Computer Labs: The i8254 Timer/Counter

2º MIEIC

Pedro F. Souto (pfs@fe.up.pt)

October 3, 2012

Lab3: The PC's Timer/Counter

- Write a set of functions:

```
int timer_test_square(unsigned long rate)
int timer_test_int(unsigned long time)
int speaker_test(unsigned long freq, unsigned long time)
```

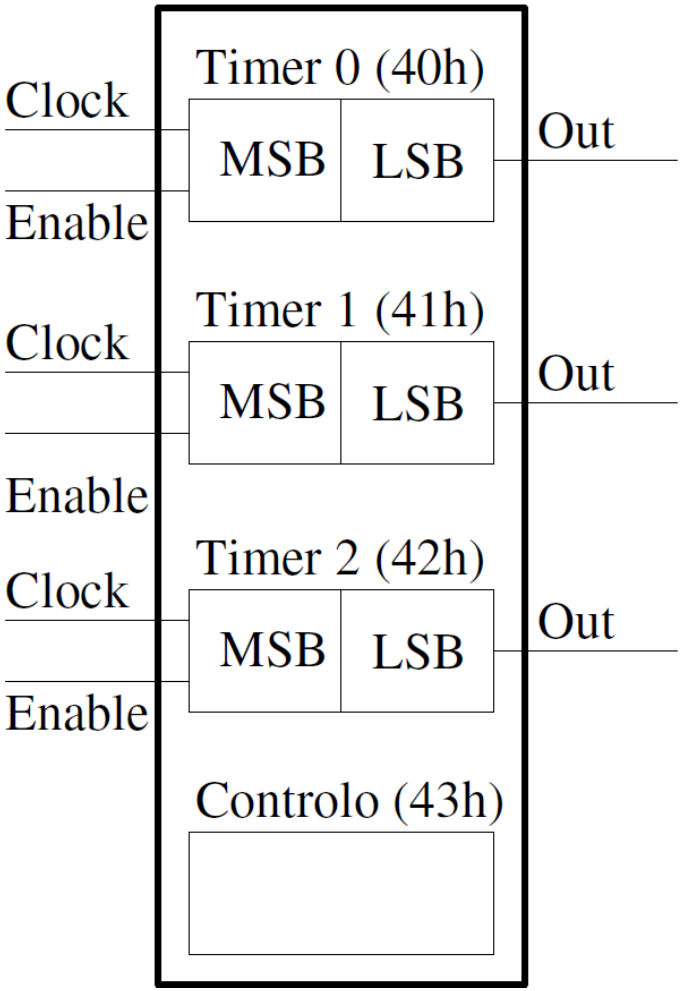
  that require programming the PC's Timer/Counter

- These functions are at a higher level than those of the previous labs
  - The idea is that you design the lower level functions (with the final project in mind)
  - In this lab we have also defined the lower level functions
- What's new?
  - Program an I/O controller: the PC's timer counter (i8254)
  - Use interrupts

The i8254

- It is a programmable timer/counter
  - Each PC has a functionally equivalent circuit, nowadays it is integrated in the so-called south-bridge
  - Allows to measure time in a precise way, independently of the processor speed
- It has 3 16-bit counters, each of which
  - May count either in binary or BCD
  - Has 6 counting modes

i8254 Block Diagram



- Three independent 16-bit counters
  - Ports 40h, 41h and 42h
  - MSB and LSB addressable separately
  - 6 counting modes
- An 8 bit-control register
  - Port `43h`
  - Programming of each counter independently

i8254 Control Word

- Written to the Control Register (`0x43`)

Bit	Value	Function
7,6		**Counter selection**
	00	0
	01	1
	10	2
5,4		**Counter Initialization**
	01	LSB
	10	MSB
	11	LSB followed by MSB
3,2,1		**Counting Mode**
	000	0
	001	1
	x10	2
	x11	3
	100	4
	101	5
0		**BCD**
	0	Binary (16 bits)
	1	BCD (4 digits)

Example

- Timer 2 in mode 3
- Couting value: 1234 = 0x04D2

Control Register: `10111110`

Timer2 LSB `0xD2`

Timer2 MSB `0x04`

i8254 Counting Modes

Mode 0 Interrupt on terminal count – for counting events

- `OUT` goes high and remains high when count reaches 0

Mode 1 Hardware retriggerable one-shot

- `OUT` goes low and remains low until count reaches 0, the counter is reloaded on a rising edge of the `ENABLE` input

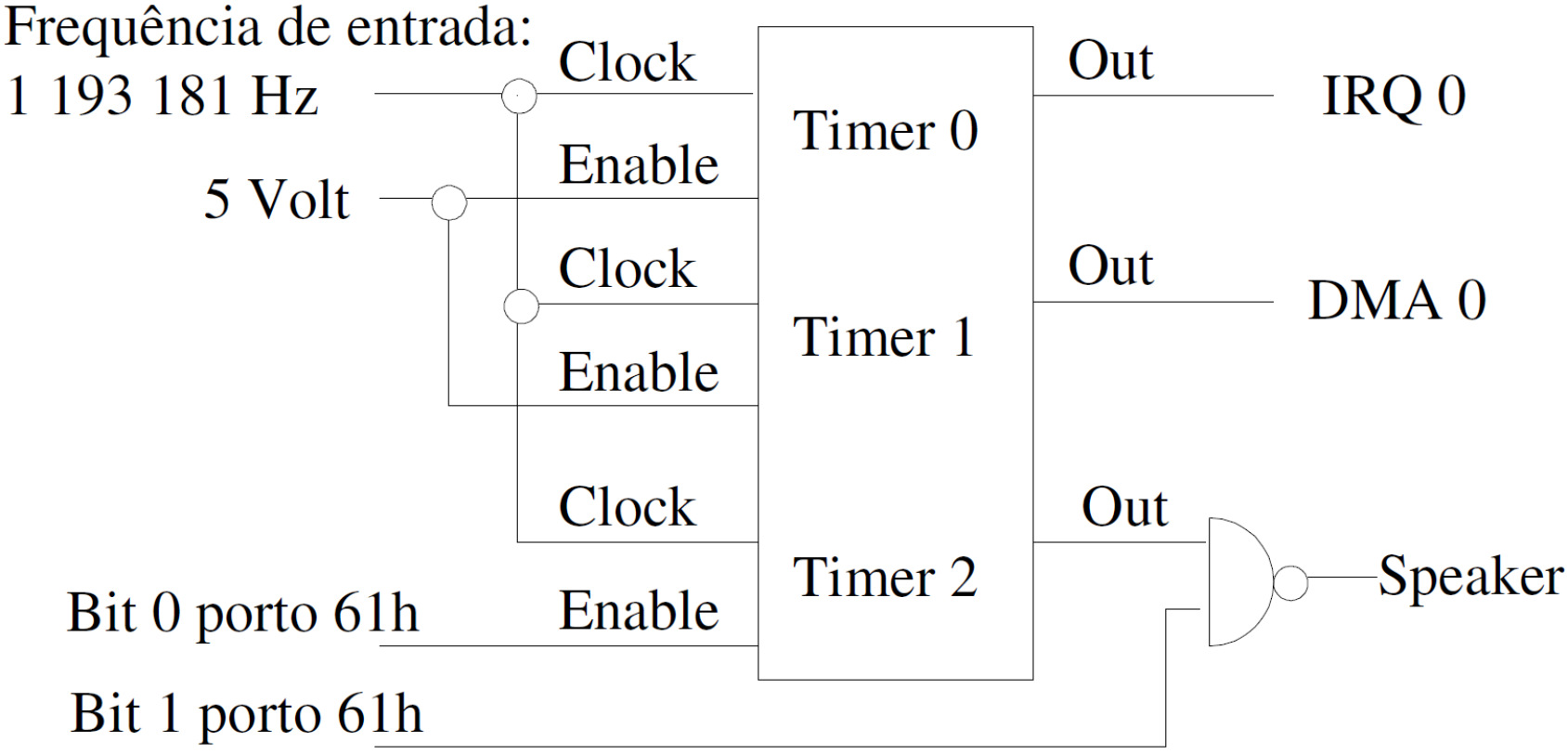
Mode 2 Rate Generator (divide-by-N counter)

- `OUT` goes low for one clock cycle when count reaches 0, the counter is reloaded with its initial count afterwards, and ...

Mode 3 Square Wave Generator – for Lab 3

- Similar to mode 2, except for the duty-cycle: `OUT` will be high for half of the cycle and low for the remaining half of the cycle

i8254: Use in the PC (1/2)



- Timer 0 is used to provide a time base.
- Timer 1 is used for DRAM refresh
  - Via DMA channel 0

  (Not sure this is still true.)
- Timer 2 is used for tone generation

i8254: Use in the PC (2/2)

- The i8254 is mapped in the I/0 address space:

```
Timer 0:           0x40
Timer 1:           0x41
Timer 2:           0x42
Control Register: 0x43
```

- Need to use `IN/OUT` assembly instructions
  - Minix 3 provides the `SYS_DEVIO` kernel call for doing I/O

```
#include <minix/syslib.h>

int sys_inb(port_t port, unsigned long *byte);
int sys_outb(port_t port, unsigned long byte);
```

- Need to write to the control register before accessing any of the timers

Minix 3 and Timer 0

- At start up, Minix 3 programs Timer 0 to generate a square wave with a fixed frequency
  - Timer 0 will generate an interrupt at a fixed rate:
    - Its output is connected to `IRQ0`
- Minix 3 uses these interrupts to measure time
  - The interrupt handler increments a global variable on every interrupt
  - The value of this variable increments at a fixed, known, rate
- Minix 3 uses this variable mainly for:
  - Keeping track of the date/time
  - Implementing SW timers

Lab 3: Part 1 - Generating a Square Wave

What to do? Change the rate at which Timer 0 generates interrupts.

```
int timer_test_square(unsigned long freq)
```

1. Write control word to configure Timer 0:
   - Select square wave mode (mode 3)
   - Preferably, LSB followed by MSB
2. Load Timer 0 with the value of the divisor to generate the frequency corresponding to the desired rate
   - Depends on the previous step

How to design it? Try to develop an API that can be used in the project.

```
int timer_set_square(unsigned char timer,
                     unsigned long freq)
```

How do we know it works? Use the `date` command.

Further Reading

- Lab 3 Script
- i8254 Data-sheet