Lab 1 - Introduction to PIC18F4550 Board, MPLAB-IDE, C-compiler and USB downloader.

Objectives

□ To illustrate the procedures to create a Microchip's PIC micro-controller project in MPLAB IDE, and to create, edit and compile a C program using MCC-18.

□ To show the steps to setup the USB link with the PIC18F4550 micro-controller, and to download a program to the micro-controller and to execute it

Introduction / Briefing

⇨ □ At the beginning of each lab session, your lab lecturer will go through a short **briefing** before you begin the experiment.

□ The discussion will help you in the MST, the lab test as well as the project. So, please pay attention and participate in the discussion.

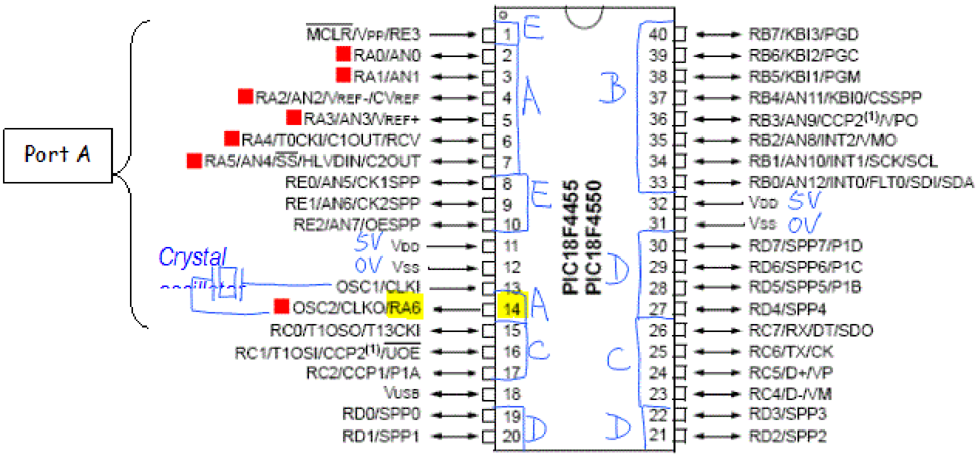
⇨ □ This lab sheet contains many **screen captures** to show you how to create a project, how to create, edit and compile a C program, and how to download a program to the micro-controller and run it. In subsequent labs, if you forget certain steps, you should refer to this lab sheet again.

⇨ □ To do this lab, the **software tools** required must already be installed on the PC.

PIC18F4550 I/O ports

□ You will learn more about the I/O ports in Chapter 3. The following is a brief summary.

□ PIC18F4550 has five I/O ports: A to E. Many pins have multiple functions. For instance, pin 14 is RA6 (Port A Pin 6) and also OSC2 (oscillator input 2).



□ The table below shows which pins can be used as general purpose I/O pins and whether they are, by default (i.e. after power on reset), analogue or digital, input or output.

Port	Available pins	Not available as general purpose I/O (- reasons)	After power on reset Need to write some data to the relevent control registers if re-configurations are required.
A	RA6-0 7 bits	RA6 (- oscillator)	RA5, 3-0: Analogue inputs (*). RA4: Digital input.
B	RB7-0 8	RB4 (- "Boot" button)	RB4-0: Analogue inputs (*). RB7-5: Digital inputs.
C	RC7-4, 2-0 4+3	RC5-4 (- USB connector)	RC7-4, 2-0: Digital inputs.
D	RD7-0 8		RD7-0: Digital inputs.
E	RE3-0 4	RE3 (- "Reset" button)	RE2-0: Analogue inputs (*). RE3: Digital input.

(P.3, 4)

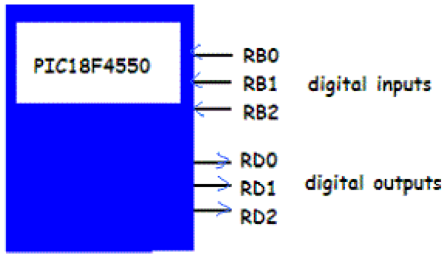
(*) ADC (Analogue to Digital Conversion) will be discussed in details in the future.

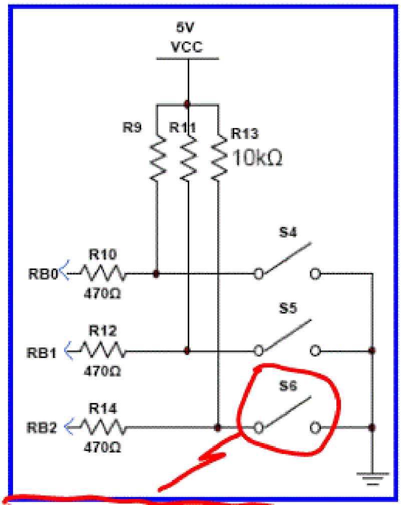
□ This lab will only involve ports B and D.

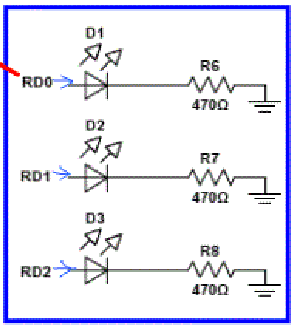
Port configuration

□ Do you know why the switches connected to RB0-2 are "active low"?

□ Do you know why the LED's connected to RD0-2 are "active high"?







*When RD0 is **HIGH** (5V),*
current flows to ground (0V)
through D1 and R6;
*hence D1 is on (**active**).*

*When RD0 is **LOW** (0V),*
no current flows to ground (0V)
through D1 and R6;
hence D1 is off (inactive).

When S6 is open (inactive):
RB2 connects to 5V through R13 and R14 -> reads HIGH.

*When S6 is closed (**active**):*
*RB2 connects to 0V through R14 -> reads **LOW**.*

□ To use port B to read the switch status (open or closed), port B must be configured as digital inputs.

□ But (referring to the table above), RB4-0 are analogue inputs after reset.

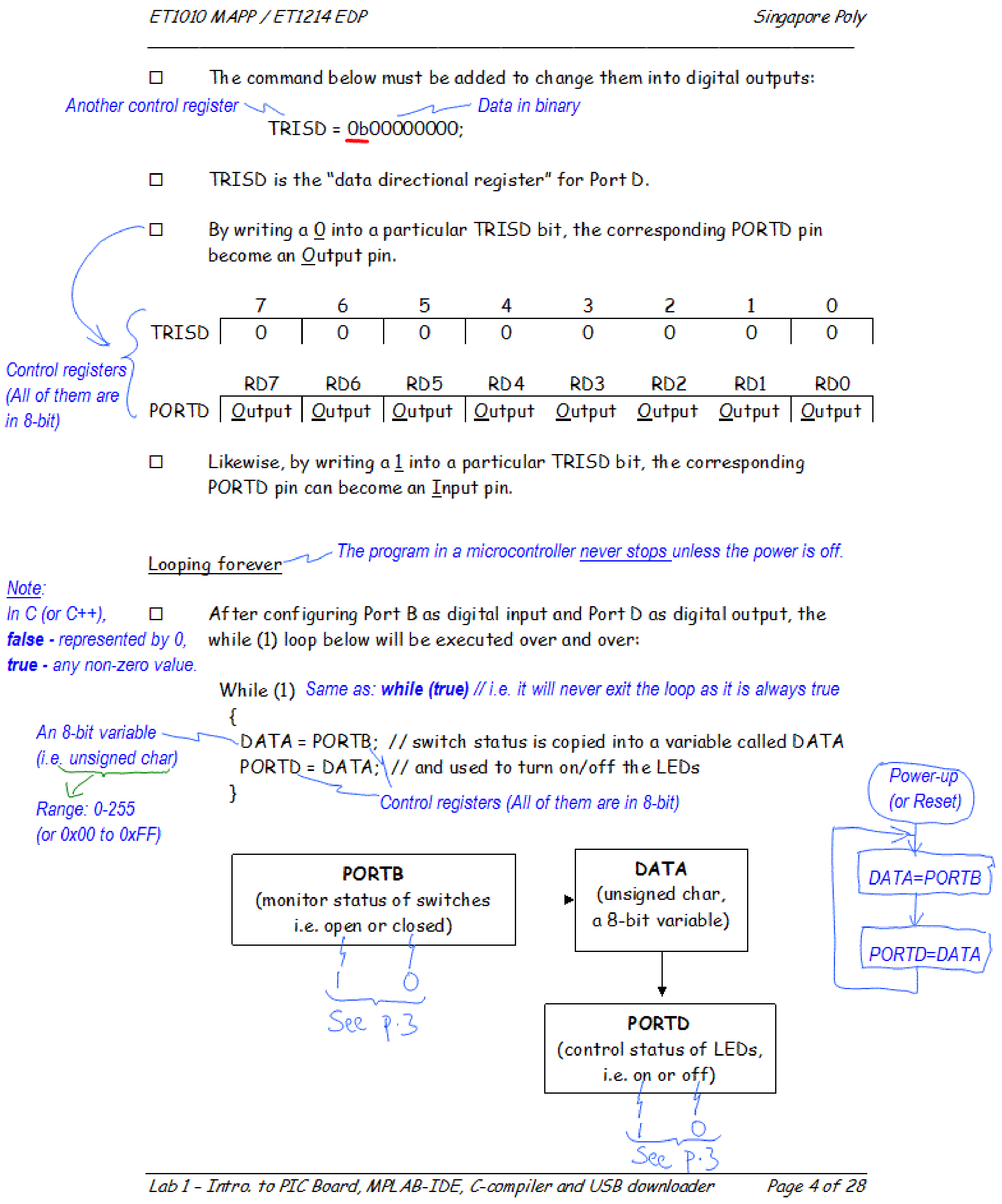
□ The command below must be added to change them into digital inputs:

One of the control registers *Data in hex (Can be in decimal or binary if it is more convenient.)*

```
ADCON1 = 0x0F;  // we will explain this in future
```

□ To use port D to control the LEDs (on or off), port D must be configured as digital outputs.

□ But, RD7-0 are digital inputs after reset.



□ The command below must be added to change them into digital outputs:

Another control register

Data in binary

```
TRISD = 0b00000000;
```

□ TRISD is the "data directional register" for Port D.

□ By writing a 0 into a particular TRISD bit, the corresponding PORTD pin become an Output pin.

	7	6	5	4	3	2	1	0
TRISD	0	0	0	0	0	0	0	0
	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0
PORTD	Output	Output	Output	Output	Output	Output	Output	Output

Control registers (All of them are in 8-bit)

□ Likewise, by writing a 1 into a particular TRISD bit, the corresponding PORTD pin can become an Input pin.

Looping forever

The program in a microcontroller never stops unless the power is off.

Note:
In C (or C++),
false - represented by 0,
true - any non-zero value.

□ After configuring Port B as digital input and Port D as digital output, the while (1) loop below will be executed over and over:

```
While (1)  Same as: while (true) // i.e. it will never exit the loop as it is always true
 {
  DATA = PORTB;  // switch status is copied into a variable called DATA
  PORTD = DATA;  // and used to turn on/off the LEDs
 }
```

An 8-bit variable (i.e. unsigned char)

Range: 0-255 (or 0x00 to 0xFF)

Control registers (All of them are in 8-bit)

PORTB (monitor status of switches i.e. open or closed)

1 0 See p.3

DATA (unsigned char, a 8-bit variable)

PORTD (control status of LEDs, i.e. on or off)

1 0 See p.3

Power-up (or Reset)

DATA=PORTB

PORTD=DATA