

# CENG4480

## Lab 3: Sound Record

**Yuzhe Ma**

2016 Fall

[yzma@cse.cuhk.edu.hk](mailto:yzma@cse.cuhk.edu.hk)



香港中文大學

The Chinese University of Hong Kong

# Overview

Timer Configuration

UART Configuration

ADC Configuration

Autocorrelation

Supplementary



# Overview

Timer Configuration

UART Configuration

ADC Configuration

Autocorrelation

Supplementary



# PC & PR

## Prescale Counter (PC)

32-bit register which controls division of PCLK. It is incremented on every PCLK.

## Prescale Register (PR)

32-bit register which specifies the maximum value of PC.

- ▶ Once PC reaches the value in PR, it will be reset.
- ▶ **TODO:** Please set the value of PR of timer 0(**T0PR**) in function `void init_timer(void)`.



# TC & MR

## Timer Counter (TC)

32-bit register which is incremented once PC reaches its terminal count.

## Match Register (MR)

32-bit register whose value is continuously compared to the TC value. When the two values are equal, actions will be triggered automatically.

- ▶ MR0-MR3 correspond to channel0-channel3.
- ▶ In function `void init_timer(void)`, `TOMR0=691`, which means the MR0 of timer 0 is set to 691.



# Relationships among PC & PR & TC

- ▶ When PC reaches the value stored in PR, TC is incremented
- ▶ PC is reset on the **next** PCLK.
- ▶ When PR = 0, TC increments on every PCLK



# Relationships among PC & PR & TC

- ▶ When PC reaches the value stored in PR, TC is incremented
- ▶ PC is reset on the **next** PCLK.
- ▶ When PR = 0, TC increments on every PCLK
- ▶ When PR = 1, TC increments every **2** PCLKs



Considering that  $PR = 0$ ,  $MR = 691$ ,  $PCLK$  is 13.824 MHz.  
Calculate interrupt frequency. How about when  $PR = 2$ ?



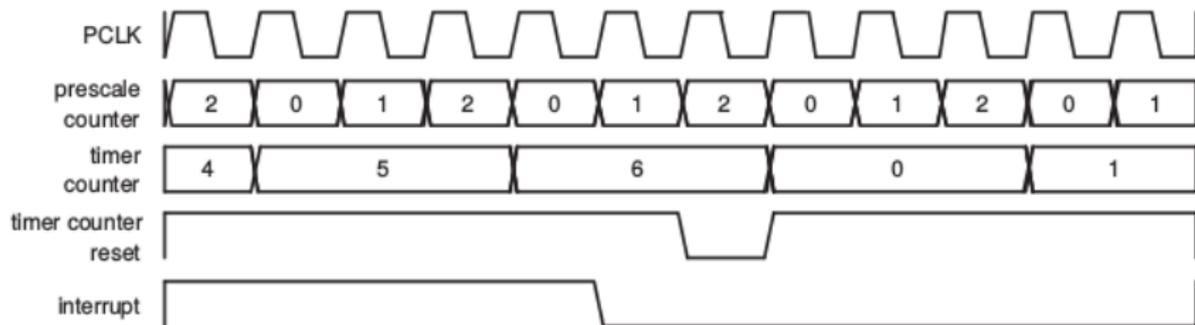
# Interrupt Register (IR)

Bit	Symbol
0	MR0 Interrupt
1	MR1 Interrupt
2	MR2 Interrupt
3	MR3 Interrupt
4	CR0 Interrupt
5	CR1 Interrupt
6	CR2 Interrupt
7	CR3 Interrupt

- ▶ Name: **T0IR** for timer 0, **T1IR** for timer 1. (This rule also applies to other timer registers)
- ▶ Setting corresponding IR bit to 1 will reset the interrupt.
- ▶ For example, '**T0IR = 0x01**' will write "1" to bit [0], which will reset the MR0 interrupt.



# An Example



The relation between PR, PCR and TC, with PR=2, MR=6



# Match Control Register (MCR)

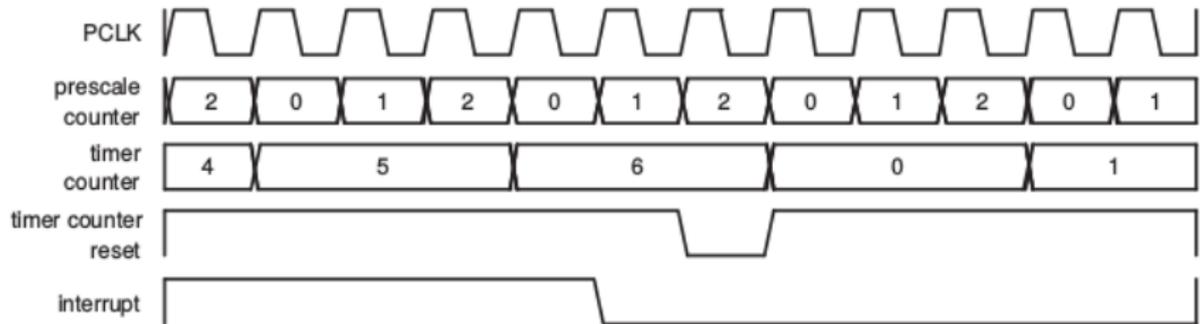
- ▶ MCR: 16-bit register which is used to control the operations to be performed when “Match Event” happens.
- ▶ Bit [0] - Bit [2] correspond to MR0. Bit [3] - Bit [5] correspond to MR1 and so on.
- ▶ Bit [13] - Bit [15] are not defined.

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 stop and TCR[0]=0 if MR0 matches the TC.
		0	Feature disabled.

**TODO:** set MCR of timer 0 in the function `void init_timer(void)`.



Considering the figure of Timer 0 above, what is the corresponding MCR value?



The relation between PR, PCR and TC, with PR=2, MR=6



# Overview

Timer Configuration

UART Configuration

ADC Configuration

Autocorrelation

Supplementary



# Divisor Latch Registers: Baudrate Calculation

- ▶ The U0DLL and U0DLM registers together form a 16 bit divisor.
- ▶ U0DLL contains the lower 8 bits of the divisor.
- ▶ U0DLM contains the higher 8 bits of the divisor.

If Fractional Divider Register(refer to datasheet) is not set, the baudrate can be calculated by

$$UART0_{baudrate} = \frac{PCLK}{16 \times (256 \times U0DLM + U0DLL)}$$

In the lab 3, we need to set the baudrate to be 57600. Please set U0DLL and U0DLM in the function `void Init_Serial_A(void)`



# Data Buffer Register & Line Status Register(LSR)

- ▶ Receiver Buffer Register(RBR): contains the oldest received byte.
- ▶ Transmit Holding Register(THR): contains the newest byte can be written via the bus interface.
- ▶ LSR is read-only, and provides status information on the TX and RX blocks.
- ▶ Bit [0] of LSR indicates whether RBR is empty or not.
- ▶ Bit [5] of LSR indicates whether THR is empty or not.



# Data Buffer Register & Line Status Register(LSR)

- ▶ Remember to check the status of RBR/THR before read/send operation.
- ▶ Based on this, fill in the condition of the “while loop” in the function `char getchar(void)` and `void sendchar(char ch)`



# Overview

Timer Configuration

UART Configuration

**ADC Configuration**

Autocorrelation

Supplementary



# A/D Control Register(ADCR)

ADCR is used for setting configuration.

- ▶ Bit [7:0]: SEL field selects which pins are sampled.
- ▶ Bit [15:8]: CLKDIV determines how much the PCLK is divided by. The divided clk is used for AD conversion.
- ▶ Bit [21]: PDN selects the ADC mode.
- ▶ Bit [26:24]: START determines when to start conversion.



# Example

In the function “`unsigned char read_sensor(int channel)`”

- ▶ `ADCR=0x1<<channel`: sets value 0 to the bit [0], which means pin 0 is used for sampling.

- ▶ `ADCR|=0x1200200`: bit [9], bit [21], bit [24] are set to 1, which define

`CLKDIV = 0x02` → sampling rate =  $\frac{PCLK}{2}$

`PDN = 1`, → operational mode

`START = 001`, → start conversion now



# A/D Data Registers(ADDR)

ADDRs have 32-bit which include the ADC result and the ADC completion flags.

- ▶ Bit [15:6]: 10 bits ADC result.
- ▶ Bit [31]: Completion flag. '1' indicates ADC is completed.

Please write codes to extract the ADC result in the same function. For simplicity, you just need to extract bit [15:8] of ADDR in this lab.



# Overview

Timer Configuration

UART Configuration

ADC Configuration

**Autocorrelation**

Supplementary



# Fundamental Frequency

- ▶ The *fundamental frequency*  $f_0$  is the lowest frequency of a periodic waveform.
- ▶ The period of fundamental frequency is  $t_0 = \frac{1}{f_0}$
- ▶ The term “lag” denotes the period expressed in samples:  $j = t_0 \times f_s$ , where  $f_s$  is sampling frequency.



# Fundamental Frequency Detection–Autocorrelation Function

Given a discrete signal  $x_n$  and the mean value  $m$ , autocorrelation  $R$  at lag  $j$  is defined as:

$$R(j) = \sum_n (x_n - m)(x_{n-j} - m)$$

- ▶ The  $x_{n-j}$  can be seen as signal  $x_n$  with a delay  $j$ .
- ▶ The larger  $R$  is, the more they “match”.
- ▶ When  $j = 0$ , autocorrelation reaches the maximum, because they are exactly matched.
- ▶ But  $j = 0$  indicates the signal is not periodic, which is not considered, so we need find  $j > 0$  that maximizes  $R$ .



# Autocorrelation

$$R(j) = \sum_n (x_n - m)(x_{n-j} - m)$$

You are going to calculate the  $R$  based on the codes in the report. Try these 2 ways and to see if there are any differences:

1.  $m$  = mean value of  $X$
2.  $m$  = minimum value of  $X$



# Overview

Timer Configuration

UART Configuration

ADC Configuration

Autocorrelation

Supplementary



# CCLK & PCLK

- ▶ CCLK: ARM processor clock frequency. The value is defined by oscillator output frequency  $F_{OSC}$  and register **PLLCFG**
- ▶ PCLK: peripheral clock. The value is defined by CCLK and register **APBDIV**.
- ▶ In lab 3,  $PCLK = \frac{F_{OSC} \times 5}{4}$ .



# Reminder

- ▶ Note that the initial value of the register is set by the file “**startup.s**”, which is the initialization file of the software you use.
- ▶ The initial value may be a little different from its reset value.

