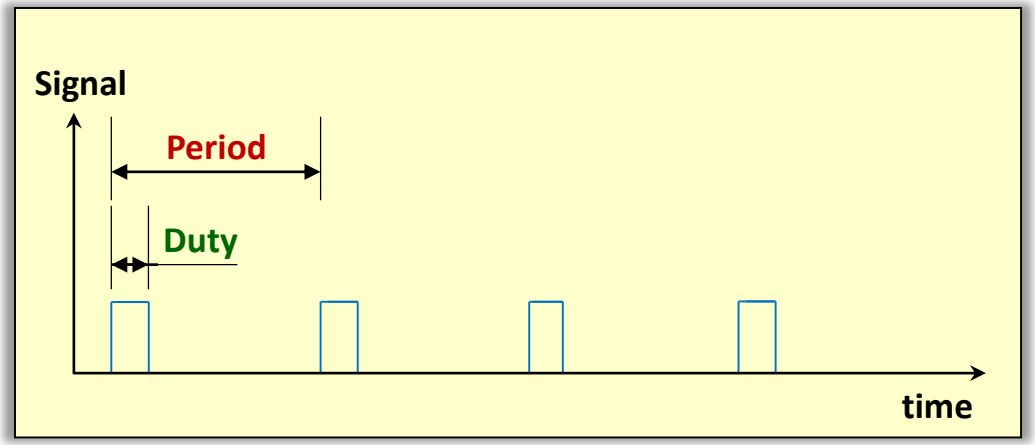
MICROCHIP

Regional Training Centers

Section 11 TCC – PWM Architecture

What's PWM

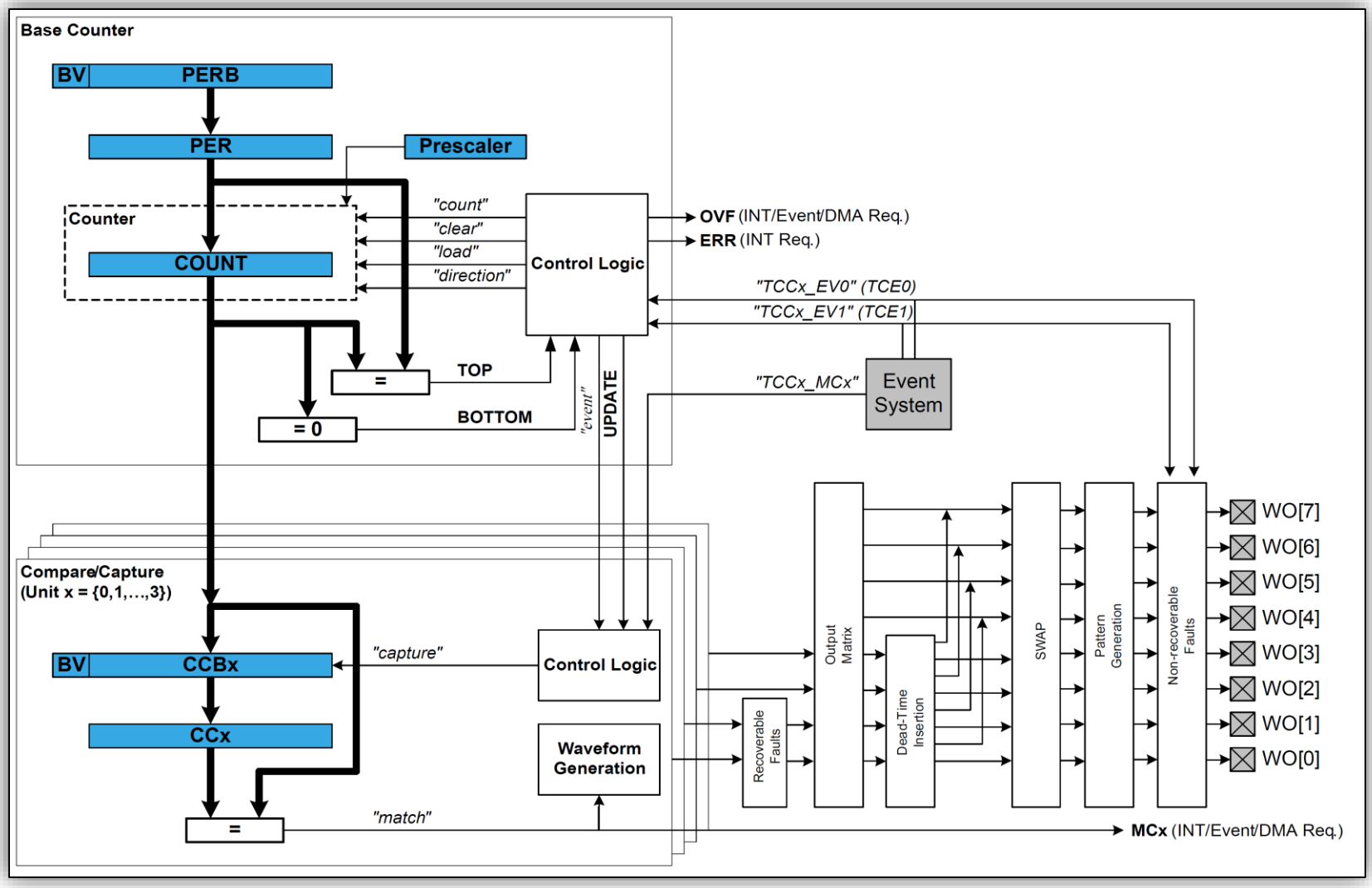
- The Pulse-width modulation (PWM) Waveform is a modulation technique used to encode a message into a pulsing signal.
- Its main use is to allow the control of the power supplied to electrical devices, ex. motor control, ballast, LED, H-bridge, power converters, and other types of power control applications.
- There have two important terms
  - **Period :**
    The single square wave duration.
  - **Duty :**
    The proportion of active time to the regular interval



SAMD21 TCC

- **Timer/Counter for Control, TCC is the TC Enhanced version.**
- **TCC provide below feature**
  - Up to four compare/capture channels
  - Waveform (PWM, Frequency generation)
  - Input capture (Event, Frequency, PWM Capture)
  - Fault protection for safe disabling
  - 3 input Event, 3 Event
  - Support Interrupt, DMA, Event

TCC Block Diagram

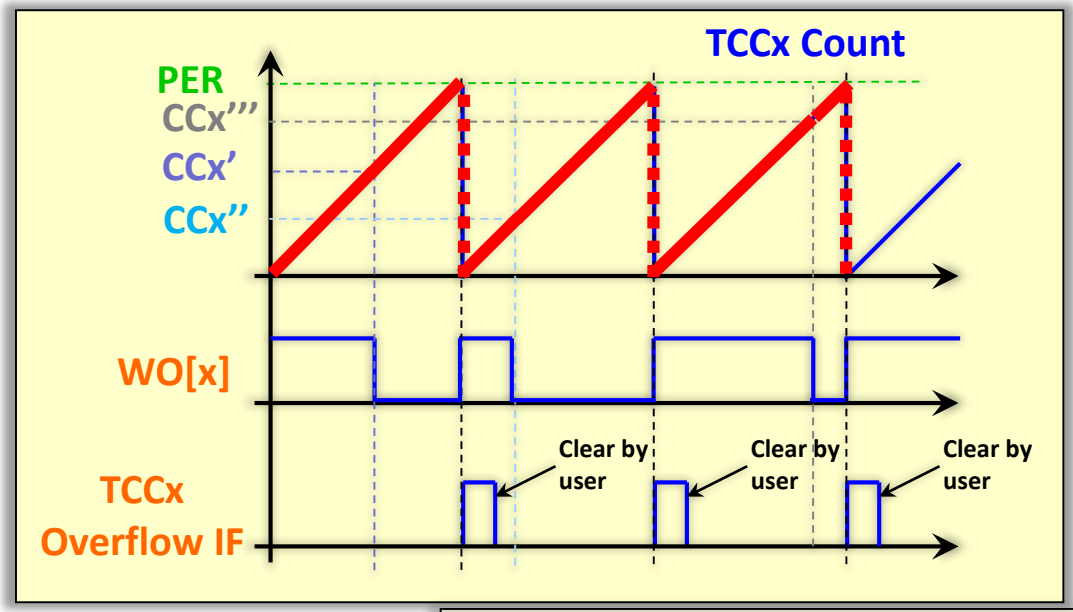


TCCx, Channel and WO[x]

- **TCCx : Timer Counter for Control peripheral**
  - TCC0, TCC1, TCC2
- **Channel : Compare Channel CCx**
  - TCC0 : 4 channel
  - TCC1 : 2 channel
  - TCC2 : 2 channel
- **WO[x] : Waveform Output Pins**
  - TCC0 : 8 pins ➔ WO[0] ~ WO[7] ➔ Channel [0, 1, 2, 3, **0, 1, 2, 3**]
  - TCC1 : 4 pins ➔ WO[0] ~ WO[3] ➔ Channel [0, 1, **0, 1**]
  - TCC2 : 2 pins ➔ WO[0] ~ WO[1] ➔ Channel [0, 1]

Normal Pulse-Width Modulation (NPWM) Mode

- For Normal Pulse-Width Modulation , the period time (T) is controlled by the PER register. WO[x] is set at start and cleared on compare match.



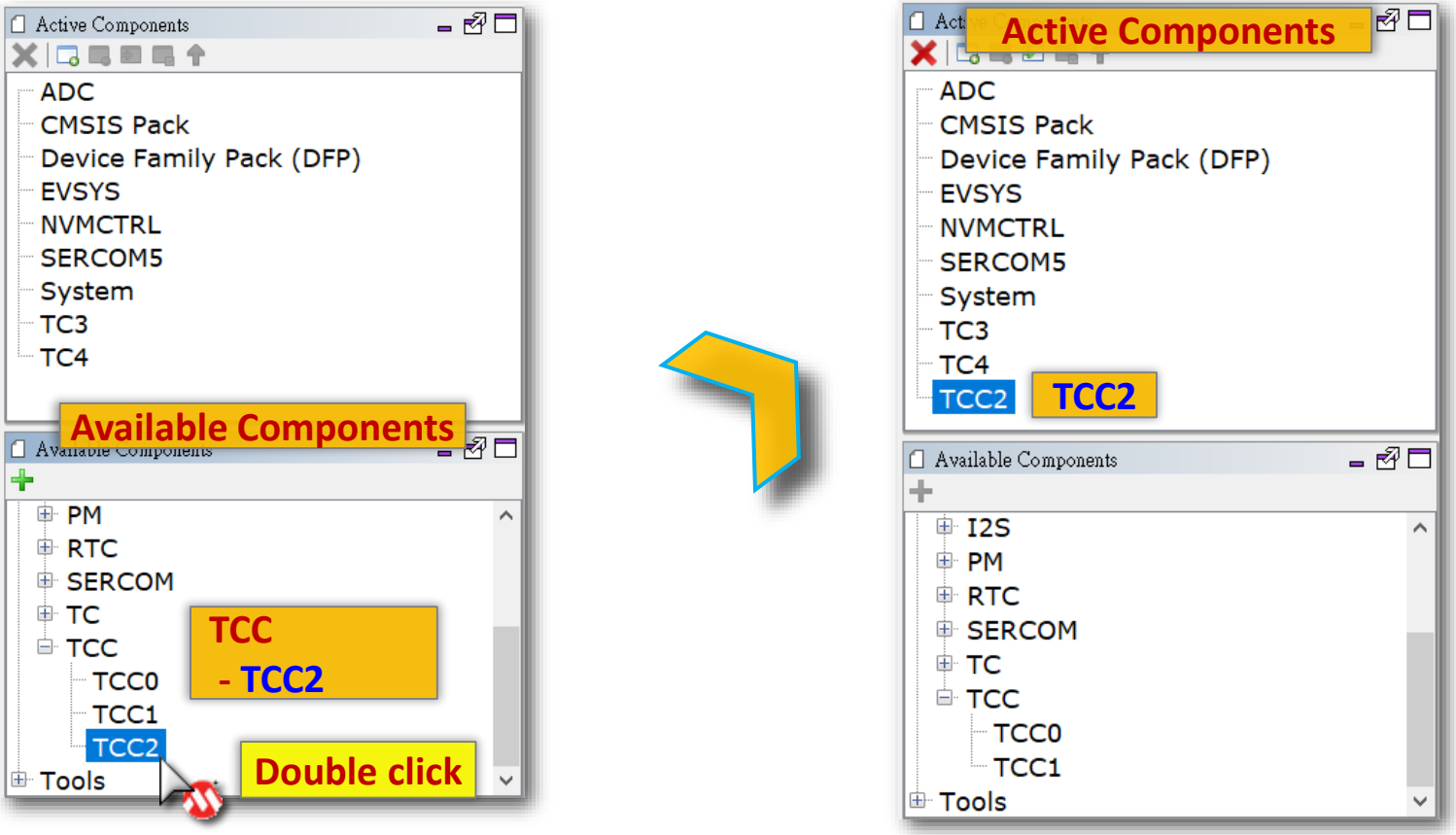
Rule

1.TCCx Counter == CCx -> WO[x] Clear to low

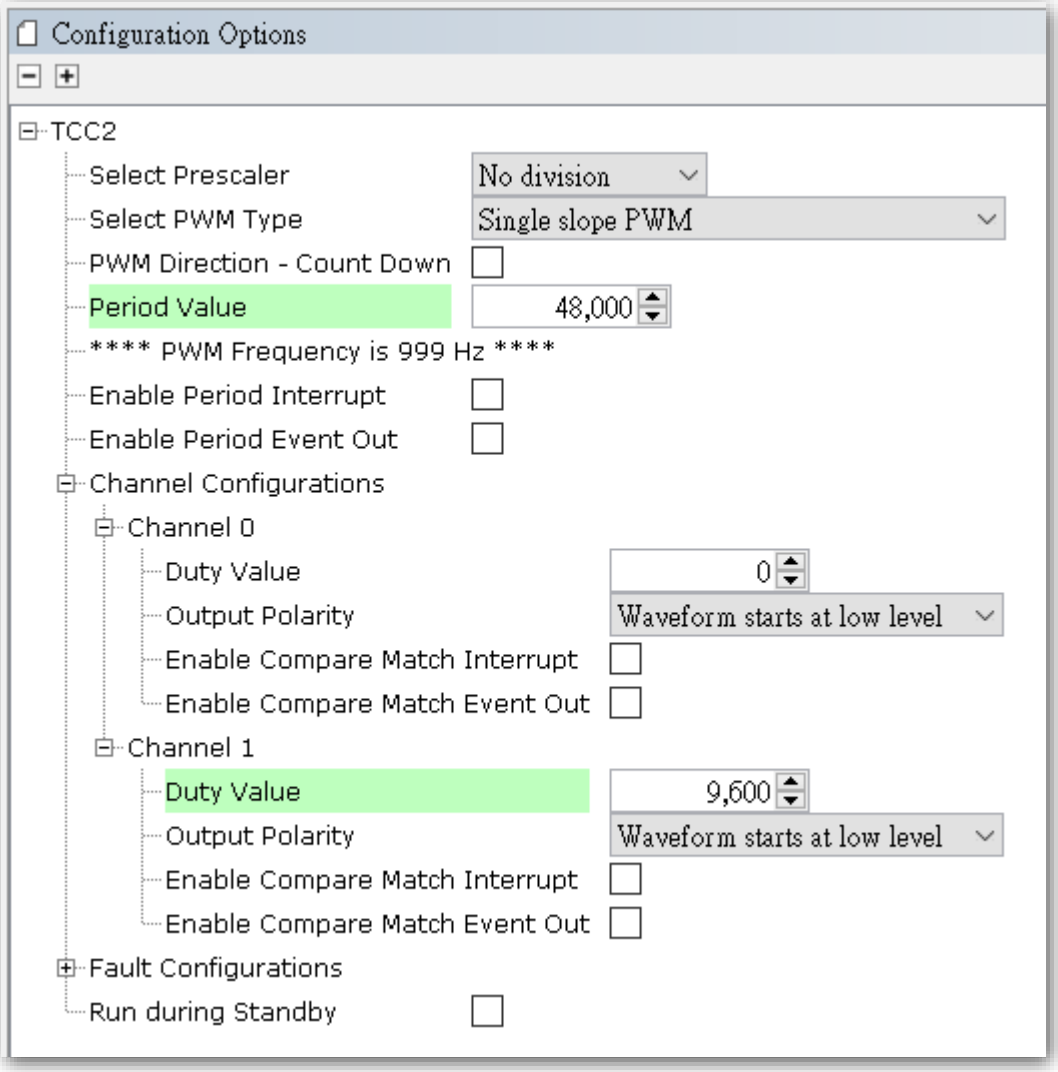
2.TCCx Counter == PER -> WO[x] Set to high

Add TCC Function using MHC

- Find TCC component in Available Components window.
- Double click TCC2 to add to Active Components window.



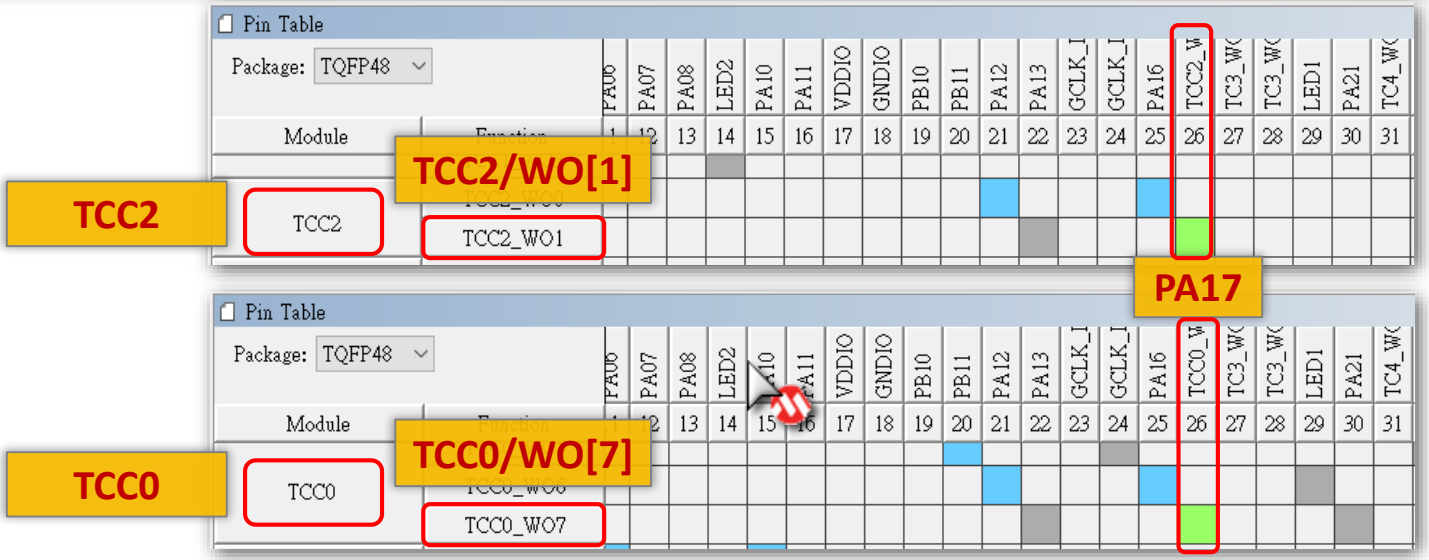
TCC Configuration Options Example



TCC PMUX Configuration Example

- **TCC could output waveform to corresponds pins of different channel.**
- **For example : PA17 as TCC2/WO[1] (TCC2 Channel 1) or TCC0/WO[7] (TCC0 Channel 3)**

Pin(1)			I/O Pin	Supply	A	B(2)(3)					C	D	E	F	G	H
SAMD21E	SAMD21G	SAMD21J			EIC	REF	ADC	AC	PTC	DAC	SERCOM(2)(3)	SERCOM-ALT	TC(4) /TCC	TCC	COM	AC/ GCLK
18	26	PA17	PA17	VDDIO	EXTINT[1]				X[5]		TCC2/WO[1]		TCC2/WO[1]	TCC0/WO[7]	TCC0/WO[7]	



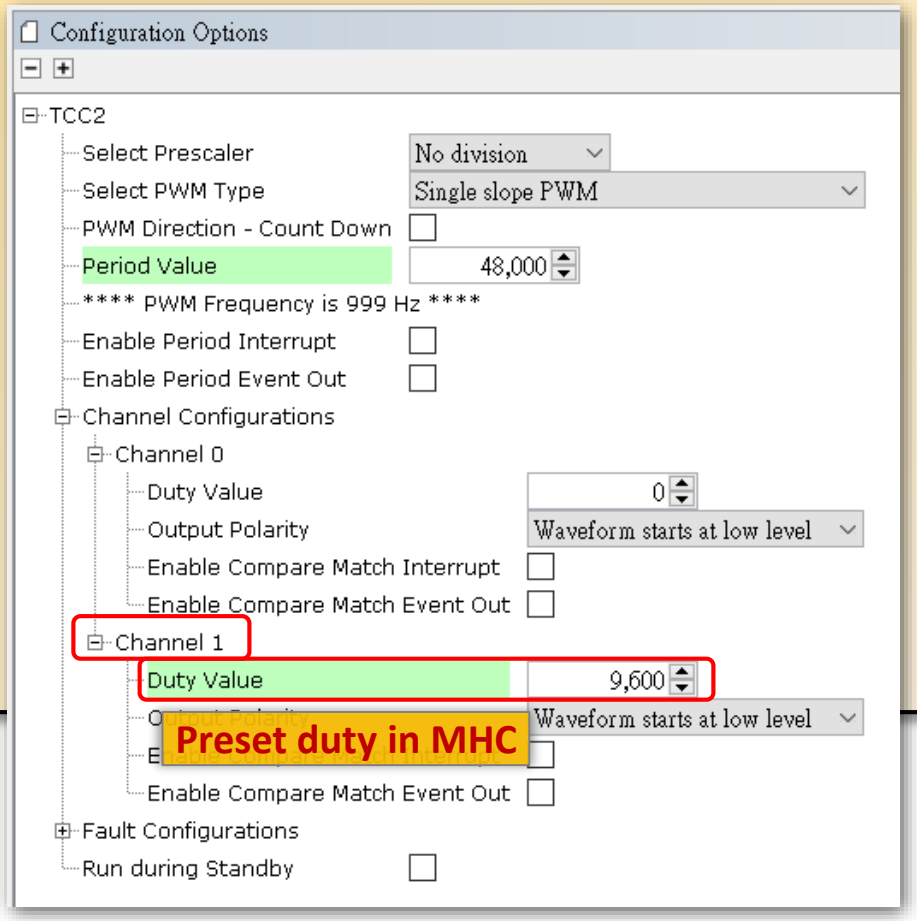
TCC Polling Interface Routines

```
/*** TCC Interface Routines***/
void TCC2_PWMInitialize(void);
void TCC2_PWMStart(void);
void TCC2_PWMStop(void);
void TCC2_PWMForceUpdate(void);
void TCC2_PWMPeriodInterruptEnable(void);
void TCC2_PWMPeriodInterruptDisable(void);
uint32_t TCC2_PWMInterruptStatusGet(void);
void TCC2_PWM16bitPeriodSet(uint16_t period);
uint16_t TCC2_PWM16bitPeriodGet(void);
void TCC2_PWM16bitCounterSet(uint16_t count);
__STATIC_INLINE void TCC2_PWM16bitDutySet(TCC2_CHANNEL_NUM channel,
                                          uint16_t duty);
```

TCC Polling Code Example

```
...

int main (void)
{
   SYS_Initialize ( NULL );
   ...
   TCC2_PWMStart();
   TCC2_PWM16bitDutySet( 1, 9600 );

   while(1)
   {
      ...
   }
}
```

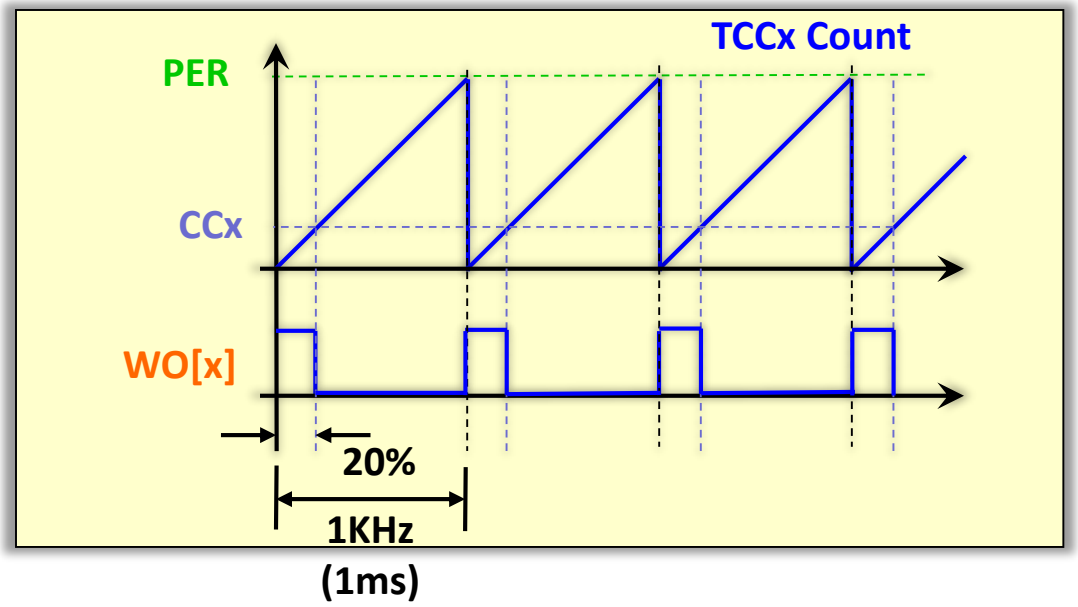
Running time update duty Channel 1, Duty 9600



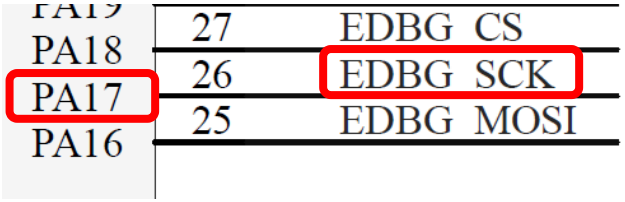
Lab14 TCC NPWM

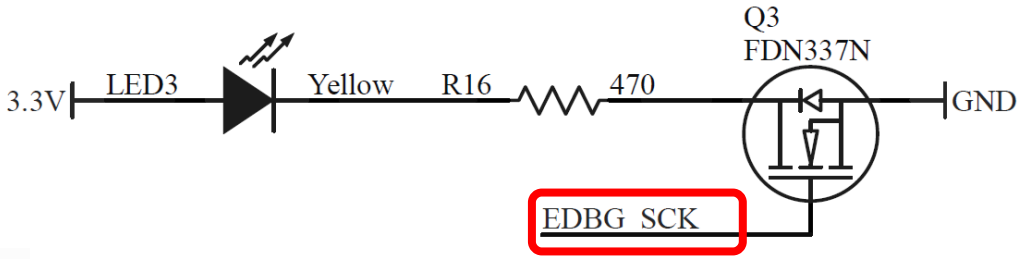
- Add TCC2 NPWM Mode (Single Slope PWM) function to your project.
- Disable LED3(PA17) GPIO function firstly.
- Configure TCC2 to output PWM waveform to LED3(PA17).
- PWM frequency is 1KHz with 20% Duty.

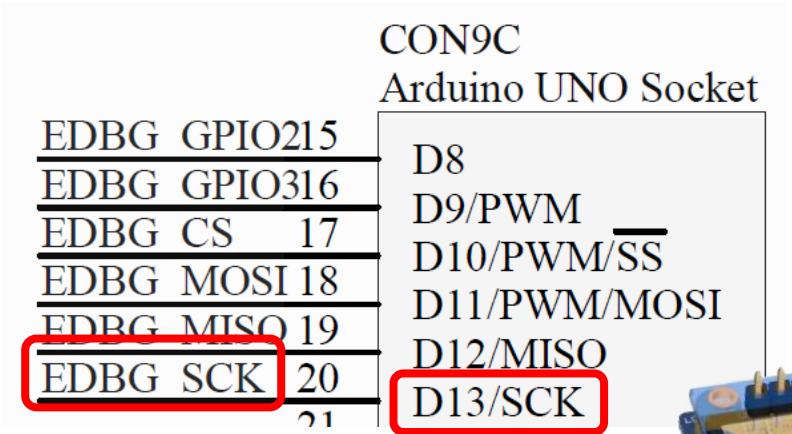
- Let's go!

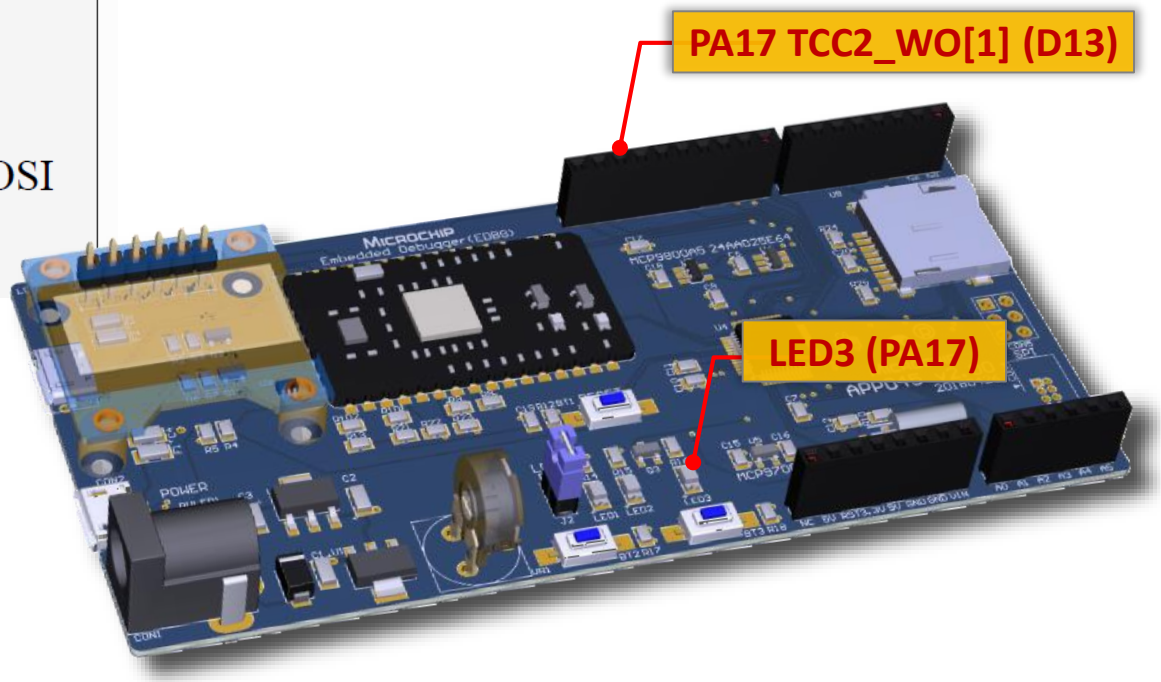


Lab14 TCC NPWM









Lab14 TCC NPWM

Step 1

Configuration Options

- TCC2
  - Select Prescaler: No division — **No division**
  - Select PWM Type: Single slope PWM — **Single slope PWM**
  - PWM Direction - Count Down
  - Period Value: 48,000 — **Period : 48000**
  - **** PWM Frequency is 999 Hz ****
  - Enable Period Interrupt
  - Enable Period Event Out
  - Channel Configurations
    - Channel 0
      - Duty Value: 0
      - Output Polarity: Waveform starts at low level
      - Enable Compare Match Interrupt
      - Enable Compare Match Event Out
    - Channel 1 — **Channel 1**
      - Duty Value: 9,600 — **Duty : 9600**
      - Output Polarity: Waveform starts at low level
      - Enable Compare Match Interrupt
      - Enable Compare Match Event Out

System Clock = 48MHz

One second counts of system clock :

48,000,000 counts

One second counts after Prescaler :

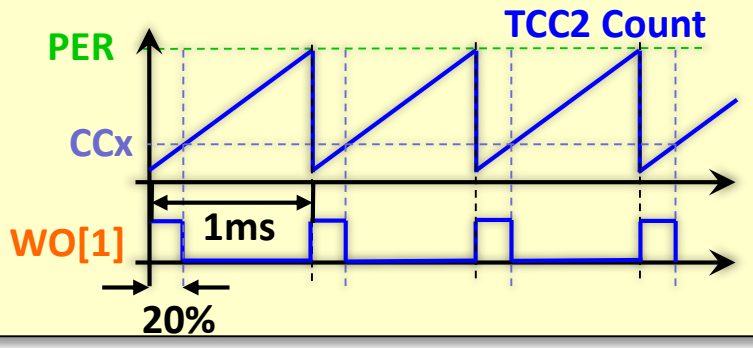
48000000 / 1 48,000,000 counts

1ms counts for LAB14 period target :

48000000 / 1000 48,000 counts

1/5ms counts for LAB14 duty target :

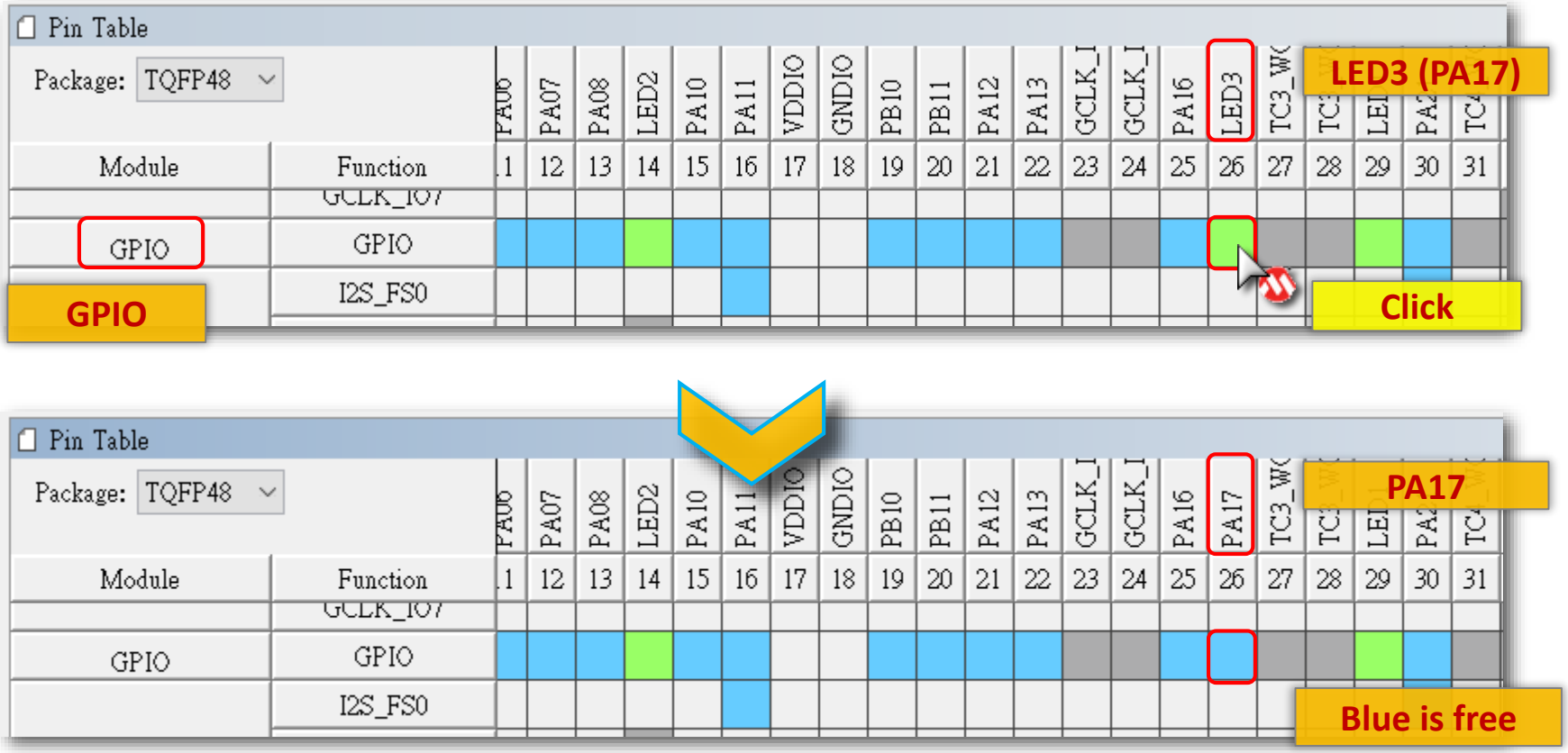
48000 / 5 9,600 counts



Lab14 TCC NPWM

Step 2

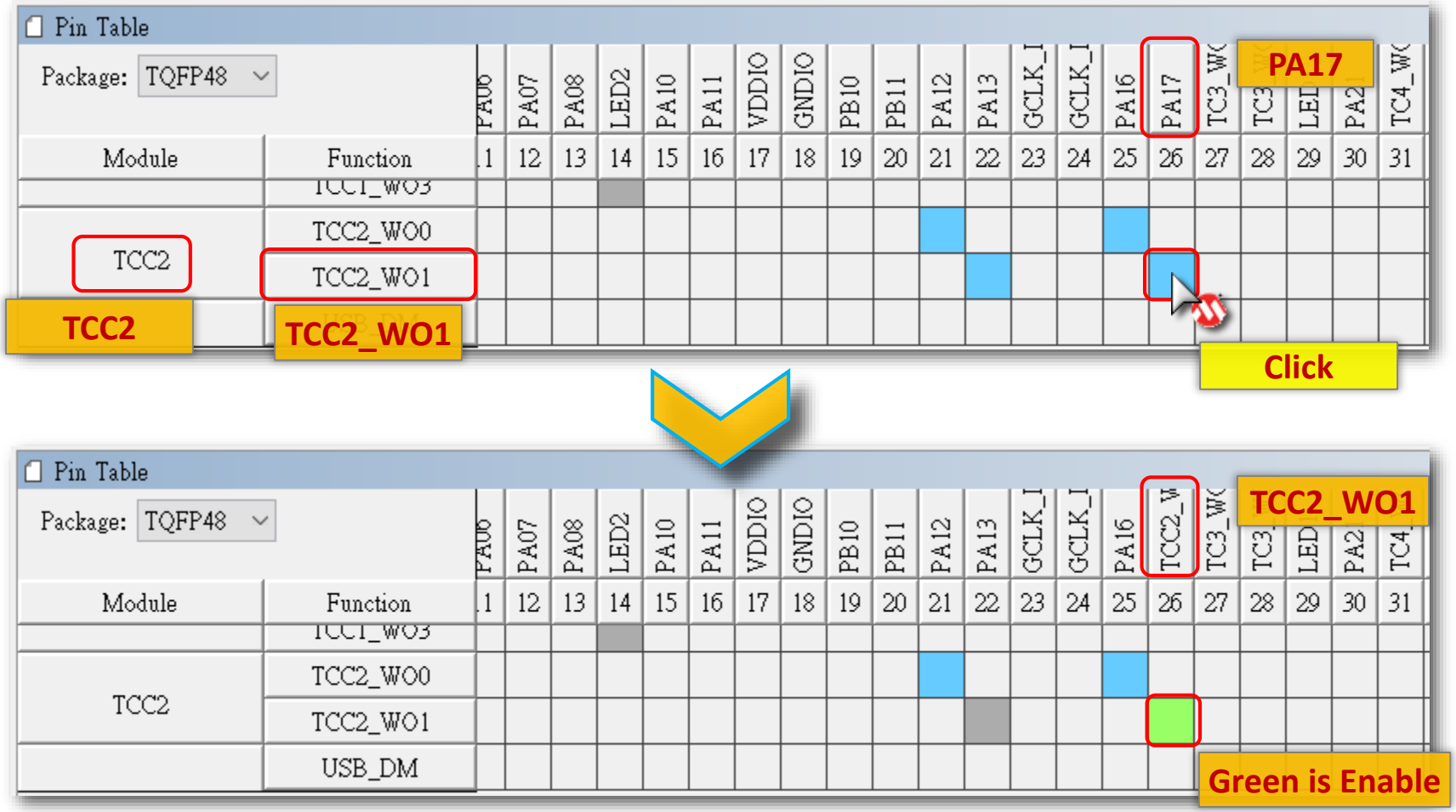
- Disable LED3(PA17) GPIO function firstly.



Lab14 TCC NPWM

Step 3

- Enable TCC2 Waveform Output 1 (TCC2_WO1).



Lab14 TCC NPWM

Step 4

- Click 

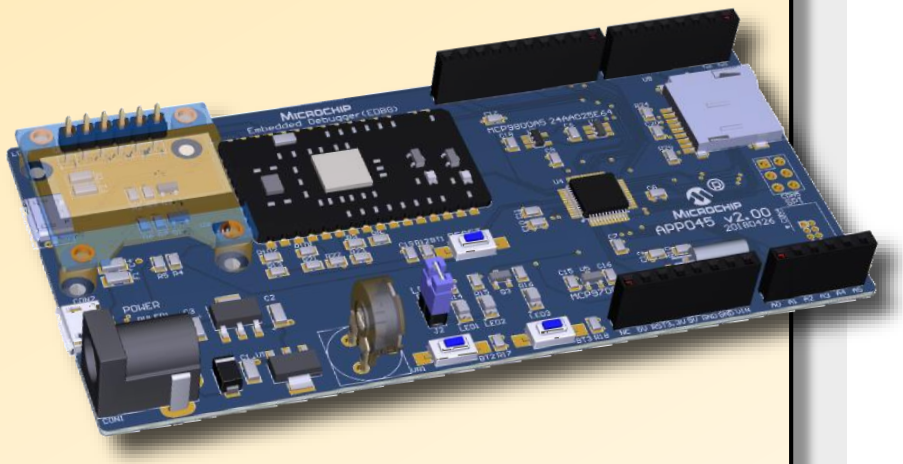
to Generate Code and save changes to MHC configure file.

Lab14 TCC NPWM

Step 5

a Add code segment to your main.c

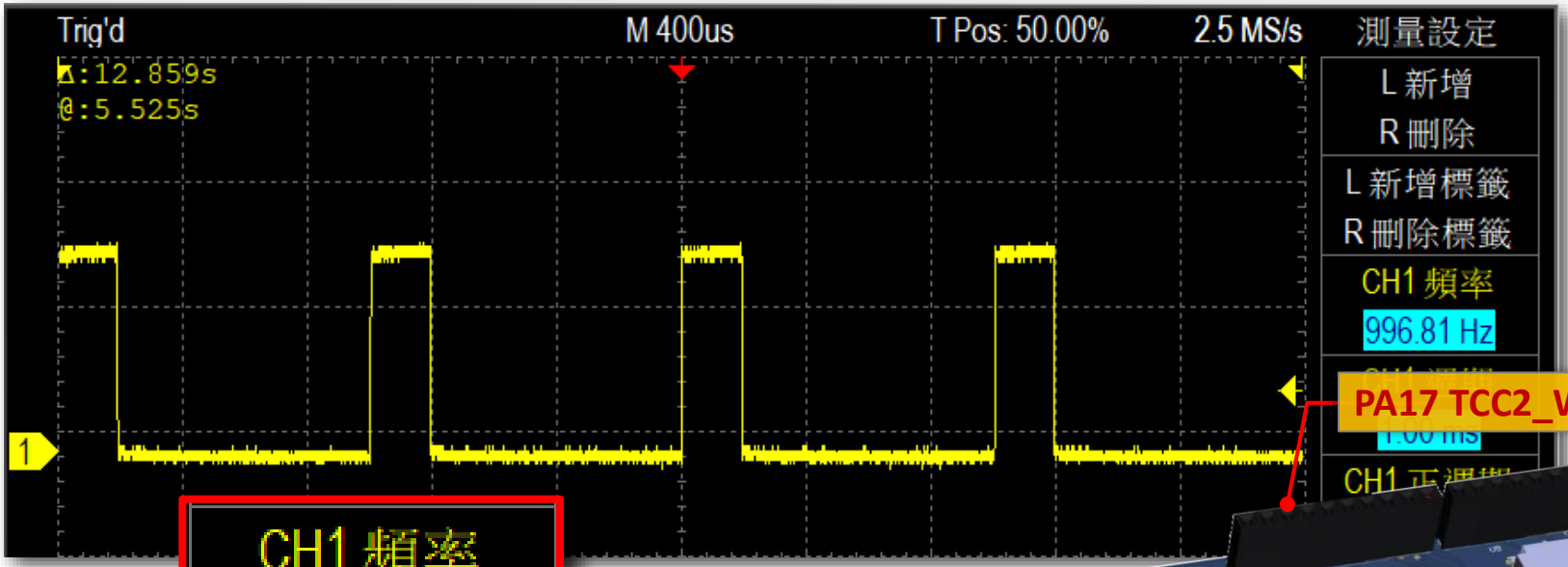
```
…

int main (void)
{
   …

   // TODO 14.01
   TCC2_PWMStart();

   while(1)
   {
      …
   }

}
```



b Program firmware to target board then observe result.

Lab14 TCC NPWM

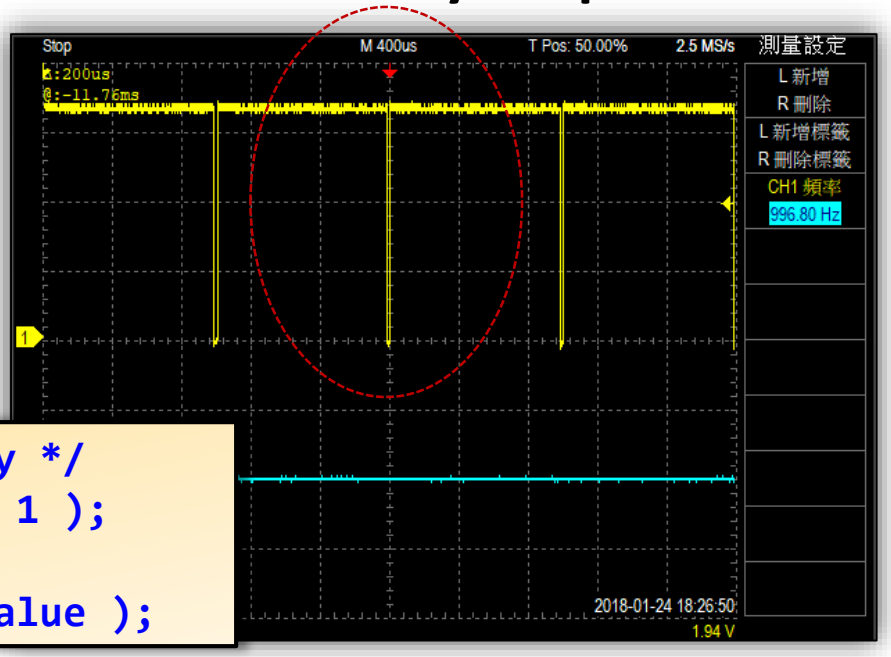
Result



CH1 頻率
996.81 Hz
CH1 週期
1.00 ms
CH1 正週期
19.14%

100% Duty Issue

- How to generate 100% duty PWM ?
  - Rule Review
    1.TCCx Counter = CCx -> Output Set to low
    2.TCCx Counter = PER -> Output Set to high
- This means CCx = PER must be set for 100% duty output.
  Which rule be executed ?
  - Result :
    Glitch !
- To prevent this issue,
  set CCx > PER for 100% duty.




```
if( DutyValue >= PER ) /* 100% duty */
    TCC2_PWM16bitDutySet( 1, PER + 1 );
else
    TCC2_PWM16bitDutySet( 1, DutyValue );
```

Lab15 TCC NPWM ADC Duty Control

- **Try use VR1 to adjust PWM duty for LED3 dimming control.**
  - VR Value = 0 -> PWM Duty 0%, VR Value = 4095 -> Duty 100%

```
if( ADC_IsCompleted )
{
  // TODO 15.01
  if (ADC_Result[0] >= 4095)
    TCC2_PWM16bitDutySet( 1, TCC2_PWM16bitPeriodGet()+1 );
  else
    TCC2_PWM16bitDutySet( 1,
                          (uint32_t)ADC_Result[0] * TCC2_PWM16bitPeriodGet()/4095 );
  ...
}
```

TCC2_ PWM_Peroid = 48000

ADC_Result[0] = 0 ~ 4095

ADC_Result[0] / 4095 = 0.0 ~ 1.0

ADC_Result[0] * TCC2_ PWM_Peroid / 4095 = 0 ~ 48000

Lab16 TCC NPWM Auto Duty Control

- **Use Timer(TC4) to control PWM duty for breathing light effect on LED3. Duty change interval 50ms with stepping 2% ~ 5%.**
  (Duty 0% -> Duty 100% -> Duty 0% -> … repeat)

```
// TODO 16.01
uint8_t Duty = 50;
int8_t DutyDistance = 2;

int main (void)
{ …
  if (TC4_IsOverflow)
  { …
    // TODO 16.02
    Duty += DutyDistance;
    if (Duty >= 100 || Duty <= 0)
      DutyDistance = -DutyDistance;

    if (Duty >= 100)
      TCC2_PWM16bitDutySet( 1, TCC2_PWM16bitPeriodGet() + 1 );
    else
      TCC2_PWM16bitDutySet( 1, ((uint32_t)Duty * TCC2_PWM16bitPeriodGet()) / 100 );
    …
```