Timers in LPC2148

Dr.R.Sundaramurthy

Department of EIE
Pondicherry Engineering College

sundar@pec.edu

Features of Timer in LPC2148

- TimerCounter0 and TimerCounter1 are functionally identical
- Each 32-bit Timer/Counter has a programmable 32-bit Prescaler.
- Counter or Timer operation
- Up to four 32-bit capture channels per timer,
  - that can take a snapshot of the timer value when an input signal transitions.
  - A capture event may also optionally generate an interrupt.
- Four 32-bit match registers that allow:
- – Continuous operation with optional interrupt generation on match.
- – Stop timer on match with optional interrupt generation.
- – Reset timer on match with optional interrupt generation.
- Up to four external outputs corresponding to match registers, with the following
- capabilities:
- – Set low on match.
- – Set high on match.
- – Toggle on match.
- – Do nothing on match.

CTCR : Count Control register

- Used to select Timer/Counter Mode.
- For Timing purpose we use this in Timer Mode.
- When the value of the CTCR is set to 0×0 Timer Mode is selected.

Bit	Symbol	Value	Description	Reset value
1:0	Counter/ Timer Mode		This field selects which rising PCLK edges can increment Timer's Prescale Counter (PC), or clear PC and increment Timer Counter (TC).	00
		00	Timer Mode: every rising PCLK edge	
		01	Counter Mode: TC is incremented on rising edges on the CAP input selected by bits 3:2.	
		10	Counter Mode: TC is incremented on falling edges on the CAP input selected by bits 3:2.	
		11	Counter Mode: TC is incremented on both edges on the CAP input selected by bits 3:2.	

Prescalar

- PR : Prescale Register (32 bit) – Stores the maximum value of Prescale counter after which it is reset.
- PC : Prescale Counter (32 bit) – This register increments on every PCLK(Peripheral clock). This register controls the resolution of the timer.
- When PC reaches the value in PR , PC is reset back to 0 and Timer Counter is incremented by 1.
- Hence if PR=0 then Timer Counter Increments on every 1 PCLK.
- If PR=9 then Timer Counter Increments on every 10th cycle of PCLK.
- Hence by selecting an appropriate prescale value we can control the resolution of the timer.

$$Fout = Fin / (PR + 1)$$

TC : Timer Counter Register (32 bit)

- This is the main counting register.
- Timer Counter increments when PC reaches its maximum value as specified by PR.
- If timer is not reset explicitly(directly) or by using an interrupt then it will act as a free running counter which resets back to zero when it reaches its maximum value which is 0xFFFFFFFF.

Total Counts = 429,49,67,296

TCR : Timer Control Register

- This register is used to enable , disable and reset TC.
- When bit0 is 1 timer is enabled and when 0 it is disabled.
- When bit1 is set to 1 TC and PC are set to zero together in sync on the next positive edge of PCLK.
- Rest of the bits of TCR are reserved.

Bit	Symbol	Description	Reset value
0	Counter Enable	When one, the Timer Counter and Prescale Counter are enabled for counting. When zero, the counters are disabled.	0
1	Counter Reset	When one, the Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of PCLK. The counters remain reset until TCR[1] is returned to zero.	0

MCR : Match Control register

- This register is used to control which all operations can be done when the value in MR matches the value in TC.

- Bits 0,1,2 are for MR0 , Bits 3,4,5 for MR1 and so on. Here is a quick table which shows the usage:
  - Bit 0 : Interrupt on MR0 i.e trigger an interrupt when MR0 matches TC. Interrupts are enabled when set to 1 and disabled when set to 0.
  - Bit 1 : Reset on MR0. When set to 1 , TC will be reset when it matched MR0. Disabled when set to 0.
  - Bit 2 : Stop on MR0. When set to 1 , TC & PC will stop when MR0 matches TC.

T0MCR

Bit	Symbol	Value	Description	Reset value
0	MR0I	1	Interrupt on MR0: an interrupt is generated when MR0 matches the value in the TC.	0
		0	This interrupt is disabled	
1	MR0R	1	Reset on MR0: the TC will be reset if MR0 matches it.	0
		0	Feature disabled.	
2	MR0S	1	Stop on MR0: the TC and PC will be stopped and TCR[0] will be set to 0 if MR0 matches the TC.	0
		0	Feature disabled.	
3	MR1I	1	Interrupt on MR1: an interrupt is generated when MR1 matches the value in the TC.	0
		0	This interrupt is disabled	
4	MR1R	1	Reset on MR1: the TC will be reset if MR1 matches it.	0
		0	Feature disabled.	
5	MR1S	1	Stop on MR1: the TC and PC will be stopped and TCR[0] will be set to 0 if MR1 matches the TC.	0
		0	Feature disabled.	
6	MR2I	1	Interrupt on MR2: an interrupt is generated when MR2 matches the value in the TC.	0
		0	This interrupt is disabled	
7	MR2R	1	Reset on MR2: the TC will be reset if MR2 matches it.	0
		0	Feature disabled.	
8	MR2S	1	Stop on MR2: the TC and PC will be stopped and TCR[0] will be set to 0 if MR2 matches the TC.	0
		0	Feature disabled.	
9	MR3I	1	Interrupt on MR3: an interrupt is generated when MR3 matches the value in the TC.	0
		0	This interrupt is disabled	
10	MR3R	1	Reset on MR3: the TC will be reset if MR3 matches it.	0
		0	Feature disabled.	
11	MR3S	1	Stop on MR3: the TC and PC will be stopped and TCR[0] will be set to 0 if MR3 matches the TC.	0

IR : Interrupt Register

- It contains the interrupt flags for 4 match and 4 capture interrupts.
- Bit0 to bit3 are for MR0 to MR3 interrupts respectively.
- And similarly the next 4 for CR0-3 interrupts.
- when an interrupt is raised the corresponding bit in IR will be set to 1 and 0 otherwise.
- Writing a 1 to the corresponding bit location will reset the interrupt – which is used to acknowledge the completion of the corresponding ISR execution.

T0IR : Interrupt Register

Bit	Symbol	Description	Reset value
0	MR0 Interrupt	Interrupt flag for match channel 0.	0
1	MR1 Interrupt	Interrupt flag for match channel 1.	0
2	MR2 Interrupt	Interrupt flag for match channel 2.	0
3	MR3 Interrupt	Interrupt flag for match channel 3.	0
4	CR0 Interrupt	Interrupt flag for capture channel 0 event.	0
5	CR1 Interrupt	Interrupt flag for capture channel 1 event.	0
6	CR2 Interrupt	Interrupt flag for capture channel 2 event.	0
7	CR3 Interrupt	Interrupt flag for capture channel 3 event.	0

Setting up & configuring Timers

- Set appropriate value in TxCTCR (Count Control Register ➔ Set as Timer)
- Define the Prescale value in TxPR (Prescale Register)

  Set Value(s) in Match Register(s) if required
- Set appropriate value in TxMCR if using Match registers / Interrupts
- Reset Timer – Which resets PR and TC
- Set TxTCR to 0×01 to Enable the Timer when required (Timer Control Register)
- Reset TxTCR to 0×00 to Disable the Timer when required (Timer Control Register)

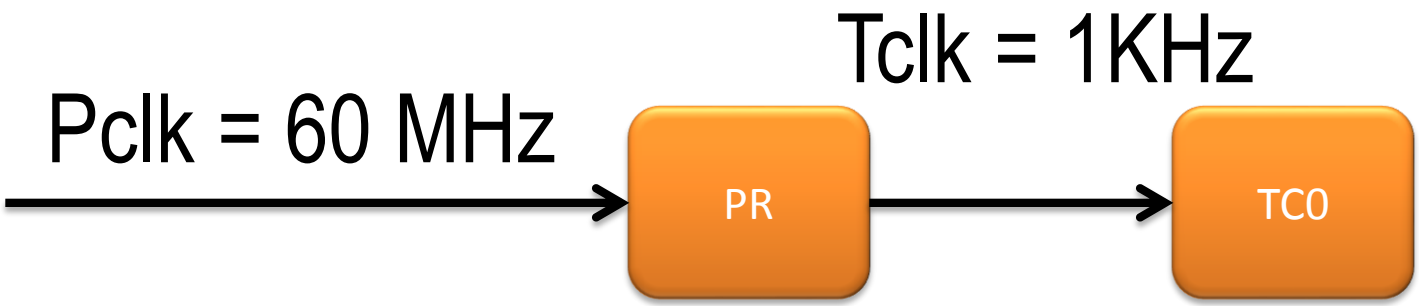
Initialize Timer

```
T0CTCR = 0x00;
T0PR = PRESCALE-1;
T0TCR = 0x02; //Reset Timer
```

Prescale (TxPR) Related Calculations:

$$PR+1 = Fin / Fout$$

For 1 ms Tick

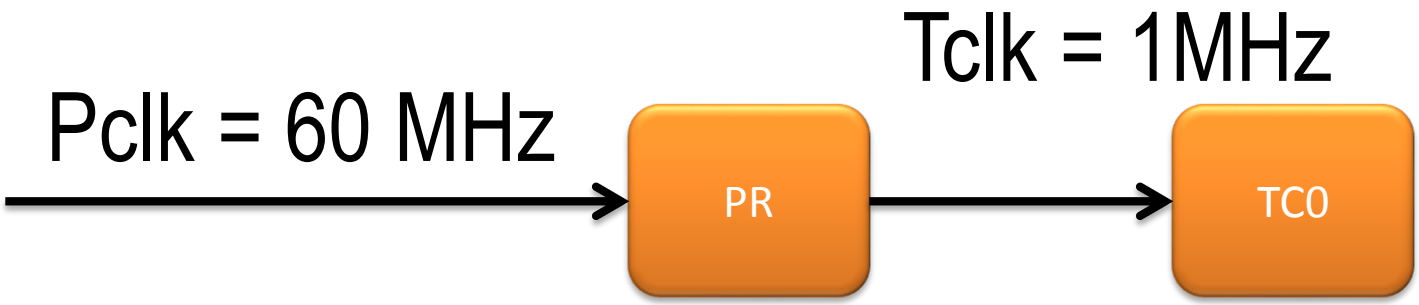


PR+1 = 60MHz/1KHz

= 60,000

PR = 59999

For 1 us Tick



PR+1 = 60MHz/1MHz

= 60

PR = 59

timer.H

- void initPLL(void);
- void delayms(int milliseconds);
- void delayus(int microseconds);

initPLL()

```
    void initPLL(void)
    {
    // To generate 60MHz from 12MHz crystal
    PLL0CFG=0X24;  // SET PSEL=2 AND MSEL=5
    PLL0CON=0X01;  //PLL IS ACTIVE BUT NOT YET CONNECT
    PLL0FEED=0XAA; //FEED SEQUENCE
    PLL0FEED=0X55;
    while((PLL0STAT & 0X400)==0);
//WAIT FOR FEED SEQUENCE TO BE INSERTED
    PLL0CON=0X03;  // PLL HAS BEEN ACTIVE AND BEING CONNECTRD
    VPBDIV=0X01;   // SET PCLK SAME AS FCCLK
    PLL0FEED=0XAA; //FEED SEQUENCE
    PLL0FEED=0X55; //FEED SEQUENCE
    }
```

delayms(unsigned int milliseconds)

- void delayMS(unsigned int milliseconds)

```
//Using Timer0
    {
      T0CTCR = 0x00 ; // configure T0 as Timer
       T0PR = 60000-1 ;
      T0TCR = 0x02; //Reset Timer

       T0TCR = 0x01; //Enable timer

       while(T0TC < milliseconds);
//wait until timer counter reaches the desired delay

       T0TCR = 0x00; //Disable timer
    }
```

delayus(unsigned int microseconds)

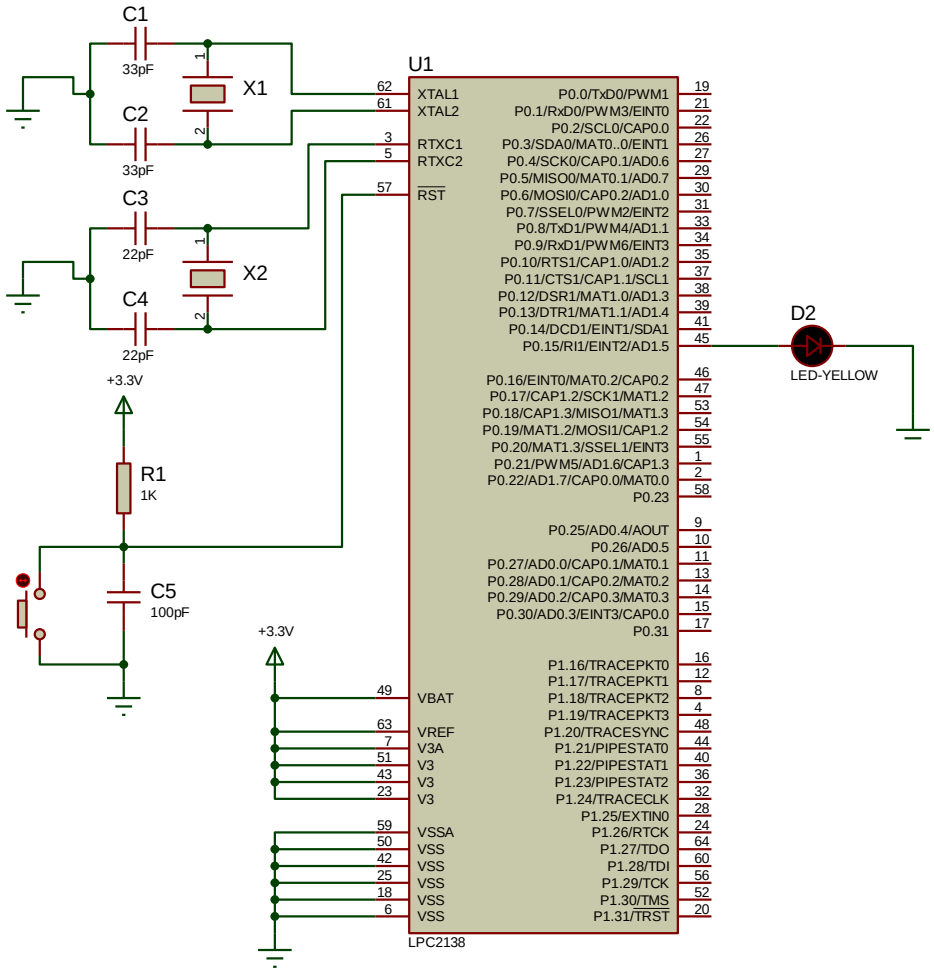
- void delayuS(unsigned int microseconds)

```
//Using Timer0
    {
      T0CTCR = 0x00 ; // configure T0 as Timer
      T0PR = 60000-1 ;
      T0TCR = 0x02; //Reset Timer

       T0TCR = 0x01; //Enable timer

       while(T0TC < microseconds);
//wait until timer counter reaches the desired delay

       T0TCR = 0x00; //Disable timer
    }
```

Example Problems

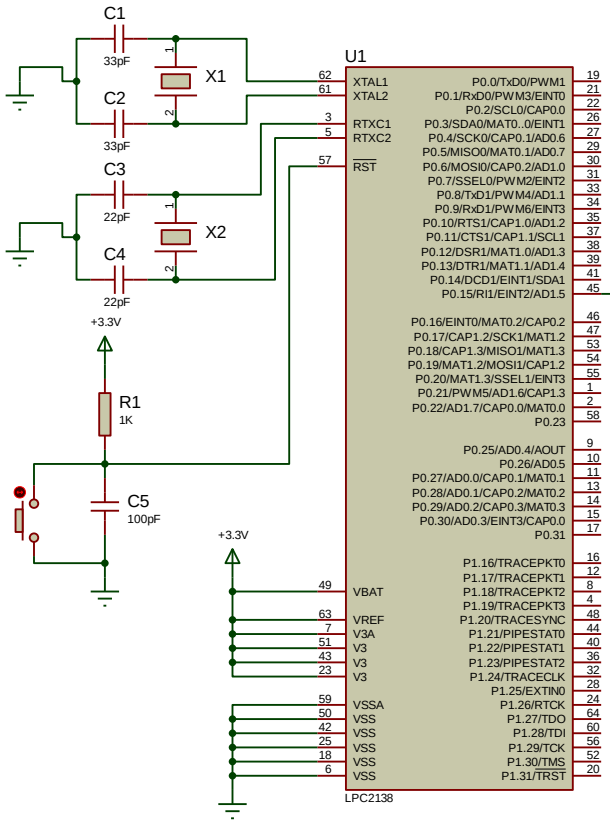
Toggle Pin @ P0.15, Ton = 1Sec, Toff=0.5Sec

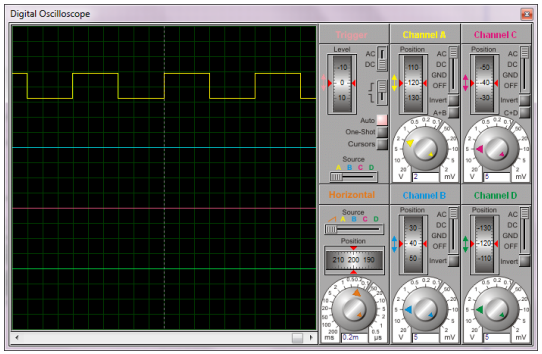
```
#include<LPC214x.h>
#include "GPIO.h"
#include "timer.h"
int main()
{
while(1)
{
     writepin(15,1);
     delayms(1000);
     writepin(15,0);
     delayms(500);

}
}
```



Generate a Square Wave at 1KHz

```
#include<LPC214x.h>
#include "GPIO.h"
#include "timer.h"
int main()
{
initPLL();
while(1)
{
    writepin(15,1);
    delayus(500);
    writepin(15,0);
    delayus(500);

}
}
```





End of Session

sundar@pec.edu