Mechatronics Design – Class#2

Liwei Lin

Professor, Dept. of Mechanical Engineering
Co-Director, Berkeley Sensor and Actuator Center
The University of California, Berkeley, CA94720
e-mail: lwlin@me.berkeley.edu
http://www.me.berkeley.edu/~lwlin
Outline

- Review on Class#1
- Lab #1
Lab #1

ME102 Lab 1: Getting Started

This lab will teach you how to setup the IDE (Integrated Development Environment) for the Arduino. These instruction have been tested for the computers available in lab. However, it is just as easy to setup the IDE on your own laptop. You may wish to do this for future lab assignments.

1. Introduction to the Arduino
   - Kit contents check
   - Arduino IDE and some terminology
   - Hello Worlds
   - Anatomy of a sketch
2. Setting up your hardware buffers
   - Digital input buffering with the 74LS14
   - Digital output buffering with the 7417
   - ADC setup with the 4342
3. Clean up and check off
   - Blinking LED with the buffered setup
4. References
Lab Kit

Kit contents check

Before we continue any further, please make sure that you have at least the following in your lab kit.

- 1 Arduino Duemilanove
- 1 USB A-B cord
- 1 breadboard
- 5 LED’s of assorted colors
- 2 tactile switches
- 1 DC motor
- 1 tiny RC servo
- 1 potentiometer
- 1 optical encoder ([datasheet](#))
- 1 pair of wire cutters
- 1 74LS14 ([datasheet](#))
- 1 DM7417 ([datasheet](#))
- 1 OPA4342 ([datasheet](#)) or TLV2374I ([datasheet](#))

If you have any problems identifying your parts, please notify a GSI. If you are missing any parts, please find Tom Clark.
Arduino
Arduino IDE

The largest benefit of the Arduino is the fact that you are not limited to programming in the lab. Instructions on how to install a copy of the Arduino IDE on your own machine can be found here (Mac, Windows and Linux).

Start the Arduino IDE by clicking the Arduino icon on the desktop.

For those of you who are familiar with microprocessor programming, the first thing you will notice is the Arduino's minimalist approach.

There are 7 shortcut commands that you can use with the Arduino IDE:

- **Serial Monitor** opens the only debugging tool you have with the Arduino. The Serial Monitor displays information passed from the Duemilanove to the computer.
- **Upload** compiles your program ("sketch" in the parlance of the creators of the Arduino, and sends it to the board if there are no compile-time errors.
- **Save** your sketch
- **Open** an existing sketch
- **New** creates a new sketch
- **Stop** interrupts compilation of your code.
- **Compile** your sketch in order to check for compile-time errors. As your sketch grows in size, it'll save time to verify that your code compiles before trying to upload it.
1. Tools → Board (select the right Arduino version)
2. Computer → Device Manager → Ports → USB Serial (COM3)
3. Tools ➔ Serial Port ➔ COM3
5. Files ➔ Examples ➔ Stubs ➔ HelloWorld (load the program)
6. Verify the Screen (successful communication)
6. File → Examples → Digital → Blink (load a new program)
Software Introduction – “HelloWorld”

```cpp
void setup() {
    Serial.begin(9600);
}

void loop() {
    Serial.println("Hello World!");
}
```

- `Serial.begin(9600);`: where to use the board
- `Serial.println("Hello World!");`: serial communication
- `void loop()`: never-ending loop
- `Serial.println`: serial print on screen

Please find out this yourself.
Software Introduction – “Blink”

```c
int ledPin = 13; // LED connected to digital pin 13
void setup() {
    pinMode(ledPin, OUTPUT);
}

void loop() {
    digitalWrite(ledPin, HIGH);
    delay(1000); // wait 1000 ms
    digitalWrite(ledPin, LOW);
    delay(1000);
}
```

Liwei Lin, University of California at Berkeley
Hardware - Setup

- Set 5Volt on one rail of the breadboard
- (the breadboard can also be powered by the Arduino board but the current is limited to 50mA by the Arduino)
- Connect the common ground (0Volt) to one rail of the breadboard
Digital output buffering with the DM7417 (datasheet) protects the Arduino. The DM7414 is a hex buffer with open collector high voltage output. An open-collector only guarantees that a LOW input results in a LOW output. There are no guarantees for a HIGH input.
Digital Output Problem – “Floating”

floating → gradually to high electrical noise
Digital Output Problem – “Short”

$V_{cc} = 3.3 \text{V}$

when grounded $\rightarrow$ short circuit

from $V_{cc} \rightarrow$ ground
Pull-Up Resistor

By using a pull-up resistor on the output end, one can supply any level of voltage within the operational specifications of the IC. For the DM7414, it is 15V.

The basic function of a pull-up resistor is to insure that given no other input, a circuit assumes a default value. Consider the circuit in “Configuration of DM7417”, when the digital signal input is HIGH, the IC acts as a high impedance part so current flows to the right. When the digital signal input is LOW, the IC becomes a current sink. IN order to prevent IC destruction, pull-up resistors are generally in the range of 10k to 47k ohms. Special cases do exist as you will see at the check-off task.
Pull-up Resistor

\[ V_{cc} \quad \text{pull up resistor} \quad 10k \sim 47k \Omega \]
Digital Output Buffer – DM7417

+5 VDC
ground

3.3 V

Output

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>sees high (3.3 V)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>sees ground</td>
</tr>
</tbody>
</table>

Output
To buffer the digital input of the Arduino, we use the 74LS14 (datasheet). The 74LS14 is a hex schmitt trigger inverter. It takes a digital input and outputs the opposite. HIGH becomes LOW, and LOW becomes HIGH. In TTL (Transistor-to-Transistor Logic), a HIGH signal is anything from 2.2V to 5V whereas a LOW signal is anything from 0V to .8V. The area from .8V to 2.2V is essentially undefined behavior. The schmitt trigger is a property of this particular chip that allows it to be slightly less responsive to noise. And finally, there are six such inverters on a single chip (hence hex).
Digital Input for DSP

Digital Input

digital input

0 - ground
1 - 3.3V

no real power
Schmitt Trigger – Comparator

Comparator

\( V_{in} \rightarrow V_{out} \)

\( V_{in} > V_{ref} \rightarrow V_{out} = V^+ \)

\( V_{in} < V_{ref} \rightarrow V_{out} = V^- \)
Schmitt Trigger – Signal & Noise

[Diagram showing voltage variations over time with annotations: 'zero crossings', 'clear signal', 'multiple crossing']

Liwei Lin, University of California at Berkeley
Schmitt Trigger – Reject Interference
Schmitt Trigger – Circuit

\[ V_{in} = V_{ref} = V_{out} \times \frac{R_1}{R_1 + R_2} \]

\[
\begin{cases}
V_{in} < V_{ref} & \Rightarrow V_{out} = V_{set} \\
V_{in} > V_{ref} & \Rightarrow V_{out} = V_{sat}
\end{cases}
\]
Schmitt Trigger – Responses

\[ V_{\text{in}} = V_{\text{set}} = V_{\text{out}} \frac{R_1}{R_1 + R_2} \]

\( V_{\text{in}} \) (temperature)
ADC (Analog-to-Digital Conversion)

We protect the ADC with a voltage follower. In your lab kits, you may either have a OPA4342 (datasheet) or a TLV2374I (datasheet). Both are suitable for our purposes.

Verify that there is unity gain or \( V_{\text{in}} = V_{\text{out}} \) for your voltage follower. You can easily do this by connecting the wiper of your potentiometer, which has been connected between 5V and ground, to the non-inverting input (+ In C). The wiper is the part that changes voltage as the knob turns.
The check off for this lab is a blinking LED that works with your buffered system.

1. Connect Pin13 to the one of the 7417’s inputs
2. Connect the corresponding output from the 7417 to the anode of the LED.
3. 3) The pull-up resistor limits current flow. With a 10k resistor, current flow through it is limited to .5mA. This is not enough to drive an LED. Given that the SN7417 can sink 40mA, use a resistor of around 1k ohms as your pull-up resistor.
4. Note that the onboard LED and your recently connected LED should blink in phase.