MICROCHIP

Regional Training Centers

Section 6
Timer

Timer or Counter ?

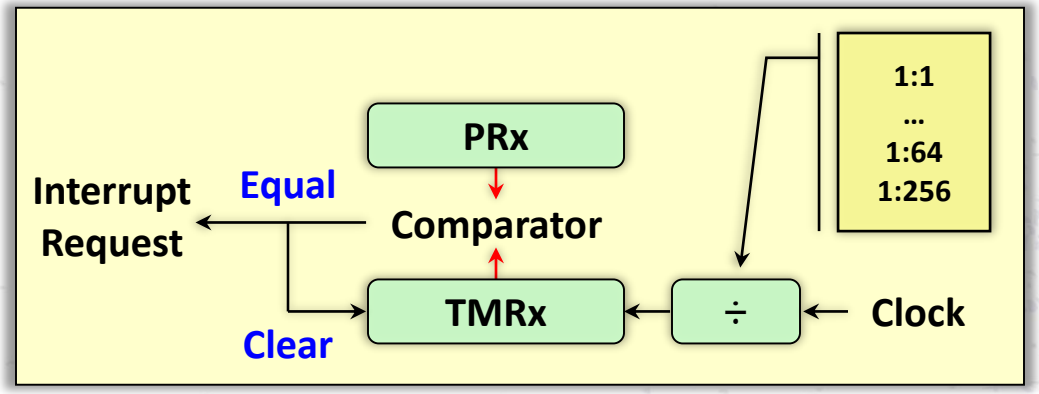
- **Counter or Timer is a clock counter, use to count how many clock into module after start.**
- **There are two kinds of generally called Timer or Counter. In fact, it's same.**
  - The Input Clock unknown : Just know how many count, called Counter.
  - The Input Clock known : Can use count and clock 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.

PIC24FJGB Timer

- The PIC24FJGB provide five Timer module
- The Timer1 module is a 16 Bits timer which can serve as the time counter for the Real-Time Clock (RTC), or operate as a free-running, interval timer/counter. Timer1 can operate in Timer, Counter and Gate Timer mode.
- The Timer2/3 and Timer4/5 modules are 32 Bits timers, which can also be configured as four independent. 16 Bits timers with selectable operating modes. Timer2/3/4/5 can operate in Timer, Counter and Gate Timer mode.

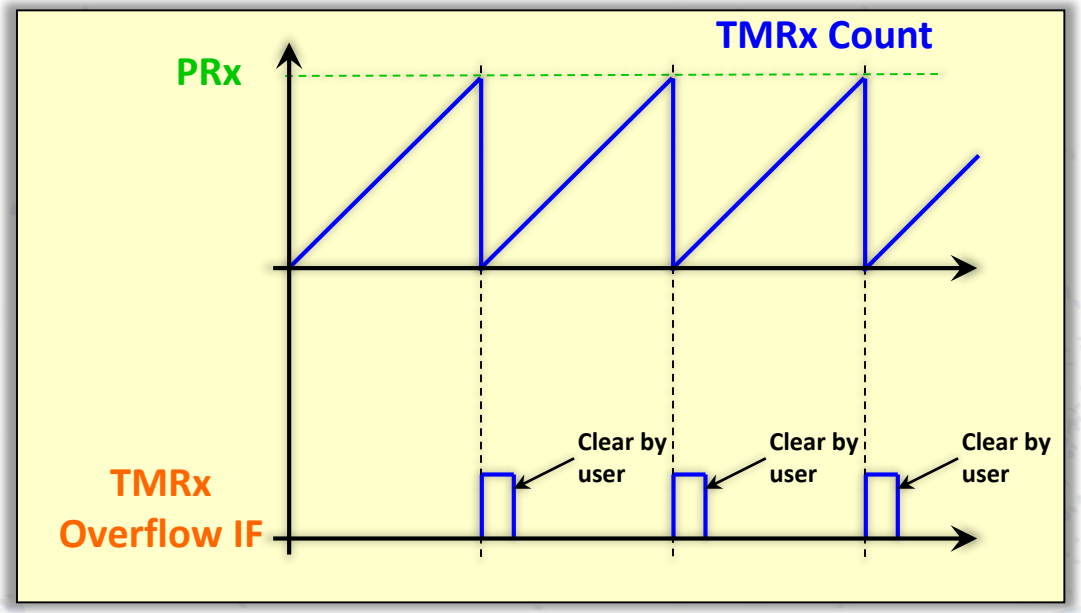
Timer/Counter Operation Mode

- Below block diagram, Just a concept.
- Clock into TMRx after prescaler. Comparator continuously compares the values of TMRx and PRx.
- Asserts an interrupt request while TMRx and PRx equal and then clear TMRx to zero.
- You can fill a value to PRx to determine period you want.



Timer/Counter Operation Mode

- TMRx increase by input clock source.
- TMRx clear automatically when counter TMRx = PRx
- set the TxIF interrupt flag.
- interrupt will be generated if enabled.



Prescaler

- Timer 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 PRx must be set to 2,000,000 (2,000,000 * 0.25uS = 500mS).
  However, this value has exceeded the acceptable range of PRx.
  At this point you can set the prescaler to reduce the count value to expand the count range.

$$\left(2^n - 1\right) \le PRx = \frac{Peroid}{Clock \times Scale}$$

$$\times \; PRx = \frac{500mS}{0.25uS \times 1}, PRx > 65535$$

$$\surd \; PRx = \frac{500mS}{0.25uS \times 64}, PRx < 65535$$

Lab4 Timer1 Polling

Lab4 Timer1 Polling

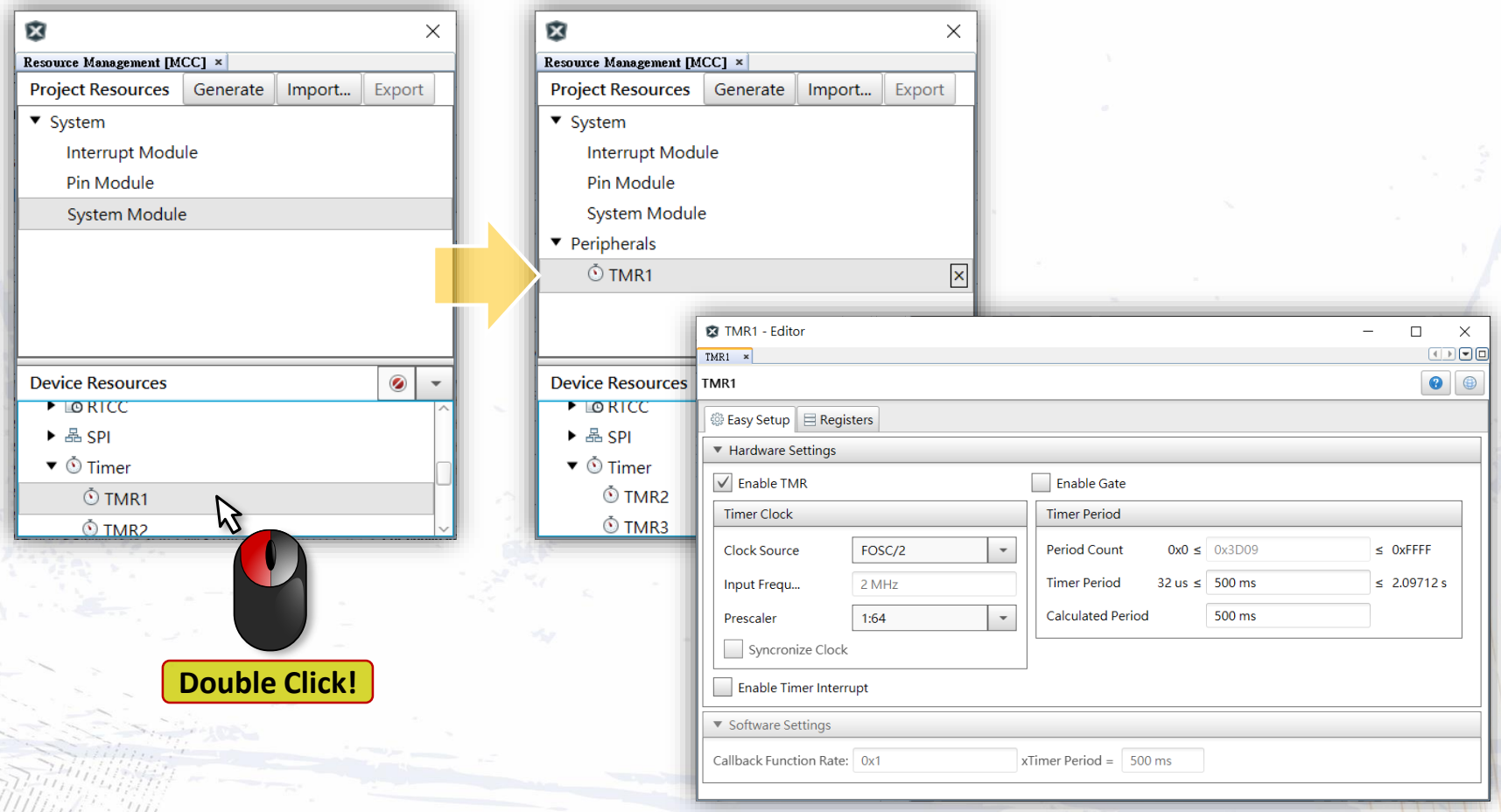
- Try to use Timer1 to replace software delay to control LEDs continues toggle.
- The Timer1 setup to timer mode, timer period is 500mS.

Lab4 Timer1 Polling Hints

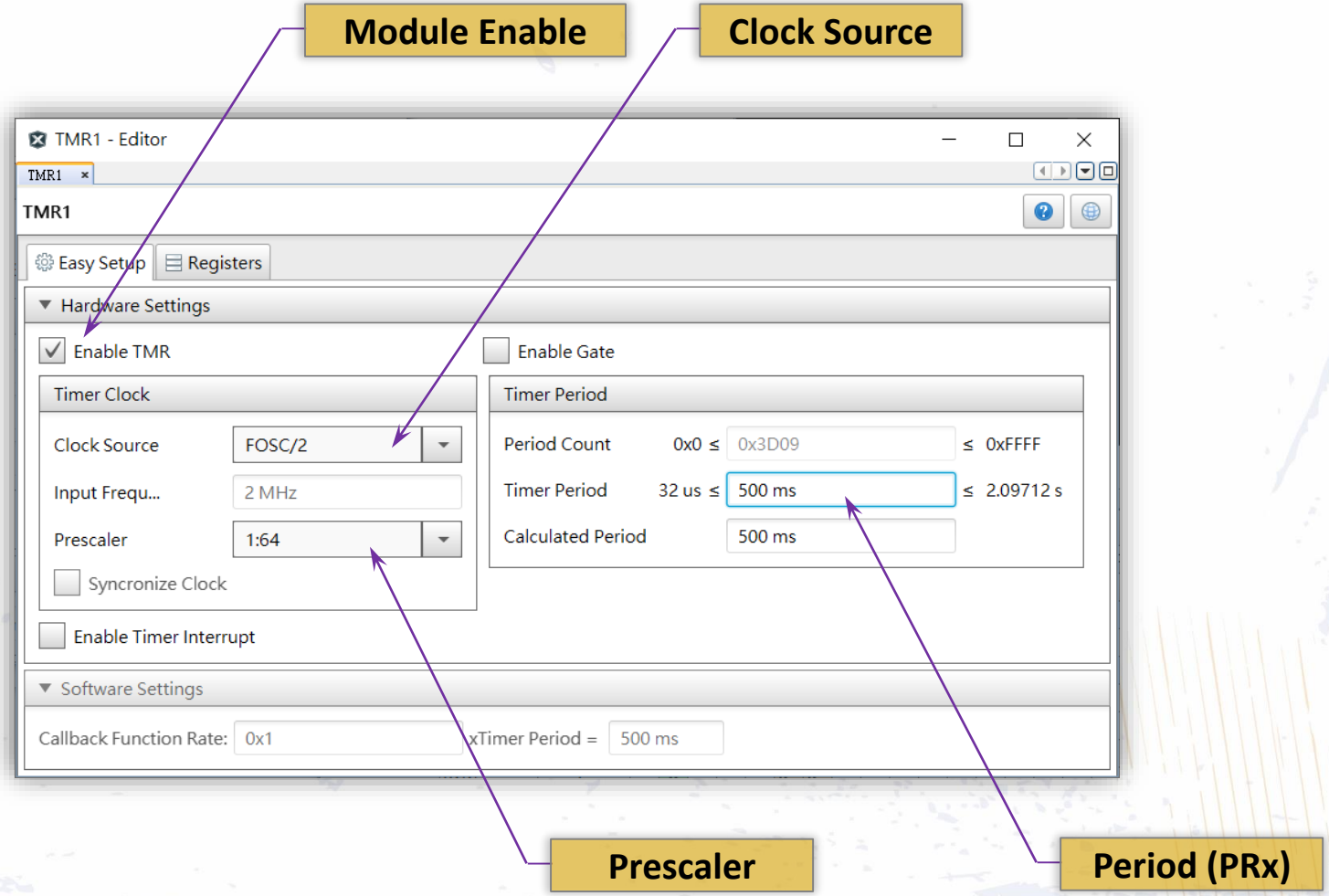
- **At this time we not to enable interrupt, so we need polling timer interrupt flag to measure timer period.**
- **Interrupt Flag must clear manually.**
  E.g. : if(_T1IF == 1) { _T1IF = 0; …; }
- **MCC provide timer status maintain & check functions,**
  - // Status maintain function, must execute at main loop or ISR.
    void TMR1_Tasks_16BitOperation(void);
  - // Status Check functions.
    bool TMR1_GetElapsedThenClear(void);
    int TMR1_SoftwareCounterGet(void);
    void TMR1_SoftwareCounterClear(void);

Lab4 Timer1 Polling Hints

- Add Timer resource for Device Resources



Lab4 Timer1 Polling Hints



Lab4 Timer1 Polling Hints

Timer Polling Example (Base on MCC Function)

```
while (1)
{
   // Check Timer 1 Status.
   if (TMR1_GetElapsedThenClear( ) )
   {
      // LEDs Toggle;
   }

   // maintain function, must execute at main loop.
   TMR1_Tasks_16BitOperation( );
}
```

Lab4 Timer1 Polling

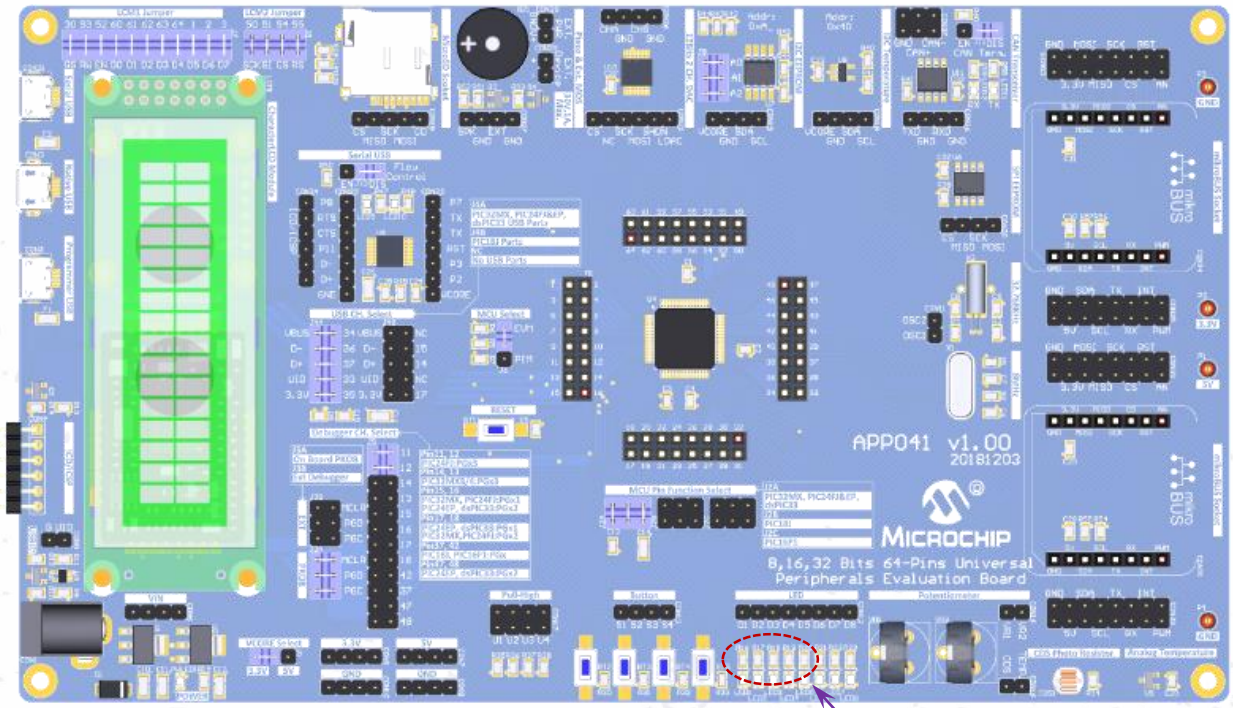
Lab4 Timer1 Polling

- Try to initial Timr1 to control LEDs continues toggle.
- Timer1 setup to timer mode, timer period is 500mS.

- **Let’s go!**

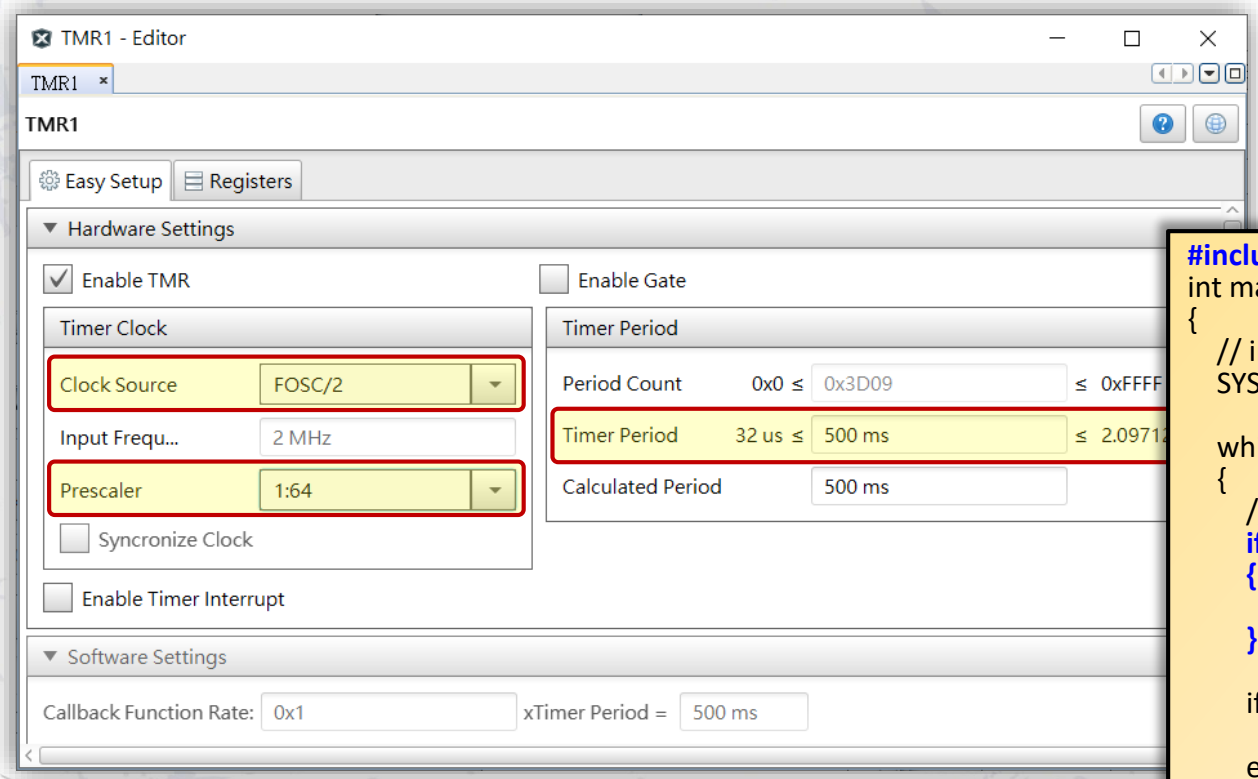
Lab4 Timer1 Polling

Result



LEDs Control by TMR1 Polling,
Toggle every 500ms.

Lab4 Timer1 Polling
MCC's Setting & Code Example




```
#include "mcc_generated_files/tmr1..h"
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();

    while (1)
    {
        // Add your application code
        if(TMR1_GetElapsedThenClear( ))
        {
            ...
        }

        if(S1_GetValue())
            D8_SetLow();
        else
            D8_SetHigh();

        TMR1_Tasks_16BitOperation( );
    }
    return -1;
}
```

Lab5 Multi-Timer Polling

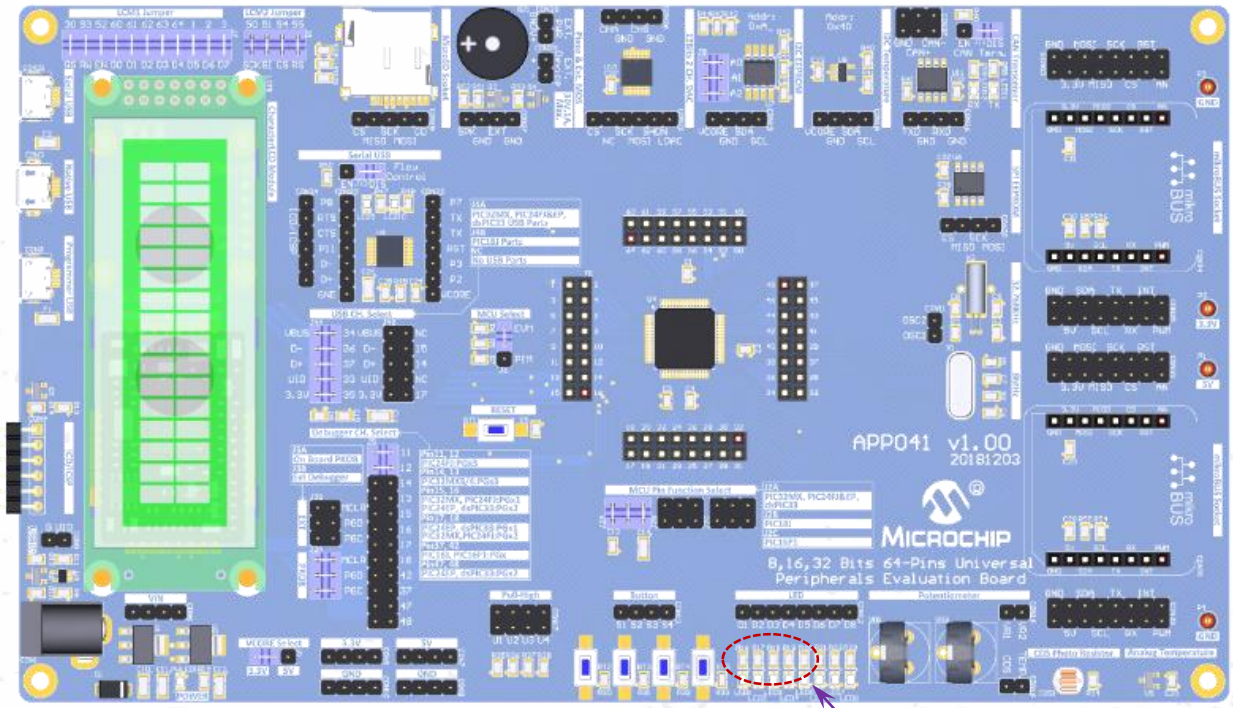
Lab5 Multi-Timer Polling

- Try to add Timer2 to control LEDs continues toggle.
- The Timer2 setup to timer mode, timer period is 1S.
- Timer1 control D1, D2, toggle period -> 500mS
  Timer2 control D3, D4, toggle period -> 1S

- **Let's go!**

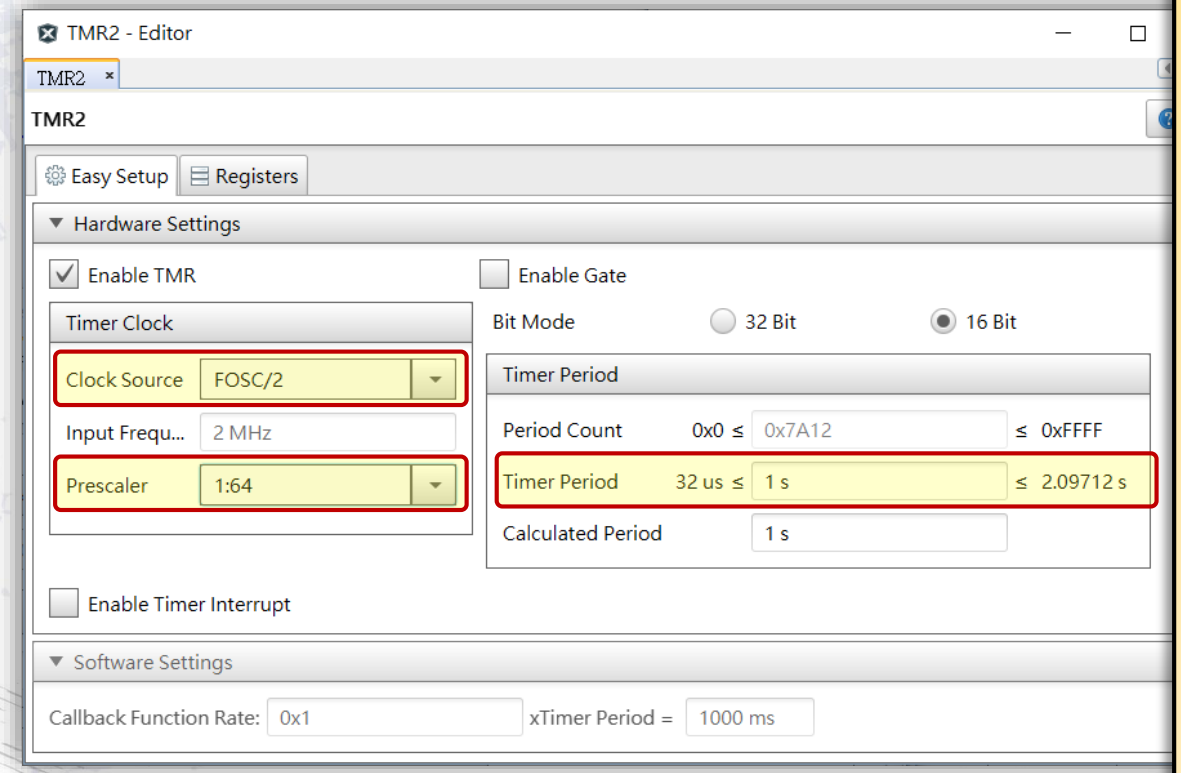
Lab5 Multi-Timer Polling

Result



D1,D2 Control by TMR1 Polling,
Toggle every 500ms.
D3,D4 Control ty TMR2 Polling,
Toggle every 1s.

Lab5 Multi-Timer Polling
MCC's Setting & Code Example




```
#include "mcc_generated_files/tmr1.h"
#include "mcc_generated_files/tmr2.h"
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();

    while (1)
    {
        // Add your application code
        if(TMR1_GetElapsedThenClear( ))
        {
            ...
        }

        if(TMR2_GetElapsedThenClear( ))
        {
            ...
        }

        if(S1_GetValue())
            D8_SetLow();
        else
            D8_SetHigh();

        TMR1_Tasks_16BitOperation( );
        TMR2_Tasks_16BitOperation( );
    }
    return -1;
}
```