**Program Code for Lab 6 and 7 Exercises**

BMEN 2151 *Introductory Medical Device Prototyping*

Prof. Steven S. Saliterman

**Exercise 6-3**

**const int LED = 10;**

**int blinks = 5; // blink 5 times;**

**bool done = false;**

**void setup()**

**{**

**pinMode(LED, OUTPUT); //set pin 10 as an OUTPUT**

**digitalWrite(LED, LOW); // Initialize off**

**}**

**void loop()**

**{**

**while (done != true)**

**{**

**for (int i = 1; i<= blinks; ++i) // ++i same as i = i+1**

**{**

**digitalWrite(LED, HIGH); // Turn on LED**

**delay(500); //Pause**

**digitalWrite(LED, LOW); // Turn off LED**

**delay(500); //Pause**

**}**

**done = true;**

**}**

**}**

**Exercise 6-4**

**const int analogPin = A0;   // pin that the potentiometer is attached to    
const int ledPin = 13;       // pin that the LED is attached to on UNO   
const int threshold = 400; // an arbitrary threshold level that's in the range of**

**// the analog input   
  
void setup()**

**{  
 pinMode(ledPin, OUTPUT); // initialize the LED pin as an output   
 Serial.begin(9600); // initialize serial communications   
}  
  
void loop()**

**{  
  int analogValue = analogRead(analogPin);  // read the value of the potentiometer   
  if (analogValue > threshold) // if the analog value is high enough, turn on the LED**

**{  
    digitalWrite(ledPin, HIGH);  
  }**

**else**

**{  
    digitalWrite(ledPin, LOW);  
  }  
 Serial.println(analogValue); // print the analog value   
 delay(1);         // delay in between reads for stability**

**}**

**Exercise 6-5**

**int button = 2; // pin to connect the button**

**int presses = 0; // variable to store number of presses**

**long time = 0; // used for debounce**

**long debounce = 100; // how many ms to "debounce"**

**const byte numPins = 8; // how many LEDs**

**int state; // used for HIGH or LOW**

**byte pins[] = {5, 6, 7, 8, 9, 10, 11, 12}; // LED Pins**

**void count() // function count the button presses**

**{**

**if(millis() - time > debounce) presses++; //debounce pushbutton**

**time = millis();**

**}**

**void setup()**

**{**

**for(int i = 0; i < numPins; i++) // set LED pins to outputs**

**{**

**pinMode(pins[i], OUTPUT);**

**}**

**pinMode(button, INPUT);**

**attachInterrupt(0, count, LOW); // pin 2 is interrupt 0 on UNO**

**}**

**void loop()**

**{**

**/\* convert presses to binary and store it as a string \*/**

**String binNumber = String(presses, BIN);**

**int binLength = binNumber.length(); //get length of string**

**if(presses <= 255) // if we have less or equal to 255 presses**

**{**

**for(int i = 0, x = 1; i < binLength; i++, x+=2)**

**{**

**if(binNumber[i] == '0') state = LOW;**

**if(binNumber[i] == '1') state = HIGH;**

**digitalWrite(pins[i] + binLength - x, state);**

**}**

**// do something when we reach 255**

**}**

**}**

**// http://www.electroschematics.com/9809/arduino-8-bit-binary-LED**

**Exercise 7-2**

**/\***

**This is a test sketch for the Adafruit assembled Motor Shield for Arduino v2**

**It won't work with v1.x motor shields! Only for the v2's with built in PWM**

**control**

**For use with the Adafruit Motor Shield v2**

**----> http://www.adafruit.com/products/1438**

**This sketch creates a fun motor party on your desk \*whiirrr\***

**Connect a unipolar/bipolar stepper to M3/M4**

**Connect a DC motor to M1**

**Connect a hobby servo to SERVO1**

**\*/**

**#include <Wire.h>**

**#include <Adafruit\_MotorShield.h>**

**#include "utility/Adafruit\_MS\_PWMServoDriver.h"**

**#include <Servo.h>**

**// Create the motor shield object with the default I2C address**

**Adafruit\_MotorShield AFMS = Adafruit\_MotorShield();**

**// Or, create it with a different I2C address (say for stacking)**

**// Adafruit\_MotorShield AFMS = Adafruit\_MotorShield(0x61);**

**// Connect a stepper motor with 200 steps per revolution (1.8 degree)**

**// to motor port #2 (M3 and M4)**

**Adafruit\_StepperMotor \*myStepper = AFMS.getStepper(200, 2);**

**// And connect a DC motor to port M1**

**Adafruit\_DCMotor \*myMotor = AFMS.getMotor(1);**

**// We'll also test out the built in Arduino Servo library**

**Servo servo1;**

**void setup() {**

**Serial.begin(9600); // set up Serial library at 9600 bps**

**Serial.println("MMMMotor party!");**

**AFMS.begin(); // create with the default frequency 1.6KHz**

**//AFMS.begin(1000); // OR with a different frequency, say 1KHz**

**// Attach a servo to pin #10**

**servo1.attach(10);**

**// turn on motor M1**

**myMotor->setSpeed(200);**

**myMotor->run(RELEASE);**

**// setup the stepper**

**myStepper->setSpeed(10); // 10 rpm**

**}**

**int i;**

**void loop() {**

**myMotor->run(FORWARD);**

**for (i=0; i<255; i++) {**

**servo1.write(map(i, 0, 255, 0, 180));**

**myMotor->setSpeed(i);**

**myStepper->step(1, FORWARD, INTERLEAVE);**

**delay(3);**

**}**

**for (i=255; i!=0; i--) {**

**servo1.write(map(i, 0, 255, 0, 180));**

**myMotor->setSpeed(i);**

**myStepper->step(1, BACKWARD, INTERLEAVE);**

**delay(3);**

**}**

**myMotor->run(BACKWARD);**

**for (i=0; i<255; i++) {**

**servo1.write(map(i, 0, 255, 0, 180));**

**myMotor->setSpeed(i);**

**myStepper->step(1, FORWARD, DOUBLE);**

**delay(3);**

**}**

**for (i=255; i!=0; i--) {**

**servo1.write(map(i, 0, 255, 0, 180));**

**myMotor->setSpeed(i);**

**myStepper->step(1, BACKWARD, DOUBLE);**

**delay(3);**

**}**

**}**