TB3236

Generating PWM Signals Using TCD with High-Frequency Input

Introduction

Author: Marius Nicolae, Microchip Technology Inc.

The AVR® DA MCU family of microcontrollers is equipped with a versatile type of timer/counter, which can be clocked by four of the frequency channels available to the user. The input frequency for the 12-bit Timer/Counter type D (TCD) varies from the 32.768 kHz input frequency provided by the Ultra Low-Power (ULP) internal or external oscillator, up to 48 MHz by using the Phase-Locked Loop (PLL) clock multiplication system. Thus, the TCD can be used to generate a wide range of Pulse-Width Modulation (PWM) signals.

The scope of this technical brief is to describe some of the TCD operating modes, emphasizing the TCD particularities and providing initialization code snippets. For a deeper understanding of the functionality, refer to the AVR DA data sheet.

This technical brief will describe two use cases where the AVR DA high-frequency sources are used as input clock for the TCD – the Internal High-Frequency Oscillator (OSCHF) and the PLL (clocked from the OSCHF):

- **Generate Two PWM Signals in One Ramp Mode:**
  The purpose of this example is to generate two PWM signals using the TCD configured in One Ramp mode and with OSCHF as input clock. OSCHF will be configured to run at 24 MHz and the PWM signals will be set as output to WOA on the PA4 pin and WOB on the PA5 pin. The PWM signal on WOA will have a 25% duty cycle and the signal on WOB will have a 30% duty cycle.
- **Generate Two PWM Signals in Two Ramp Mode:**
  The scope of this example is to generate two PWM signals using the TCD configured in Two Ramp mode with PLL as input clock. The PLL will be configured to run at 48 MHz and the PWM signals will be set as output to WOA on the PA4 pin and WOB on the PA5 pin. The PWM signal on WOA will have a 10% duty cycle and the signal on WOB will have a 20% duty cycle.

Note: In each example, the PA4 and PA5 pins are used for the PWM signals generation. The code examples were developed using the AVR128DA48 Curiosity Nano development board.

Table of Contents

1. Relevant Devices

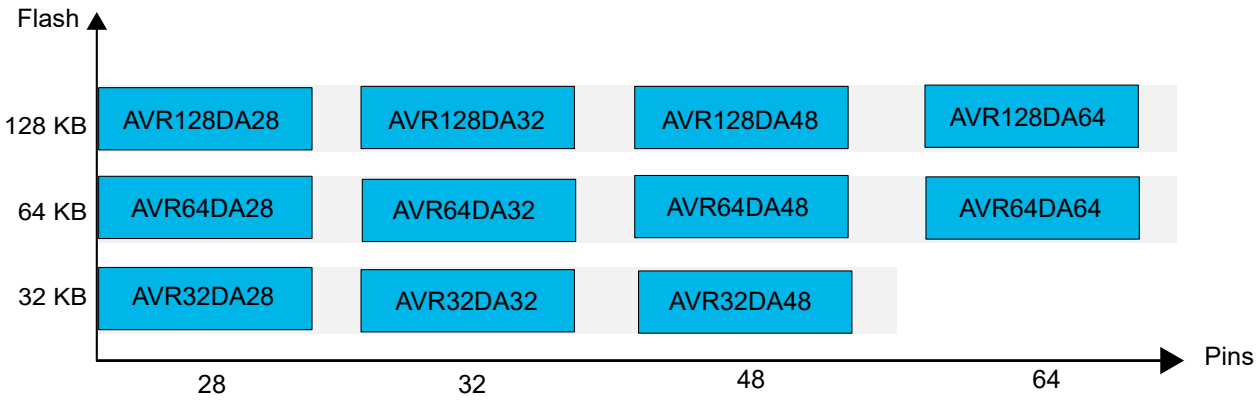
This chapter lists the relevant devices for this document.

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 12-bit Timer/Counter type D (TCD) is a high-performance waveform generator that consists of an asynchronous counter, a prescaler, and compare, capture and control logic. The TCD contains a counter that can run on a clock which is asynchronous to the peripheral clock. It contains compare logic that can generate two independent outputs with optional dead time.

Figure 2-1. Timer/Counter Type-B Block Diagram

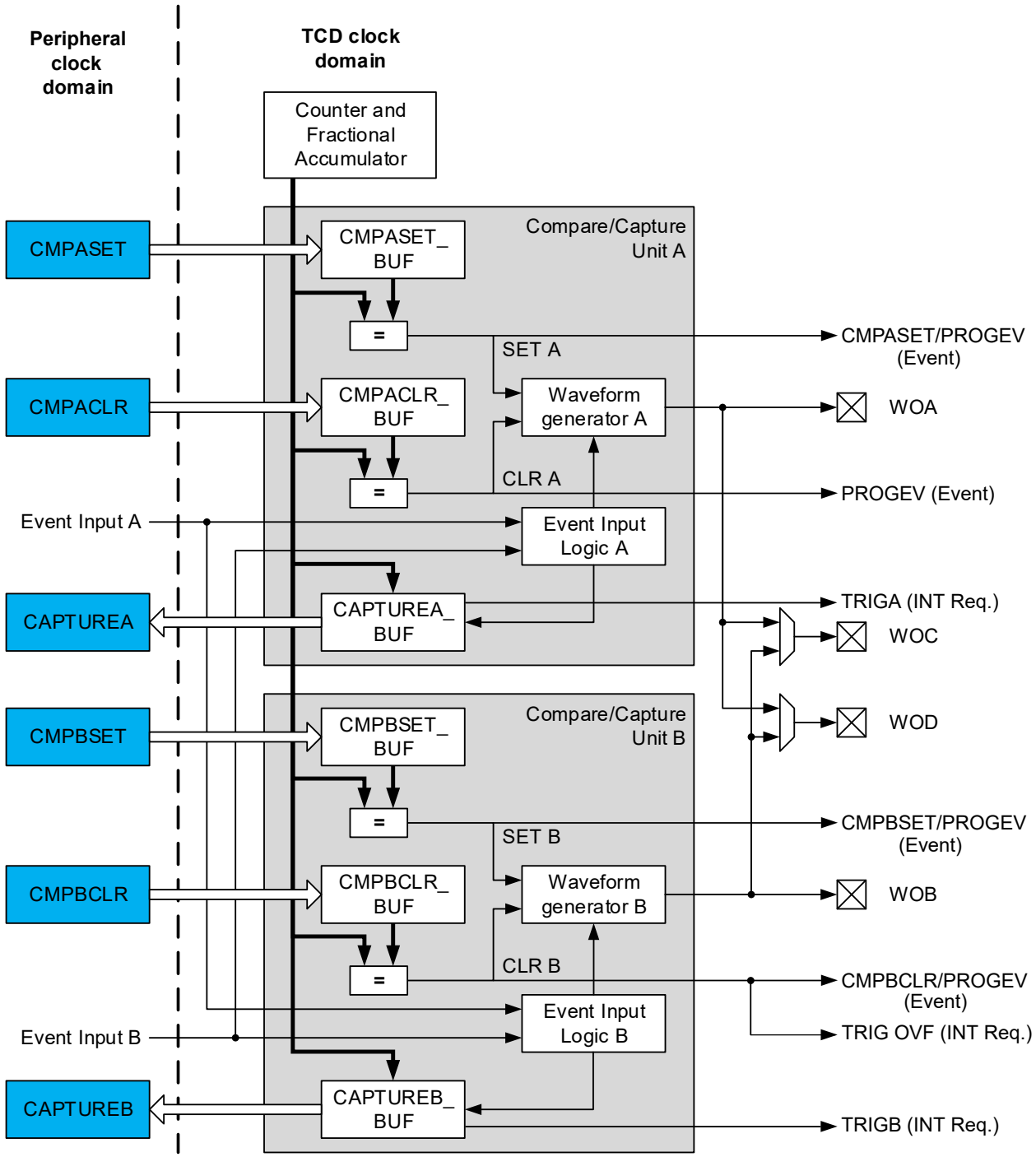
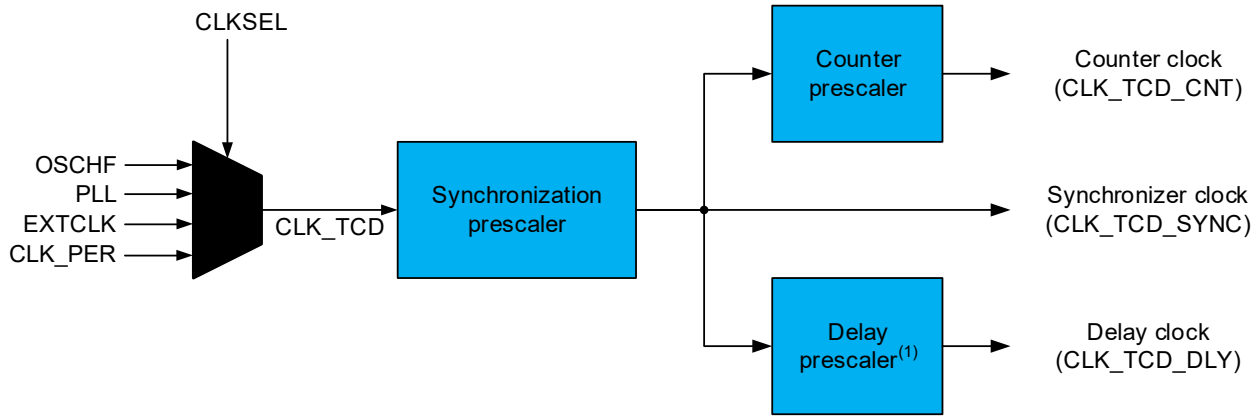


Figure 2-2. Clock Selection and Prescalers Overview



1. Used by input blanking/delay event out.

3. Generating Two PWM Signals in One Ramp Mode

Use case description: The TCD will be initialized and configured to run in One Ramp mode, have the OSCHF as input clock running at 24 MHz, and generate two PWM signals – WOA on the PA4 pin, and WOB on the PA5 pin. The PWM signal on WOA will have a 25% duty cycle and the signal on WOB will have a 30% duty cycle.

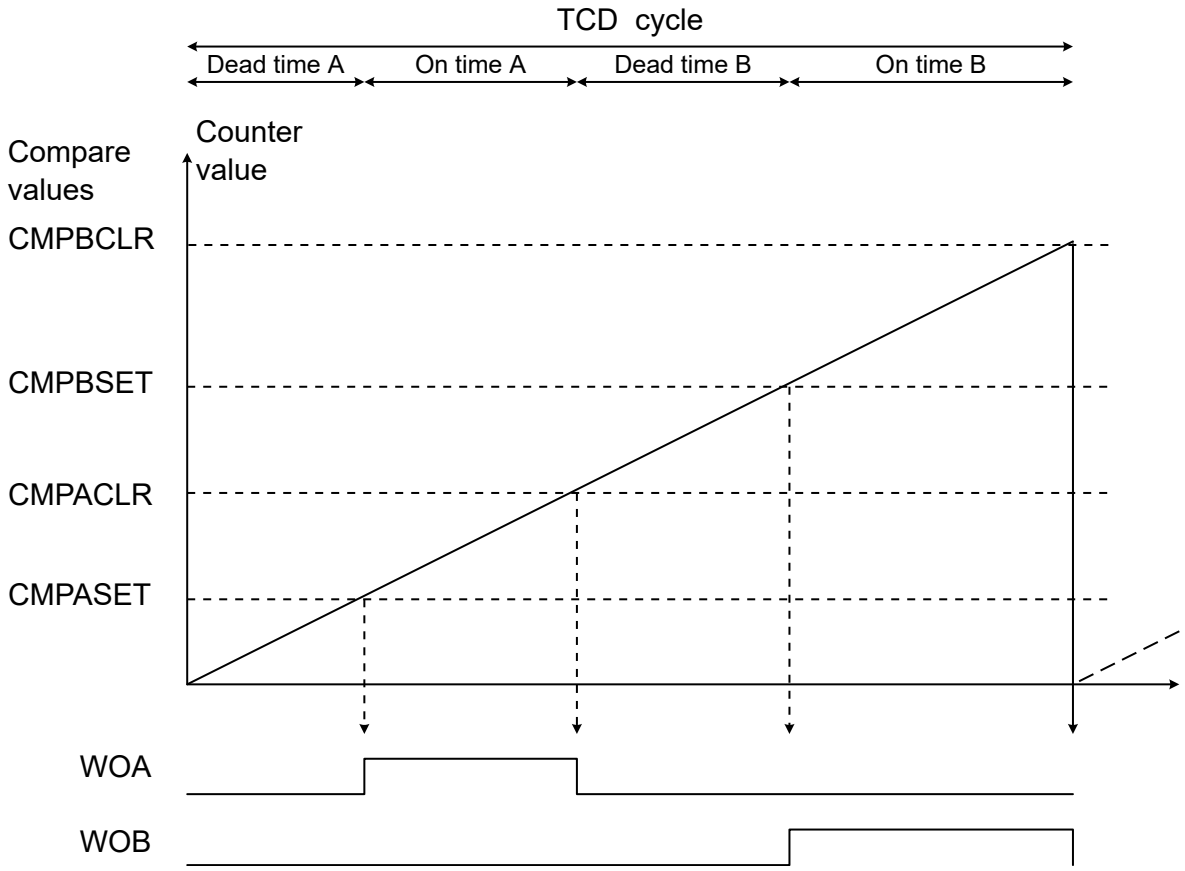
Result: The TCD will generate one PWM signal with a 25% duty cycle on WOA (PA4 pin) and one PWM signal with a 30% duty cycle on WOB.

The TCD can be configured to run in One Ramp mode, where the counter's value increments until it reaches the CMPBCLR value. Then, the TCD cycle is completed and the counter restarts from `0x000`, beginning a new TCD cycle. The TCD cycle period is:

$$T_{TCD\_cycle} = \frac{(CMPBCLR + 1)}{f_{CLK\_TCD\_CNT}}$$

In this configuration example, nonoverlapping outputs will be generated, so the case where CMPASET < CMPACLR < CMPBSET < CMPBCLR will be used.

Figure 3-1. One Ramp Mode



Configuring the Main Clock

There are four clock sources for the TCD:

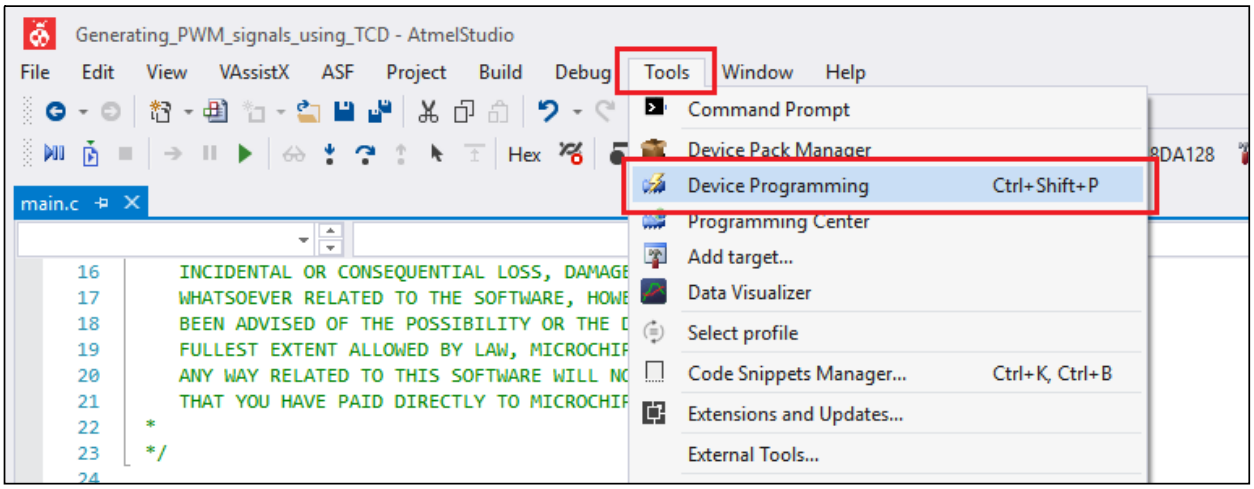
- OSCHF
- PLL
- EXTCLK
- CLK_PER

In this example of the TCD configuration, the OSCHF clock source will be the input to CLK_MAIN and the TCD. The following configurations must be made to have the CPU and the TCD run at 24 MHz, having OSCHF as input clock.

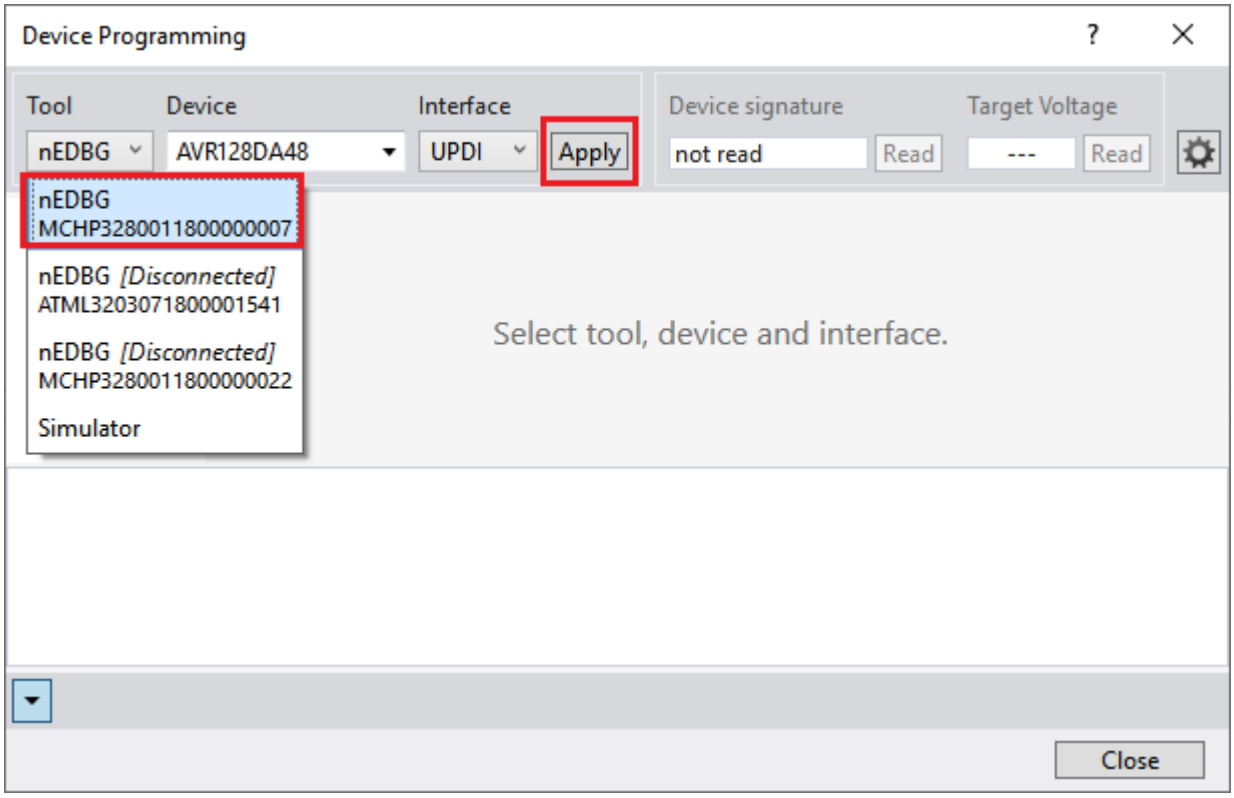
In the example code available in 6. Appendix, the main clock initialization will be done in the `CLK_Init()` function.

Setting the Default Clock Source to OSCHF (Optional)
This can be done by using Atmel Studio and following the steps below:

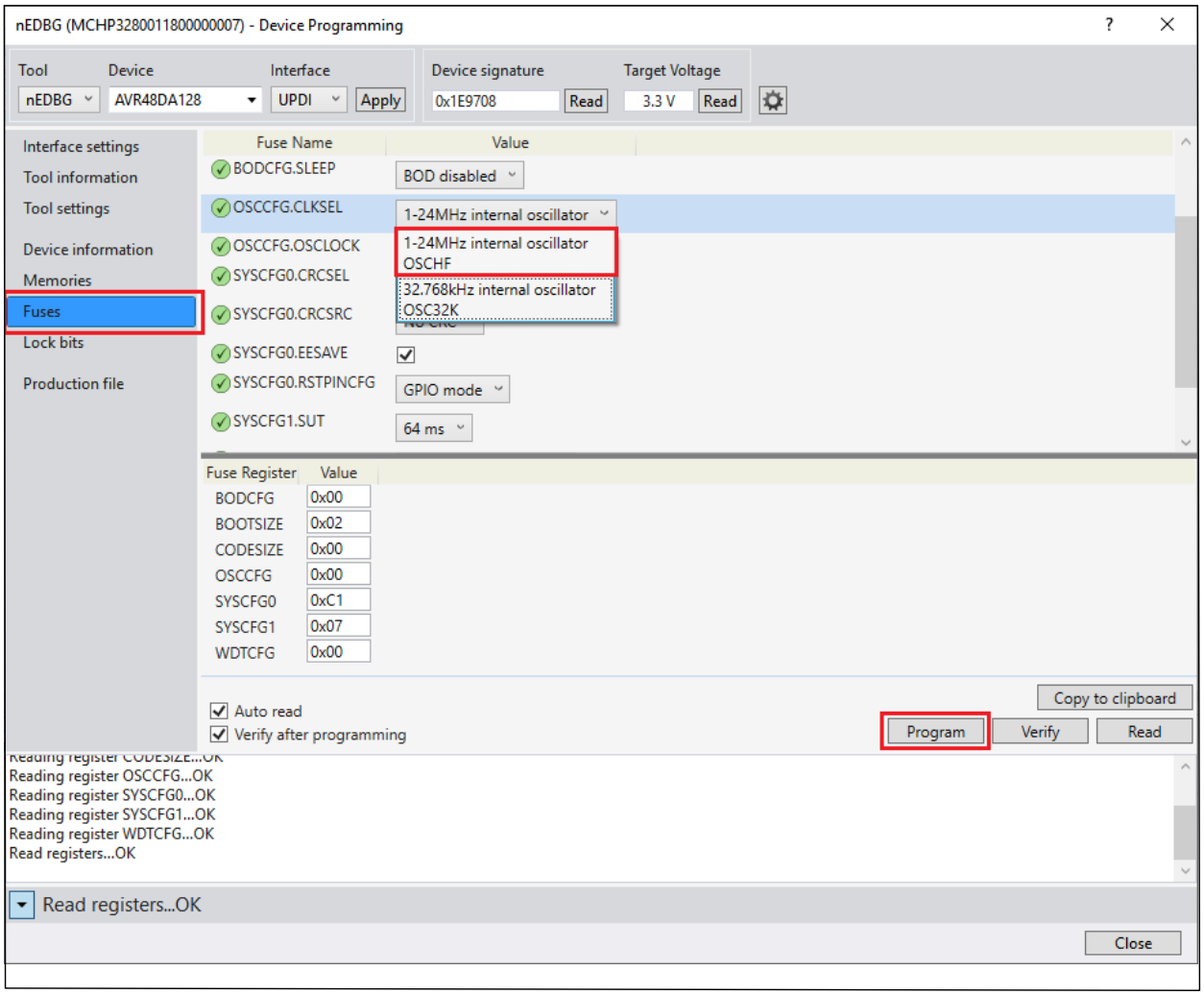
a. Click **Tools** → **Device Programming**.



b. Select the attached development board and click **Apply**.



c. Select the **Fuses** tab from the left-hand side and select **1-24 MHz internal oscillator OSCHF** for **OSCCFG.CLKSEL**, and then click **Program**.



Changing the clock source to OSCHF and the configuration for running at 24 MHz are described in the following three steps:

1. Set OSCHF as clock source for the main clock.

Figure 3-2. CLKCTRL.MCLKCTRLA Register Configuration

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

Bits 3:0 – CLKSEL[3:0] Clock Select
This bit field selects the source for the Main Clock (CLK_MAIN).

Value	Name	Description
0x0	OSCHF	Internal High-Frequency Oscillator
0x1	OSC32K	32.768 kHz Internal Oscillator
0x2	XOSC32K	32.768 kHz External Crystal Oscillator
0x3	EXTCLK	External clock
Other	-	Reserved

The Main Clock Control A register is protected by the Configuration Change Protection (CCP) mechanism, requiring a timed-write procedure for changing the register content. To write to the CCP-protected registers, the following API must be used:

```
_PROTECTED_WRITE(register, value);
```

OSCHF must be selected, which means that the CLKSEL bit field must be set to value `0x0`. This translates into the following code:

```
_PROTECTED_WRITE(CLKCTRL.MCLKCTRLA, CLKCTRL_CLKSEL_OSCHF_gc);
```

2. Wait for the clock switch process to complete.

Figure 3-3. CLKCTRL.MCLKSTATUS Register

Bit	7	6	5	4	3	2	1	0
			PLLS	EXTS	XOSC32KS	OSC32KS	OSCHFS	SOSC
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 0 – SOSC Main Clock Oscillator Changing

Value	Description
0	The clock source for CLK_MAIN is not undergoing a switch
1	The clock source for CLK_MAIN is undergoing a switch and will change as soon as the new source is stable

The clock switching process is indicated by the SOSC bit. The program must halt during an undergoing switch of the clock source, so a wait until the switch is over will be implemented.

```
while (CLKCTRL.MCLKSTATUS & CLKCTRL_SOSC_bm)
    {
        ;
    }
```

3. Set the OSCHF to run at 24 MHz.

Figure 3-4. CLKCTRL.OSCHFCTRLA Register Configuration

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

Bits 5:2 – FRQSEL[3:0] Frequency Select
This bit field selects the output frequency of the oscillator.

Value	Name	Description
0x0	1 MHz	1 MHz output
0x1	2 MHz	2 MHz output
0x2	3 MHz	3 MHz output
0x3	4 MHz	4 MHz output (default)
0x4	-	Reserved
0x5	8 MHz	8 MHz output
0x6	12 MHz	12 MHz output
0x7	16 MHz	16 MHz output
0x8	20 MHz	20 MHz output
0x9	24 MHz	24 MHz output
Other	-	Reserved

The default Reset value of the FRQSEL bit field in the OSCHFCTRLA register is `0x3`, which means that the default frequency value is 4 MHz. To obtain the 24 MHz desired frequency, the content of the FRQSEL bit field must be changed to `0x9`. This bit field uses the CCP mechanism, so a protected write must be performed. The following code will select 24 MHz output for OSCHF:

```
_PROTECTED_WRITE(CLKCTRL.OSCHFCTRLA, CLKCTRL_FREQSEL_24M_gc);
```

Configuring PA4 and PA5 Pins as Output

The PA4 and PA5 pins must be configured as output pins for the WOA and WOB PWM signals. The following code snippet sets the PA4 and PA5 pins as output low.

```
PORTA.DIRSET |= PIN4_bm | PIN5_bm;
PORTA.OUTSET |= PIN4_bm | PIN5_bm;
```

In the example code available in 6. Appendix, the pins initialization will be done in the `PORT_Init()` function.

Configuring the TCD Input Clock and Operation Mode

To generate the two PWM signals using the TCD configured in One Ramp mode and with OSCHF as input clock, the following registers must be changed:

- TCD0.CTRLA
- TCD0.CTRLB
- TCD0.CMPASET
- TCD0.CMPACLR
- TCD0.CMPBSET
- TCD0.CMPBCLR

In the example code available in 6. Appendix, the TCD initialization will be done in the `TCD_Init()` function.

1. Select the Waveform Generation mode and configure the TCD.

Figure 3-5. TCD0.CTRLB Register Configuration

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

Bits 1:0 – WGMODE[1:0] Waveform Generation Mode
These bits select the waveform generation.

Value	Name	Description
0x0	ONERAMP	One Ramp mode
0x1	TWORAMP	Two Ramp mode
0x2	FOURRAMP	Four Ramp mode
0x3	DS	Dual Slope mode

To use TCD0 in One Ramp mode, the WGMODE bit field in the TCD0.CTRLB register must be set to `0x0`. The following code snippet configures TCD0 in One Ramp mode:

```
TCD0.CTRLB |= TCD_WGMODE_ONERAMP_gc;
```

In One Ramp mode, the TCD0.CMPASET and TCD0.CMPACLR registers are used for setting the ‘Dead time A’ and ‘On time A’ for the WOA signal; in addition, the TCD0.CMPBSET and TCD0.CMPBCLR registers are used for setting ‘Dead time B’ and ‘On time B’ for the WOB signal.

As the TCD0 counter continuously increases and overflows, setting CMPASET < CMPACLR < CMPBSET < CMPBCLR will result in nonoverlapping outputs during the on time.

Since the TCD is a 12-bit timer/counter, it ranges from 0 to 4095 (4096 steps), corresponding to `0x000` to `0xFFF`. For a 25% PWM duty cycle, WOA must have an on time of 1024 clock cycles (defined below by the `ON_TIME_CYCLES_WOA` macro), corresponding to `0x400` in hexadecimal format:

$$ON\_TIME\_CYCLES\_WOA = \frac{25}{100} \times 4096 = 1024 = 0x400$$

This means that the difference between the value of TCD0.CMPACLR and TCD0.CMPASET must be 1024. For the purpose of this use case exemplification, the start of the on time for WOA will be 1023 (defined below by the `ON_TIME_START_WOA` macro).

$$TCD0.CMPASET = ON\_TIME\_START\_WOA = 1023 = 0x3FF$$

$$TCD0.CMPACLR = ON\_TIME\_START\_WOA + ON\_TIME\_CYCLES\_WOA = 1023 + 1024 = 2047 = 0x7FF$$

For a 30% PWM duty cycle, WOB must have an on time of 1228 clock cycles (defined below by the `ON_TIME_CYCLES_WOB` macro), corresponding to `0x508` in hexadecimal format:

$$ON\_TIME\_CYCLES\_WOB = \frac{30}{100} \times 4096 = \sim 1288 = 0x508$$

This means the difference between the value of TCD0.CMPBCLR and TCD0.CMPBSET must be 1288. For the purpose of this use case exemplification, the start of the on time for WOB will be 2457 (defined below by the `ON_TIME_START_WOB` macro).

$$TCD0.CMPBSET = ON\_TIME\_START\_WOB = 2457 = 0x999$$

$$TCD0.CMPBCLR = ON\_TIME\_START\_WOB + ON\_TIME\_CYCLES\_WOB = 2457 + 1288 = 3745 = 0xEA1$$

The following code snippet initializes TCD0.CMPASET, TCD0.CMPACLR, TCD0.CMPBSET, and TCD0.CMPBCLR with the corresponding values for generating the PWM signals with 25% and 30% duty cycles.

```
#define ON_TIME_START_WOA   0x3FF
#define ON_TIME_CYCLES_WOA  0x400
#define ON_TIME_START_WOB   0x999
#define ON_TIME_CYCLES_WOB  0x508

TCD0.CMPASET = ON_TIME_START_WOA;
TCD0.CMPACLR = ON_TIME_START_WOA + ON_TIME_CYCLES_WOA;
TCD0.CMPBSET = ON_TIME_START_WOB;
TCD0.CMPBCLR = ON_TIME_START_WOB + ON_TIME_CYCLES_WOB;
```

2. Enable the waveform channels as output.

Figure 3-6. TCD0.FAULTCTRL Register Configuration

Bit	7	6	5	4	3	2	1	0
	CMPDEN	CMPCEN	CMPBEN	CMPAEN	CMPD	CMPC	CMPB	CMPA
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 4, 5, 6, 7 – CMPEN Compare x Enable
These bits enable the waveform from compare as output on the pin.

Bits 0, 1, 2, 3 – CMP Compare x Value
These bits set the default state of the compare waveform output.

For generating the PWM signals, the two output channels, WOA and WOB, must be enabled. Additionally, to exemplify this use case, the Default state of the two waveform outputs will be high. Since the TCD0.FAULTCTRL register is under Configuration Change Protection, the CMPAEN, CMPBEN, CMPB and CMPA bits must be written using the `_PROTECTED_WRITE` API.

The following code snippet enables the output channels and sets the waveform output to high:

```
_PROTECTED_WRITE(TCD0.FAULTCTRL, TCD_CMPAEN_bm | TCD_CMPA_bm
                                 | TCD_CMPBEN_bm | TCD_CMPB_bm);
```

3. Check if the TCD is ready for enabling.

Figure 3-7. TCD0.STATUS Register

Bit	7	6	5	4	3	2	1	0
	PWMACTB	PWMACTA					CMDRDY	ENRDY
Access	R/W	R/W					R	R
Reset	0	0					0	0

Bit 0 – ENRDY Enable Ready
This status bit tells when the ENABLE value in TCDn.CTRLA is synced to the TCD domain and is ready to be written to again.
The following actions clear the ENRDY bit:

1. Writing to the ENABLE bit in TCDn.CTRLA.
2. TCDn.CTRLE DISEOC strobe.
3. Going into BREAK in an On-Chip Debugging (OCD) session while the Debug Run (DBGCTRL) bit in TCDn.DBGCTRL is '0'.

To enable the TCD, first, it must be checked if it is ready. The following code snippet implements a wait until the TCD is ready to be enabled:

```
while (!(TCD0.STATUS & TCD_ENRDY_bm))
    {
        ;
    }
```

4. Select the input clock source and enable the TCD.

Figure 3-8. TCD0.CTRLA Register Configuration

Bit	7	6	5	4	3	2	1	0
		CLKSEL[1:0]		CNTPRES[1:0]		SYNCPRES[1:0]		ENABLE
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 6:5 – CLKSEL[1:0] Clock Select
The Clock Select bits select the clock source of the TCD clock.

Value	Name	Description
0x0	OSCHF	Internal High-Frequency Oscillator
0x1	PLL	PLL
0x2	EXTCLK	External clock
0x3	CLK_PER	Peripheral clock

Bit 0 – ENABLE Enable
When writing to this bit, it will automatically be synchronized to the TCD clock domain.
This bit can be changed as long as the synchronization of this bit is not ongoing. See the Enable Ready (ENRDY) bit in the Status (TCDn.STATUS) register.
This bit is not enable-protected.

Value	Name	Description
0	NO	The TCD is disabled.
1	YES	The TCD is enabled and running.

The following code snippet will select OSCHF for the input frequency and will enable the TCD:

```
TCD0.CTRLA |= TCD_CLKSEL_OSCHF_gc | TCD_ENABLE_bm;
```

View Code Example on GitHub
Click to browse repository

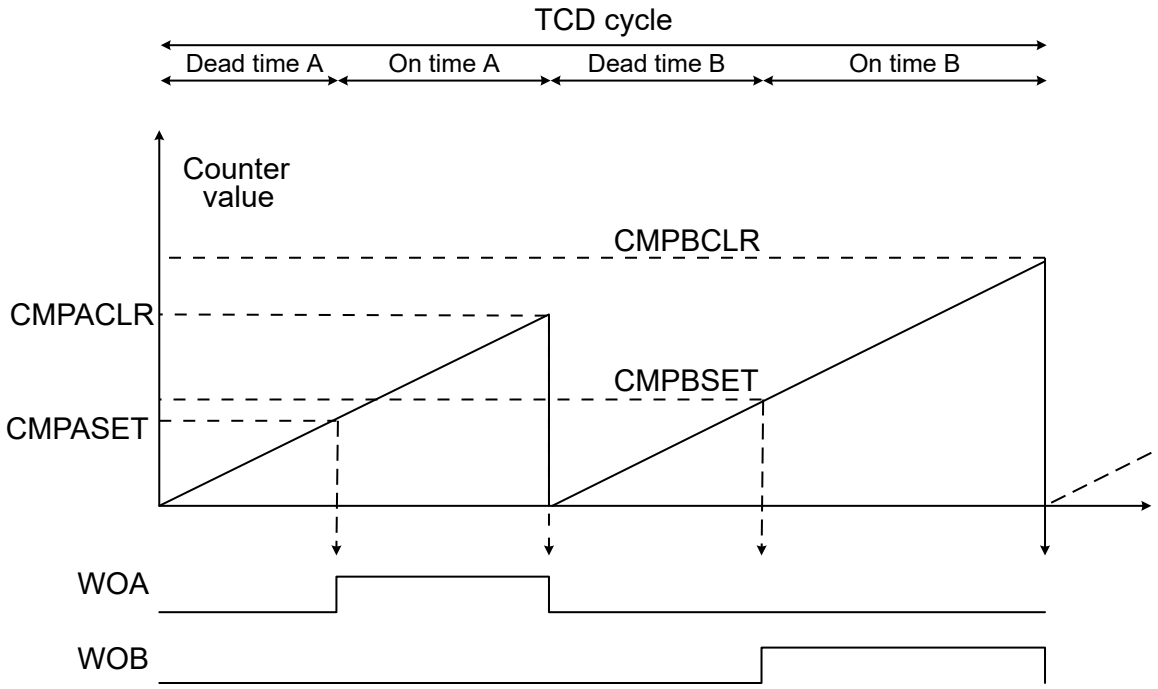
Tip: The full code example is also available in 6. Appendix.

4. Generating Two PWM Signals in Two Ramp Mode

Use case description: The TCD will be initialized and configured to run in Two Ramp mode, to have the PLL as input clock running at 48 MHz, and to generate two PWM signals – WOA on the PA4 pin and WOB on the PA5 pin. The PWM signal on WOA will have a 10% duty cycle and the signal on WOB (PA5 pin) will have a 20% duty cycle.

Result: The TCD will generate one PWM signal with a 10% duty cycle on WOA (the PA4 pin) and one PWM signal with a 20% duty cycle on WOB.

Figure 4-1. Two Ramp Mode



In Two Ramp mode, the TCD counts up until it reaches the CMPACLR value, then it resets and counts up until it reaches the CMPBCLR value. Then, the TCD cycle is completed, and the counter restarts from `0x000`.

$$T_{TCD\_cycle} = \frac{(CMPACLR + 1 + CMPBCLR + 1)}{f_{CLK\_TCD\_CNT}}$$

Configuring the Main Clock

To obtain the maximum input frequency for the TCD from the PLL, the OSCHF will be configured to run at the highest frequency (24 MHz). The maximum frequency achievable by the PLL is 48 MHz, so a multiplication factor of 2x will be used for the PLL.

Furthermore, the OSCHF will also serve as clock source for `CLK_MAIN`.

In the example code available in 6. Appendix, the main clock initialization will be done in the `CLK_Init()` function.

1. Set OSCHF as clock source for the main clock. The following code snippet will switch the main clock to the OSCHF oscillator.

```
_PROTECTED_WRITE(CLKCTRL.MCLKCTRLA, CLKCTRL_CLKSEL_OSCHF_gc);
```

2. Wait for the clock switch process to complete. The following code snippet will demonstrate how to wait for the clock source switching process to finish.

```
while (CLKCTRL.MCLKSTATUS & CLKCTRL_SOSC_bm)
    {
        ;
    }
```

3. Set the OSCHF to run at 24 MHz. The following code snippet will set the OSCHF frequency to 24 MHz.

```
_PROTECTED_WRITE(CLKCTRL.OSCHFCTRLA, CLKCTRL_FREQSEL_24M_gc);
```

4. Configure the PLL settings – a multiplication factor of 2x will be chosen as the maximum PLL frequency is 48 MHz. The PLL system will not yet be active, but will start and generate when requested as a clock source by the TCD.

```
_PROTECTED_WRITE(CLKCTRL.PLLCTRLA, CLKCTRL_MULFAC_2x_gc);
```

Configuring PA4 and PA5 Pins as Output

The PA4 and PA5 pins must be configured as output pins for the WOA and WOB PWM signals. The following code snippet sets PA4 and PA5 pins as output low.

```
PORTA.DIRSET |= PIN4_bm | PIN5_bm;
PORTA.OUTSET |= PIN4_bm | PIN5_bm;
```

Note: In the example code available in 6. Appendix, the pins initialization will be done in the `PORT_Init()` function.

Configuring the TCD Input Clock and Operation Mode

To Generate the two PWM signals using the TCD configured in One Ramp mode and with OSCHF as input clock, the following registers must be changed:

- TCD0.CTRLA
- TCD0.CTRLB
- TCD0.CMPASET
- TCD0.CMPACLR
- TCD0.CMPBSET
- TCD0.CMPBCLR

In the example code available in 6. Appendix, the TCD initialization will be done in the `TCD_Init()` function.

1. Select the Waveform Generation Mode and configure the TCD.

Figure 4-2. TCD0.CTRLB Register Configuration

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

Bits 1:0 – WGMODE[1:0] Waveform Generation Mode
These bits select the waveform generation.

Value	Name	Description
0x0	ONERAMP	One Ramp mode
0x1	TWORAMP	Two Ramp mode
0x2	FOURRAMP	Four Ramp mode
0x3	DS	Dual Slope mode

To use TCD0 in One Ramp mode, the WGMODE bit field in the TCD0.CTRLB register must be set to `0x0`. The following code snippet configures TCD0 in Two Ramp mode:

```
TCD0.CTRLB |= TCD_WGMODE_TWORAMP_gc;
```

Since the TCD is a 12-bit timer/counter, it ranges from 0 to 4095 (4096 steps), corresponding to `0x000` to `0xFFF`. For a 10% duty cycle, WOA must have an on time of 409 clock cycles (defined below by the 'ON_TIME_CYCLES_WOA' macro), corresponding to `0x199` in hexadecimal format:

$$ON\_TIME\_CYCLES\_WOA = \frac{10}{100} \times 4096 = 409 = 0x199$$

This means that the difference between the value of TCD0.CMPACLR and TCD0.CMPASET must be 409. In this use case example, the start of the on time for WOA will be 1023 (defined below by the `ON_TIME_START_WOA` macro).

$$TCD0.CMPASET = ON\_TIME\_START\_WOA = 1023 = 0x3FF$$
$$TCD0.CMPACLR = ON\_TIME\_START\_WOA + ON\_TIME\_CYCLES\_WOA = 1023 + 409 = 1423 = 0x598$$

For a 20% duty cycle, WOB must have an on time of 819 clock cycles (defined below by the `ON_TIME_CYCLES_WOB` macro), corresponding to `0x333` in hexadecimal format:

$$ON\_TIME\_CYCLES\_WOB = \frac{20}{100} \times 4096 = 819 = 0x333$$

This means that the difference between the value of TCD0.CMPBCLR and TCD0.CMPBSET must be 819. In this use case example, the start of the on time for WOB will be 1023 (defined below by the `ON_TIME_START_WOB` macro).

$$TCD0.CMPBSET = ON\_TIME\_START\_WOB = 1023 = 0x3FF$$
$$TCD0.CMPBCLR = ON\_TIME\_START\_WOB + ON\_TIME\_CYCLES\_WOB = 1023 + 819 = 1842 = 0x732$$

The following code snippet initializes TCD0.CMPASET, TCD0.CMPACLR, TCD0.CMPBSET, and TCD0.CMPBCLR with the corresponding values for generating the PWM signals with 10% and 20% duty cycles.

```
#define ON_TIME_START_WOA   0x3FF
#define ON_TIME_CYCLES_WOA  0x199
#define ON_TIME_START_WOB   0x3FF
#define ON_TIME_CYCLES_WOB  0x333

TCD0.CMPASET = ON_TIME_START_WOA;
TCD0.CMPACLR = ON_TIME_START_WOA + ON_TIME_CYCLES_WOA;
TCD0.CMPBSET = ON_TIME_START_WOB;
TCD0.CMPBCLR = ON_TIME_START_WOB + ON_TIME_CYCLES_WOB;
```

2. Enable the waveform channels as output.
   The following code snippet enables the output channels and sets the waveform output to high:

```
_PROTECTED_WRITE(TCD0.FAULTCTRL, TCD_CMPAEN_bm | TCD_CMPA_bm
                                 | TCD_CMPBEN_bm | TCD_CMPB_bm);
```

3. Check if the TCD is ready for enabling.
   The following code snippet implements a wait until the TCD is ready to be enabled.

```
while (!(TCD0.STATUS & TCD_ENRDY_bm))
    {
        ;
    }
```

4. Select the input clock source and enable the TCD.

Figure 4-3. TCD0.CTRLA Register Configuration

Bit	7	6	5	4	3	2	1	0
		CLKSEL[1:0]		CNTPRES[1:0]		SYNCPRES[1:0]		ENABLE
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 6:5 – CLKSEL[1:0] Clock Select
The Clock Select bits select the clock source of the TCD clock.

Value	Name	Description
0x0	OSCHF	Internal High-Frequency Oscillator
0x1	PLL	PLL
0x2	EXTCLK	External clock
0x3	CLK_PER	Peripheral clock

Bit 0 – ENABLE Enable
When writing to this bit, it will automatically be synchronized to the TCD clock domain.
This bit can be changed as long as the synchronization of this bit is not ongoing. See the Enable Ready (ENRDY) bit in the Status (TCDn.STATUS) register.
This bit is not enable-protected.

Value	Name	Description
0	NO	The TCD is disabled.
1	YES	The TCD is enabled and running.

The following code snippet will select OSCHF for the input frequency and will enable the TCD:

```
TCD0.CTRLA |= TCD_CLKSEL_PLL_gc | TCD_ENABLE_bm;
```

View Code Example on GitHub
Click to browse repository

Tip: The full code example is also available in 6. Appendix.

5. References

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

6. Appendix

Example 6-1. TCD One Ramp Mode with OSCHF Code Example

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

#define ON_TIME_START_WOA   0x3FF
#define ON_TIME_CYCLES_WOA  0x400
#define ON_TIME_START_WOB   0x999
#define ON_TIME_CYCLES_WOB  0x508

void CLK_Init(void);
void PORT_Init(void);
void TCD_Init(void);

void CLK_Init(void)
{
    /* Set OSCHF as main clock source */
    _PROTECTED_WRITE(CLKCTRL.MCLKCTRLA, CLKCTRL_CLKSEL_OSCHF_gc);

    /* Wait for main clock oscillator changing to finish */
    while (CLKCTRL.MCLKSTATUS & CLKCTRL_SOSC_bm)
    {
        ;
    }

    /* Change OSCHF frequency to 24 MHz */
    _PROTECTED_WRITE(CLKCTRL.OSCHFCTRLA, CLKCTRL_FREQSEL_24M_gc);
}

void PORT_Init(void)
{
    /* Configure PORT A PIN4 and PIN5 as output low */
    PORTA.DIRSET |= PIN4_bm | PIN5_bm;
    PORTA.OUTSET |= PIN4_bm | PIN5_bm;
}

void TCD_Init(void)
{
    /* Select the One Ramp mode */
    TCD0.CTRLB |= TCD_WGMODE_ONERAMP_gc;

    /* Load the compare and clear registers */
    TCD0.CMPASET = ON_TIME_START_WOA;
    TCD0.CMPACLR = ON_TIME_START_WOA + ON_TIME_CYCLES_WOA;

    TCD0.CMPBSET = ON_TIME_START_WOB;
    TCD0.CMPBCLR = ON_TIME_START_WOB + ON_TIME_CYCLES_WOB;

    /* Enable the PWM channels */
    _PROTECTED_WRITE(TCD0.FAULTCTRL, TCD_CMPAEN_bm | TCD_CMPA_bm);
    _PROTECTED_WRITE(TCD0.FAULTCTRL, TCD_CMPBEN_bm | TCD_CMPB_bm);

    /* Wait for TCD to be ready for enabling */
    while (!(TCD0.STATUS & TCD_ENRDY_bm))
    {
        ;
    }
```

```
    /* Select OSCHF as input clock and enable TCD */
    TCD0.CTRLA |= TCD_CLKSEL_OSCHF_gc | TCD_ENABLE_bm;
}

int main(void)
{
    CLK_Init();
    PORT_Init();
    TCD_Init();

    while (1)
    {
        ;
    }
}
```

Example 6-2. TCD Two Ramp Mode with PLL Code Example

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

#define ON_TIME_START_WOA   0x3FF
#define ON_TIME_CYCLES_WOA  0x199
#define ON_TIME_START_WOB   0x3FF
#define ON_TIME_CYCLES_WOB  0x333

void CLK_Init(void);
void PORT_Init(void);
void TCD_Init(void);

void CLK_Init(void)
{
    /* Set OSCHF as main clock source */
    _PROTECTED_WRITE(CLKCTRL.MCLKCTRLA, CLKCTRL_CLKSEL_OSCHF_gc);

    /* Wait for main clock oscillator changing to finish */
    while (CLKCTRL.MCLKSTATUS & CLKCTRL_SOSC_bm)
    {
        ;
    }

    /* Change OSCHF frequency to 24 MHz */
    _PROTECTED_WRITE(CLKCTRL.OSCHFCTRLA, CLKCTRL_FREQSEL_24M_gc);

    /* Set the multiplication factor for PLL to 2x */
    _PROTECTED_WRITE(CLKCTRL.PLLCTRLA, CLKCTRL_MULFAC_2x_gc);
}

void PORT_Init(void)
{
    /* Configure PORT A PIN4 and PIN5 as output low */
    PORTA.DIRSET |= PIN4_bm | PIN5_bm;
    PORTA.OUTSET |= PIN4_bm | PIN5_bm;
}

void TCD_Init(void)
{
    /* Select the Two Ramp mode */
    TCD0.CTRLB |= TCD_WGMODE_TWORAMP_gc;

    /* Load the compare and clear registers */
    TCD0.CMPASET = ON_TIME_START_WOA;
    TCD0.CMPACLR = ON_TIME_START_WOA + ON_TIME_CYCLES_WOA;
    TCD0.CMPBSET = ON_TIME_START_WOB;
    TCD0.CMPBCLR = ON_TIME_START_WOB + ON_TIME_CYCLES_WOB;

    /* Enable the PWM channels */
    _PROTECTED_WRITE(TCD0.FAULTCTRL, TCD_CMPAEN_bm | TCD_CMPA_bm);
    _PROTECTED_WRITE(TCD0.FAULTCTRL, TCD_CMPBEN_bm | TCD_CMPB_bm);

    /* Wait for TCD to be ready for enabling */
    while (!(TCD0.STATUS & TCD_ENRDY_bm))
    {
        ;
    }

    /* Select PLL as input clock and enable TCD */
    TCD0.CTRLA |= TCD_CLKSEL_PLL_gc | TCD_ENABLE_bm;
}

int main(void)
```

```
{
    CLK_Init();
    PORT_Init();
    TCD_Init();

    while (1)
    {
        ;
    }
}
```

7. Revision History

Doc. Rev.	Date	Comments
C	05/2020	Fixed code snippets; Updated AVR® MCU DA (AVR-DA) to AVR® DA MCU, and AVR-DA to AVR DA, per latest trademarking
B	03/2020	Updated repository links. Updated AVR-DA to AVR® MCU DA (AVR-DA), per latest trademarking.
A	02/2020	Initial document release

The Microchip Website

Microchip provides online support via our website at http://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 http://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: http://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

your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PackeTime, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TempTrackr, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, FlashTec, Hyper Speed Control, HyperLight Load, IntelliMOS, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, Vite, WinPath, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BlueSky, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, INICnet, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

ISBN: 978-1-5224-6075-6

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit http://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: http://www.microchip.com/support Web Address: http://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: 45-4450-2828 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