MICROCHIP

TB3233

Using ZCD to Implement Special Functions

Introduction

Author: Gheorghe Turcan, Microchip Technology Inc.

The Microchip AVR® DA MCU family of microcontrollers features up to three Zero-Cross Detectors (ZCD) with flexible input selection, requiring only one external component, and configurable output (interrupts on rising/falling or both edges, event generation, and output inversion).

This technical brief explains the concepts of the ZCD and its implementation in the AVR DA family of microcontrollers with the following use cases:

- **Free-Running to External I/O**
  This example shows how to use the ZCD to detect when the zero value of an analog signal is reached (for example, the zero-cross of the Alternating Current mains electricity).
- **AC Signal Frequency Detection**
  This example describes how to use the ZCD together with a timer to measure the pulse duration and/or the period of analog signals with minimal intervention of the CPU.
- **Active Bridge Control Signal Generation**
  This example describes how to use the ZCD together with the CPU to create a Pulse Width Modulated (PWM) and inverted PWM depiction of the Alternating Current (AC) signal, to control an active bridge.

Note: The code examples were developed using the AVR128DA48 Curiosity Nano board.

Table of Contents

1. Relevant Devices

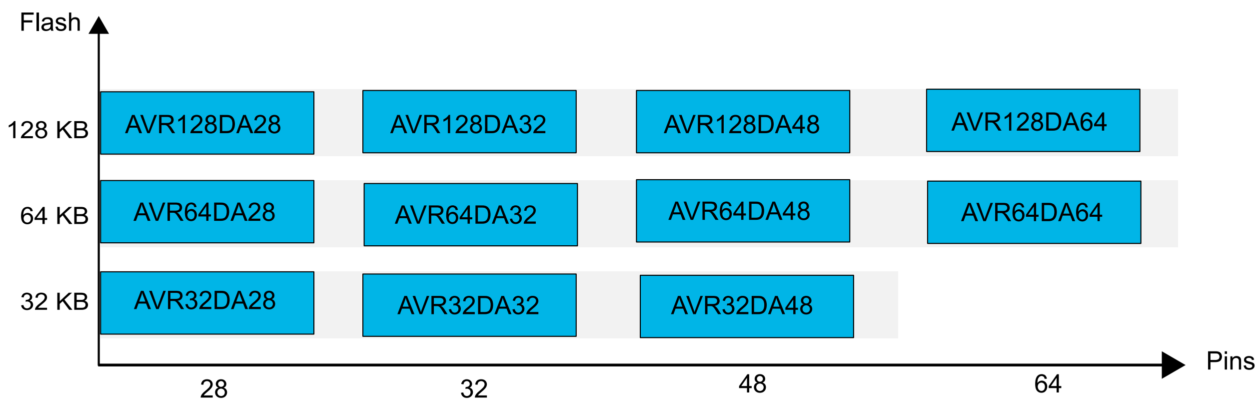
This section lists the relevant devices for this document.

1.1 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 1-1. AVR® DA Family Overview



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

2. Overview

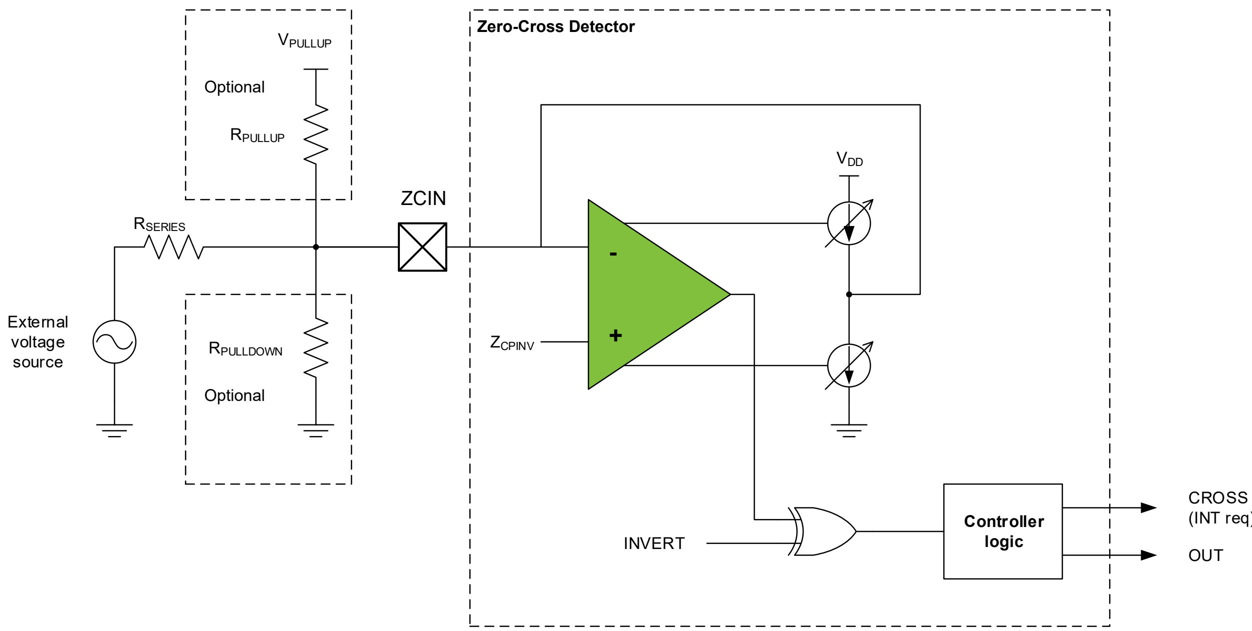
The ZCD detects when an alternating voltage crosses through a threshold voltage level near ground potential. The threshold is the zero-cross reference voltage, Z_{CPINV}. The connection from the ZCD input pin (ZCIN) to the alternating voltage must be made through a series current-limiting resistor (R_{SERIES}). The ZCD applies either a current source or current sink to the ZCD input pin to maintain a constant voltage on the pin, thereby preventing the pin voltage from forward biasing the electrostatic discharge (ESD) protection diodes in the device.

When the applied voltage is greater than the reference voltage, the ZCD sinks current. When the applied voltage is less than the reference voltage, the ZCD sources current, thus allowing connection to high voltages without a resistance divider, and only through a series resistance.

The ZCD can be used when monitoring an alternating waveform for, but not limited to, the following purposes:

- Period measurement
- Accurate long-term time measurement
- Dimmer phase-delayed drive
- Low-EMI cycle switching

Figure 2-1. Zero-Cross Detector Block Diagram



The ZCD requires a current-limiting resistor in series (R_{SERIES}) with the external voltage source. If the peak amplitude (V_{PEAK}) of the external voltage source is expected to be stable, the resistor value must be chosen such that a 300 μA resistor current results in a voltage drop equal to the expected peak voltage. The power rating of the resistor must be at least the mean square voltage divided by the resistor value. The STATE bit in the ZCDn.STATUS register indicates whether the input signal is above or below the reference voltage, Z_{CPINV}. By default, the STATE bit is '1' when the input signal is above the reference voltage and '0' when the input signal is below the reference voltage. The polarity of the STATE bit can be reversed by writing the INVERT bit to '1' in the ZCDn.CTRLA register. The INVERT bit will also affect the ZCD interrupt polarity.

The actual voltage at which the ZCD changes state is the zero-cross reference voltage. Because this reference voltage is slightly offset from the ground, the zero-cross event generated by the ZCD will occur either early or late with respect to the true zero-crossing. This offset time can be compensated by adding a pull-up or pull-down biasing resistor to the ZCD input pin. A pull-up resistor is used when the external voltage source is referenced to ground, and a pull-down resistor is used when the voltage is referenced to V_{DD}, as shown in Figure 2-1.

The following equations help calculate the external components value:

Equation 2-1. External Resistor Value Calculation

$$R_{SERIES} = \frac{V_{PEAK}}{3x10^{-4}}$$

When the External Voltage source is referenced to ground:

Equation 2-2. ZCD Pull-Up Resistor Value Calculation

$$R_{PULLUP} = \frac{R_{SERIES}(V_{PULLUP} - Z_{CPINV})}{Z_{CPINV}}$$

When the External Voltage source is referenced to V_{DD}:

Equation 2-3. ZCD Pull-Down Resistor Value Calculation

$$R_{PULLDOWN} = \frac{R_{SERIES} x Z_{CPINV}}{V_{DD} - Z_{CPINV}}$$

In Idle sleep mode, the ZCD will continue to operate as normal.

In Standby sleep mode, the ZCD is disabled by default. If the Run in Standby (RUNSTDBY) bit in the Control A (ZCDn.CTRLA) register is written to '1', the ZCD will continue to operate as normal with interrupt generation, event generation, and ZCD output on pin even if CLK_PER is not running in Standby sleep mode. In Power-Down sleep mode, the ZCD is disabled, including its output to pin.

The ZCD can generate the following events:

Table 2-1. ZCD Event Generator

Generator Name		Description	Event Type	Generator Clock Domain	Length of Event
Peripheral	Event				
ZCDn	OUT	ZCD output level	Level	Asynchronous	Determined by the ZCD output level

The ZCD has no event inputs.

3. Free-Running to External I/O

This example shows a basic initialization and link to I/O ports of the ZCD peripheral. This example is used to obtain PWM representation of a sinusoidal AC signal, and the ZCD is working independently of the CPU.

How to select the external components

The AC signal was obtained from the signal generation capability of the oscilloscope. It has a sinusoidal 5V peak-to-peak amplitude and is offset by one volt from ground to compensate for Z_{CPINV}. Using Equation 2-1, the selected R_{SERIES} value is approximately 10 kΩ. The connection setup is depicted in Figure 3-2.

How to set up the internal circuit

For basic operation, the following steps are taken:

1. Configure the desired input pin in the PORT peripheral as an analog pin with digital input buffer disabled. Internal pull-up and pull-down resistors must also be disabled.

   To allow the zero-cross detection, the ZCD input has to be connected to the sensing 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 AVR128DA48, PORTD pin 1 (PD1) is used as ZCD0 negative input, as depicted in the figure below.

   Figure 3-1. ZCD I/O Connections

VQFN48/TQFP48	PIN Name	Special	ADC0	ZCDn
3	PA7	CLKOUT		0,OUT 1,OUT 2,OUT
19	PC7			0,OUT 1,OUT 2,OUT
21	PD1		AIN1	0,ZCIN
33	PE3		AIN11	1,ZCIN

   This translates to the following code:

   ```
   PORTD.PIN1CTRL |= PORT_ISC_INPUT_DISABLE_gc;
   ```

2. Optional: Enable the output pin by writing a '1' to the Output Enable (OUTEN) bit in the Control A (ZCDn.CTRLA) register and set the desired pin as an output.
   For the AVR128DA48, PORTA pin 7 (PA7) is used as ZCD0 output, as depicted in the figure above.

   This translates to the following code:

   ```
   ZCD0.CTRLA = ZCD_OUTEN_bm;
   PORTA.DIRSET |= PIN7_bm;
   PORTA.OUTSET |= PIN7_bm;
   ```

3. Enable the ZCD by writing a '1' to the ENABLE bit in ZCDn.CTRLA.
   After the ZCD is enabled, there is a start-up time during which the output of the ZCD may be invalid. Refer to the *Electrical Characteristics* section in the device data sheet for the typical value of the start-up time.

   Enabling the ZCD can be done using the following line of code:

   ```
   ZCD0.CTRLA = ZCD_ENABLE_bm;
   ```

   All this can be encapsulated into one initialization function (e.g., 'ZCD_Init') to keep the code easier to follow.

The oscilloscope measurements from Figure 3-3 depicts the input and output of the ZCD as it works in Free-Running mode measured from the I/O pins of the MCU.

Figure 3-2. ZCD Connected as Free-Running to External I/O

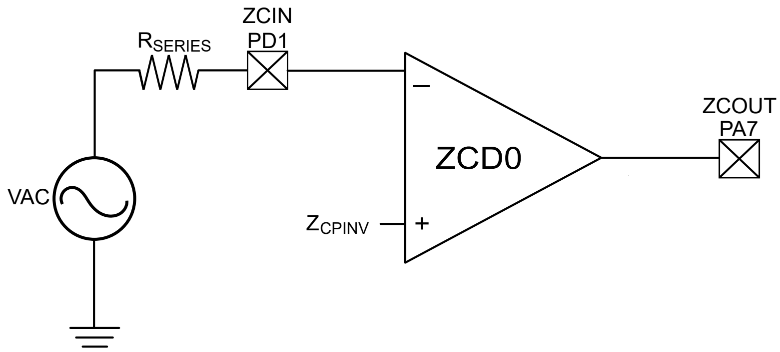
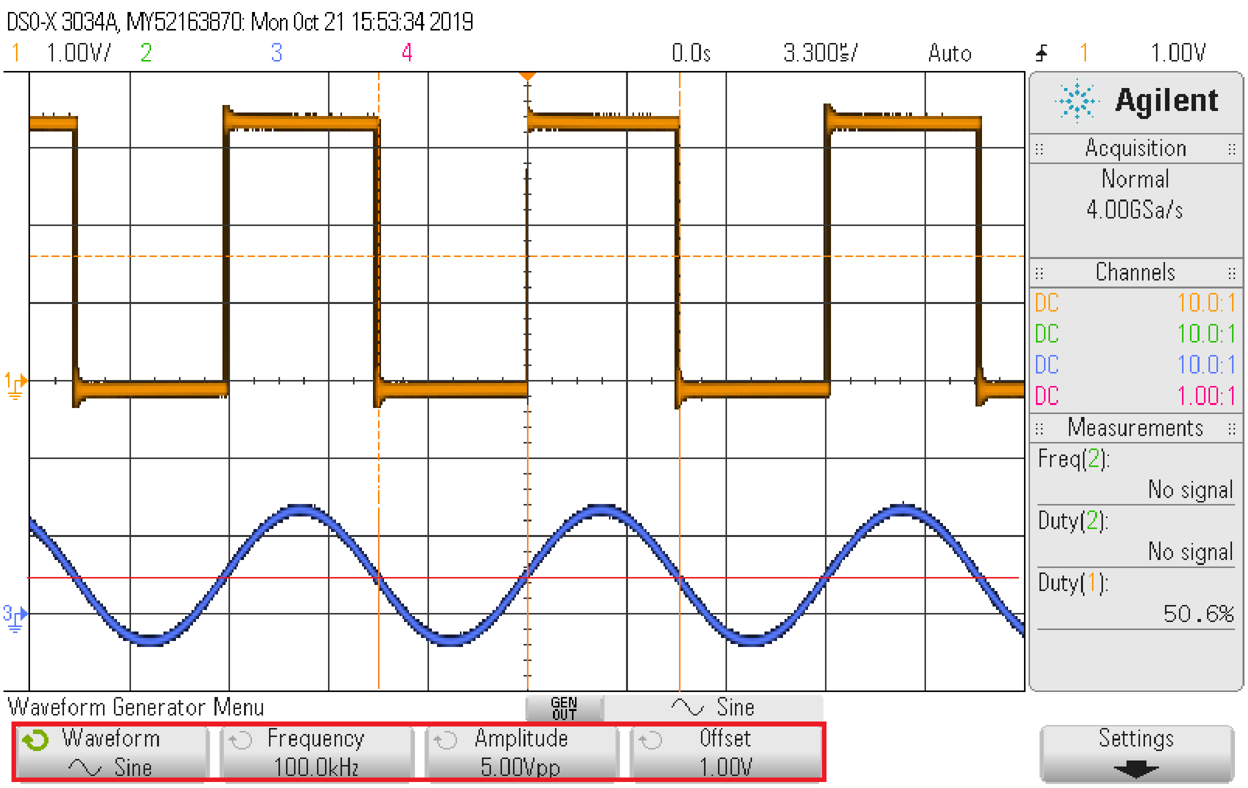


Figure 3-3. Oscilloscope Measurement of the Free-Running ZCD; CH1 – ZCD_OUT, CH3 – ZCD_IN



The STATE bit in the ZCDn.STATUS register indicates whether the input signal is above or below the reference voltage, Z_{CPINV}. By default, the STATE bit is '1' when the input signal is above the reference voltage and '0' when the input signal is below the reference voltage. The polarity of the STATE bit can be reversed by writing the INVERT bit to '1' in the ZCDn.CTRLA register. The INVERT bit will also affect the ZCD interrupt polarity.

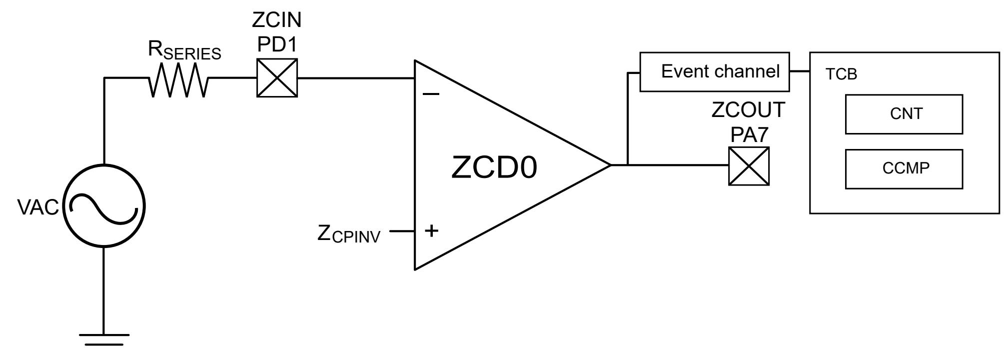
View Code Example on GitHub
Click to browse repository

4. AC Signal Frequency Detection

The AVR DA family features an Event System (EVSYS), enabling inter-peripheral communication without using interrupts or CPU. It allows a change in one peripheral (the event generator) to trigger actions in other peripherals (the event users) through event channels. The EVSYS provides short latency-free and predictable communication between peripherals and can reduce the complexity, size, and execution time of the software, to save power.

The next application example shows an implementation of duration/frequency measurement for an AC input signal, with minimal usage of microcontroller power. It uses the EVSYS to route the signals from the ZCD output through an event channel to Timer Counter B (TCB) event input. To do this, the EVSYS must be configured properly. This measurement can be done only with ZCD, as the Analog Comparator does not allow negative voltages on its inputs.

Figure 4-1. AC Signal Frequency Measurement



The first step in the EVSYS configuration is to set the ZCD output as an event generator for channel 0:

Figure 4-2. EVSYS.CHANNEL - Set ZCD0 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

GENERATOR			Async/Sync	Description	Channel Availability
Value	Name				
	Peripheral	Output			
0x30	ZCD0	OUT	Async	ZCD output level	All channels
0x31	ZCD1(1)				
0x32	ZCD2(1)				

For EVSYS channel 0, the Channel Generator Selection register must be loaded with `0x30` to enable ZCD0 as event generator:

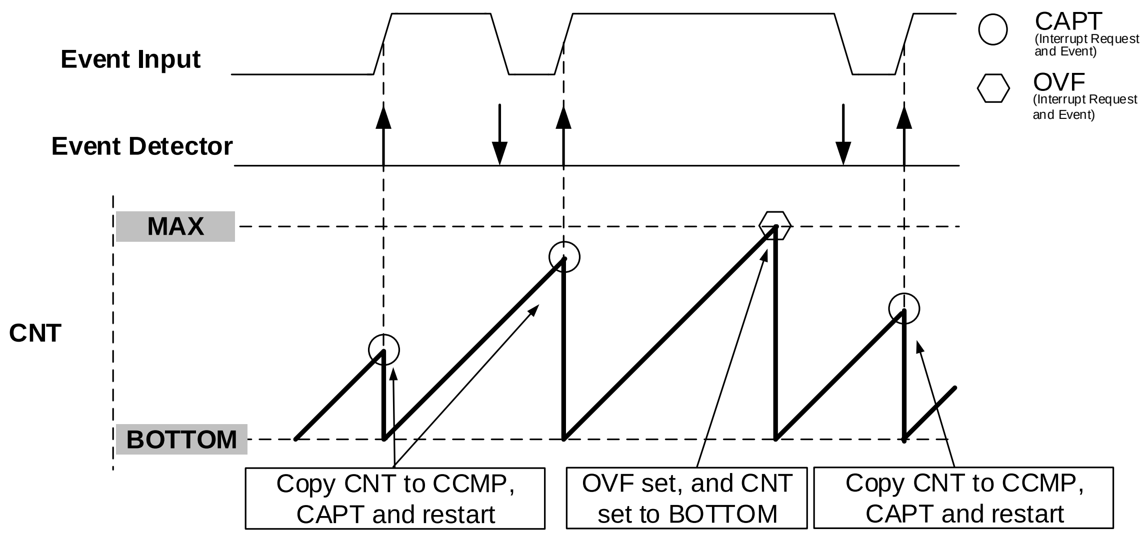
```
EVSYS.CHANNEL0 = EVSYS_CHANNEL0_ZCD0_gc;
```

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

```
EVSYS.USERTCB0CAPT = EVSYS_CHANNEL00_bm;
```

To enable frequency measurement, the TCB is configured in Frequency Measurement mode, having the EVSYS as input, which is used to route the ZCD0 output through event channel 0 to the TCB event input. In the Input Capture Frequency Measurement mode, the TCB captures the counter value and restarts on either a positive or negative edge of the event input signal. The CAPT Interrupt flag is automatically cleared after the low byte of the Compare/Capture (TCBn.CCMP) register has been read. An OVF interrupt and event is generated when the CNT reaches maximum value.

Figure 4-3. TCB - Input Capture Frequency Measurement



The following code provides a basic initialization for the TCB in Frequency Measurement mode with EVSYS as input:

```
void TCB_Init (void)
{
    TCB0.CTRLB = TCB_CNTMODE_FRQ_gc;
    TCB0.EVCTRL = TCB_CAPTEI_bm;
    TCB0.INTCTRL = TCB_CAPT_bm;
    TCB0.CTRLA = TCB_CLKSEL_DIV2_gc
    | TCB_ENABLE_bm
    | TCB_RUNSTDBY_bm;
}
```

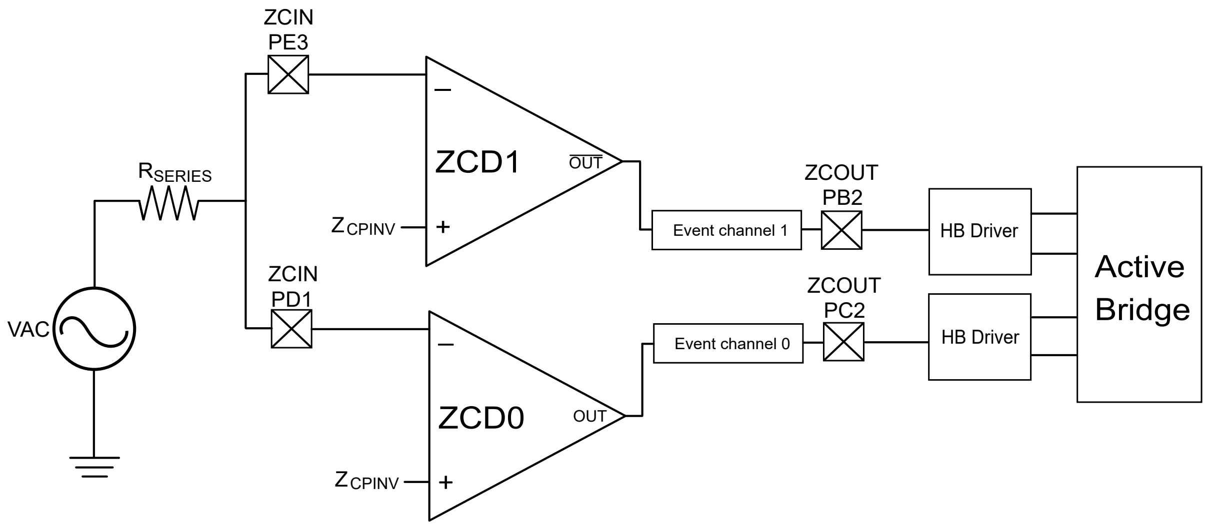
For this example, the MCU internal clock is set to 4 MHz, and the clock select of the TCB is divided by 2 to be able to count until 40 Hz without having an overflow. If the same result is needed with a 20 MHz internal clock, then Timer Counter A (TCA) can be used to divide the clock further with its prescaler and use the TCA clock as a clock source to the TCB.

View Code Example on GitHub
Click to browse repository

5. Active Bridge Control Signal Generation

This application example shows the implementation of an active bridge driven with the help of the ZCD peripherals, see Figure 5-1. This solution uses no CPU intervention and can adapt to frequency changes because there are only internal analog connections and functions. It uses the EVSYS to route the signals from the ZCD output through an event channel to an I/O pin. To do this, the EVSYS must be configured properly.

Figure 5-1. Active Bridge Driver Implementation with ZCDs



The first step in the EVSYS configuration is to set the ZCD0 output as an event generator for channel 0, and the ZCD1 inverted output as an event generator for channel 1. For EVSYS channels 0 and 1, the channel generator selection register must be loaded with `0x30` to enable the ZCD0 as an event generator. To trigger events on I/O pins, the EVSYS event user must be connected to channels 0 and 1:

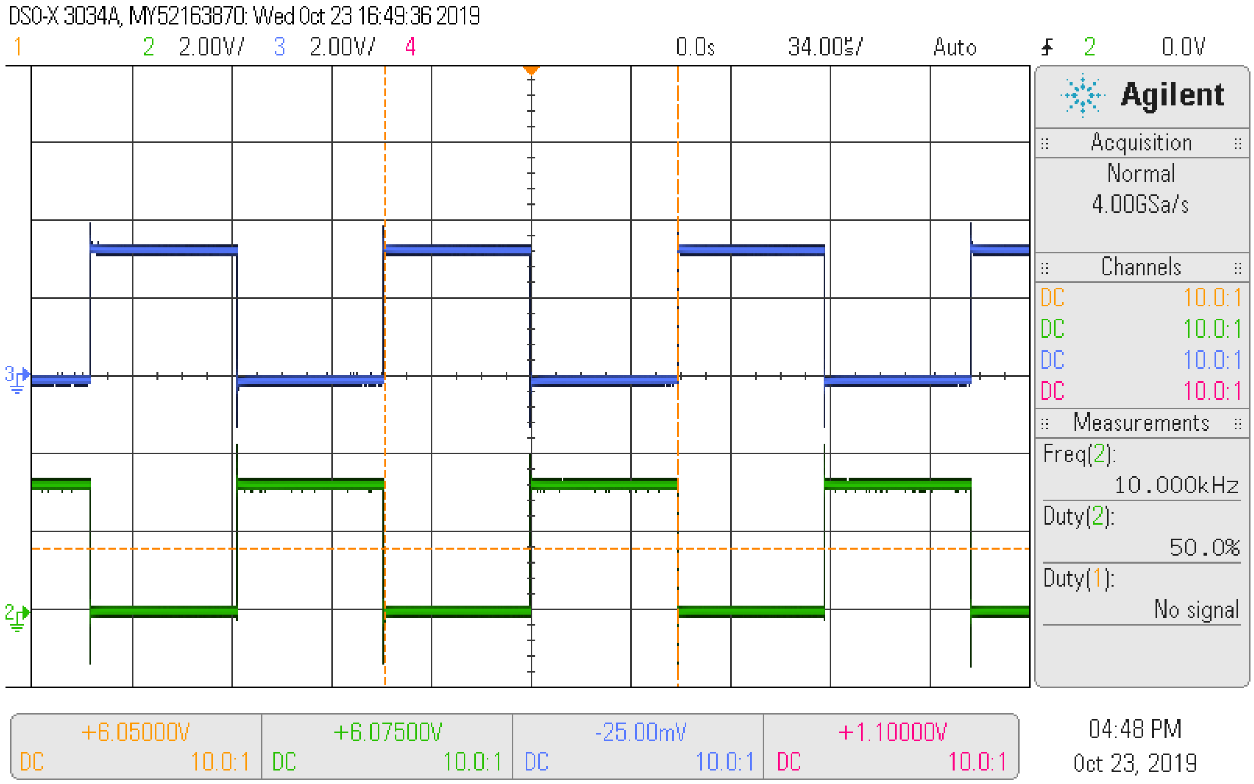
```
EVSYS.CHANNEL0 = EVSYS_CHANNEL0_ZCD0_gc;
EVSYS.USEREVSYSEVOUTB = EVSYS_CHANNEL00_bm;
EVSYS.CHANNEL1 = EVSYS_CHANNEL1_ZCD1_gc;
EVSYS.USEREVSYSEVOUTC = EVSYS_CHANNEL01_bm;
```

Then the input and output on the desired pins can be enabled:

```
PORTE.PIN3CTRL = PORT_ISC_INPUT_DISABLE_gc;

PORTD.PIN1CTRL = PORT_ISC_INPUT_DISABLE_gc;

PORTB.OUTSET |= PIN2_bm;
PORTB.DIRSET |= PIN2_bm;

PORTC.OUTSET |= PIN2_bm;
PORTC.DIRSET |= PIN2_bm;
```

The oscilloscope measurements (refer to Figure 5-2) show a successful implementation that adapts instantly to frequency changes. The I/O pins cannot drive the transistors directly, so a half-bridge or full-bridge driver with dead-band delay capabilities is necessary.

Figure 5-2. Driving Signals for the Active Bridge Implementation



View Code Example on GitHub
Click to browse repository

6. References

1. *AVR128DA28/32/48/64 Preliminary Data Sheet.*
2. *AVR128DA48 Curiosity Nano Board User's Guide.*

7. Appendix

Example 7-1. ZCD Free-Running Code Example

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

void ZCD0_Init(void);
void PORT_Init(void);

/* Initialization of the Zero Cross Detector */
void ZCD0_Init(void)
{
    ZCD0.CTRLA = ZCD_ENABLE_bm       /* Enable the ZCD */
               | ZCD_OUTEN_bm;    /* Enable the output of the ZCD */
}

void PORT_Init(void)
{
    /* Disable digital input buffer and pull-up resistor on PD1*/
    PORTD.PIN1CTRL |= PORT_ISC_INPUT_DISABLE_gc;

    /* Enable PA7 to become a gateway for an internal signal */
    PORTA.DIRSET |= PIN7_bm;
    PORTA.OUTSET |= PIN7_bm;
}

int main(void)
{
    ZCD0_Init();
    PORT_Init();

    while(1)
    {
        ;
    }
}
```

Example 7-2. ZCD Frequency Detection Code Example

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

uint16_t signal_frequency = 0;

void CLK_Init (void);
void PORT_Init (void);
void ZCD0_Init (void);
void TCB0_Init (void);
void EVSYS_Init (void);

void CLK_Init(void)
{
    /* Select internal HF oscillator */
    _PROTECTED_WRITE (CLKCTRL.MCLKCTRLA, CLKCTRL_CLKSEL_OSCHF_gc);
    /* Set internal clock to 4 Mhz */
    _PROTECTED_WRITE (CLKCTRL.OSCHFCTRLA, CLKCTRL_FREQSEL_4M_gc);
}

void PORT_Init(void)
{
    /* Disable digital input buffer and pull-up resistor on PD1*/
    PORTD.PIN1CTRL = PORT_ISC_INPUT_DISABLE_gc;

    /* Enable PA7 to become a gateway for an internal signal */
    PORTA.OUTSET |= PIN7_bm;
    PORTA.DIRSET |= PIN7_bm;
}

/* Initialization of the Zero Cross Detector */
void ZCD0_Init(void)
```

```
{
    ZCD0.CTRLA = ZCD_ENABLE_bm    /* Enable the ZCD */
            | ZCD_OUTEN_bm;    /* Enable the output of the ZCD */
}

/* Initialize the TCB in pulse width-frequency measurement mode,
   input from ZCD through Event System */
void TCB0_Init(void)
{
    TCB0.CTRLB = TCB_CNTMODE_FRQ_gc;    /* Input Capture Frequency */
    TCB0.EVCTRL = TCB_CAPTEI_bm;        /* Event Input Enable: enabled */
    TCB0.INTCTRL = TCB_CAPT_bm;         /* Capture or Timeout: enabled */

    TCB0.CTRLA = TCB_CLKSEL_DIV2_gc     /* CLK_PER/2 (From Prescaler) - This
is needed to be
                                             able to count to 40 Hz with a 4
Mhz system clock */
            | TCB_ENABLE_bm       /* Enable: enabled */
            | TCB_RUNSTDBY_bm;    /* Run Standby: enabled */
}

/* Enable event generation from ZCD to TCB */
void EVSYS_Init(void)
{
    /* Zero-cross detector 0 out linked to event channel 0 */
    EVSYS.CHANNEL0 = EVSYS_CHANNEL0_ZCD0_gc;
    /* TCB uses event channel 0 */
    EVSYS.USERTCB0CAPT = EVSYS_CHANNEL00_bm;
}

int main(void)
{
    CLK_Init();
    TCB0_Init();
    EVSYS_Init();
    ZCD0_Init();
    PORT_Init();

    while(1)
    {
        ;
    }
}

ISR(TCB0_INT_vect)
{
    /* The frequency is stored in the CCMP register
       An equation can be used to transform the value in Hz */
    signal_frequency = TCB0.CCMP;
}
```

Example 7-3. ZCD Active Bridge Code Example

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

void ZCD0_Init(void);
void ZCD1_Init(void);
void PORT_Init(void);
void EVSYS_Init(void);

/* Initialization of the IO Ports */
void PORT_Init(void)
{
    /* Disable digital input buffer and pull-up resistor on PE3*/
    PORTE.PIN3CTRL = PORT_ISC_INPUT_DISABLE_gc;

    /* Disable digital input buffer and pull-up resistor on PD1*/
    PORTD.PIN1CTRL = PORT_ISC_INPUT_DISABLE_gc;

    /* SET PB2 as an output*/
    PORTB.OUTSET |= PIN2_bm;
    PORTB.DIRSET |= PIN2_bm;
```

```
    /* SET PC2 as an output*/
    PORTC.OUTSET |= PIN2_bm;
    PORTC.DIRSET |= PIN2_bm;
}

/* Initialization of the Zero Cross Detector */
void ZCD0_Init(void)
{
    ZCD0.CTRLA = ZCD_ENABLE_bm;     /* Enable the ZCD0 */
}

void ZCD1_Init(void)
{
    ZCD1.CTRLA = ZCD_ENABLE_bm      /* Enable the ZCD1 */
            | ZCD_INVERT_bm;    /* Invert the ZCD1 OUTPUT */
}

/* Enable event generation from ZCD to Event system OUTPUT I/O pins */
void EVSYS_Init(void)
{
    /* Zero-cross detector 0 out linked to event channel 0 */
    EVSYS.CHANNEL0 = EVSYS_CHANNEL0_ZCD0_gc;
    /* Event system output B (PINB2) uses event channel 0 */
    EVSYS.USEREVSYSEVOUTB = EVSYS_CHANNEL00_bm;

    /* Zero-cross detector 1 out linked to event channel 1 */
    EVSYS.CHANNEL1 = EVSYS_CHANNEL1_ZCD1_gc;
    /* Event system output C (PINC2) uses event channel 1 */
    EVSYS.USEREVSYSEVOUTC = EVSYS_CHANNEL01_bm;
}

int main(void)
{
    EVSYS_Init();
    ZCD0_Init();
    ZCD1_Init();
    PORT_Init();

    while(1)
    {
        ;
    }
}
```

8. Revision History

Document Revision	Date	Comments
D	05/2020	Updated AVR® MCU DA (AVR-DA) to AVR® DA MCU, and AVR-DA to AVR DA, per latest trademarking
C	03/2020	Updated AVR-DA to AVR® MCU DA (AVR-DA), per latest trademarking
B	03/2020	Updated repository links
A	02/2020	Initial document release

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

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with

Trademarks

ISBN: 978-1-5224-6076-3

Quality Management System

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

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
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	**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	**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	**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