THE HONG KONG POLYTECHNIC UNIVERSITY

Department of Electronic and Information Engineering

EIE3105 Integrated Project (Part I)

Laboratory Exercise 1: Assembly and C Programming

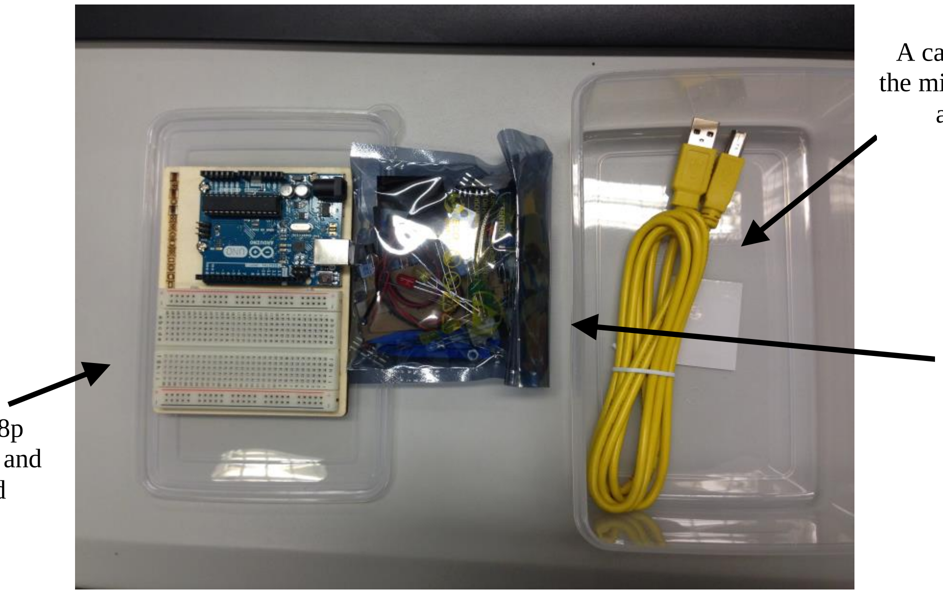
(Deadline: Check the course information)

Objective:

1. To familiarize students with the use of an AVR microcontroller.
2. To develop assembly and C programs under the Arduino platform.

Equipment:

Atmel Studio 6 (software)
The Arduino Starter Kit (hardware)



A cable to connect the microcontroller to a computer

Components

An ATmeg328p microcontroller and a breadboard

Important Notices:

1. You must read the ATmega328p datasheet carefully before you do the laboratory exercises. You have three ways to access the datasheet:
   a. Download the pdf file from the course web site;
   b. Download the pdf file from drive K in a computer in CF105 or CF504.
   c. Use a web searching engine (e.g., Google) to search the keyword "ATmega328p datasheet" and get the pdf file.
2. You leave your student identity card to our technicians to borrow one box of the kit. They will return your card to you when you return the box. Note that you need to return the box on or before 1:20 p.m.; otherwise, our technicians will pass your card to the instructor and you will have penalties when it happens.

3. You are not required to keep using the kit. You need it only when you want to test your program or/and demonstrate your work.

Introduction:

This experiment introduces some simple applications of using an AVR microcontroller to perform some arithmetic and I/O operations. Students are required to develop assembly and C programs and build up some simple electronic circuits for such simple applications.

Atmel Studio 6 is the integrated development platform (IDP) for developing and debugging Atmel AVR microcontroller (MCU) applications. The Atmel Studio 6 IDP gives you a seamless and easy-to-use environment to write, build and debug your applications written in C/C++ or assembly code. Atmel Studio 6 supports all 8- and 32-bit AVR MCUs. It also connects seamlessly to Atmel debuggers and development kits.

Arduino is an open-source physical computing platform based on a simple I/O board and a development environment that implements the Processing/Wiring language. Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer.

The Arduino Start Kit walks you through the basics of using the Arduino in a hands-on way. You will learn through building several creative projects. The kit includes a selection of the most common and useful electronic components.

Procedure:

Section A: Create your first simple application

1. Open Atmel Studio: Double click Atmel Studio 6.2.

2. Go to main menu. Select “Tools” → “External Tools”. Type in “Arduino Uno” in the field “Title”. Moreover, type in the following paths into the fields “Command” and “Arguments”:

   Command

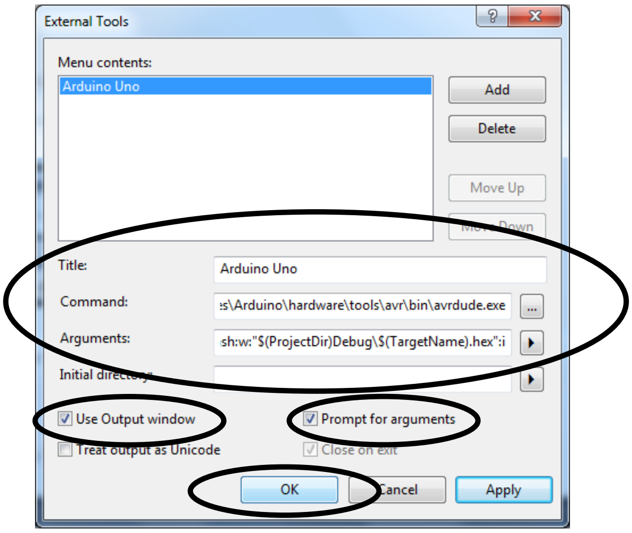
   ```
   C:\Program Files\Arduino\hardware\tools\avr\bin\avrdude.exe
   ```

   ```
   -C"C:\Program Files\Arduino\hardware\tools\avr\etc\avrdude.conf"
   -v -v -v -patmega328p -carduino -P\\.\COM4 -b115200 -D -
   Uflash:w:"$(ProjectDir)Debug\$(TargetName).hex":i
   ```

   Arguments

   Note that “COM4” in the above path shows the port that the Arduino Uno plugs in your computer. Thus, later, when you plug in the Arduino Uno to your computer, you need to check the port is “COM4” or not (Double click the icon “Computer” and Select “Properties” → “Device Manager” → “Ports”. Check the position of the port for the Arduino Uno). You may need to change the port number if it is not “COM4”. Moreover, you may go to the course web site to download the text file “ArduinoUno.txt” which stores these two paths. Then you can copy and paste them into such two fields. Note that this file can also be found in “K:\Arduino”. Finally, in some computers, the directory "Arduino" may not be installed in the directory "Program Files" but "Program Files (x86)". If it happens, please change the name of the directory "Program Files" in the command and the argument accordingly.

After that, click the checkboxes “Use Output window” and “Prompt for arguments” and then click the button “OK”. When you finish the setting, you will see a new field “Arduino Uno” is created under the menu item “Tools”. Note that the above setting is required to go through once it is the first time to open Atmel Studio.



Title: Arduino Uno

Command: :s\Arduino\hardware\tools\avr\bin\avrdude.exe

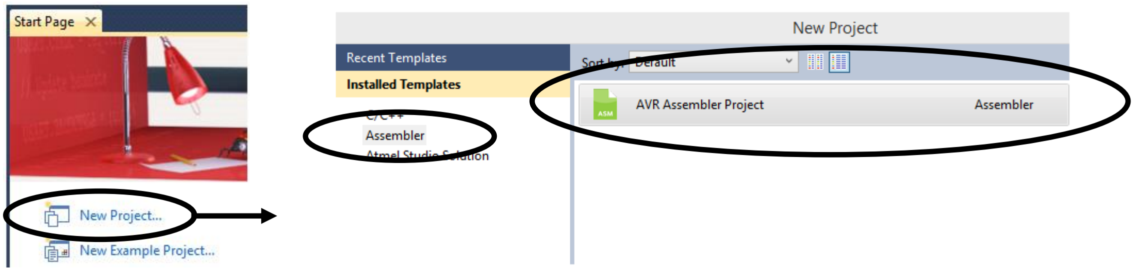
Arguments: sh:w:"$(ProjectDir)Debug\$(TargetName).hex":i

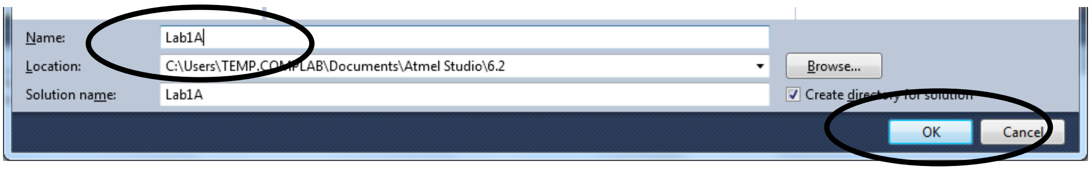
Initial directory:

☑ Use Output window ☑ Prompt for arguments

□ Treat output as Unicode ☑ Close on exit

3. Go to the front page or the main menu to create a new project. Now we are going to create a new assembly program. Thus, select “Assembler” → “AVR Assembler Project” and then type in the project name (i.e., “Lab1A”). After that, click the button “OK”.





Name: Lab1A

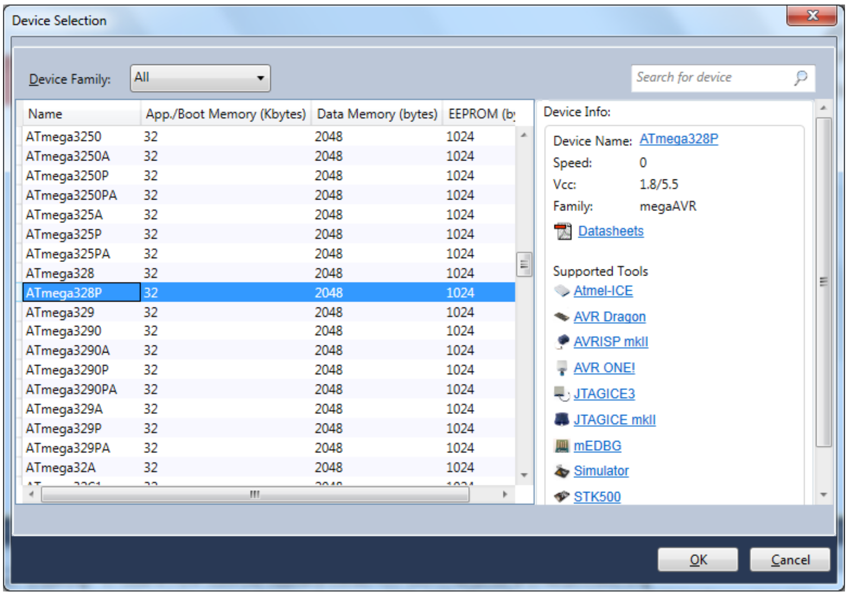
Location: C:\Users\TEMP.COMPLAB\Documents\Atmel Studio\6.2

Solution name: Lab1A

☑ Create directory for solution

Check the location. This is the directory that you can access your created files.

4. Select “ATmega328p” and then click the button “OK”.



Name	App./Boot Memory (Kbytes)	Data Memory (bytes)	EEPROM (b
ATmega3250	32	2048	1024
ATmega3250A	32	2048	1024
ATmega3250P	32	2048	1024
ATmega3250PA	32	2048	1024
ATmega325A	32	2048	1024
ATmega325P	32	2048	1024
ATmega325PA	32	2048	1024
ATmega328	32	2048	1024
ATmega328P	32	2048	1024
ATmega329	32	2048	1024
ATmega3290	32	2048	1024
ATmega3290A	32	2048	1024
ATmega3290P	32	2048	1024
ATmega3290PA	32	2048	1024
ATmega329A	32	2048	1024
ATmega329P	32	2048	1024
ATmega329PA	32	2048	1024
ATmega32A	32	2048	1024

Device Info:

Device Name: ATmega328P
Speed: 0
Vcc: 1.8/5.5
Family: megaAVR
Datasheets

Supported Tools
- Atmel-ICE
- AVR Dragon
- AVRISP mkII
- AVR ONE!
- JTAGICE3
- JTAGICE mkII
- mEDBG
- Simulator
- STK500

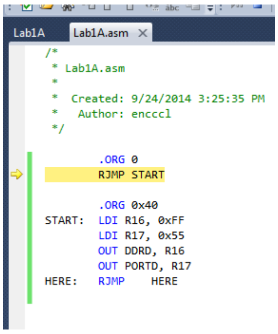
5. Type in the following program into “Lab1A.asm”.

```
              .ORG 0
              RJMP START

              .ORG 0x40
       START: LDI    R16, 0xFF
              LDI    R17, 0x55
              OUT    DDRD, R16
              OUT    PORTD, R17
       HERE:  RJMP   HERE
```

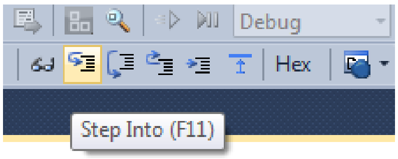
6. Go to the main menu. Select “Build” → “Build Solution” to compile the program.

7. Go to the main menu. Select “Debug” → “Start Debugging and Break” to execute the program step by step. Then a dialogue box will be displayed for users to choose “Selected debugger/programmer”. It will be displayed once only when a new project is created. Select “Simulator” in the field “Selected debugger/programmer” and then go to the main menu again to select “Start Debugging and Break”.

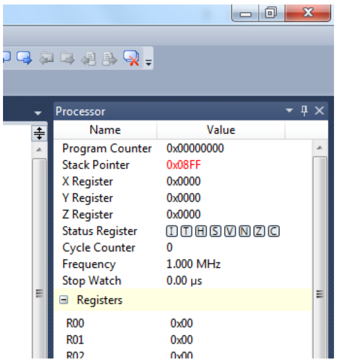
8. The following screen will be displayed. You can press F11 or click the button “Step Into” to execute the program step by step.



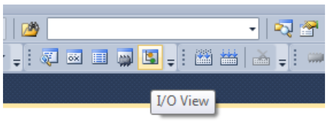
```
/*
 * Lab1A.asm
 *
 *  Created: 9/24/2014 3:25:35 PM
 *   Author: encccl
 */

        .ORG 0
        RJMP START

        .ORG 0x40
START:  LDI R16, 0xFF
        LDI R17, 0x55
        OUT DDRD, R16
        OUT PORTD, R17
HERE:   RJMP    HERE
```

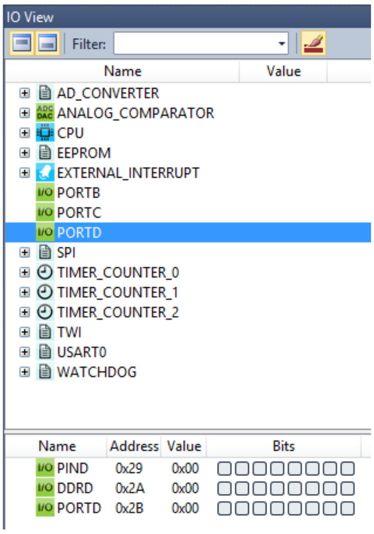


9. On the right hand side of the screen, you can check the values of different registers. If you want to check the status of different ports, you can click the icon “I/O View”. In this program, check the changes of PIND, DDRD and PORTD.



Name	Value
Program Counter	0x00000000
Stack Pointer	0x08FF
X Register	0x0000
Y Register	0x0000
Z Register	0x0000
Status Register	ITHSVNZC
Cycle Counter	0
Frequency	1.000 MHz
Stop Watch	0.00 µs
⊟ Registers	
R00	0x00
R01	0x00
R02	0x00

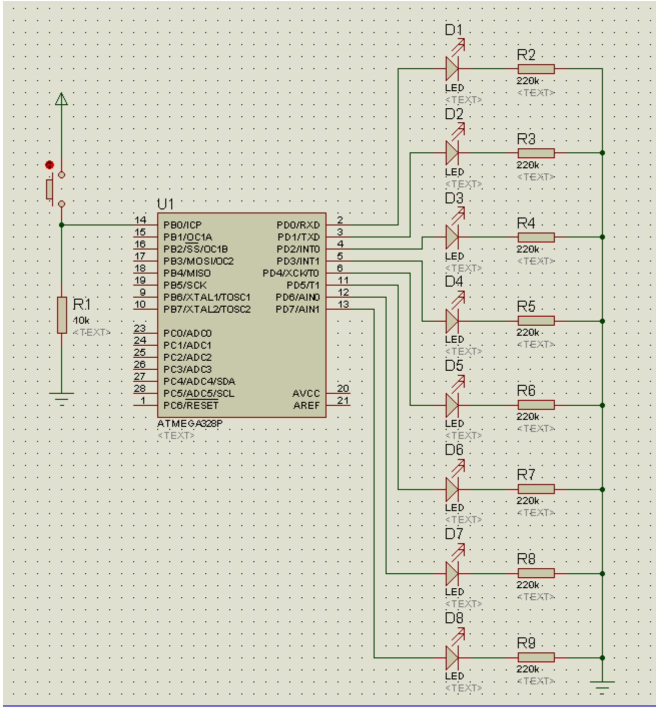




Name	Value
⊞ AD_CONVERTER	
⊞ ANALOG_COMPARATOR	
⊞ CPU	
⊞ EEPROM	
⊞ EXTERNAL_INTERRUPT	
PORTB	
PORTC	
PORTD	
⊞ SPI	
⊞ TIMER_COUNTER_0	
⊞ TIMER_COUNTER_1	
⊞ TIMER_COUNTER_2	
⊞ TWI	
⊞ USART0	
⊞ WATCHDOG	

Name	Address	Value	Bits
PIND	0x29	0x00	□□□□□□□□
DDRD	0x2A	0x00	□□□□□□□□
PORTD	0x2B	0x00	□□□□□□□□

10. Build the circuit below on the Arduino Uno and the breadboard. Refer to the pinout diagram in the appendix. Note that “220k” should be “220”. It means 220Ω.



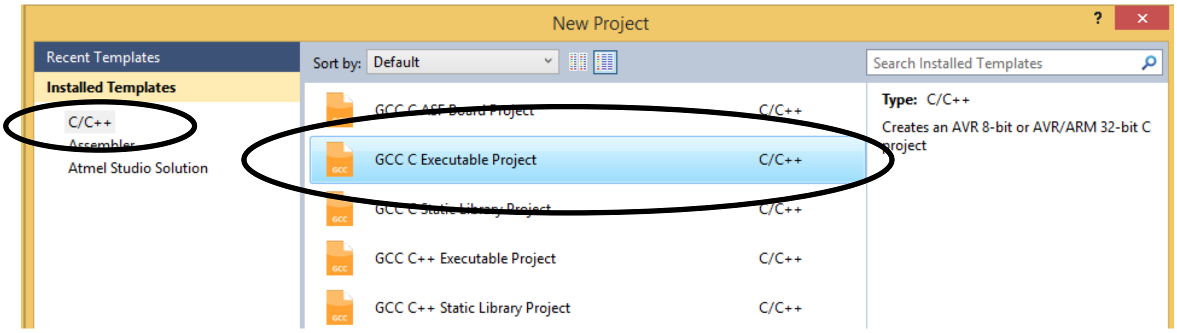
11. Go to the main menu. Select “Tools” → “Arduino Uno” to burn the program into the Arduino Uno. You should see some LEDs are on. Note that you should remove any connections to pin 0 (RX) and pin 1 (TX) of Arduino Uno before you burn your program into it; otherwise, you may not burn your program successfully.

12. Repeat the above steps but this time Port D is toggled between 0x55 and 0xAA with 1 to 2 seconds time delay. You should use NOP (No operation) and looping to generate the delay. Show the calculation how to generate the delay. Note that the clock frequency of

the Arduino Start Kit is 16 MHz. When your program works properly, you should see all LEDs connected to Port D are flashing.

13. Repeat Step 12 by writing a C program.

You should go to the main menu and create a new project. Then select “C/C++” and then “GCC C Executable Project” to create a new project. After that, select “ATmega328p”.


New Project
?
Recent Templates
Installed Templates
C/C++
Assembler
Atmel Studio Solution
Sort by: Default
Search Installed Templates
GCC C ASF Board Project C/C++
GCC C Executable Project C/C++
GCC C Static Library Project C/C++
GCC C++ Executable Project C/C++
GCC C++ Static Library Project C/C++
Type: C/C++
Creates an AVR 8-bit or AVR/ARM 32-bit C project

Note that the I/O ports of the AVR microcontroller are bit-accessible but the AVR C compiler does not support this feature. Thus, you should use a mask to set, clear and check some bits in the I/O ports.

Section B: Check the value of SREG

Do the following arithmetic operations and check the value of SREG in each arithmetic operation:

0x75 – 0x23
0x8C + 0x74
20 + 44

Explain why some flags of SREG are turned on after executing the above operations. Put your answer into a word document and submit it.

Section C: Simple applications (C programs)

1. Connect a switch to PB0 and the pin diagram is shown in Step 10 of Section A. Connect a LED to the switch. When the switch is pressed, the LED is on. When the switch is released, the LED is off.

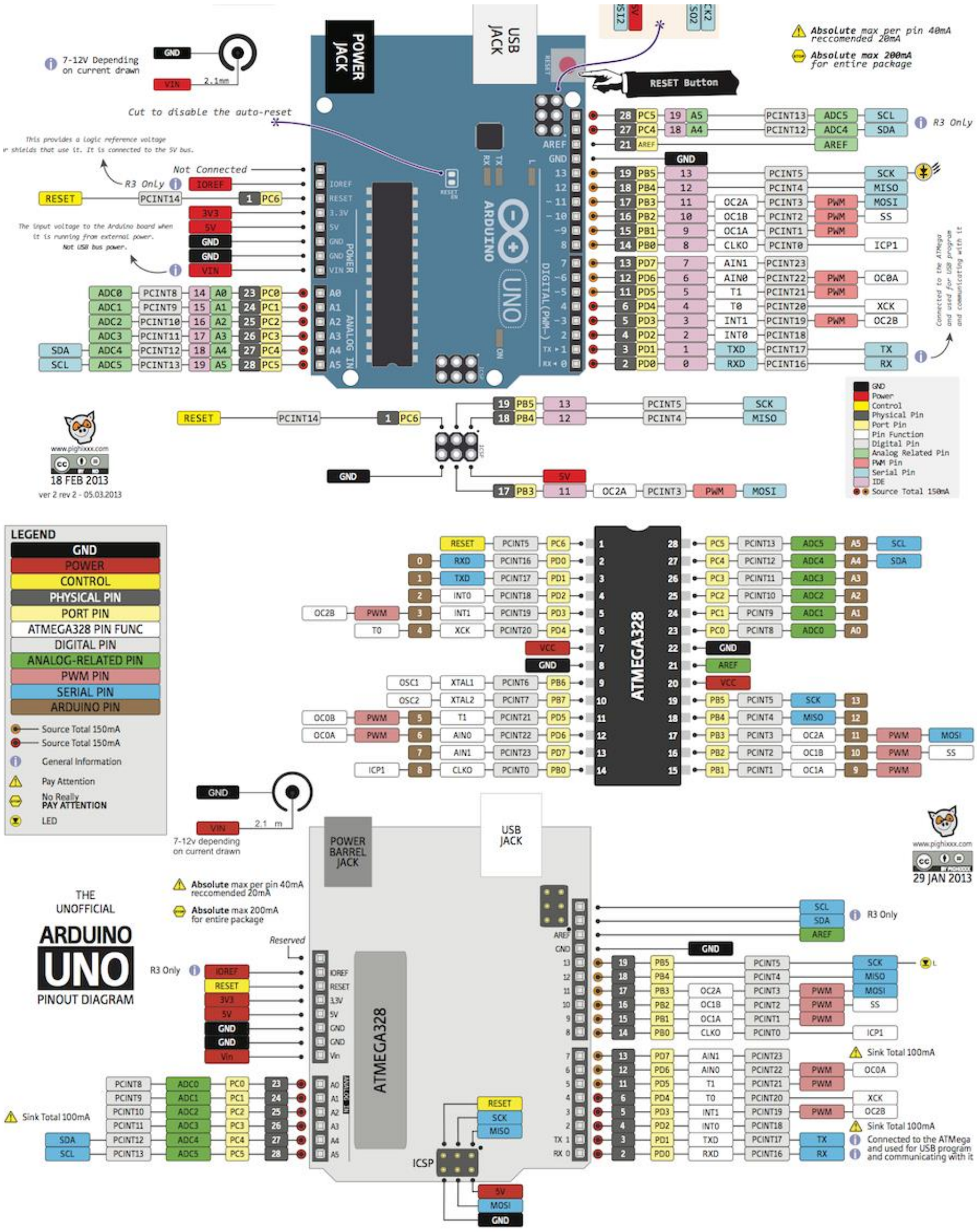
2. Use the circuit in Section A and the connection in Step 10. There are two states in the switch: State 0 and 1. When it is in State 0, all LEDs connected to Port D are off. When it is in State 1, all LEDs are on. At the beginning, the switch is in State 0. When the switch is pressed and it is in State 0, it goes to State 1. When the switch is pressed and it is in State 1, it goes to State 0.

Demonstrate your results to our tutor or technicians. It includes the programs in Section A (Step 12 and 13) and C (Step 1 and 2).

Instructions:

1. You are required to demonstrate your programs to our tutor or technicians.
2. Zip all programs (including the whole projects) in Section A and C, and your answers in Section B (a word document) to a single file. Submit it to Blackboard.
3. Deadline: **<u>Check the course information</u>**.

Appendix: Arduino Uno Pinout Diagrams



Ivan Lau
Lawrence Cheung
August 2016