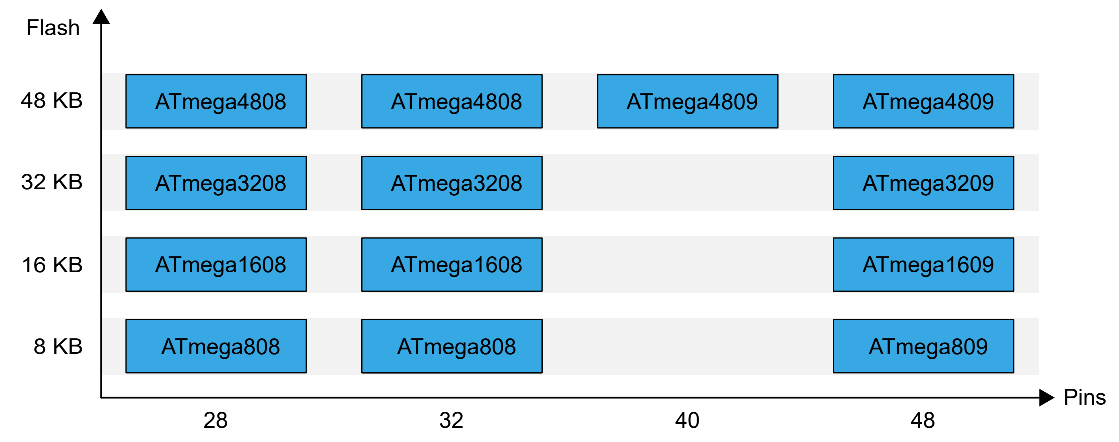
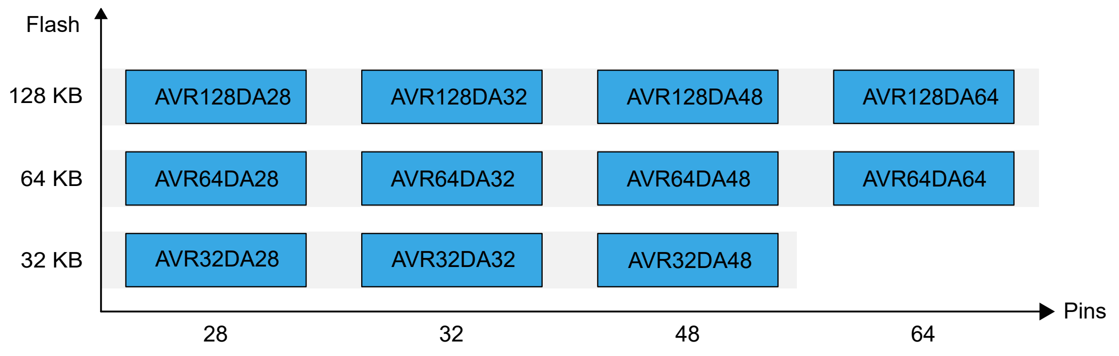


Figure 1-4. AVR® DA Family Overview

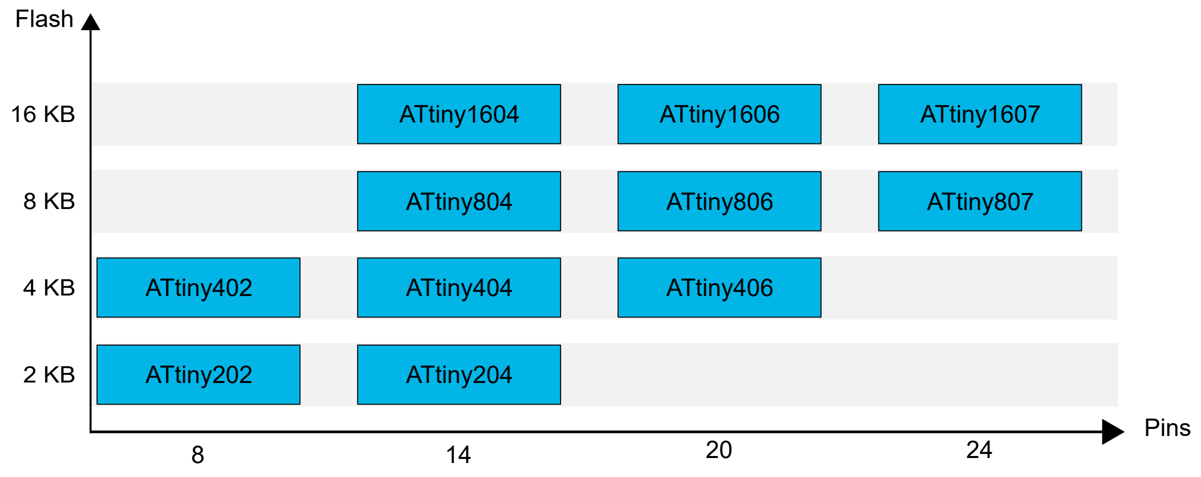


1.1 tinyAVR® 0-series

The figure below shows the tinyAVR® 0-series devices, laying out pin count variants and memory sizes:

- Vertical migration upwards is possible without code modification, as these devices are pin-compatible and provide the same or more features
- Horizontal migration to the left reduces the pin count and, therefore, the available features

Figure 1-5. tinyAVR® 0-series Overview



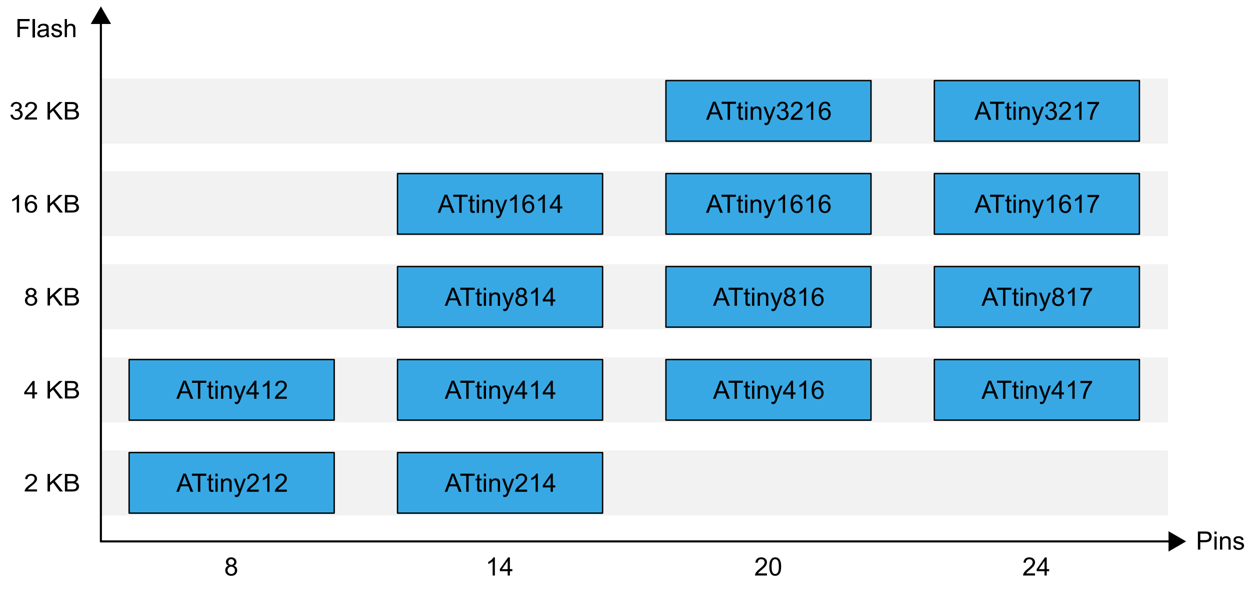
Devices with different Flash memory sizes typically also have different SRAM and EEPROM.

1.2 tinyAVR® 1-series

The figure below shows the tinyAVR® 1-series devices, laying out pin count variants and memory sizes:

- Vertical migration upwards is possible without code modification, as these devices are pin compatible and provide the same or more features. Downward migration may require code modification due to fewer available instances of some peripherals.
- Horizontal migration to the left reduces the pin count and therefore, the available features.

Figure 1-6. tinyAVR® 1-series Overview



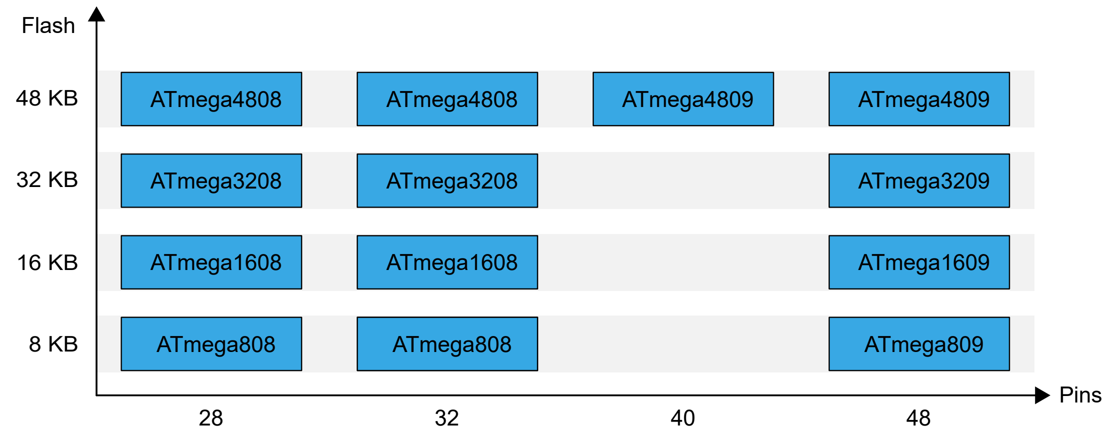
Devices with different Flash memory size typically also have different SRAM and EEPROM.

1.3 megaAVR® 0-series

The figure below shows the megaAVR® 0-series devices, laying out pin count variants and memory sizes:

- Vertical migration is possible without code modification, as these devices are fully pin and feature compatible
- Horizontal migration to the left reduces the pin count and, therefore, the available features

Figure 1-7. megaAVR® 0-series Overview



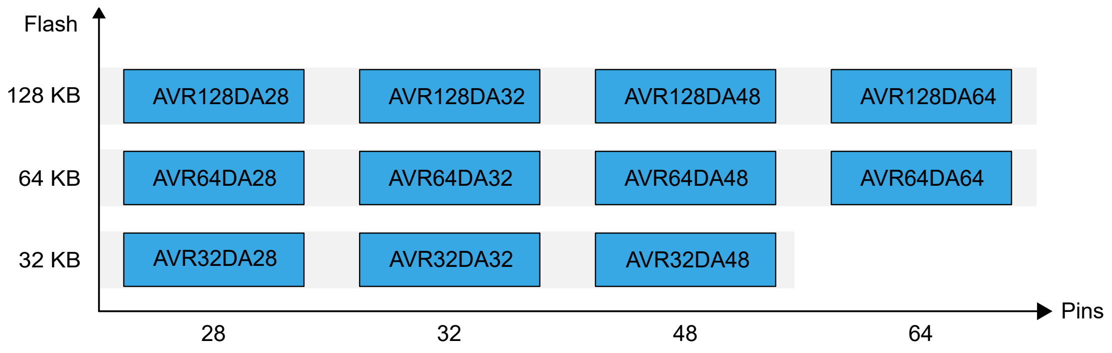
Devices with different Flash memory sizes typically also have different SRAM and EEPROM.

1.4 AVR® DA Family Overview

The figure below shows the AVR® DA devices, laying out pin count variants and memory sizes:

- Vertical migration is possible without code modification, as these devices are fully pin and feature compatible
- Horizontal migration to the left reduces the pin count, and therefore, the available features

Figure 1-8. AVR® DA Family Overview



Devices with different Flash memory sizes typically also have different SRAM.

2. Overview

The AC peripheral compares two analog input voltages and outputs a signal level indicating when one of the inputs is higher than the other. An AC is basically an amplifier without feedback and thus has very high gain.

Figure 2-1. Analog Comparator Block Diagram

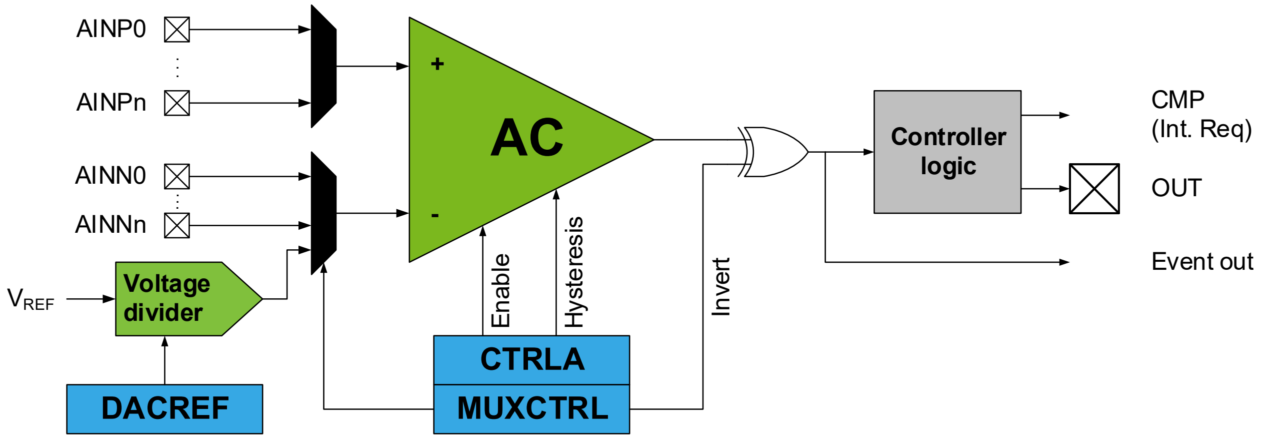


Figure 2-1 shows the block diagram of the analog comparator in tinyAVR® 0- and 1-series, megaAVR® 0-series and AVR® Dx devices. It compares the voltage levels on two inputs and gives a digital output based on this comparison.

The dynamic behavior of the AC can be adjusted by a hysteresis feature. The hysteresis can be customized to optimize the operation for each application. The input selection includes analog port pins and internally generated inputs.

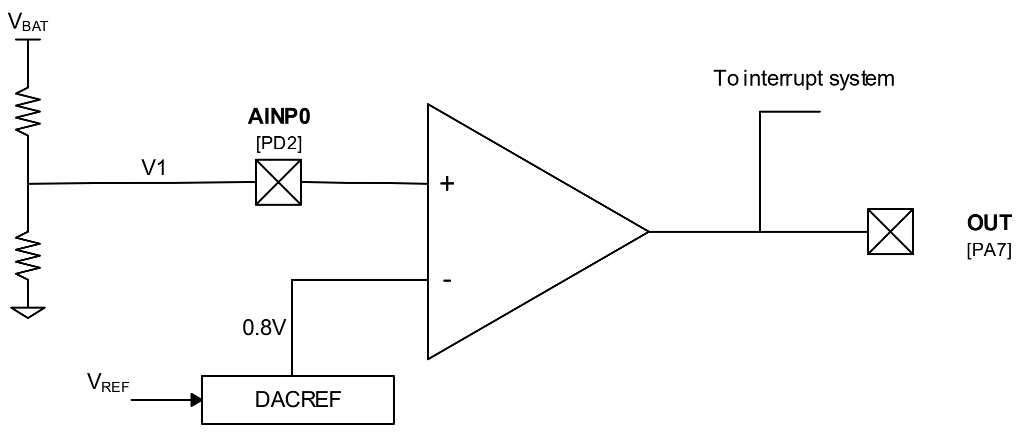
The comparator has one positive input and one negative input. The positive input may be chosen from a selection of analog input pins. The negative input may be chosen from a selection of analog input pins or internal inputs, such as a band gap reference voltage (DACREF). The digital output from the comparator is '1' when the difference between the positive and the negative voltage is positive, and '0' otherwise.

The AC can be configured to generate interrupt requests and/or events upon several different combinations of input change. The AC output can be delivered on an output pin, to be used by external devices.

3. Level Crossing Detector

This example shows a basic initialization and setup for the AC peripheral. The application monitors an analog input signal, compares it to a fixed voltage, and notifies the user via interrupt and an output pin every time the input signal crosses the fixed voltage level. The comparator can be used to monitor battery voltage (or any other DC level).

Figure 3-1. Analog Comparator as Voltage Monitor



To monitor an external voltage, the AC input must be connected to this voltage using an I/O pin. This pin needs to have the digital input buffer and the pull-up resistor disabled, to have the highest possible input impedance. For the ATmega4809, PORTD pin 2 (PD2/AINP0) is used as AC positive input:

Figure 3-2. AC Positive Input

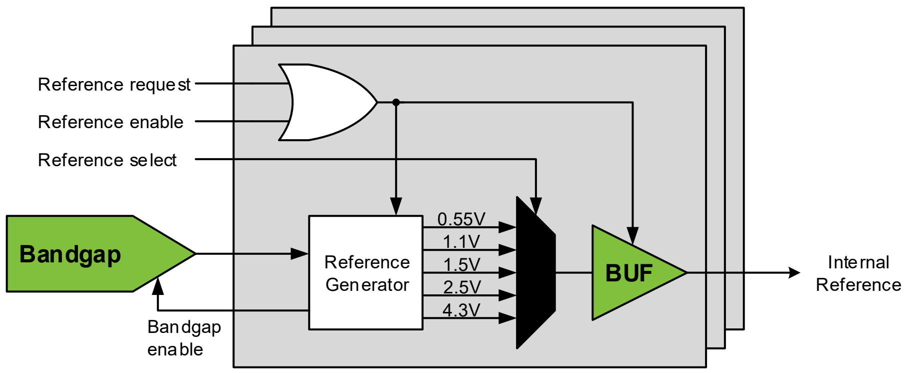
QFN48/ TQFP48	Pin name (1,2)	Special	ADC0	AC0
20	PD0		AIN0	
21	PD1		AIN1	P3
22	PD2		AIN2	P0
23	PD3		AIN3	N0

This translates to the following code:

```
PORTD.PIN2CTRL = PORT_ISC_INPUT_DISABLE_gc;
```

In the selected application, the AC uses the analog pin as positive input and an internal reference for its negative input. The Voltage Reference (V_{REF}) peripheral must be configured before using the internal voltage reference on negative input.

Figure 3-3. V_{REF} Block Diagram



The V_{REF} provides control registers for selecting between multiple internal reference levels. The internal references are generated from an internal band gap.

Figure 3-4. VREF.CTRLA - AC Voltage Reference Selection

Bit	7	6	5	4	3	2	1	0
		ADC0REFSEL[2:0]				AC0REFSEL[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

bits 2:0 AC0REFSEL[2:0]: AC0 Reference Select bits
These bits select reference voltage for AC0.

Value	Name	Description
0x0	0V55	0.55V
0x1	1V1	1.1V
0x2	2V5	2.5V
0x3	4V3	4.3V
0x4	1V5	1.5V
0x5	-	Reserved
0x6	-	Reserved
0x7	AVDD	AVDD

A value of 1.5V is selected as reference voltage:

```
VREF.CTRLA = VREF_AC0REFSEL_1V5_gc;
```

To enable V_{REF} voltage generation, the output buffer must be enabled after the selection of reference voltage. Do this by setting the AC0REFEN bit from the VREF Control B register:

Figure 3-5. VREF.CTRLB - Enable the AC0REFEN bit

Bit	7	6	5	4	3	2	1	0
							ADC0REFEN	AC0REFEN
Access							R/W	R/W
Reset							0	0

bit 0 AC0REFEN: AC0 DACREF Reference Force Enable bit
Writing a '`1`' to this bit forces the voltage reference for AC0 DACREF to be enabled, even if it is not requested.

Writing a '`0`' to this bit allows to automatic enable/disable of the reference source when not requested.

This translates to the following code:

```
VREF.CTRLB = VREF_ADC0REFEN_bm;
```

After configuring the peripherals and the modules required by the AC, the selection of inputs is done using the MUXCTRLA register as follows:

Figure 3-6. AC Input Selection

Bit	7	6	5	4	3	2	1	0
	INVERT			MUXPOS[1:0]			MUXNEG[1:0]	
Access	R/W			R/W	R/W		R/W	R/W
Reset	0			0	0		0	0

bits 4:3 MUXPOS[1:0]: Positive Input MUX Selection bits
Writing to this bit field selects the input signal to the positive input of the AC.

Value	Name	Description
0x0	AINP0	Positive Pin 0
0x1	AINP1	Positive Pin 1
0x2	AINP2	Positive Pin 2
0x3	AINP3	Positive Pin 3

bits 1:0 MUXNEG[1:0]: Negative Input MUX Selection bits
Writing to this bit field selects the input signal to the negative input of the AC.

Value	Name	Description
0x0	AINN0	Negative pin 0
0x1	AINN1	Negative pin 1
0x2	AINN2	Negative pin 2
0x3	DACREF	Internal DAC reference

The positive input pin 0 (AINP0) and internal DAC reference are used as AC inputs:

```
AC0.MUXCTRLA = AC_MUXPOS_PIN0_gc | AC_MUXNEG_DACREF_gc;
```

The analog value used by AC (V_{DACREF}) is derived from internal reference using the DACREF register, and the output voltage is defined by:

$$V_{DACREF} = \frac{DACREF}{256} \times V_{REF}$$

The DACREF value can be calculated using the following macro:

```
#define DACREF_VALUE              (VDACREF * 256 / VREF)
```

where V_{DACREF} represents the desired value on the analog comparator input and V_{REF} represents the value selected as internal reference (1.5 Volts).

Important: Configure the DACREF register to select 0.8V on the negative input to implement this application. The user must select a proper combination of R1 and R2 resistors (see Figure 3-1) in such a way that when battery voltage low threshold is reached, voltage V1 will be 0.8V:

$$V1 = 0.8V = V_{BAT} \times \frac{R2}{(R1 + R2)}$$

Assuming $V_{BAT} = 3\text{V}$ and $R2 = 10\ \text{k}\Omega$, the value of R1 must be $27.5\ \text{k}\Omega$

The output of the comparator is present on the external pin, and the interrupts are enabled on the negative edge to notify the application when the critical level is reached. That is done using the CTRLA register:

Figure 3-7. AC0.CTRLA - Set AC, Enable Interrupts and Output

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY	OUTEN	INTMODE[1:0]		LPMODE	HYSMODE[1:0]		ENABLE
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

bit 6 OUTEN: Analog Comparator Output Pad Enable bit
Writing this bit to '1' makes the OUT signal available on the pin.

bits 5:4 INTMODE[1:0]: Interrupt Modes bits
Writing to these bits selects what edges of the AC output triggers an interrupt request.

Value	Name	Description
`0x0`	BOTHEDGE	Both negative and positive edge
`0x1`	-	Reserved
`0x2`	NEGEDGE	Negative edge
`0x3`	POSEDGE	Positive edge

bit 0 – ENABLE: Enable AC bit
Writing this bit to '1' enables the AC.

These settings translate to the following code:

```
AC0.CTRLA = AC_ENABLE_bm | AC_INTMODE_NEGEDGE_gc | AC_OUTEN_bm;
```

The AC peripheral interrupt must be enabled to complete the setup and make the selected interrupt available to the application. Do this by using the CMP bit from the INTCTRL register:

Figure 3-8. AC0.INTCTRL - Enable AC Interrupt

Bit	7	6	5	4	3	2	1	0
								CMP
Access								R/W
Reset								0

bit 0 CMP: Analog Comparator Interrupt Enable bit
Writing this bit to ‘`1`’ enables Analog Comparator Interrupt.

Tip: The full code example is available in the Appendix section.

View the ATmega4809 Code Example on GitHub
Click to browse repository

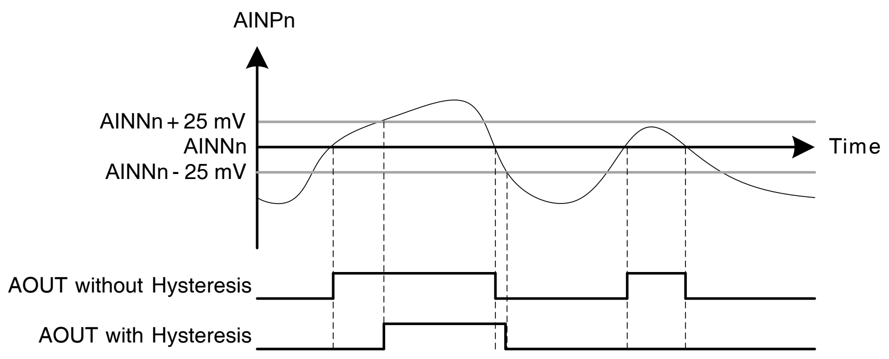
An MCC generated code example for AVR128DA48 with the same functionality as the one described in this section can be found here:

View the AVR128DA48 Code Example on GitHub
Click to browse repository

4. Preventing False Spike Detection

This example demonstrates the hysteresis feature of the AC peripheral that helps in avoiding frequent toggling of the AC when the positive input oscillates very close to the negative input level. This application is similar to the voltage level detector application, but additionally it has the Hysteresis mode enabled.

Figure 4-1. Analog Comparator Response With and Without Hysteresis Enabled



Configure the hysteresis by writing to the Hysteresis Mode Select (HYSMODE[1:0]) bit field in the Control A register:

Figure 4-2. AC0.CTRLA - Configure Hysteresis Mode

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY	OUTEN	INTMODE[1:0]		LPMODE	HYSMODE[1:0]		ENABLE
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

bits 2:1 HYSMODE[1:0]: Hysteresis Mode Select bits
Writing these bits select the hysteresis mode for the AC input.

Value	Name	Description
0x0	NONE	No hysteresis
0x1	SMALL	Small hysteresis
0x2	MEDIUM	Medium hysteresis
0x3	LARGE	Large hysteresis

A medium hysteresis is used (25 mV), which translates in the following sequence of code:

```
AC0.CTRLA |= AC_HYSMODE_25mV_gc;
```

Tip: The full code example is also available in the Appendix section.

View the ATmega4809 Code Example on GitHub
Click to browse repository

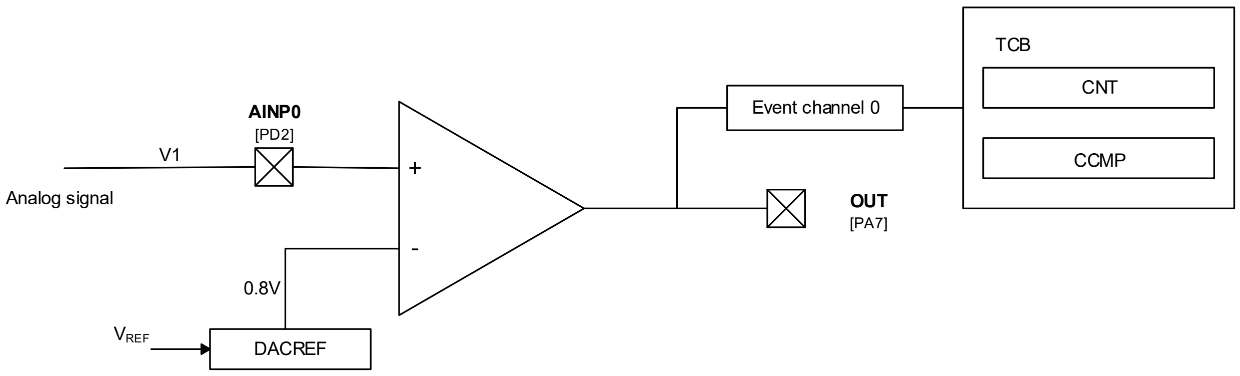
An MCC generated code example for AVR128DA48 with the same functionality as the one described in this section can be found here:

View the AVR128DA48 Code Example on GitHub
Click to browse repository

5. Analog Signal Pulse Duration/Frequency Measurements

The tinyAVR® 0- and 1-series, megaAVR® 0-series and AVR® Dx devices feature Event System (EVSYS), a simple but powerful system that allows autonomous control of peripherals without any use of interrupts, CPU, or DMA resources. It allows a change in one peripheral (the event generator) to trigger actions in other peripherals (the event users) through event channels. It provides short and predictable response times between peripherals and can reduce the complexity, size, and execution time of the software, to save power.

Figure 5-1. Analog Signal Duration/Frequency Measurement Block Diagram



The application example in this chapter shows an implementation of duration/frequency measurement for an analog input signal, with minimal usage of microcontroller power. It uses the Event System to route the signals from the AC output through an event channel to Timer Counter B (TCB) event input. Configure the Event System properly to do this.

The first step in the Event System configuration is to set the AC output as event generator for channel 0:

Figure 5-2. EVSYS.CHANNEL - Set AC Output as Event Generator for Channel 0

Bit	7	6	5	4	3	2	1	0
	GENERATOR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

bits 7:0 GENERATOR[7:0]: Channel Generator Selection bits

Value		Description
...		
0x10-0x13	CCL_LUTn	Event channel connected to LUTn
0x20	AC0	Event channel connected to AC0
0x24	ADC0	Event channel connected to ADC0
...		

For EVSYS channel 0, the channel generator selection register must be loaded with `0x20` to enable analog comparator as event generator:

```
EVSYS.CHANNEL0 = EVSYS_GENERATOR_AC0_OUT_gc;
```

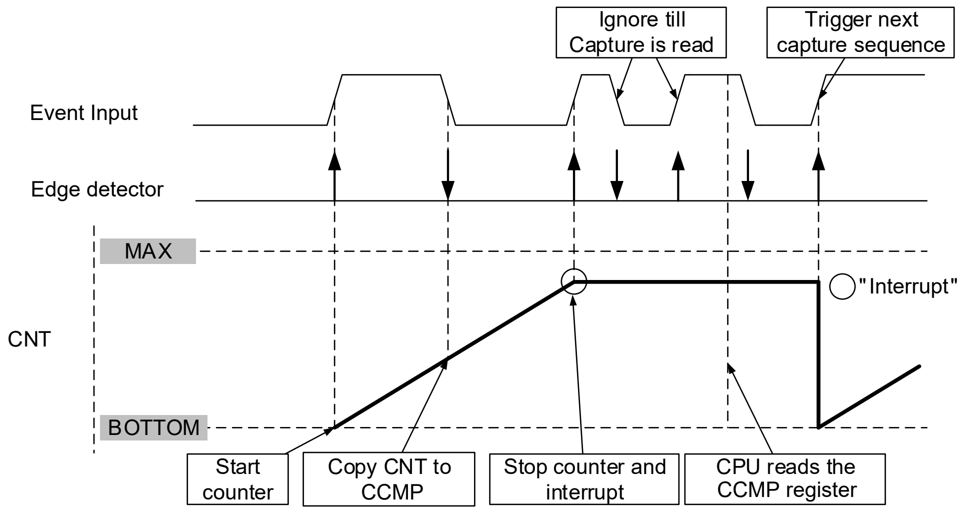
To trigger events on TCB input, the TCB event user must be connected to channel 0:

```
EVSYS.USERTCB0 = EVSYS_CHANNEL_CHANNEL0_gc;
```

To enable pulse and period measurements, the TCB is configured in Pulse-Width Measurement mode, having the Event System as input. The Event System is used to route AC output through event channel 0 to TCB event input.

In Pulse-Width Measurement mode, the TCB will start counting when a positive edge is detected on the event input signal. On the next falling edge, the count value is captured. The counter stops when the second rising edge of the event input signal is detected, and this will set the Interrupt flag. Reading the capture will clear the Interrupt flag. When the Capture register is read, or the Interrupt flag cleared, the TCB becomes ready for a new capture sequence. Therefore, it is recommended to read the Counter register before the Capture register since it is reset to zero at the next positive edge:

Figure 5-3. TCB - Input Capture Frequency and Pulse-Width Measurement


Ignore till Capture is read
Trigger next capture sequence
Event Input
Edge detector
MAX
"Interrupt"
CNT
BOTTOM
Start counter
Copy CNT to CCMP
Stop counter and interrupt
CPU reads the CCMP register

The following code provides basic initialization for TCB in Pulse-Width Measurement mode with Event System as input:

```
int8_t TIMER0_init()
{

    TCB0.CTRLB = TCB_CNTMODE_FRQPW_gc;
    TCB0.EVCTRL = TCB_CAPTEI_bm;

    TCB0.INTCTRL = TCB_CAPT_bm;
    TCB0.CTRLA = TCB_CLKSEL_CLKDIV2_gc
               | TCB_ENABLE_bm
               | TCB_RUNSTDBY_bm;

}
```

Tip: The full code example is also available in the Appendix section.

View the AVR128DA48 Code Example on GitHub
Click to browse repository

An MCC generated code example for AVR128DA48 with the same functionality as the one described in this section can be found here:

View the AVR128DA48 Code Examples on GitHub
Click to browse repository

6. References

1. AVR128DA48 product page: www.microchip.com/wwwproducts/en/AVR128DA48
2. AVR128DA48 Curiosity Nano Evaluation Kit product page: https://www.microchip.com/Developmenttools/ProductDetails/DM164151
3. AVR128DA28/32/48/64 Data Sheet
4. Getting Started with the AVR® DA Family
5. ATmega4809 product page: www.microchip.com/wwwproducts/en/ATMEGA4809
6. megaAVR® 0-series Family Data Sheet
7. ATmega809/1609/3209/4809 – 48-Pin Data Sheet megaAVR® 0-series
8. AN2451 - Getting Started with Core Independent Peripherals on AVR® Microcontrollers (DS00002451)

7. Revision History

Document Revision	Date	Comments
B	02/2021	Updated the GitHub repository links and the *References* section. Added the *AVR® DA Family Overview* and *Revision History* sections. Added MCC versions for each use case, running on AVR128DA48. Other minor corrections.
A	05/2019	Initial document release.

8. Appendix

Example 8-1. Level Crossing Detector Source Code

```
#include <avr/io.h>
#include <avr/interrupt.h>

/* set DACREF to 0.8 Volts for Vref = 1.5Volts */
#define DACREF_VALUE    (0.8 * 256 / 1.5)

void PORT0_init (void);
void AC0_init(void);

ISR(AC0_AC_vect)
{
    /* Insert AC interrupt handling code here */

    /* The interrupt flag has to be cleared manually */
    AC0.STATUS = AC_CMP_bm;
}

void PORT0_init (void)
{
    /* Positive Input - Disable digital input buffer */
    PORTD.PIN2CTRL = PORT_ISC_INPUT_DISABLE_gc;
    /*Enable output buffer on PA7*/
    PORTA |= PIN7_bm;
}

void AC0_init(void)
{

    /* Negative input uses internal reference - voltage reference should be
enabled */
    VREF.CTRLA = VREF_AC0REFSEL_1V5_gc;      /* Voltage reference at 1.5V */
    VREF.CTRLB = VREF_AC0REFEN_bm;           /* AC0 DACREF reference enable:
enabled */

    AC0.DACREF = DACREF_VALUE;               /* Set DAC voltage reference */

    /*Select proper inputs for comparator*/
    AC0.MUXCTRLA = AC_MUXPOS_PIN0_gc         /* Positive Input - Analog
Positive Pin 0 */
                | AC_MUXNEG_DACREF_gc;     /* Negative Input - DAC Voltage
Reference */

    AC0.CTRLA = AC_ENABLE_bm                 /* Enable analog comparator */
              | AC_OUTEN_bm;                 /* Output Buffer Enable: enabled */

    AC0.INTCTRL = AC_CMP_bm;                 /* Analog Comparator 0 Interrupt
enabled */

}

int main(void)
{
    PORT0_init();
    AC0_init();
    sei();                                   /* Global interrupts enabled */

    while (1)
    {
        ;
    }
}
```

Example 8-2. Preventing False Spike Detection Source Code

```
#include <avr/io.h>
#include <avr/interrupt.h>

/* set DACREF to 0.8 Volts for Vref = 1.5Volts */
#define DACREF_VALUE    (0.8 * 256 / 1.5)


void PORT0_init (void);
void AC0_init (void);

ISR(AC0_AC_vect)
{
    /* Insert AC interrupt handling code here */

    /* The interrupt flag has to be cleared manually */
    AC0.STATUS = AC_CMP_bm;
}

void PORT0_init (void)
{
    /* Positive Input - Disable digital input buffer */
    PORTD.PIN2CTRL = PORT_ISC_INPUT_DISABLE_gc;
    /*Enable output buffer on PA7*/
    PORTA |= PIN7_bm;
}

void AC0_init (void)
{

    /* Negative input uses internal reference - voltage reference should be
enabled */
    VREF.CTRLA = VREF_AC0REFSEL_1V5_gc;      /* Voltage reference at 1.5V */
    VREF.CTRLB = VREF_AC0REFEN_bm;           /* AC0 DACREF reference enable:
enabled */

    AC0.DACREF = DACREF_VALUE;               /* Set DAC voltage reference */

    /*Select proper inputs for comparator*/
    AC0.MUXCTRLA = AC_MUXPOS_PIN0_gc         /* Positive Input - Analog
Positive Pin 0 */
                | AC_MUXNEG_DACREF_gc;   /* Negative Input - DAC Voltage
Reference */

    AC0.CTRLA = AC_ENABLE_bm                 /* Enable analog comparator */
            | AC_HYSMODE_25mV_gc         /* Enable hysteresis @25mV  */
            | AC_OUTEN_bm;               /* Output Buffer Enable: enabled */

    AC0.INTCTRL = AC_CMP_bm;                 /* Analog Comparator 0 Interrupt
enabled */

}

int main(void)
{
    PORT0_init();
    AC0_init();
    sei();                                   /*Global interrupts enabled */

    while (1)
    {
        ;
    }
}
```

Example 8-3. Analog Signal Pulse Duration/Frequency Measurement Source Code

```
#include <avr/io.h>

/* set DACREF to 0.8 Volts for Vref = 1.5Volts */
#define DACREF_VALUE    (0.8 * 256 / 1.5)

void PORT0_init (void);
void EVENT_SYSTEM_init (void);
void AC0_init (void);
void TIMER0_init (void);


void PORT0_init (void)
{
    /* Positive Input - Disable digital input buffer */
    PORTD.PIN2CTRL = PORT_ISC_INPUT_DISABLE_gc;
    /*Enable output buffer on PA7*/
    PORTA |= PIN7_bm;
}

void AC0_init (void)
{

    /* Negative input uses internal reference - voltage reference should be
enabled */
    VREF.CTRLA = VREF_AC0REFSEL_1V5_gc;      /* Voltage reference at 1.5V */
    VREF.CTRLB = VREF_AC0REFEN_bm;           /* AC0 DACREF reference enable:
enabled */

    AC0.DACREF = DACREF_VALUE;               /* Set DAC voltage reference */

    /*Select proper inputs for comparator*/
    AC0.MUXCTRLA = AC_MUXPOS_PIN0_gc         /* Positive Input - Pin 0 */
                 | AC_MUXNEG_DACREF_gc;      /* Negative Input - DAC Voltage
Reference */

    AC0.CTRLA = AC_ENABLE_bm                 /* Enable analog comparator */
              | AC_OUTEN_bm;                 /* Output Buffer Enable: enabled */

    AC0.INTCTRL = 0;                         /* Analog Comparator 0 Interrupt
disabled */

}

/*Init TCB in pulse width-frequency measurement mode, input from Analog
Comparator through Event System */
void TIMER0_init (void)
{

    TCB0.CTRLB = TCB_CNTMODE_FRQPW_gc;       /* Input Capture Frequency and
Pulse-Width measurement */
    TCB0.EVCTRL = TCB_CAPTEI_bm;             /* Event Input Enable: enabled */

    TCB0.INTCTRL = TCB_CAPT_bm;              /* Capture or Timeout: enabled */

    TCB0.CTRLA = TCB_CLKSEL_CLKDIV2_gc       /* CLK_PER/2 (From Prescaler) */
              | TCB_ENABLE_bm                /* Enable: enabled */
              | TCB_RUNSTDBY_bm;             /* Run Standby: enabled */

}

/* Enable event generation from Analog comparator to TCB */
void EVENT_SYSTEM_init (void)
{
    EVSYS.CHANNEL0 = EVSYS_GENERATOR_AC0_OUT_gc;     /* Analog Comparator 0 out
linked to Event Channel 0 */
    EVSYS.USERTCB0 = EVSYS_CHANNEL_CHANNEL0_gc;      /* TCB uses Event Channel
0 */
}

int main(void)
{
    uint16_t signal_pulse = 0, signal_period = 0;
    PORT0_init();
    AC0_init();
    EVENT_SYSTEM_init();
    TIMER0_init();

    while(1)
    {
        if (TCB0.INTFLAGS)
        {
            /**
            * First read the CNT register
            * The interrupt flag is cleared by writing 1 to it, or when the
Capture register
            * is read in Capture mode
            */
            signal_period = TCB0.CNT;
            signal_pulse  = TCB0.CCMP;
        }
    }
}
```

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Microchip Devices Code Protection Feature

ISBN: 978-1-5224-7538-5

Quality Management System

For information regarding Microchip’s Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4485-5910
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-72400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-72884388

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820