Table of Contents

Part 1

1. Working with the development environment and target platform.
2. Modify the program so that LED is on for 2 seconds and off for 1 second and repeats.
3. Modify the program so that the LED has two 1 second blinks followed by two 2 second blinks and repeats.
4. blink the LED Brick rather than the onboard LED.
5. LED Brick can be controlled by the Button Brick.
6. Processor Outputting Your Stuff

Part 2

1. Writing to the LCD
2. Working With the LCD and Peripherals
3. Communicating Between Two Microprocessors
4. Working With the Two Processors and Peripherals

Abstract

This lab is mainly about controlling LED, learn how to connect UNO with other peripherals, and communicate between two microprocessors.

Introduction

The first part has three main purposes. To introduce the C language, to introduce the Arduino UNO microprocessor board, and to work with the basic Blink program. The second part of the lab we will work with LCD, learning how send information from one microprocessor to a second to control a peripheral device attached to the second microprocessor.

During this project we will use C language, LED, LCD, Arduion UNO microcontroller, Electronic Brick , button switch , etc.

Discussion of the lab

1.Brief Design Specification

Part 1

1.Requirements: Working with the development environment and target platform.

Input: select Blink, verify and upload it.

Output: the LED on the board start blinking

2.Requirements: Modify the program so that LED is on for 2 seconds and off for 1 second and repeats.

Input: change delay to (2000)(two seconds)

Output: LED is on for 2 seconds and off for 1 second

3.Requirements: Modify the program so that the LED has two 1 second blinks followed by two 2 second blinks and repeats.

Input: modify the delay between LED, HIGH and LOW.

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

4.Requirements: blink the LED Brick rather than the onboard LED.

Input: put Red Brick on UNO, insert Brick LED in D9 pin (as the code set)

Output: LED Brick blinks

5.Requirements: LED Brick can be controlled by the Button Brick.

Input: connect brick button to D10 pin (as the code set)

Output: LED goes out when press the button

6.Requirements: Processor Outputting Your Stuff

Input: serial.print(total\_price())

Output: get the total price from serial monitor.

Part 2

1.Requirements:

2.Requirements: design a program that starts by writing the character ‘X’ to position row 0 and column 0 on the LCD. Next, design the code that will read the states of the two buttons. When button1 is pressed, move the character ‘X’ to the left by 1 position (column). When button2 is pressed, move the character ‘X’ to the right by 1 position (column).

Input: put Brick on UNO, connect BUS1 to LCD

Output: x shows, two buttons control x’s position

3.Requirements: Communicating Between Two Microprocessors

Input: connect two USB port to two UNO, separately connect them with Brick than connect them using pin4 and pin5.

Output: open monitor, we can see master send information and slave catches them.

4.Requirements: make the button on the master microprocessor control the LED on the slave processor.

Input: connect button to pin 10 on the master processor, connect LED to pin 9 on the slave processor

Output: when press the button, LED will blink, and LED stop blinking when you release the button

2.Hardware Implementation

Program3,4:

Serial Monitor

LED

Brick Chasis

Arduino UNO

9

Program5

Button

Serial Monitor

10

LED

Brick Chasis

Arduino UNO

9

Problem 8

LCD

Serial Monitor

Arduino UNO

Brick Chasis

BUS1

Program9

BUTTON1

LCD

Serial Monitor

Brick Chasis

Arduino UNO

BUS1

BUTTON2

Program11

Serial Monitor

Arduino UNO1

Arduino UN21

4,5

LED

BUTTON

I2 C

BUTTON

3.Software Implementation

Function of program9 is controlling ‘x’ on LCD by button1 and button2. So after initializing the position of ‘x’ (lcd.(begin(16,2)), we set button1 to pin 9 and button2 to pin 10, if you press button1 and coordinate (the position of ‘x’) is greater than zero, then ‘x’ moves to the left, if press button2 and coordinate is less than 15 then ‘x’ moves to the right. When the ‘x’ reaches the border of the screen and you keep moving it, the code of movement won’t execute, in other word, it won’t move.

Function of program11 is controlling the Brick LED on the slave processor by press the button on the master processor. I initialized a variable on as false, when the button was pressed, on was assigned true, when release the button on was assigned false. We use Wire.beginTransmission(4) (transmit to device using pin4 and pin5), use Wire.write(on) sending signal to slave processor, use Wire.endTransmission() to stop transmitting, then enter the function of receiveEvent() in slave processor, light or turn off the LED by the value of variable on.

Test Plan

In part 1, because of the requirement is relatively obvious so that basically we just see if the LED is blinking or the monitor to verify the code, all the test measurements are presented on the Inputs and Outputs of each requirement above.

In part 2, in program 4 I added serial.println(on) in slave port to check out if slave has received the right command.

Presentation, discussion, and analysis of the results

Part 1

1. : the LED on the board start blinking (through command turn the LED on)
2. : LED is on for 2 seconds and off for 1 second (change the delay time)
3. : LED has two 1 second blinks followed by two 2 second blinks and repeats.
4. : LED Brick blinks
5. LED goes out when press the button
6. get the total price from serial monitor.

Part 2

1. LCD shows the names of group members.

2. x shows, two buttons control x’s position

3. open monitor, we can see master send information and slave catches them.

4. when press the button, LED will blink, and LED stop blinking when you release the button

Analysis of any errors

We met some problems in part1 4 program in which Brick LED didn’t blink, it turns out the Brick LED didn’t work. And at the last program I don’t think connect pin5 between to processor is necessary, causing communication breaks, I changed the code lots of times before I found it.

Summary and conclusion

The project included controlling onboard LED using Arduino, controlling LED Brick by Button Brick, Printing variables on the UNO, printing letters to the LCD, setting cursor on LCD and control it by button bricks, and controlling peripherals on slave processor by button on master processor.

Appendices