Lab 1 Introducing the Lab Environment

Leqi Ding
dlq991118@sjtu.edu.cn

Ming Cheng
chengming@sjtu.edu.cn

Wanda Guo
wdguo0424@gmail.com

July 18, 2019

Contents

Abstract

As the solutions to any complex problems can be traced back to the simplest ones, most of the implements of projects in engineering nowadays can date back to those on the simple, little board – the Arduino UNO board. Therefore, lab1 aims to enable us to get familiar with the Arduino IDE and the implementation process of specific functions on the board in the C language. There are two parts in this lab, with the first one aiming to introduce the C language and the Arduino UNO microprocessor board based upon the Atmel Atmega 328P microcomputer associated with Arduino development environment as the interaction between the microprocessor and peripheral devices. The second part uses the LCD (Liquid Crystal Display) as hardware, with the purpose of implementing the interaction between two microprocessors in order to control a peripheral device attached to the second microprocessor.

Keywords: Arduino UNO microprocessor board, the C language, peripheral device, interaction

1 Introduction

Lab1 consists of two parts.

There are mainly two purposes of part1. The first one is to introduce the C language and the Arduino UNO microprocessor board based upon the Atmel Atmega 328P microcomputer and the associated Arduino development environment. The second one is to work with the basic Blink program, make some modifications on that to implement some specific functions and interface the microprocessor to two peripheral devices – a button and an LED in order to gain the knowledge of sending output from a program to display device.

Part2 is to gain the knowledge of sending information from one microprocessor to the other to control a peripheral device attached to the second microprocessor. There are the LCD (Liquid Crystal Display) and Arduino as the hardware, while the C language and the associated development environment as the software.

2 Discussion of the lab

2.1 Brief Design Specification:

Part 1

We need to familiarize with C and Arduino and configure the Arduino IDE for further work. Then, by running examples and appropriately modifying them, we enable LED to blink for a set period and to be controlled by a button as well as use the peripheral devices to get expected outputs.

As for program 1 and 2, we use LED on board as output. For program 3, 4 and 5, we use LED brick as the output, instead of LED on board. Especially for program 5, a button is used as input. Finally, for program 6 and 7, we use the Serial Monitor as output. For program 1 to 4, we use delay statement to implement a LED brick or onboard LED blinking for a set period. For program 5, a button is used to control LED brick by judging the status of switch circularly. For program 6 and 7, we use an internal tool, the serial monitor, to display the result of computation. Troubleshooting work mainly lies on the logical debugging.

Part 2

As a further experiment based on part 1, we are required to use LCD to get a chain of characters output and then use a button to control the shift of character displayed on the LCD. Then, we try to implement communication between two Arduino UNO boards and similarly, connect peripherals to the circuit. For program 8, 9 and 10, LCD brick is regarded as output. And especially for program 10, we use two buttons as input. When it comes to program 11, the output is Serial Monitor on both the master board and slave board. The output of program 12 is also LED brick.

The function of program 8, 9 and 10 is to display expected characters. Especially for program 10, we use two buttons to control the shift of the character X. Program 11 and 12 implement communication between two boards. In this part, troubleshooting work lies on the debugging to implement turning on and off the LED, which involves judging status of buttons and the procedure of program uploading.

2.2 Hardware Implementation

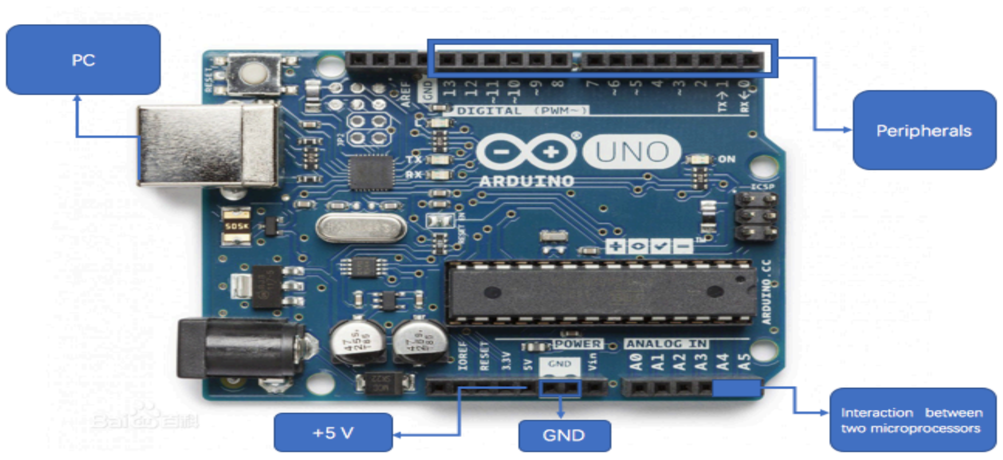

Figure 1: the schematic diagram of the hardware implementation

The schematic diagram of the hardware implementation is as above. On the Arduino board, port 0 to 13 are used to connect to the peripherals, such as the LED, the button and so on. Besides, port A4 and A5 are used to implement the interaction between two microprocessors. In addition, +5V source is connected into the board through port 5V, while the port GND should be connected to the ground, obviously. Last, in order to enable the whole board to work, it is indispensable to connect the board to PC, enabling the interaction between PC and the board as well.

2.3 Software Implementation

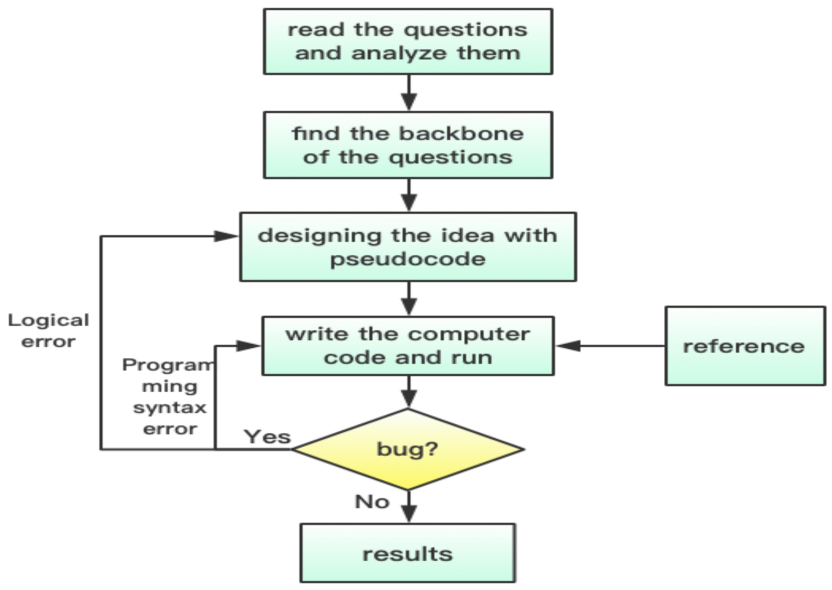
read the questions and analyze them
find the backbone of the questions
designing the idea with pseudocode
write the computer code and run
reference
bug?
Yes
No
results
Logical error
Programming syntax error

Figure 2: Structural diagram of software

Initially, we read the questions, analyze them, and find the backbone of the questions. We use the reverse-analysis method to start from the final problem and gradually analyze the functions to be implemented in each step until the problem can be solved by the lowest level statements. After designing the idea with pseudocode, we write it into the computer code and run it. We observe the results of the codes, use the debugger to debug if possible, and finally solve the whole problem. In the meanwhile, if we need to use the knowledge that has not been learnt yet, we will turn to the references for help and apply related functions and algorithms to the code.

Part1 Porgram1: digitalWrite($LED_BUILTIN$, HIGH); enables to turn the LED on by enabling the voltage HIGH while digitalWrite($LED_BUILTIN$, LOW); does the opposite. delay(2000); sets the time with unit as millisecond.

Program2: Analyses are the same as program1.

Program 3&4: int LED = 8; sets the port of LED which is associated with the hardware. pinMode(LED, OUTPUT); initializes digital pin $LED_BUILTIN$ as the output. The rest analyses are the same as previous programs.

Program5: int Button = 9; sets the port of the button which is associated with the hardware. if (digitalRead(Button)) indicates the button is pressed so that the button can control the LED.

Program6: Function Serial.print(”hello world ”); enables the string to be printed without enabling the cursor to go to the next line, while function Serial.println(”how ya doin”); enables the string to be printed with enabling the cursor to go to the next line.

Program7: const double COSTPERPERSON = 977.50; defines a constant named COSTPERPERSON, enabling to change the value conveniently if possible. The formula TotalCost = COSTPERPERSON * 0.95 * 1.095 * 9; calculates the result.

Part2 Program8: const int rs = 2, en = 3, d4 = 4, d5 = 5, d6 = 6, d7 = 7, d8 = 8; LiquidCrystal lcd(rs, en, d4, d5, d6, d7, d8); initializes the library by associating any needed LCD interface pin with the Arduino pin number it is connected to. lcd.noDisplay(); turns off the display while lcd.display();turns on.

Program9: Function lcd.setCursor(); sets the location of the string to display.

Program10: digital- Read(Button1)means button1 is pressed and a ‘!’ ahead of it means the opposite. $InitialPos = (InitialPos + 1)\%16$; calculates the change of the position, including special situation.

3 Test Plan

Part1: In program 1 and 2, the duration of LED’s each state (on and off) needs to be recorded. In program 3, 4 and 5, what needs to record is the duration of the state of LED Brick rather than LED on board, while whether the button can control the LED should also be tested in program 5. In the meanwhile, in program 6 and 7 we need to verify whether the specific string is properly printed in Serial Monitor.

Part2: The string printed on the LCD should be checked in program 8, 9 and 10, while whether the button can control the LCD also needs to be tested in program 10. In program 11 and 12, two processors should be used: one is master, the other is slave. We should test whether the slave can be controlled by the master successfully, by watching the Serial Monitor and the state of LED.

4 Presentation, discussion, and analysis of the results

Part1 Porgram1: The LED is on for 2 seconds and off for 1 second and repeats.

Program2: The LED has two 1 second blinks followed by two 2 second blinks and repeats.

Program 3&4: The LED Brick blinks like the phenomena in program 1&2, respectively.

Program5: The LED Brick can be controlled by the Button Brick. The LED is on as soon as Button is pressed.

Program6: “hello world 9” is printed in the Serial Monitor while “how ya doin” is printed in the next line, and it repeats every second.

Program7: The total price $9,151.56 is printed in the window, and it repeats every second.

Part2 Program8: The LCD displays ”hello, world” successfully, and it blinks every 0.5 seconds.

Program9: The LCD displays “MING” on the top-left while “WANDA” on the top-right and “LEQI” on the bottom-left.

Program10: When button1 is pressed, move the character ‘X’ to the left by one position. (When ‘X’ reaches the leftmost position, the next position is rightmost) When button2 is pressed, move the character ‘X’ to the right by one position. (When ‘X’ reaches the rightmost position, the next position is leftmost) When both two buttons are pressed, the position of ‘X’ does not change.

5 Analysis of any errors

Part 1:As most of the programs are examples or simple to modify, most of the errors appear when working on program 7. When coding, we once added constant defining statement in the loop, which caused an error. We solved this by moving the statement to the front of the program.

Part 2:We mainly met 2 problems while working on program 10. The first one is the clearing problem while the character shifts. According to the references, we added ‘lcdclear’ statement into the loop and solved this problem. The second one lies in the failure of port on the Bus Shield. When testing, unexpected characters appeared but our code had no logical problem. After that, when we selected port 8 as one of the inputs, we find it broken. After we changed the input to port 10, certain character vanished and the expected results appeared. That verifies that our assumption that the port doesn’t work is true.

In program 12, we met two problems. The first one is no matter whether the BUTTON is pressed or not, we cannot change the status of LED: it just kept on or off. What we did first was to test the software. However, we did not find any error in the code. Then, we checked the Arduino board to ensure whether there was something wrong in the hardware. After checking all the wires we used, we found the wire between pins A4 on each of the two boards did not connect well. That is where the real problem is.

The other problem is that we cannot control the LED with the BUTTON successfully. No matter whether the BUTTON is pressed or not, LED just blinks. Because we had confirmed that the program is correct, we carefully read the instructions again and changed the order we uploaded the files of the master and slave. Then the result satisfied the expected one.

6 Summary and Conclusion

In the project, we get familiar with the Arduino board and the C language. In part 1, we use the function digitalWrite() and delay statements to implement the blinking function and the control of LED by judging statements. In part 2, we use ‘lcd.clear’ statement to implement the shifting of a certain character, and implement the interaction between the slave and the master.

In a nutshell, we successfully set up the working environment, implemented the blinking of LED, enabled to display a certain character on LCD and achieved the communication between two Arduino UNO boards.

When completing the implementation of programs 11 and 12, we found some references about the principle of I2C Protocol, which acts as an important role in the communication of Arduino boards. Simply put, in the I2C communication protocol, we have two devices, a master and a slave, which are interconnected through two lines: a data line and a clock line. In Arduino Uno, the data line is Analog pin 4(A4) and the clock line is Analog pin 5(A5). This, however, is different for different boards. When the clock pin goes from low to high, one bit of data is transferred via the data pin. The slave board may then either send back data via the same data pin or perform a task. The first eight bits, however, are reserved for the address of the slave Arduino board to which the master sends values. As a recommendation of the project, we think it essential to add more introductive content to Arduino and correct some errors of instructions in the Lab Intro document.

7 Appendix

Problem1

```
// Onboard LED is on for 2 seconds and off for 1 second and repeats.
/*
Turns an LED on for one second, then off for one second, repeatedly.
*/
// the setup function runs once when you press reset or power the board
void setup()

// initialize digital pin LED_BUILTIN as an output.
pinMode (LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop()
{
//LED is on for 2 seconds and off for 1 second and repeats.
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(2000); // wait for two seconds
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
```

Problem 2

```
// Onboard LED has 2 1 second blinks followed by 2 2 second blinks and repeats.
// the setup function runs once when you press reset or power the board
void setup()
{
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop()
{
// the LED has two 1 second blinks followed by two 2 second blinks and repeats.
//two 1 second blinks
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
//two 2 second blinks
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(2000); // wait for two seconds
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(2000); // wait for two seconds
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(2000); // wait for two seconds
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(2000); // wait for two seconds
}
```

Problem 3

```
// LED Brick is on for 2 seconds and off for 1 second and repeats.
```

```
int LED = 8;//define the 8th digital pin for LED brick
// the setup function runs once when you press reset or power the board
void setup()
{
// initialize LED Brick as an output.
pinMode(LED, OUTPUT);
}
// the loop function runs over and over again forever
void loop()
{
digitalWrite(LED, HIGH); // turn the LED on (HIGH is the voltage level)
delay(2000); // wait for two seconds
digitalWrite(LED, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
```

Problem 4

```
// LED Brick has 2 1 second blinks followed by 2 2 second blinks and repeats.
// the setup function runs once when you press reset or power the board
int LED = 8;//define the 8th digital pin for LED brick
void setup()
{
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED, OUTPUT);
}
// the loop function runs over and over again forever
void loop()
{
digitalWrite(LED, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
digitalWrite(LED, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
digitalWrite(LED, HIGH); // turn the LED on (HIGH is the voltage level)
delay(2000); // wait for two seconds
digitalWrite(LED, LOW); // turn the LED off by making the voltage LOW
delay(2000); // wait for two seconds
digitalWrite(LED, HIGH); // turn the LED on (HIGH is the voltage level)
delay(2000); // wait for two seconds
digitalWrite(LED, LOW); // turn the LED off by making the voltage LOW
delay(2000); // wait for two seconds
}
```

Problem 5

```
//Can the LED Brick can be controlled by the Button Brick.
int Button = 9; //define the 9th digital pin for button brick
int LED = 8; //define the 8th digital pin for LED brick
void setup()
{
pinMode(LED,OUTPUT); //set the LED pin for digital output
pinMode(Button,INPUT); //set the Button pin for digital input
}
void loop()
{
if (digitalRead(Button)) // if button pressed
digitalWrite(LED,HIGH); // light the LED
else // if not pressed
digitalWrite(LED,LOW); // turn off the LED
}
```

Problem 6

```
//display information on the screen
int i = 9;//assign 9 to variable i
void setup()
{
// start serial port at 9600 bps and wait for port to open:
Serial.begin(9600);

void loop()
{
Serial.print(”hello world ”); //display ”hello world ” on the screen
Serial.println(i); //display the value of i, then go on to the next line
delay(1000); //wait for a second
Serial.println(”how ya doin”); //display ”how ya doin” on the screen, then go on to the next line
delay(1000); //wait for a second
}
```

Problem 7

```
//Print the results for homework 1 Problem 1 on the UNO.
const double COSTPERPERSON = 977.50; //define COSTPERPERSON to present the cost of each person, then it is a constant variable.
double TotalCost = 0; //define TotalCost to present the total cost.
void setup()
{
// start serial port at 9600 bps and wait for port to open:
Serial.begin(9600);
}
void loop()
{
TotalCost = COSTPERPERSON * 0.95 * 1.095 * 9;//compute the value of the total cost
Serial.print(”The price is: $”);//display ”The price is: $” on the screen
Serial.println(TotalCost);//display the value of the TotalCost, then go on to the next line
delay(1000);//wait for a second
}
```

Problem 8

```
//Team Names printed to the LCD
// include the library code:
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 2, en = 3, d4 = 4, d5 = 5, d6 = 6, d7 = 7, d8 = 8;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7, d8);
//To use BUS1 to drive the LCD, we need to change the LiquidCrystal lcd(12, 11, 5, 4, 3, 2); into LiquidCrystal lcd(2, 3, 4, 5, 6, 7, 8);.
```

```
void setup()
{
// set up the LCD’s number of columns and rows:
lcd.begin(16, 2);
// Print ”hello world!” to the LCD.
lcd.print(”hello, world!”);
}
void loop()
{
// Turn off the display:
lcd.noDisplay();
delay(500);//wait for 0.5 second
// Turn on the display:
lcd.display();
delay(500);//wait for 0.5 second
}
```

Problem 9

```
//cursor on LCD will be incrementally moved by one space, left or right, by signals from button bricks
// include the library code:
#include <LiquidCrystal.h>
#include <stdio.h>
//#include<iostream>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 2, en = 3, d4 = 4, d5 = 5, d6 = 6, d7 = 7, d8 = 8;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7, d8);
//To use BUS1 to drive the LCD, we need to change the LiquidCrystal lcd(12, 11, 5, 4, 3, 2); into LiquidCrystal lcd(2, 3, 4, 5, 6, 7, 8);.
void setup()
{
// set up the LCD’s number of columns and rows:
lcd.begin(16, 2);
// Print a message to the LCD.
// use the setCursor() method to reposition the cursor
lcd.setCursor(0,0);//topleft
lcd.print(”MING”);
lcd.setCursor(11,0);//topright
lcd.print(”WANDA”);
lcd.setCursor(0,1);//bottomleft
lcd.print(”LEQI”);
}
void loop()
{
// Turn off the display:
lcd.noDisplay();
delay(500);//wait for 0.5 second
// Turn on the display:
lcd.display();
delay(500);//wait for 0.5 second
}
```

Problem 10

//Can LED brick on slave be turned ON or OFF by signals over I^2C from a button brick on the master

```
// include the library code:
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 2, en = 3, d4 = 4, d5 = 5, d6 = 6, d7 = 7, d8 = 8;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7, d8);
int InitialPos = 0;
int Button1 = 9; //define the 9th digital pin for button1 brick
int Button2 = 10; //define the 10th digital pin for button2 brick
void setup()
{
pinMode(Button2,INPUT); //set the Button2 pin for digital input
pinMode(Button1,INPUT); //set the Button1 pin for digital input
// set up the LCD's number of columns and rows:
lcd.begin(16, 2);
// Print a message to the LCD.
lcd.setCursor(0,0);//topleft
lcd.print("X");//print "X" on the screen
}
void loop()
{
// Turn on the display:
lcd.display();
delay(500);
lcd.noDisplay();
// Turn off the display
if (!digitalRead(Button1) && digitalRead(Button2)) // if button1 pressed, button2 not pressed
{
if (0 == InitialPos)
InitialPos = 15;// if the previous positon of "X" is 0, the next postion is 15
else
InitialPos -= 1;// leftshift
}
if (!digitalRead(Button2) && digitalRead(Button1)) // if button2 pressed ,button1 not pressed
{
InitialPos = (InitialPos + 1) % 16;// rightshift }
if (digitalRead(Button2) && digitalRead(Button1)) // if both button1 and button2 pressed
{
//just keep in place }
lcd.clear();//clear the previous state
lcd.setCursor(InitialPos,0);
lcd.print("X");//print "X" in the new position
}
```

Problem 11

```
//code for master
#include <Wire.h>
void setup()
{
Wire.begin(); // join i2c bus (address optional for master)
pinMode(12,INPUT);
Serial.begin(9600); // start serial for output
}
byte x = 0;
void loop()
```

```
{
if(digitalRead(12))
{
Wire.beginTransmission(4); // transmit to device #4
Wire.write(”x is ”); // sends five bytes
Wire.write(x); // sends one byte
Wire.endTransmission(); // stop transmitting
Serial.print(”x is ”);
Serial.println(x);
x++;
delay(500);
}
}
//code for slave
#include <Wire.h>
void setup()
{
Wire.begin(4); // join i2c bus with address #4
Wire.onReceive(receiveEvent); // register event
pinMode(9,OUTPUT);
Serial.begin(9600); // start serial for output
}
void loop()
{
delay(100);
}
// function that executes whenever data is received from master
// this function is registered as an event, see setup()
void receiveEvent(int howMany)
{
while(1 <Wire.available()) // loop through all but the last
{
char c =Wire.read(); // receive byte as a character
Serial.print(c); // print the character }
int x = Wire.read(); // receive byte as an integer
Serial.println(x); // print the integer
if(2*(x/2)==x) digitalWrite(9,HIGH);
else digitalWrite(9,LOW);
}
```

References

[1] The C Programming Language, Brian W. Kernighan, Dennis M. Ritchie

[2] https://www.arduino.cc [Online]