**// Include Libraries**

**#include "Arduino.h"**

**#include "DCMotor.h"**

**#include "Wire.h"**

**#include "SPI.h"**

**#include "Adafruit\_SSD1306.h"**

**#include "Adafruit\_GFX.h"**

**#include "Encoder.h"**

**#include "Button.h"**

**#include "Servo.h"**

**// Pin Definitions**

**#define DCMOTOR\_PIN\_COIL1 4**

**#define OLED128X32\_PIN\_RST 5**

**#define ROTARYENCI\_PIN\_CLK 2**

**#define ROTARYENCI\_PIN\_D 3**

**#define ROTARYENCI\_PIN\_S1 6**

**#define SERVO360MICRO1\_1\_PIN\_SIG 7**

**#define SERVO360MICRO2\_2\_PIN\_SIG 8**

**#define WHEELENCODER\_INT\_PIN\_CH\_A 18**

**#define WHEELENCODER\_INT\_PIN\_CH\_B 19**

**// Global variables and defines**

**long rotaryEncIOldPosition = 0;**

**// object initialization**

**DCMotor dcMotor(DCMOTOR\_PIN\_COIL1);**

**#define SSD1306\_LCDHEIGHT 32**

**Adafruit\_SSD1306 oLed128x32(OLED128X32\_PIN\_RST);**

**Encoder rotaryEncI(ROTARYENCI\_PIN\_D,ROTARYENCI\_PIN\_CLK);**

**Button rotaryEncIButton(ROTARYENCI\_PIN\_S1);**

**Servo servo360Micro1\_1;**

**Servo servo360Micro2\_2;**

**// define vars for testing menu**

**const int timeout = 10000; //define timeout of 10 sec**

**char menuOption = 0;**

**long time0;**

**// Setup the essentials for your circuit to work. It runs first every time your circuit is powered with electricity.**

**void setup()**

**{**

**// Setup Serial which is useful for debugging**

**// Use the Serial Monitor to view printed messages**

**Serial.begin(9600);**

**while (!Serial) ; // wait for serial port to connect. Needed for native USB**

**Serial.println("start");**

**oLed128x32.begin(SSD1306\_SWITCHCAPVCC, 0x3C); // initialize with the I2C addr 0x3C (for the 128x32)**

**oLed128x32.clearDisplay(); // Clear the buffer.**

**oLed128x32.display();**

**rotaryEncIButton.init();**

**pinMode(ROTARYENCI\_PIN\_S1, INPUT\_PULLUP);**

**menuOption = menu();**

**}**

**// Main logic of your circuit. It defines the interaction between the components you selected. After setup, it runs over and over again, in an eternal loop.**

**void loop()**

**{**

**if(menuOption == '1') {**

**// DC Motor - Hobby Motor Gear - Test Code**

**// The DC motor will turn on and off for 4000ms (4 sec)**

**dcMotor.on(200); // 1. turns on**

**delay(4000); // 2. waits 4000 milliseconds (4 sec). change the value in the brackets (4000) for a longer or shorter delay.**

**dcMotor.off(); // 3. turns off**

**delay(4000); // 4. waits 4000 milliseconds (4 sec). change the value in the brackets (4000) for a longer or shorter delay.**

**}**

**else if(menuOption == '2') {**

**// Monochrome 128x32 I2C OLED graphic display - Test Code**

**oLed128x32.setTextSize(1);**

**oLed128x32.setTextColor(WHITE);**

**oLed128x32.setCursor(0, 10);**

**oLed128x32.clearDisplay();**

**oLed128x32.println("Circuito.io Rocks!");**

**oLed128x32.display();**

**delay(1);**

**oLed128x32.startscrollright(0x00, 0x0F);**

**delay(2000);**

**oLed128x32.stopscroll();**

**delay(1000);**

**oLed128x32.startscrollleft(0x00, 0x0F);**

**delay(2000);**

**oLed128x32.stopscroll();**

**}**

**else if(menuOption == '3') {**

**// Rotary Encoder Module - Test Code**

**//Read encoder new position**

**long rotaryEncINewPosition = rotaryEncI.read();**

**bool rotaryEncIButtonVal = rotaryEncIButton.onPress();**

**if (rotaryEncINewPosition != rotaryEncIOldPosition || rotaryEncIButtonVal) {**

**rotaryEncIOldPosition = rotaryEncINewPosition;**

**Serial.print(F("Pos: "));**

**Serial.print(rotaryEncINewPosition);**

**Serial.print(F("\tButton: "));**

**Serial.println(rotaryEncIButtonVal);**

**}**

**}**

**else if(menuOption == '4') {**

**// Continuous Rotation Micro Servo - FS90R #1 - Test Code**

**// The servo will rotate CW in full speed, CCW in full speed, and will stop with an interval of 2000 milliseconds (2 seconds)**

**servo360Micro1\_1.attach(SERVO360MICRO1\_1\_PIN\_SIG); // 1. attach the servo to correct pin to control it.**

**servo360Micro1\_1.write(180); // 2. turns servo CW in full speed. change the value in the brackets (180) to change the speed. As these numbers move closer to 90, the servo will move slower in that direction.**

**delay(2000); // 3. waits 2000 milliseconds (2 sec). change the value in the brackets (2000) for a longer or shorter delay in milliseconds.**

**servo360Micro1\_1.write(0); // 4. turns servo CCW in full speed. change the value in the brackets (0) to change the speed. As these numbers move closer to 90, the servo will move slower in that direction.**

**delay(2000); // 5. waits 2000 milliseconds (2 sec). change the value in the brackets (2000) for a longer or shorter delay in milliseconds.**

**servo360Micro1\_1.write(90); // 6. sending 90 stops the servo**

**delay(2000); // 7. waits 2000 milliseconds (2 sec). change the value in the brackets (2000) for a longer or shorter delay in milliseconds.**

**servo360Micro1\_1.detach(); // 8. release the servo to conserve power. When detached the servo will NOT hold it's position under stress.**

**}**

**else if(menuOption == '5') {**

**// Continuous Rotation Micro Servo - FS90R #2 - Test Code**

**// The servo will rotate CW in full speed, CCW in full speed, and will stop with an interval of 2000 milliseconds (2 seconds)**

**servo360Micro2\_2.attach(SERVO360MICRO2\_2\_PIN\_SIG); // 1. attach the servo to correct pin to control it.**

**servo360Micro2\_2.write(180); // 2. turns servo CW in full speed. change the value in the brackets (180) to change the speed. As these numbers move closer to 90, the servo will move slower in that direction.**

**delay(2000); // 3. waits 2000 milliseconds (2 sec). change the value in the brackets (2000) for a longer or shorter delay in milliseconds.**

**servo360Micro2\_2.write(0); // 4. turns servo CCW in full speed. change the value in the brackets (0) to change the speed. As these numbers move closer to 90, the servo will move slower in that direction.**

**delay(2000); // 5. waits 2000 milliseconds (2 sec). change the value in the brackets (2000) for a longer or shorter delay in milliseconds.**

**servo360Micro2\_2.write(90); // 6. sending 90 stops the servo**

**delay(2000); // 7. waits 2000 milliseconds (2 sec). change the value in the brackets (2000) for a longer or shorter delay in milliseconds.**

**servo360Micro2\_2.detach(); // 8. release the servo to conserve power. When detached the servo will NOT hold it's position under stress.**

**}**

**else if(menuOption == '6')**

**{**

**// Disclaimer: The Photoelectric Speed Wheel Encoder is in testing and/or doesn't have code, therefore it may be buggy. Please be kind and report any bugs you may find.**

**}**

**if (millis() - time0 > timeout)**

**{**

**menuOption = menu();**

**}**

**}**

**// Menu function for selecting the components to be tested**

**// Follow serial monitor for instrcutions**

**char menu()**

**{**

**Serial.println(F("\nWhich component would you like to test?"));**

**Serial.println(F("(1) DC Motor - Hobby Motor Gear"));**

**Serial.println(F("(2) Monochrome 128x32 I2C OLED graphic display"));**

**Serial.println(F("(3) Rotary Encoder Module"));**

**Serial.println(F("(4) Continuous Rotation Micro Servo - FS90R #1"));**

**Serial.println(F("(5) Continuous Rotation Micro Servo - FS90R #2"));**

**Serial.println(F("(6) Photoelectric Speed Wheel Encoder"));**

**Serial.println(F("(menu) send anything else or press on board reset button\n"));**

**while (!Serial.available());**

**// Read data from serial monitor if received**

**while (Serial.available())**

**{**

**char c = Serial.read();**

**if (isAlphaNumeric(c))**

**{**

**if(c == '1')**

**Serial.println(F("Now Testing DC Motor - Hobby Motor Gear"));**

**else if(c == '2')**

**Serial.println(F("Now Testing Monochrome 128x32 I2C OLED graphic display"));**

**else if(c == '3')**

**Serial.println(F("Now Testing Rotary Encoder Module"));**

**else if(c == '4')**

**Serial.println(F("Now Testing Continuous Rotation Micro Servo - FS90R #1"));**

**else if(c == '5')**

**Serial.println(F("Now Testing Continuous Rotation Micro Servo - FS90R #2"));**

**else if(c == '6')**

**Serial.println(F("Now Testing Photoelectric Speed Wheel Encoder - note that this component doesn't have a test code"));**

**else**

**{**

**Serial.println(F("illegal input!"));**

**return 0;**

**}**

**time0 = millis();**

**return c;**

**}**

**}**

**}**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\* Circuito.io is an automatic generator of schematics and code for off**

**\* the shelf hardware combinations.**

**\* Copyright (C) 2016 Roboplan Technologies Ltd.**

**\* This program is free software: you can redistribute it and/or modify**

**\* it under the terms of the GNU General Public License as published by**

**\* the Free Software Foundation, either version 3 of the License, or**

**\* (at your option) any later version.**

**\* This program is distributed in the hope that it will be useful,**

**\* but WITHOUT ANY WARRANTY; without even the implied warranty of**

**\* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the**

**\* GNU General Public License for more details.**

**\* You should have received a copy of the GNU General Public License**

**\* along with this program. If not, see <http://www.gnu.org/licenses/>.**

**\* In addition, and without limitation, to the disclaimers of warranties**

**\* stated above and in the GNU General Public License version 3 (or any**

**\* later version), Roboplan Technologies Ltd. ("Roboplan") offers this**

**\* program subject to the following warranty disclaimers and by using**

**\* this program you acknowledge and agree to the following:**

**\* THIS PROGRAM IS PROVIDED ON AN "AS IS" AND "AS AVAILABLE" BASIS, AND**

**\* WITHOUT WARRANTIES OF ANY KIND EITHER EXPRESS OR IMPLIED. ROBOPLAN**

**\* HEREBY DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT**

**\* NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, TITLE, FITNESS**

**\* FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, AND THOSE ARISING BY**

**\* STATUTE OR FROM A COURSE OF DEALING OR USAGE OF TRADE.**

**\* YOUR RELIANCE ON, OR USE OF THIS PROGRAM IS AT YOUR SOLE RISK.**

**\* ROBOPLAN DOES NOT GUARANTEE THAT THE PROGRAM WILL BE FREE OF, OR NOT**

**\* SUSCEPTIBLE TO, BUGS, SECURITY BREACHES, OR VIRUSES. ROBOPLAN DOES**

**\* NOT WARRANT THAT YOUR USE OF THE PROGRAM, INCLUDING PURSUANT TO**

**\* SCHEMATICS, INSTRUCTIONS OR RECOMMENDATIONS OF ROBOPLAN, WILL BE SAFE**

**\* FOR PERSONAL USE OR FOR PRODUCTION OR COMMERCIAL USE, WILL NOT**

**\* VIOLATE ANY THIRD PARTY RIGHTS, WILL PROVIDE THE INTENDED OR DESIRED**

**\* RESULTS, OR OPERATE AS YOU INTENDED OR AS MAY BE INDICATED BY ROBOPLAN.**

**\* YOU HEREBY WAIVE, AGREE NOT TO ASSERT AGAINST, AND RELEASE ROBOPLAN,**

**\* ITS LICENSORS AND AFFILIATES FROM, ANY CLAIMS IN CONNECTION WITH ANY OF**

**\* THE ABOVE.**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**