

Power Up Timers

(1) Power up timer

1. Power: In the PCONP register (Table 46), set bits PCTIM0/1/2/3. On reset, Timer0/1 are enabled (PCTIM0/1 = 1), and Timer2/3 are disabled (PCTIM2/3 = 0). Chapter 21, pp 499;
2. Table 46. Power Control for Peripherals register (PCONP - address 0x400F C0C4)

```
int main(void)
{
    /* Power up timer0 */
    LPC_SC->PCONP |= BIT(1);
    /* Clear all interrupts */
    LPC_TIM0->IR = 0x3F;
    /* Set prescaler based on sysclk to get
    1000us minimum interval */
    switch (PCLKSEL_TIMER0(LPC_SC-
    >PCLKSEL0)) {
    case 0x00:
        LPC_TIM0->PR = SystemCoreClock / 4000;
        break;
    }
}
```

Fig. 1

Example: based on table 46, power up Timer 1, write a C-code to realize this task

Bit	Symbol	Description
0	-	Reserved.
1	PCTIM0	Timer/Counter 0 power/clock control bit.
2	PCTIM1	Timer/Counter 1 power/clock control bit.
3	PCUART0	UART0 power/clock control bit.

```
/* Power up timer1 */
LPC_SC->PCONP |= BIT(2);
```

Or

```
LPC_SC->PCONP |= 0x4;
```

How would you power up both Timer 0 and 1?

Clear All Timer INTs

(2) LPC_TIM0->IR Interrupt Register; (3) Set Prescaler

Table 427. Interrupt Register - addresses 0x4000 4000, IR consists of 4 bits for the match interrupts, if interrupt generated then its corresponding bit will set high. chapter 21, pp. 502; you will need to clear it for the next interrupt.

Example: based on table 427, clear TIM3->IR, write a C-code to realize this task

```
int main(void)
{
    /* Power up timer0 */
    LPC_SC->PCONP |= BIT(1);
    /* Clear all interrupts */
    LPC_TIM0->IR = 0x3F;
    /* Set prescaler based on sysclk to get
    1000us minimum interval */
    switch (PCLKSEL_TIMER0(LPC_SC->PCLKSEL0)) {
    case 0x00:
        LPC_TIM0->PR = SystemCoreClock / 4000;
        break;
    }
```

Bit	Symbol	Description
0	MR0 Interrupt	Interrupt flag for match channel 0.
1	MR1 Interrupt	Interrupt flag for match channel 1.
2	MR2 Interrupt	Interrupt flag for match channel 2.
3	MR3 Interrupt	Interrupt flag for match channel 3.

LPC_TIM0->IR |= 0x4;

How to clear both Timer 3 and 4?
Can you change the above c code to the following for the same clear function?

LPC_TIM0->IR = 0x4;

Fig. 1

Timing Waveforms and Prescaler PR

(3) Set Prescaler

Note: 4 important registers, PR, PC, TC, and MR

Two steps: first, PCLKSEL register, Peripheral Clock Selection register, selects Timer0 clock; Section 4.7.3, pp. 57;

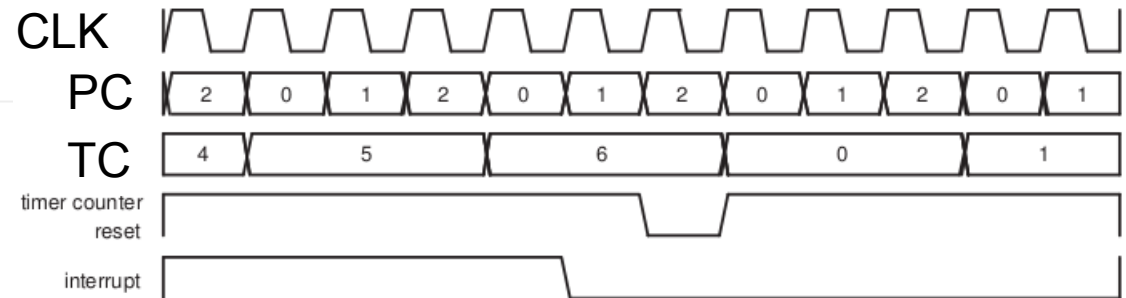
2nd, LPC_TIM0->PR Prescaler register, When the Prescale Counter (PC) is equal to PR, Timer Counter (TC) is incremented by 1;

e.g,

TC is incremented every PR+1 cycles of PCLK.

Bit	Symbol	Description
1:0	PCLK_WDT	Peripheral clock selection for WDT.
3:2	PCLK_TIMER0	Peripheral clock selection for TIMER0.
5:4	PCLK_TIMER1	Peripheral clock selection for TIMER1.

```
int main(void)
{
    /* Power up timer0 */
    LPC_SC->PCONP |= BIT(1);
    /* Clear all interrupts */
    LPC_TIM0->IR = 0x3F;
    /* Set prescaler based on sysclk to
    get 1000us minimum interval */
    switch (PCLKSEL_TIMER0(LPC_SC-
    >PCLKSEL0)) {
    case 0x00:
        LPC_TIM0->PR = SystemCoreClock / 4000;
        break;
    }
}
```



Example:

Given PR = 2, MR = 6; An interrupt generated on match.

(1) at every clock, PC -> PC+1;

(2) when PC = PR, TC -> TC+1 ;

(3) when TC = MR, INT generated.

Fig. 1

Set MR, MCR and TCR

(4) set match register MR and match control register MCR

```
/* Set the interval at Match Control 0 */
LPC_TIM0->MR0 = TIMER_INTERVAL;
/* Enable Match Control 0 */
LPC_TIM0->MCR = BIT(0) | BIT(1);
/* Disable Timer as default */
LPC_TIM0->TCR = 0;
/* Enable Timer IRQ */
NVIC_EnableIRQ(TIMER0_IRQn);
/* Set P2.10 as EINT0 function */
LPC_PINCON->PINSEL4 &= ~GEN_MASK(2, 20);
LPC_PINCON->PINSEL4 |= BIT(20);
/* Enable falling edge trigger for P2.10 */
LPC_GPIOINT->IO2IntEnF |= BIT(10);
/* Edge trigger for EINT0 */
LPC_SC->EXTMODE |= BIT(0);
/* Enable EINT IRQ */
NVIC_EnableIRQ(EINT0_IRQn);

printf("Main Function: Waiting for
interrupt\n");

while(1);

return 0;
}
```

MR values are continuously compared to the Timer Counter value. When matched, action takes place as one of the following three: an interrupt, reset the Timer Counter, or stop the timer. Actions controlled by the settings in the MCR register.

Table 430 (pp. 505). Match Control Register (T[0/1/2/3]MCR - addresses 0x4000 4014

Bit	Symbol	Value	Description
0	MR0I	1	Interrupt on MR0: an interrupt is generated when MR0 matches the value in the TC.
		0	This interrupt is disabled
1	MR0R	1	Reset on MR0: the TC will be reset if MR0 matches it.
		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	Feature disabled.

Timer Counter registers (T0TC – T3TC) is incremented when the prescale counter reaches its terminal count. Chapter 21, pp. 504.