



MICROCHIP

Regional Training Centers

Section 6

Timer/Counter, TC Architecture

Timer or Counter?

- ❖ **Counter or Timer is a clock counter, use to count how many clock into module after start.**
- ❖ **There are two kinds of generally called Timer or Counter. In fact, it's same.**
 - 📖 The Input Clock unknown : Just know how many count, called Counter.
 - 📖 The Input Clock known : Can use count and clock period to calculate time, called Timer.
- ❖ **Timer can set a notify condition, like overflow, equal some value etc.**
 - 📖 If interrupt disable : user must check flag, as soon as possible.
 - 📖 If interrupt enable : Timer will notify CPU, if condition is true.

Timer or Counter?

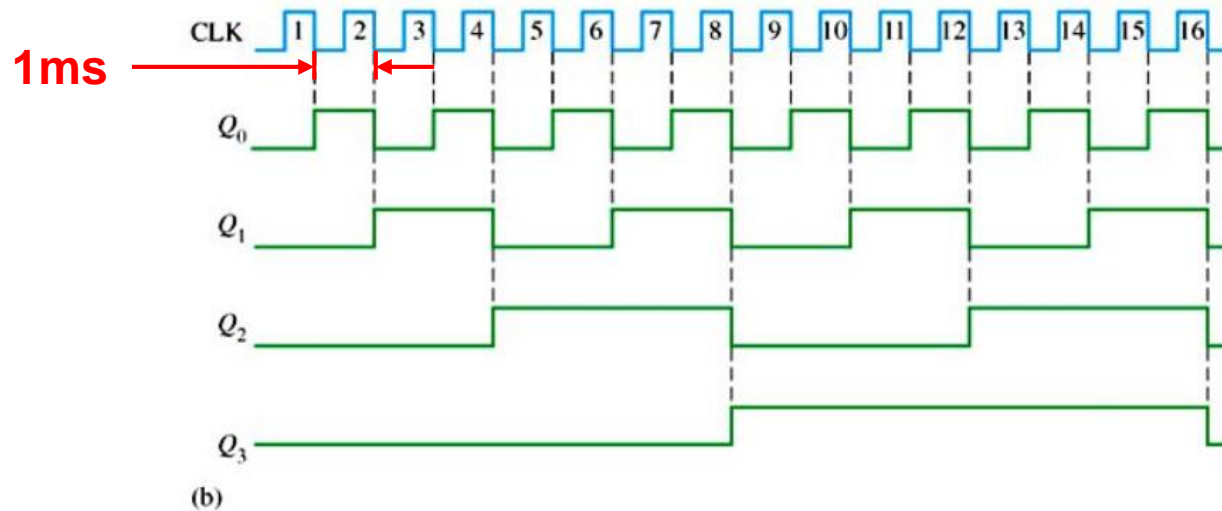
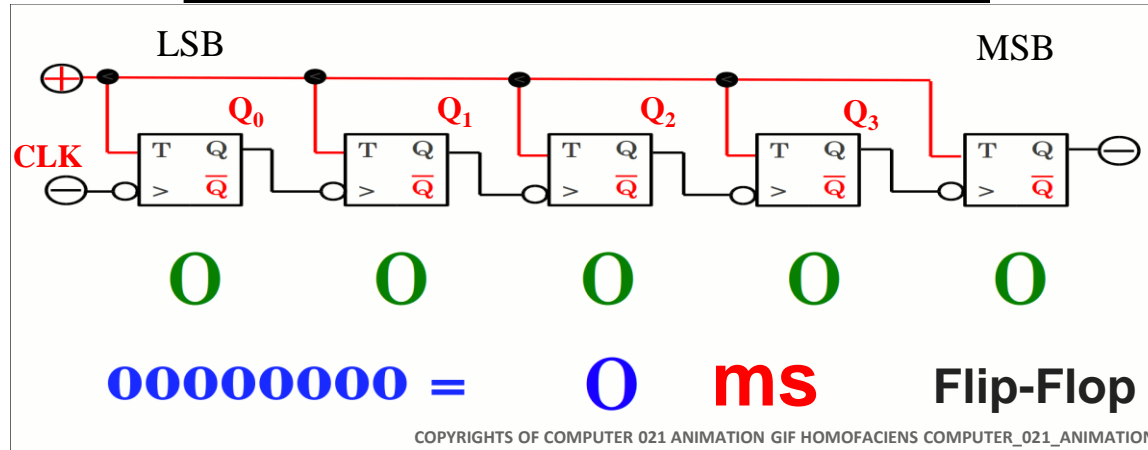


Figure 9--5 Four-bit asynchronous binary counter and its timing diagram.

6

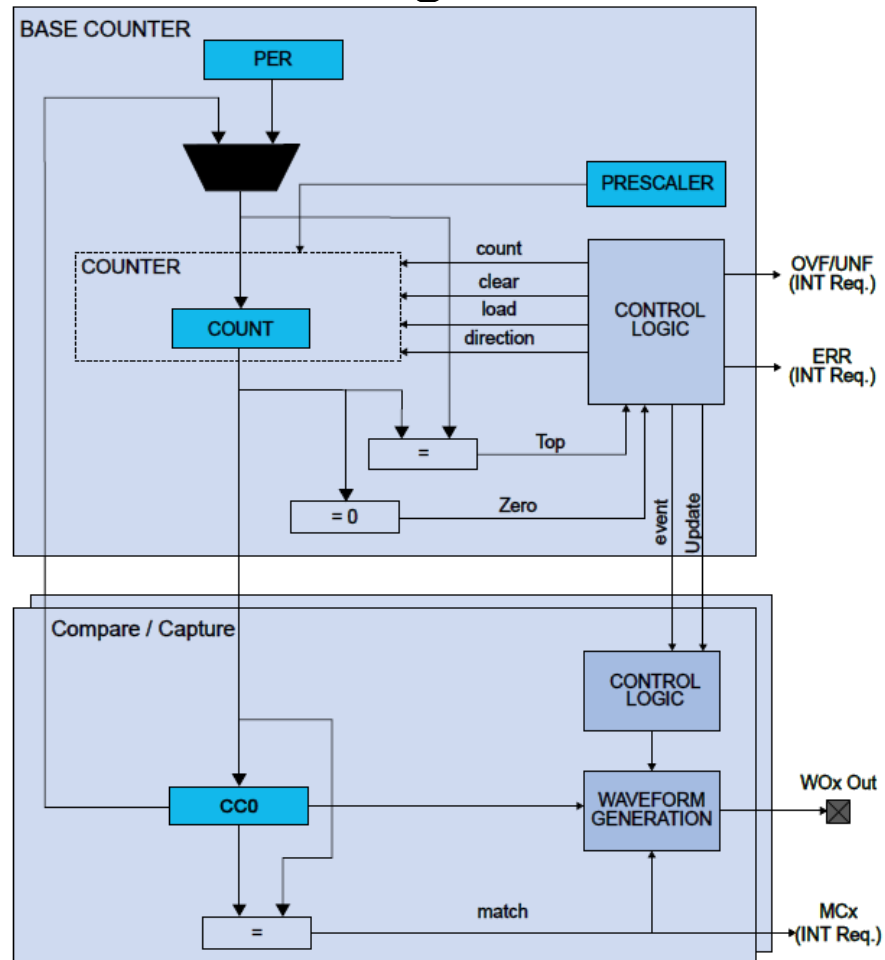
Copyright ©2003 by Pearson Education, Inc.
Upper Saddle River, New Jersey 07458
All rights reserved.

SAMD21 Timer/Counter

- The SAMD21 Timer/Counter, TC module consists 3 function Counter, Compare and Capture.
- The counter can be set to count events, or it can be configured to count clock pulses. The counter, together with the compare/capture channels, can be configured to timestamp input events, allowing capture of frequency and pulse width.
- It can also perform waveform generation, such as frequency generation and pulse-width modulation (PWM).

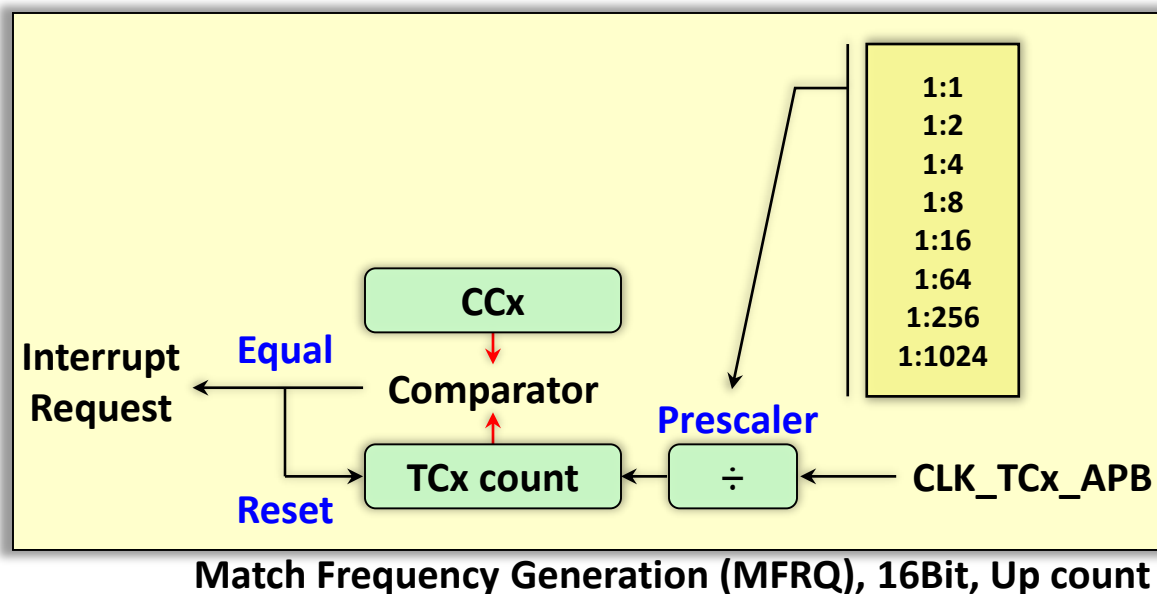
TC Block Diagram

■ SAMD21G18A TC Block Diagram



SAMD21 Timer/Counter

- ❖ Please refer to the block diagram, Just a concept. The real and detail block diagram please refer to the following slide.
- ❖ Clock into TCx after prescaler. Comparator continuously compares the values of **TCx** and **CCx**. Asserts a into the an interrupt request when equal, and resets TCx to zero.
- ❖ You can fill a value to CCx to determine period you want.



Prescaler

- TC is 16 Bits counter , counting range from 0 to 65,535. If you need more than the count. You must adjust prescaler to expand the count range.

- For example:

if the clock in is 0.25uS, to reach a 500mS period, the CC0 must be set to 2,000,000 (2,000,000 * 0.25uS = 500mS).

However, this value has exceeded the acceptable range of CC0.

At this point you can set the prescaler to reduce the count value to expand the count range.

$$(2^n - 1) \leq CC0 = \frac{Period}{Clock \times Scale}$$

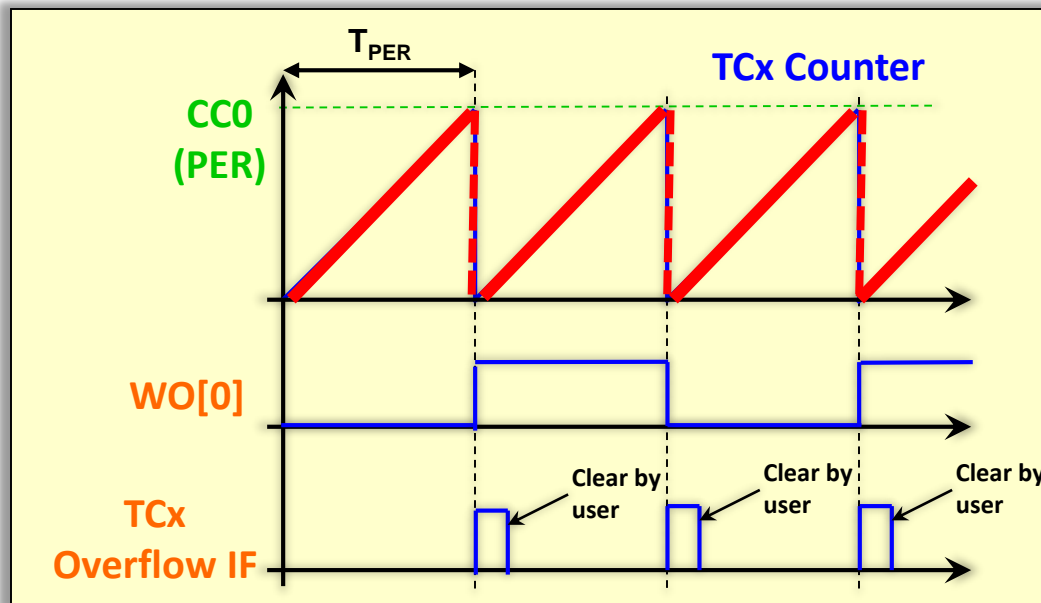
2,000,000	$CC0 = \frac{500mS}{0.25uS \times 1}$	$CC0 > 65,535$
31,250	$CC0 = \frac{500mS}{0.25uS \times 64}$	$CC0 < 65,535$

Timer/Counter

- ❖ **TC provide 4 difference for Timer, Counter and Compare operation mode**
 - ❖ **Normal Frequency Generation (NFRQ)**
toggled on each compare match
 - ❖ **Match Frequency Generation (MFRQ)**
toggles on each update condition
 - ❖ **Normal Pulse-Width Modulation Operation (NPWM)**
toggles on each match & update condition
 - ❖ **Match Pulse-Width Modulation Operation (MPWM)**
toggles on each match & update condition, Count direction reverse.

Match Frequency Generation (MFRQ) Mode

- For Match Frequency Generation, the period time (T_{PER}) is controlled by the **CC0** register instead of PER.
WO[0] toggles on each update condition.



Rule

TCx Counter == CC0 -> WO[0] Toggle

Docking MHC windows

Return to Harmony and docking windows as below.

The screenshot displays the MPLAB Harmony Configurator interface with several windows docked. The 'Active Components' window on the left shows the 'CMSIS Pack' and 'Device Family Pack (DFP)' under the 'System' category. The 'Available Components' window below it lists various peripherals like AC, DSU, EIC, I2S, PM, RTC, and SERCOM. The 'Pin Settings' window in the center shows a table of pins with columns for Pin Number, Pin ID, Custom Name, Function, Mode, Direction, Latch, Pull Up, and Pull Down. The 'Configuration Options' window on the right shows a tree view of system configuration options. The 'Pin Table' window at the bottom shows a grid mapping modules to pins.

Active Components

Available Components


Pin Setting

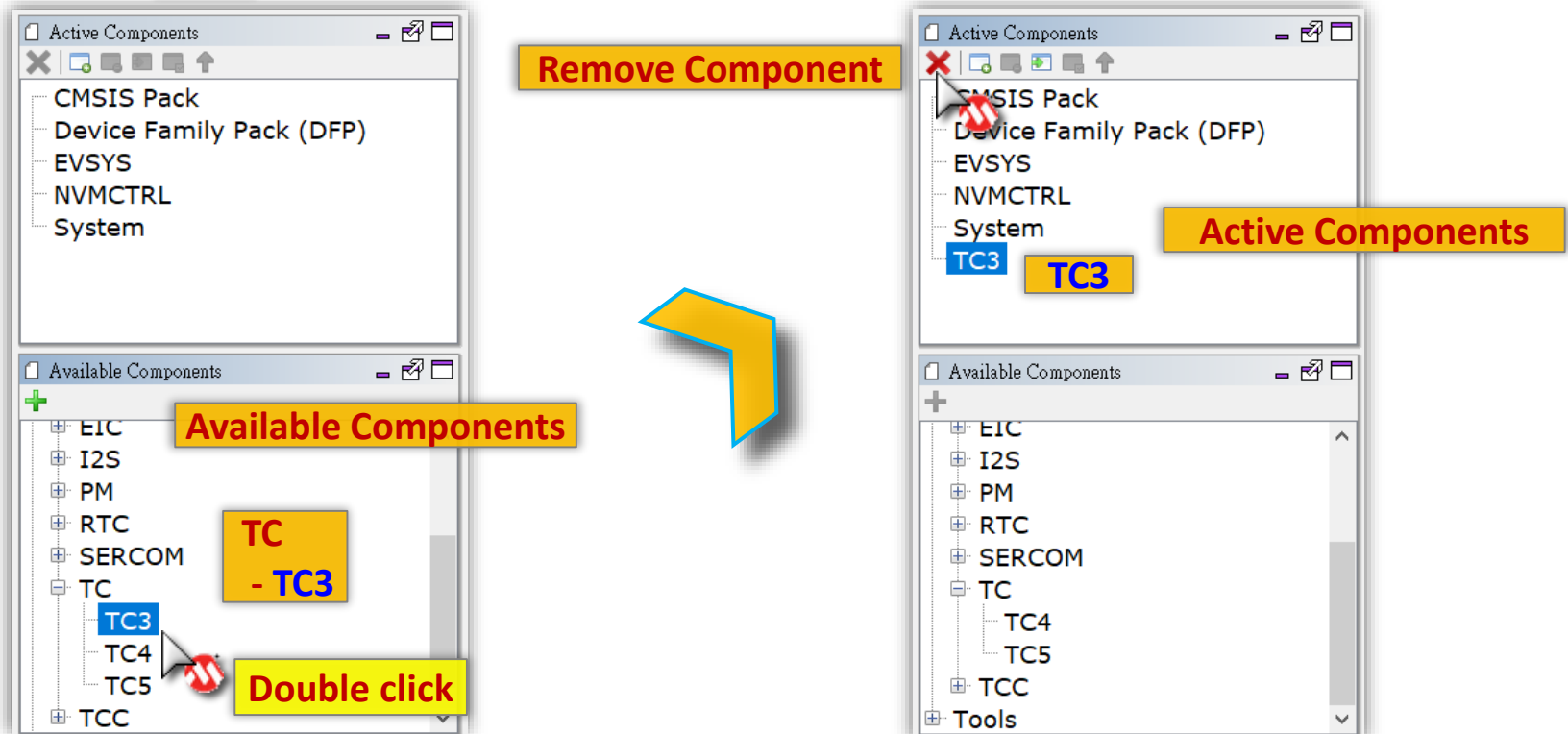
Configuration Options

Pin Table

Module	Function	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48		
AC	AC_AIN0																																																		
	AC_AIN1																																																		
	AC_AIN2																																																		
	AC_AIN3																																																		
	AC_CMP0																																																		
	AC_CMP1																																																		
ADC_AIN0																																																			
ADC_AIN1																																																			

Add TC Function using MHC

- Find **TC3** component in Available Components window.
- Double click **TC3** to add to Active Components window.
- Click  could remove component from Active window.



TC Configuration Options Example

The screenshot shows the 'Configuration Options' dialog box for a Timer Counter (TC3). The 'Counter Mode' is set to 'Counter in 16-bit mode'. The 'Select Prescaler' is set to 'Prescaler: GCLK_TC/256', which is highlighted in green. A yellow callout box points to this setting with the text: 'Value different to default will change to green color'. The 'Operating Mode' is set to 'Compare'. Under the 'Compare' section, 'Waveform Mode' is set to 'Match Frequency', 'Counter Direction' is 'UP Count', and 'Period Value' is '18,750', all highlighted in green. A yellow callout box points to the 'Period Value' with the text: 'Right click the value could Clear User Setting'. A right-click context menu is open over the 'Period Value' field, showing options: 'Clear User Setting' (highlighted in blue), 'Display Mode', and 'Show User Manual Entry'. The menu title is 'TC_CTRLA_PRESCALER'. Other options like 'Compare Value' (24), 'Enable One-Shot Mode', 'Invert Output WO[0]', 'Invert Output WO[1]', and 'Enable Period Interrupt' are also visible. The 'Events' and 'Sleep Configurations' sections are partially visible at the bottom.

Configuration Options

TC3

- Counter Mode: Counter in 16-bit mode
- Select Prescaler: Prescaler: GCLK_TC/256
- ****Timer resolution is 5333.33333333 nS****
- Operating Mode: Compare
- Compare
 - Waveform Mode: Match Frequency
 - Counter Direction: UP Count
 - Period Value: 18,750
 - **** Timer Period is 100000.0 us ****
 - Compare Value: 24
 - Enable One-Shot Mode: ☐
 - Invert Output WO[0]: ☐
 - Invert Output WO[1]: ☐
 - Enable Period Interrupt: ☐
- Events
- Sleep Configurations
 - Run during Standby: ☐

TC_CTRLA_PRESCALER

- Clear User Setting
- Display Mode
- Show User Manual Entry

Configuration Options Help

TC polling Interface Routines

```
/** TC3 Interface Routines**/  
void TC3_CompareInitialize( void );  
void TC3_CompareStart( void );  
void TC3_CompareStop( void );  
uint32_t TC3_CompareFrequencyGet( void );  
void TC3_Compare16bitPeriodSet( uint16_t period );  
uint16_t TC3_Compare16bitPeriodGet( void );  
uint16_t TC3_Compare16bitCounterGet( void );  
void TC3_Compare16bitCounterSet( uint16_t count );  
TC_COMPARE_STATUS TC3_CompareStatusGet( void );
```

TC Interrupt Flag Event

- The TC has the following interrupt sources:
 - ▢ **Overflow/Underflow (OVF)**
Counter overflow/underflow
 - ▢ **Match or Capture Channel x (MCx)**
Compare match or capture
 - ▢ **Capture Overflow Error (ERR)**
New capture interrupt
occurs while interrupt flag is set.
 - ▢ **Synchronization Ready (SYNCRDY)**
Read/Write synchronization ready.
- **How to get and polling TC overflow interrupt flag status ?**

TC Polling Code Example

TC MFRQ Example

```
...
TC3_CompareStart();

while(1)
{
    if ( TC3_REGS->COUNT16.TC_INTFLAG & TC_COMPARE_STATUS_OVERFLOW )
    {
        TC3_REGS->COUNT16.TC_INTFLAG = TC_COMPARE_STATUS_OVERFLOW;

        LED1_Toggle();
    }
}
```

Bit 0 – OVF Overflow Interrupt Flag

Writing a '0' to this bit has no effect.

Writing a '1' to this bit clears the Overflow interrupt flag.

OR

```
...
TC3_CompareStart();

while(1)
{
    if ( TC3_CompareStatusGet() & TC_COMPARE_STATUS_OVERFLOW )
    {
        LED1_Toggle();
    }
}
```

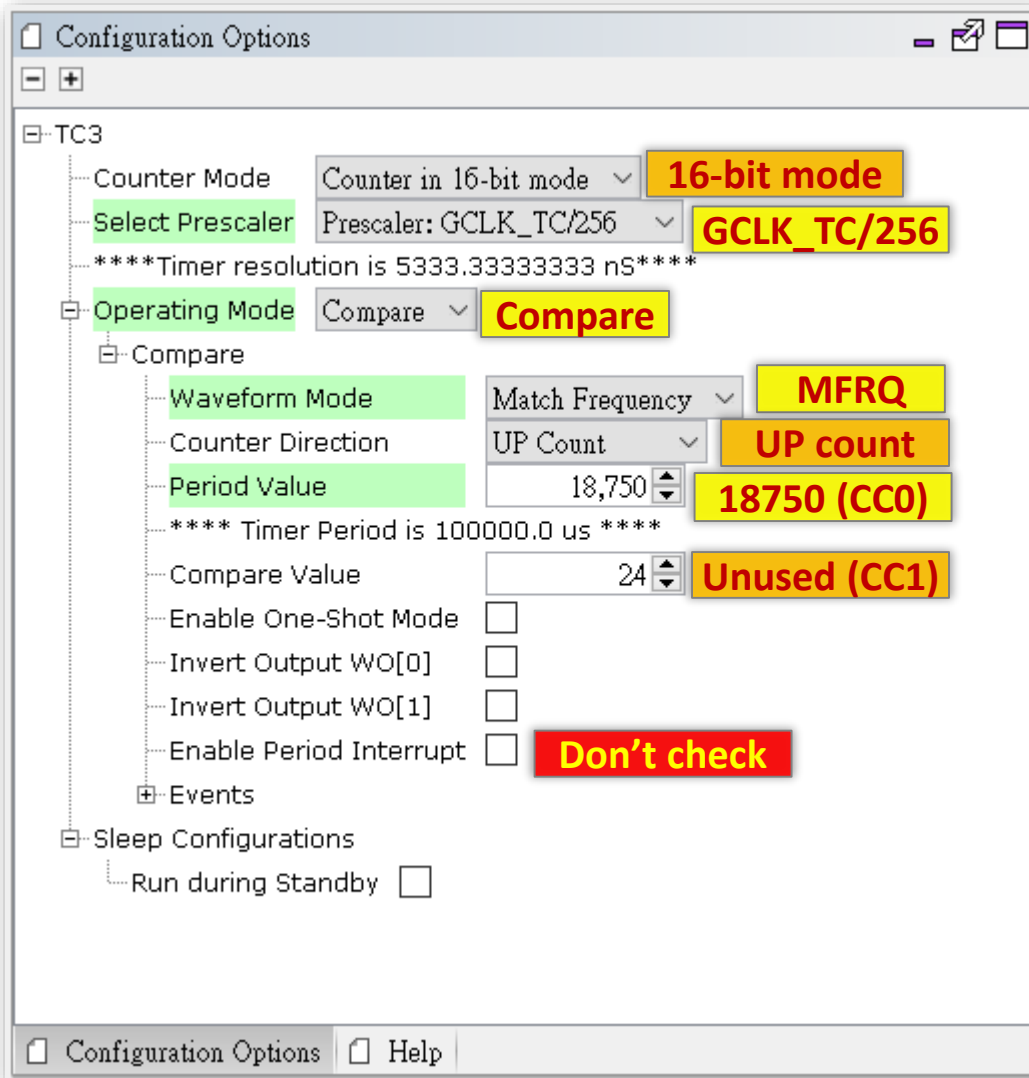
Lab3 TC MFRQ Polling

- Try to use **TC3** to replace software delay to control **LED1(PA20)** continues toggle.
- The TC3 setup to **MFRQ**, 16 bit mode, Timer period is **100mS**.
- Enable waveform output of TC3 on pin **PA18** and **PA19**.

 **Let's go!**

Lab3 TC MFRQ Polling

Step 1

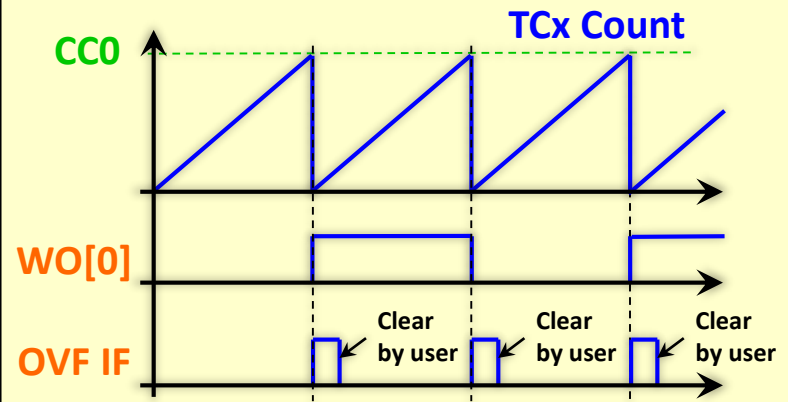


System Clock = 48MHz

One second counts of system clock :
48,000,000 counts

One second counts after Prescaler :
 $48000000 / 256$ 187,500 counts

100ms counts for LAB3 target :
 $187500 / 10$ 18,750 counts



Lab3 TC MFRQ Polling

Step 2

- TC support waveform output through WO[x] pin depend on different mode.

The image shows two screenshots of the Pin Table configuration tool, illustrating the process of enabling TC3_WO0 and TC3_WO1 pins for waveform output.

Top Screenshot: The Pin Table is shown for Package: TQFP48. The TC3 module is selected. The TC3_WO0 and TC3_WO1 functions are highlighted. The PA18 and PA19 pins are highlighted. A yellow box labeled "Click" points to the PA18 and PA19 pins.

Bottom Screenshot: The Pin Table is shown for Package: TQFP48. The TC3 module is selected. The TC3_WO0 and TC3_WO1 functions are highlighted. The PA18 and PA19 pins are highlighted. A yellow box labeled "Green is Enable" points to the PA18 and PA19 pins.

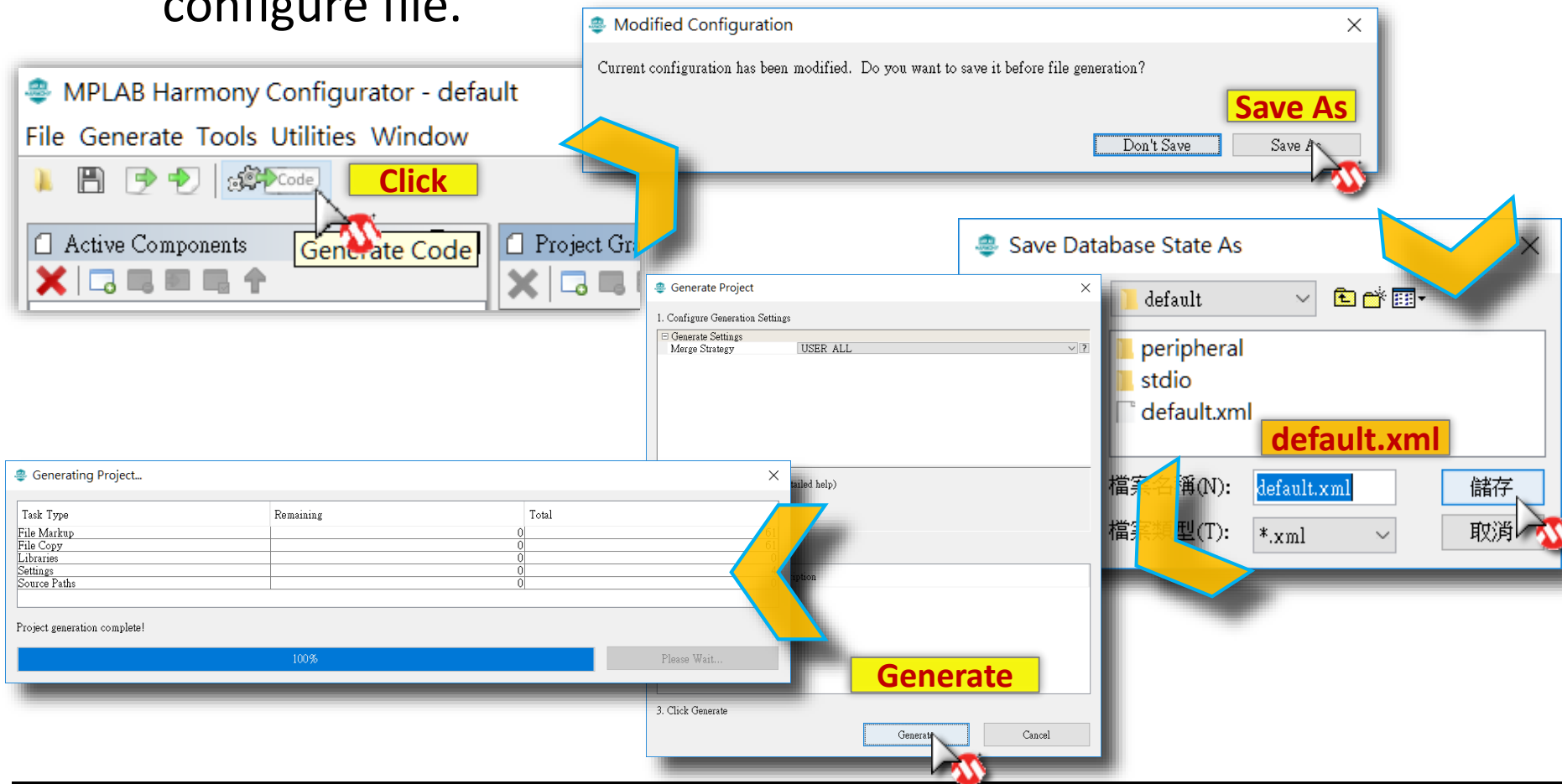
The Pin Table structure is as follows:

Module	Function	PA18	PA19	LED1
TC3	TC3_WO0			
TC3	TC3_WO1			

Lab3 TC MFRQ Polling

Step 3

- Click  to Generate Code and save changes to MHC configure file.



Lab3 TC MFRQ Polling

Step 4

a Add code segment to your main loop

```
...
int main (void)
{
    SYS_Initialize ( NULL );

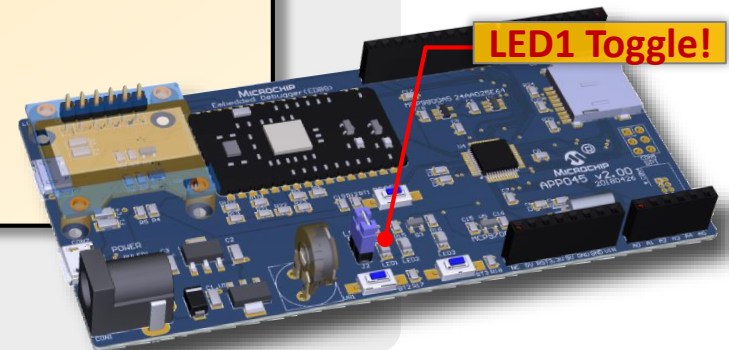
    // TODO 3.01
    TC3_CompareStart();

    while(1)
    {
        // TODO 3.02
        if ( TC3_CompareStatusGet() &
            TC_COMPARE_STATUS_OVERFLOW )
        {
            LED1_Toggle();
        }
        ...
    }
}
```

Period Control

LED1 Toggle!

b Program firmware to target board then observe result.

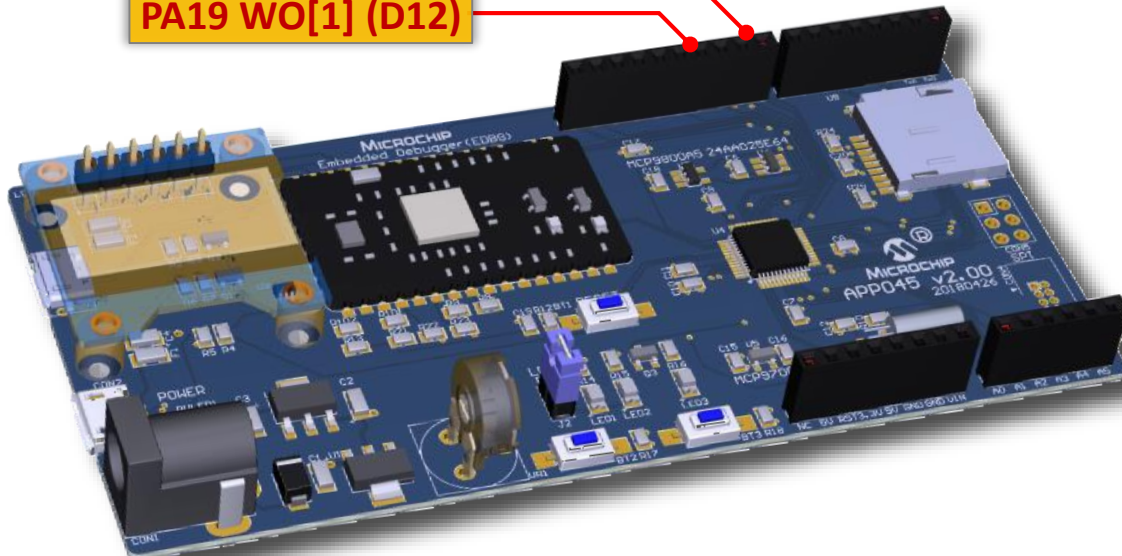


Lab3 TC MFRQ Polling

Waveform Output

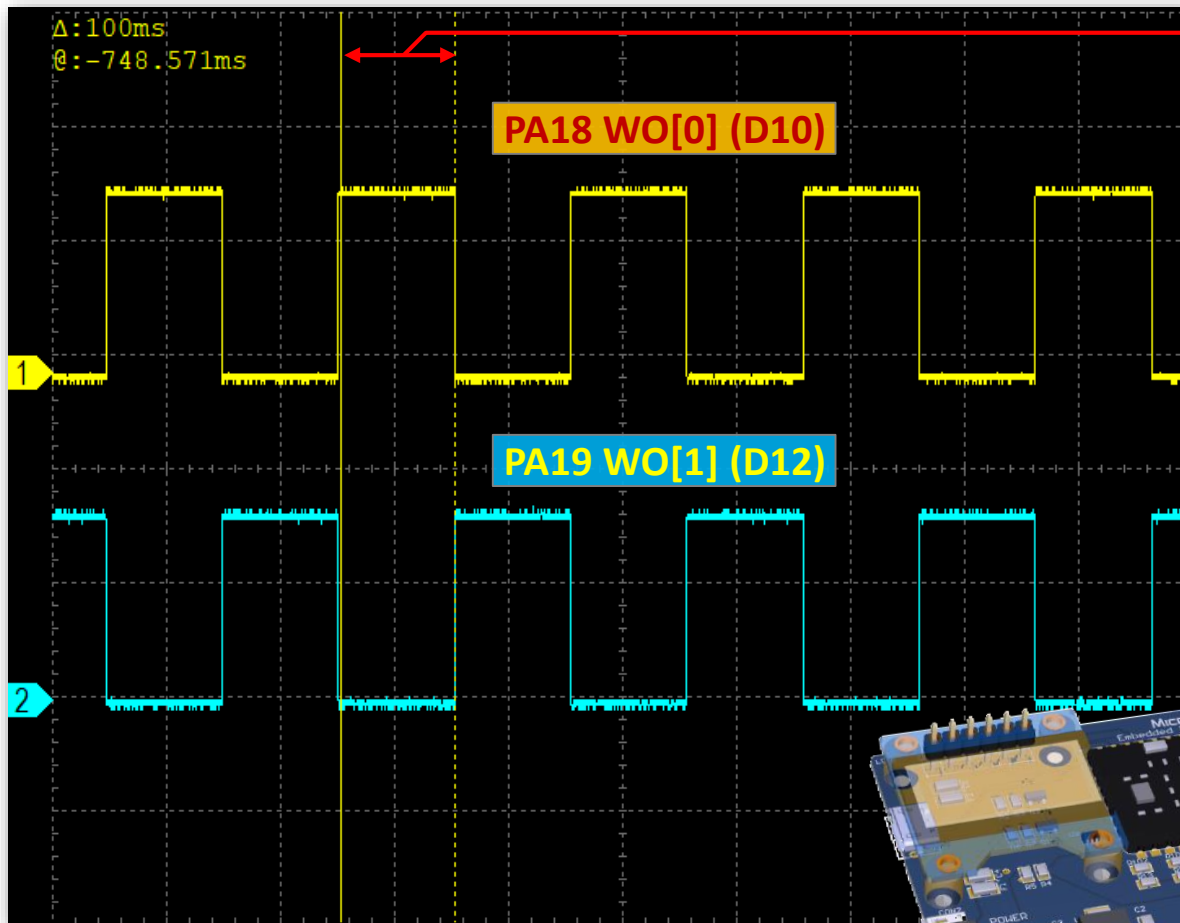
PA21	29	PA20	EDBG GPIO215		CON9C
PA20	28	EDBG MISO	EDBG GPIO316		Arduino UNO Socket
PA19	27	EDBG CS	EDBG CS 17		D8
PA18	26	EDBG SCK	EDBG MOSI 18		D9/PWM
PA17	25	EDBG MOSI	EDBG MISO 19		D10 PWM/SS
			EDBG SCK 20		D11/PWM/MOSI
			21		D12 MISO
					D13/SCK

PA18 WO[0] (D10)
PA19 WO[1] (D12)



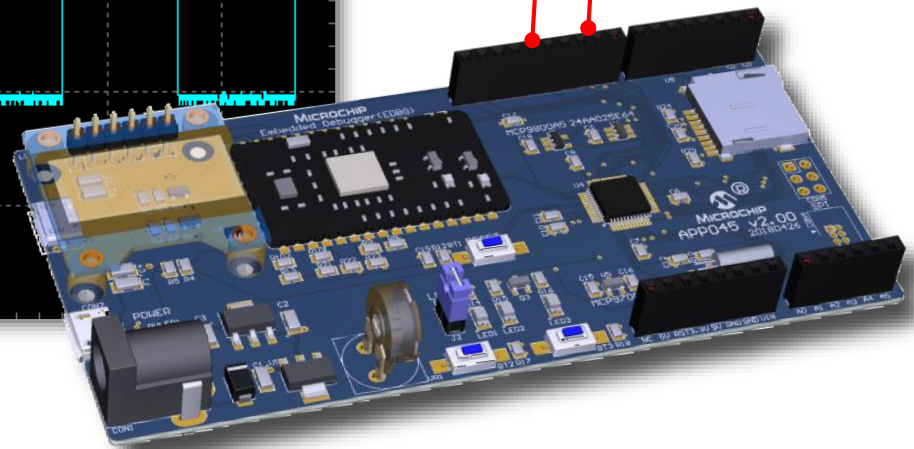
Lab3 TC MFRQ Polling

Waveform Output



Δ:100ms
@:-748.571ms

PA19 WO[1] (D12)
PA18 WO[0] (D10)




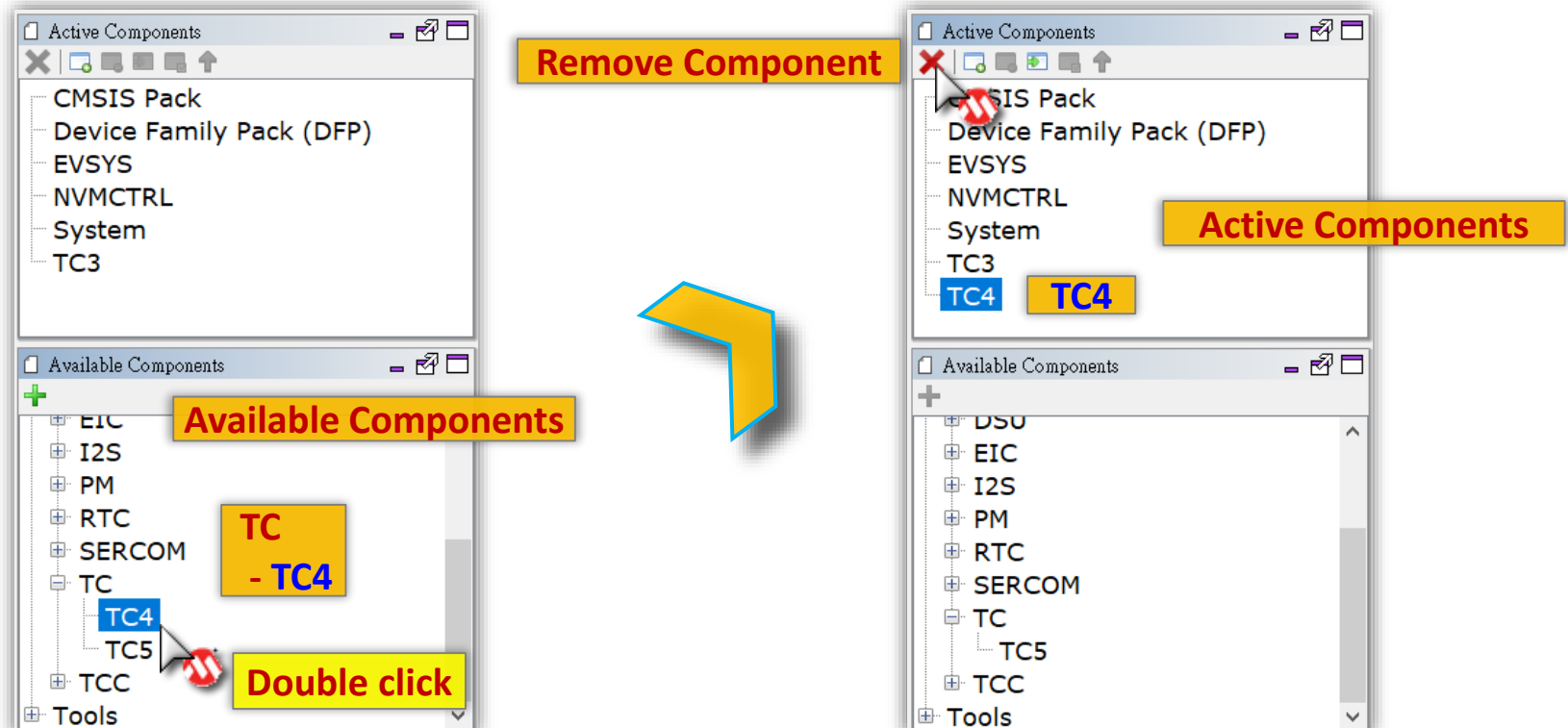
Lab4 TCs MFRQ Polling

- Try to add **TC4** to replace software delay to control **LED2(PA09)** continues toggle.
- All setting same as TC3 except period.
- The **TC4** set timer period to **50ms**.
- Enable waveform output of TC4 on pin **PA22** and **PA23**.
Then you can measure the signal at PA22 and PA23 also.

Lab4 TCs MFRQ Polling

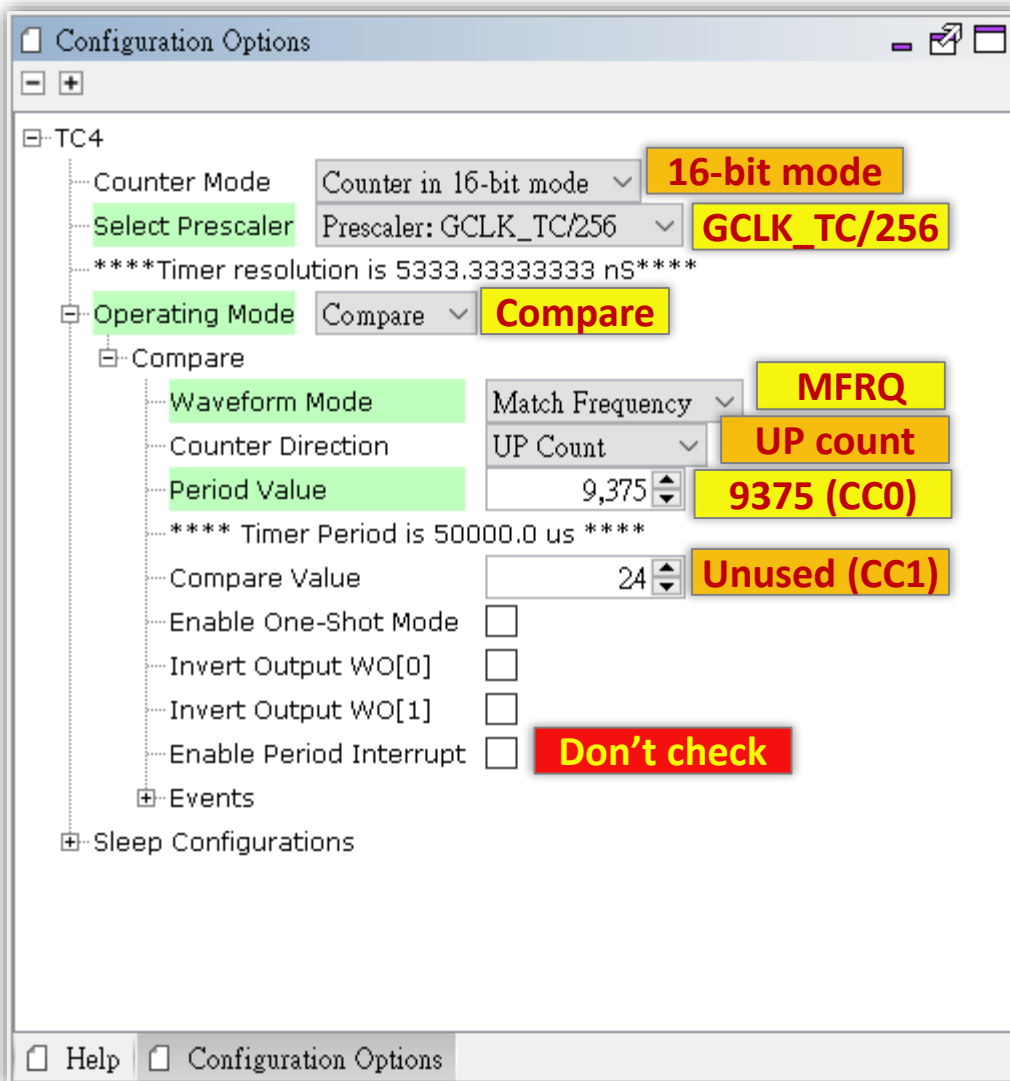
Step 1

- Find **TC4** component in Available Components window.
- Double click **TC4** to add to Active Components window.
- Click  could remove component from Active window.



Lab4 TCs MFRQ Polling

Step 2



System Clock = 48MHz

One second counts of system clock :

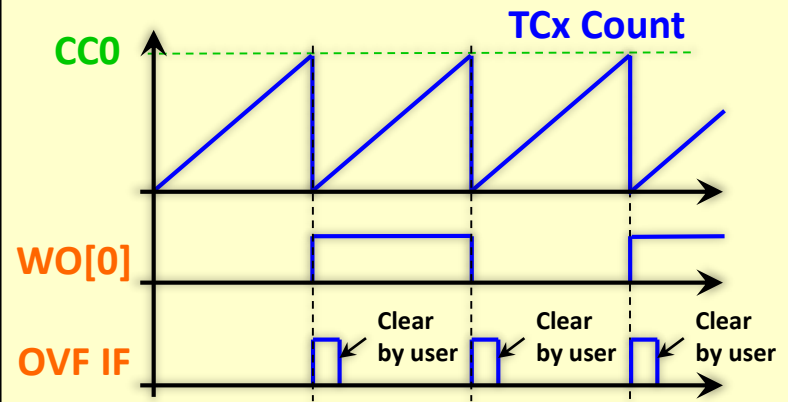
48,000,000 counts

One second counts after Prescaler :

48000000 / 256 187,500 counts

50ms counts for LAB4 target :

187500 / 20 9,375 counts



Lab4 TCs MFRQ Polling

Step 3

- TC support waveform output through WO[x] pin depend on different mode.

PA22 / PA23

Click

TC4

TC4_WO0

TC4_WO1

TC4_WO0

TC4_WO1

Green is Enable

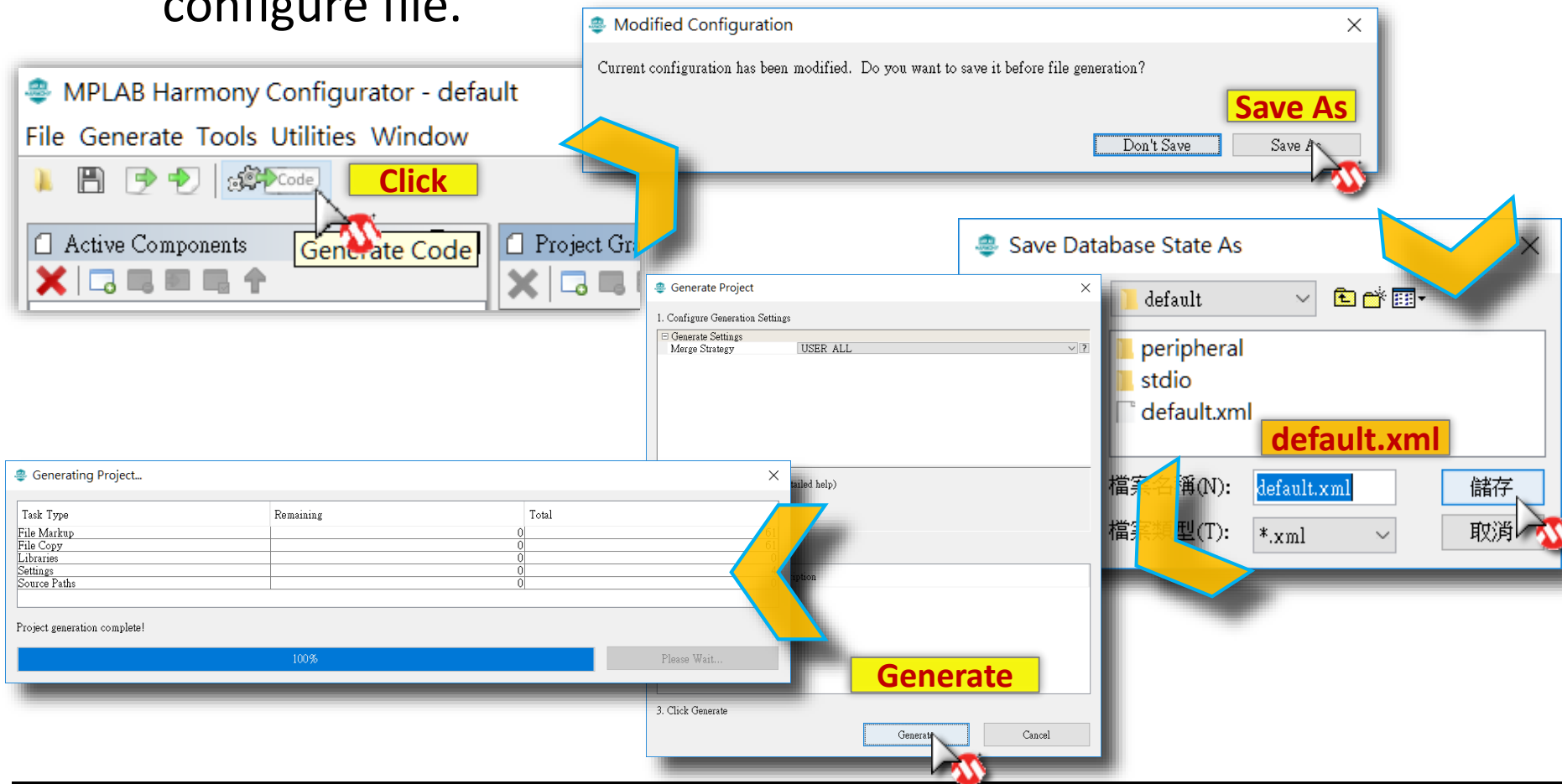
Module	Function	0	21	22	23	24	25	26	27	28	29	30	31	32	33	34
TC3	TC3_WO0															
	TC3_WO1															
TC4	TC4_WO0															
	TC4_WO1															

Module	Function	0	21	22	23	24	25	26	27	28	29	30	31	32	33	34
TC3	TC3_WO0															
	TC3_WO1															
TC4	TC4_WO0															
	TC4_WO1															

Lab4 TCs MFRQ Polling

Step 4

- Click  to Generate Code and save changes to MHC configure file.



Lab4 TCs MFRQ Polling

Step 5

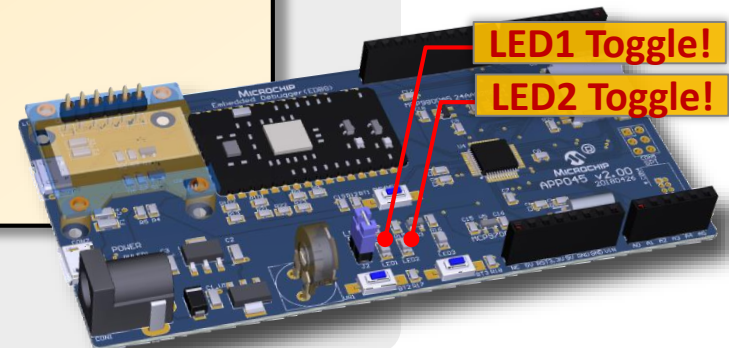
a Add code segment to your main loop

```
int main (void)
{
    SYS_Initialize ( NULL );
    ...
    // TODO 4.01
    TC4_CompareStart();

    while(1)
    {
        ...
        // TODO 4.02
        if ( TC4_CompareStatusGet() &
            TC_COMPARE_STATUS_OVERFLOW )
        {
            LED2_Toggle();
        }
        ...
    }
}
```

Period Control

b Program firmware to target board then observe result.



LED1 Toggle!

LED2 Toggle!

Lab4 TCs MFRQ Polling

Waveform Output

PA25	33	D-
PA24	32	SCL
PA23	31	SDA
PA22	30	EDBG GPIO1
PA21	29	PA20

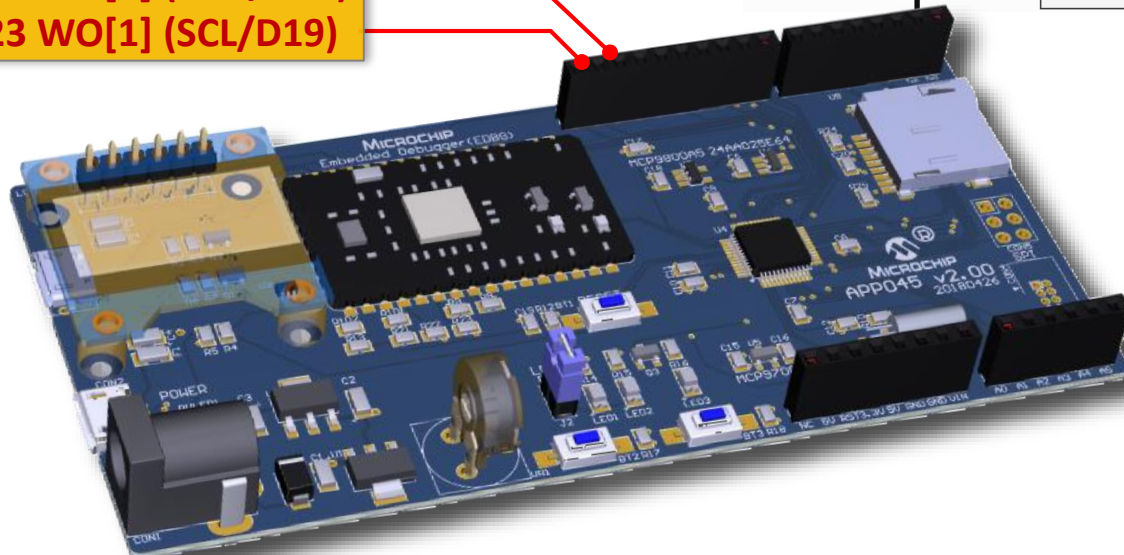
EDBG GPIO215
EDBG GPIO316
EDBG CS 17
EDBG MOSI18
EDBG MISO 19
EDBG SCK 20

PA03	22
SDA	23
SCL	24

CON9C
Arduino UNO Socket

D8
D9/PWM
D10/PWM/SS
D11/PWM/MOSI
D12/MISO
D13/SCK
GND
AREF
D18/SDA
D19/SCL

PA22 WO[0] (SDA/D18)
PA23 WO[1] (SCL/D19)



Lab4 TCs MFRQ Polling

Waveform Output

