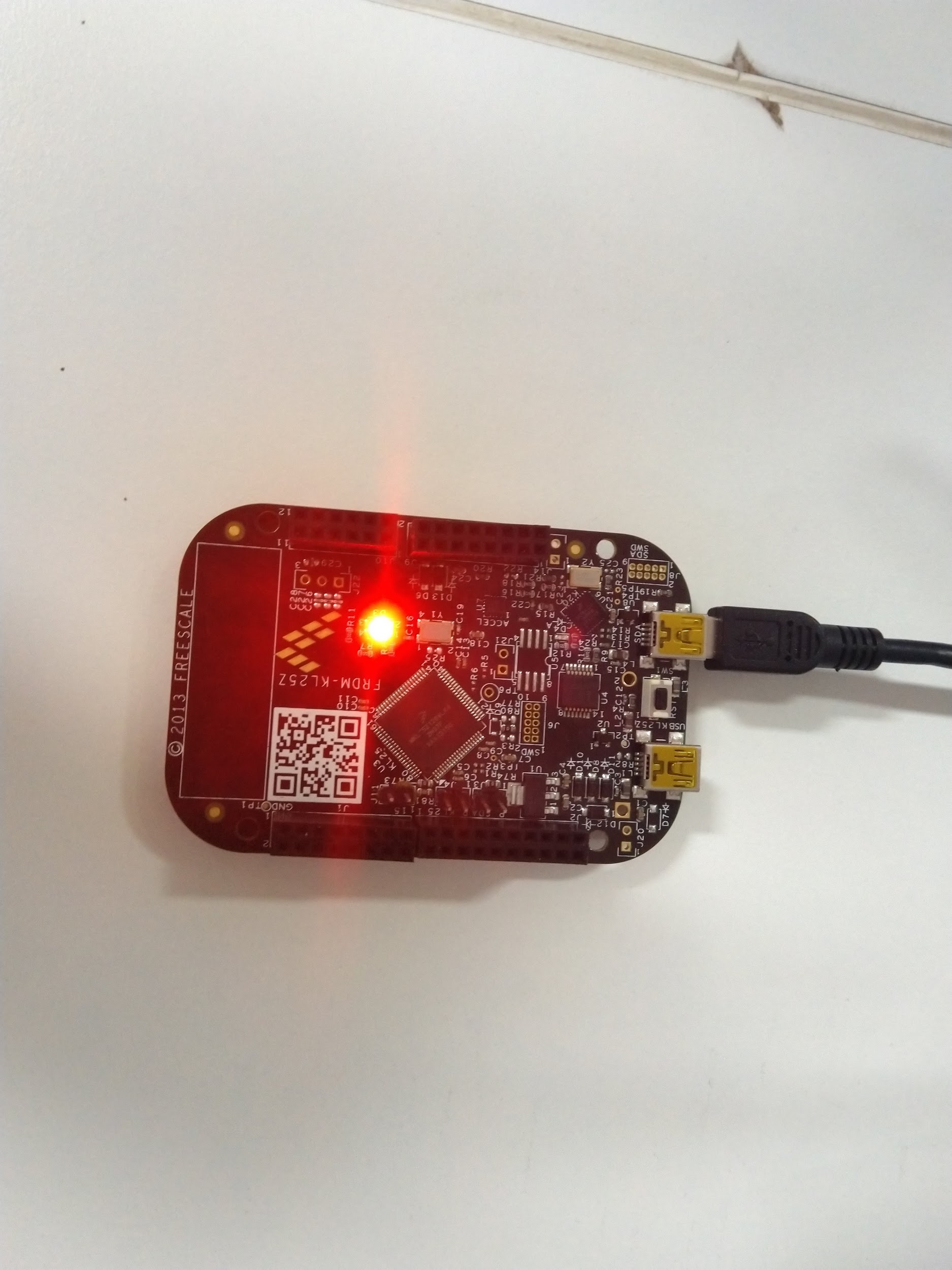
**Lab Report**

****

**Embedded Systems**

**Himanshu Rai**

**111601032 4th October, 2019**

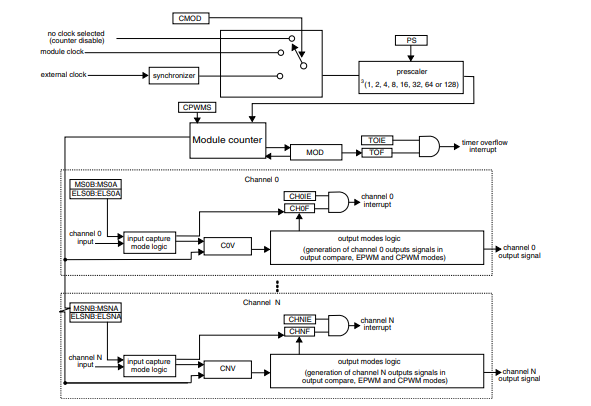
**Objective**

Using Timer/PWM module (TPM) to make the lights on FRDM boards blink.

**Theory**

**Timer/PWM module**

The TPM (Timer/PWM Module) is a two to eight channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low power modes.



**Figure -** TPM block diagram

**Using TPM on FRDM-KL25Z**

We would be using TPM0, so at first we need to set it up before using it. The following steps describes the setup process.

* First of all we need to disable the clock gating (enable the clock) for the *TPM* module. For this, the 24th bit in the *SCGC\_6 (system clock gating control register 6)* must be set.
* Next we must configure the clock source for *TPM.* We would be using *MCGFLLCLK* or *MCGPLLCLK/2 clock.* For this we must set the 25th and 24th bit in *SIM\_SOPT2 register (system options register)* to 01.
* Before configuring *TPM0,* we must disable its counter. For this the 4th and 3rd bit in *TPM0\_SC* (*status and control) register* must be set to 00.
* We know that

Frequency of overflow = Timer clock frequency / (1 + MOD)

We have,

frequency of overflow = 100Hz

timer clock frequency = 21MHz

So, MOD value would be 20999. We must set this value in *TPM0\_MOD* register.

* We need to use *edge-aligned* PWM with *high-true pulses* (clear output on match, set output on reload). For this TPM0\_C1SC[5:2] should be set as 1010. Note that all the bits must be set at once, because some configurations are not allowed, which may lead to error if we do bit by bit setting.
* Next we enable the clock (LPTPM counter increments on every LPTPM counter clock) and select prescale factor to 1 by setting *TPM0\_SC register* to 8 (decimal).
* Finally we will activate PORTD by setting 12th bit in *SIM\_SCGC5 register* and use *TPM0\_CH1 (channel 1 of TPM0),* through PORT D1, by setting its *PCR’s (pin control register),* 10th to 8th bits to 100.

The above steps sets up the TPM0 module to be used through PORT D1, and light up the LED. To change its brightness we can vary the duty cycle through *CnV register (channel (n) value).*

**Procedure**

The above mentioned steps is written in the form of a function that initialises the TPM module. Another function is written to vary the duty cycle for the TPM.

The initialisation function is called once, at the beginning of the program, then at regular intervals the function to vary the duty cycle is called which makes the LED to blink.

The source code is at the end of the report.

**Results**

The desired outputs were obtained and demonstrated during the lab.

**Source Code**

*#include<MKL25Z4.h>*

**void** init\_TPM**()** **{**

*// we would be using TPM0, channel 0 (Timer/PWM module)*

*// disabling clock gating (enabling clock) for TPM0*

SIM**->**SCGC6 **|=** **(1** **<<** **24);**

*// selecting clock source for TPM counter clock as MCGFLLCLK clock or MCGPLLCLK/2*

*// set SIM\_SOPT2[25:24] to 01*

SIM**->**SOPT2 **|=** **(1** **<<** **24);**

SIM**->**SOPT2 **&=** **~(1** **<<** **25);**

*// first disable the TPM0 counter*

*// TPM0\_SC[4:3] = 00*

TPM0**->**SC **&=** **~(3** **<<** **3);**

*// (MOD + 1) \* frequency\_of\_overflows = timer\_clock\_frequency*

*// frequency\_of\_overflows = 100Hz*

*// timer\_clock\_frequency = 21MHz*

TPM0**->**MOD **=** **20999;**

*// configuring the channel for Edge-aligned (TPM0\_C1SC[5:4] = 10) PWM High-true pulses*

*// (TPM0\_C1SC[3:2] = 10) (clear Output on match, set Output on reload)*

*// use TPM0->CONTROLS[1].CnSC or TPM0\_SC to configure TMP0\_C1SC register*

*// note that all the values must be set at one go*

**int** tmp **=** TPM0**->**CONTROLS**[1].**CnSC**;**

tmp **|=** **(1** **<<** **5);**

tmp **&=** **~(1** **<<** **4);**

tmp **|=** **(1** **<<** **3);**

tmp **&=** **~(1** **<<** **2);**

TPM0**->**CONTROLS**[1].**CnSC **=** tmp**;**

*// configuring for 75% duty cycle (1 means off and 0 means on)*

TPM0**->**CONTROLS**[1].**CnV **=** TPM0**->**MOD **/** **4;** *// (TPM0->MOD / 4)*

*// see manual for further details*

TPM0**->**SC **=** **0**b01000**;**

*// activating PORTD*

SIM**->**SCGC5 **|=** **1** **<<** **12;**

*// configuring PORTD1 for TPM0\_CH1 (PORTD\_PCR1[10:8] = 100)*

PORTD**->**PCR**[1]** **|=** **(1** **<<** **10);**

PORTD**->**PCR**[1]** **&=** **~(3** **<<** **8);**

**}**

**void** decrease\_brigtness**()** **{**

**int** interval **=** **10;**

**if(**TPM0**->**CONTROLS**[1].**CnV **>=** TPM0**->**MOD **-** interval**)**

TPM0**->**CONTROLS**[1].**CnV **=** TPM0\_MOD **/** **4;**

**else** TPM0**->**CONTROLS**[1].**CnV **+=** interval**;**

**}**

**void** delay**(int** t**)** **{** **while(**t**--);** **}**

**int** main**()** **{**

SystemCoreClockUpdate**();** *// updating the clock from PLL*

init\_TPM**();**

**while(1)** **{**

decrease\_brigtness**();** delay**(10);**

**}**

**}**