// Timer Programming using PR registers without passing parameter technique

```
#include "lpc17xx.h"

void timer0_init(void);
void delayms(void);


int main ( )
{
timer0_init();
LPC_GPIO0->FIODIR = 0xffffffff;
  while(1)
{
  LPC_GPIO0->FIOSET = 0xffffffff; //LEDs ON
  delayms(); //1000 milliseconds
  LPC_GPIO0->FIOCLR = 0xffffffff;  //LEDs OFF
  delayms();  //1000 milliseconds
}

}
void timer0_init(void)
 {
LPC_SC->PCONP |=(1<<1); //enable POWER to TIM0
LPC_SC->PCLKSEL0 &=~(3<<2);//Pclk =system clk/4 i.e. 100MHz/4 = 25MHz
LPC_TIM0->CTCR  = 0;    //timer mode
LPC_TIM0->PR= 1<<20; //for 1ms
 }

void delayms(void)
{
LPC_TIM0->TCR = 2; //reset timer
LPC_TIM0->TCR = 1;    //enable timer
while (LPC_TIM0->TC < 1);
}
```

<u>Calculation of Pre-scale value</u>

Time period of one clock cycle -> Tpclk = 1/pclk
Timer rate or Timer resolution -> Trate = PR +1/pclk
From this, we can calculate the PR value for various timer resolutions. For example, if we want 1 milli second resolution on a timer with 25 MHz PCLK, then
PR = $(25 * 10^6 * 1 * 10^{-3}) - 1 = 24999$
OR PR = Pclk/timer rate = 25MHZ/1ms=25MHz/1KHz =25000000/1000 = 25000-1=24999

Logic: **PC = PC + 1 for every Pclk**
When PC = PR, TC = TC + 1

// Timer Programming using PR registers based on passing parameter technique

```
#include "lpc17xx.h"

void delayms(unsigned  int  milliseconds);
void timer0_init(void);

int main ( )
{
timer0_init();
LPC_GPIO0->FIODIR = 0xffffffff;
  while(1)
{
  LPC_GPIO0->FIOSET = 0xffffffff; //LEDs ON
  delayms(1000); //1000 milliseconds
  LPC_GPIO0->FIOCLR = 0xffffffff;  //LEDs OFF
  delayms(1000);  //1000 milliseconds
}
}
void timer0_init(void)
 {
LPC_SC->PCONP |=(1<<1); //enable POWER to TIM0
LPC_SC->PCLKSEL0 &=~(0x3<<3);//Pclk =system clk/4 i.e. 100MHz/4 = 25MHz
LPC_TIM0->CTCR  = 0;    //timer mode
LPC_TIM0->PR= 24999; //for 1ms
 }

void delayms(unsigned int milliseconds)
{
 LPC_TIM0->TCR = 2; //reset timer
 LPC_TIM0->TCR = 1;    //enable timer
 while (LPC_TIM0->TC < milliseconds);
}
```

// Timer Programming using MR registers

```
#include "lpc17xx.h"

void wait (void);

int main (void)
        {
          LPC_SC->PCONP |=(1<<1); //enable POWER to TIM0
          LPC_SC->PCLKSEL0 &=~(0x3<<3);//Pclk =system clk/4 i.e. 100MHz/4 = 25MHz
          LPC_TIM0->MR0 = 24999; //load number in the match register
          LPC_TIM0->MCR = 0x04;                    // stop timer on match //
          LPC_TIM0->PR  = 0x8;             // set prescaler to zero //=8; //prescaclare

          LPC_GPIO0->FIODIR |= (1<<0);         // LEDs on PORT0 are output //
          while(1)
          {
         LPC_GPIO0->FIOPIN ^=(1<<0);
          wait();
           }
            }

        void wait(void)
                 {
                   LPC_TIM0->TCR = 1;   //start the timer
                   while(!(LPC_TIM0->TC == LPC_TIM0->MR0));     //until T0Tc = MR0
                   LPC_TIM0->TCR = 2;  //reset the counter
                   LPC_TIM0->TC = 0;  //make the timer count reg = 0
                 }
```

Logic :

Timer rate = MR0 X pclk in time/PR

// Timer Programming using IR registers

```
#include "LPC17xx.h"
int main (void)
{
        LPC_SC->PCONP |= (1 << 1);     // Power up Timer 0
        LPC_SC->PCLKSEL0 |= (1 << 2);  // Clock for timer = CCLK
        LPC_TIM3->CTCR = 0; // Timer mode
        LPC_TIM0->MR0 = (1<<20);  // Suitable value for LED blinking
        LPC_TIM0->MCR |= (3<<0);  // generate Interrupt and reset timer on Match

        NVIC_EnableIRQ(TIMER0_IRQn);  // Enable timer interrupt
        LPC_TIM0->TCR |= (1 << 0);     // Start timer

        LPC_SC->PCONP |= ( 1 << 15 ); // power up GPIO
        LPC_GPIO1->FIODIR |= (1 << 29); // LED is connected to P1.29

        while(1)
        {
                //do nothing
        }

    }

void TIMER0_IRQHandler (void)
{
  if((LPC_TIM0->IR & 0x01) == 0x01)      // if MR0 interrupt
  {
     LPC_TIM0->IR |= (1 << 0);   // Clear MR0 interrupt flag
     LPC_GPIO1->FIOPIN ^=(1 << 29);     // Toggle the LED
  }
}
```

//Counter programming on CAP0.0 for incrementing

```
#include <stdio.h>
#include "lpc17xx.h"

int main (void)
{
LPC_PINCON->PINSEL3 |=  (3 << 20); //p1.26 as input capture mode   pinsel3 (21-20) 11 =3= cap0.0
LPC_GPIO0->FIODIR |= 0xffffffff;          /* LEDs on PORT0 are output */
LPC_TIM0->CTCR = 1;     //timer as counter
LPC_TIM0->TC = 0;
LPC_TIM0->CCR = 1;       //increments on rising edge
LPC_TIM0->TCR=1; //start counter

 while(1)
 {
   LPC_GPIO0->FIOPIN=LPC_TIM0->CR0;  //output  the count value

    }

   }
```

//Counter programming on CAP0.0 and CAP1.0 for two way counting, one is incrementing another Decrementing

```
#include <stdio.h>
#include "lpc17xx.h"
 uint32_t x,y,z;
int main (void)
{
LPC_PINCON->PINSEL3 |=  (3<<20); //p1.26 as input capture mode   pinsel3 (21-20) 11 =3= cap0.0
LPC_PINCON->PINSEL3 |=  (3 << 4); //p1.18 as input capture mode   pinsel3 (5-4) 11 =3= cap1.0
LPC_GPIO0->FIODIR |= 0xffffffff;          /* LEDs on PORT0 are output */
LPC_TIM0->CTCR = 1;     //timer as counter
LPC_TIM0->TC = 0;
LPC_TIM0->CCR = 1;       //increments on rising edge
LPC_TIM0->PR = 0;  //no pre-scaller
LPC_TIM0->TCR=1; //start counter


LPC_TIM1->CTCR = 1;     //timer as counter
LPC_TIM1->TC = 0;
LPC_TIM1->CCR = 1;       //increments on rising edge  //10 increment on falling edge //11 increment on
both
LPC_TIM1->PR = 0;  //no pre-scaller
LPC_TIM1->TCR=1; //start counter
 while(1)
 {

   x=LPC_TIM0->CR0;     //output the count value
   y=LPC_TIM1->CR0;
          z=x-y;//one way count another way decrease //z=x+y for two way count
   LPC_GPIO0->FIOPIN=z;//z<<16;  //output  the count value

   }

  }
```

// //**Counter programming on CAP0.0 and based on the count value switch on the particular LEDs**

```
#include <stdio.h>
#include "lpc17xx.h"
unsigned int counter_value,x;
int main (void)
{
LPC_PINCON->PINSEL3 |=  (3 << 20); //p1.26 as input capture mode   pinsel3 (21-20) 11 =3= cap0.0
LPC_GPIO0->FIODIR = 0xffffffff; /* LEDs on PORT0 are output */
LPC_GPIO2->FIODIR = 0xffffffff;
LPC_TIM0->CTCR = 1;     //timer as counter
LPC_TIM0->TC = 0;
LPC_TIM0->CCR = 1;       //increments on rising edge
LPC_TIM0->TCR=1; //start counter

 while(1)
 {
          LPC_GPIO0->FIOPIN=LPC_TIM0->CR0;  //output  the count value
          counter_value =LPC_TIM0->TC;// read the count value

          switch(counter_value)
          {
          case (10):
          LPC_GPIO2->FIOSET =(1<<0);
          break;

          case (20):
          LPC_GPIO2->FIOSET =(1<<8);
          break;

          case (30):
          LPC_GPIO2->FIOSET =(1<<16);
          break;

          case (40):
          LPC_GPIO2->FIOSET =(1<<24);
          break;

          case (50):
          LPC_GPIO2->FIOSET =(1<<31);
          break;
                    }
}
}
```

//Timer **programming to generate square wave on port0**

```
#include "lpc17xx.h"

void delayms();
void timer0_init(void);

int main ( )
{
timer0_init();
LPC_GPIO0->FIODIR |= (1<<0);
  while(1)
{
  LPC_GPIO0->FIOPIN ^= (1<<0); //square wave generation
  delayms(); //1000 milliseconds

}
}
void timer0_init(void)
 {
LPC_SC->PCONP |=(1<<1); //enable POWER to TIM0
LPC_SC->PCLKSEL0 &=~(3<<2);//Pclk =system clk/4 i.e. 100MHz/4 = 25MHz
LPC_TIM0->CTCR  = 0;    //timer mode
LPC_TIM0->PR= 3999; //for 1ms

 }

void delayms()
{
LPC_TIM0->TCR = 2; //reset timer
LPC_TIM0->TCR = 1;    //enable timer
while (LPC_TIM0->TC < 1);
}
```