**// 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);

}

**Calculation of Pre-scale value**

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 \* 106 \* 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);

}