#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#define OLED\_RESET 4

Adafruit\_SSD1306 display(OLED\_RESET);

#include <Servo.h> // Include servo library

#include <boarddefs.h>

#include <IRremote.h>

#include <IRremoteInt.h>

#include <ir\_Lego\_PF\_BitStreamEncoder.h>

#define RECV\_PIN 7

IRrecv irrecv(RECV\_PIN);

Servo servoLeft; // Declare left and right servos

Servo servoRight;

enum ir\_command {Undefined, Forward, Reverse, Stop,

LeftSpin, RightSpin, LeftForward, RightForward, LeftReverse, RightReverse,

Record, Replay1, Replay2, Replay3, Replay4};

void setup() // Built-in initialization block

{

servoRight.attach(12);

servoLeft.attach(11);

irrecv.enableIRIn();

irrecv.blink13(true);

pinMode(6,OUTPUT);

digitalWrite(6,LOW);

Serial.begin(9600);

Serial.println("Project: IR remote-controlled robot car");

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C); // initialize with the I2C addr for the 128x64 OLED

display.display(); // show splashscreen

delay(2000);

display.clearDisplay(); // clears the screen and

}

void loop(){

decode\_results results;

enum ir\_command ir\_key;

//char \*banner = "ROBOT CAR";

byte bx = display.width()/2;

byte by = display.height()/2;

if (irrecv.decode(&results)){

switch (results.decode\_type){

case NEC: Serial.print("NEC"); break ;

case SONY: Serial.print("SONY"); break ;

case RC5: Serial.print("RC5"); break ;

case RC6: Serial.print("RC6"); break ;

case DISH: Serial.print("DISH"); break ;

case SHARP: Serial.print("SHARP"); break ;

case JVC: Serial.print("JVC"); break ;

case SANYO: Serial.print("SANYO"); break ;

case MITSUBISHI: Serial.print("MITSUBISHI"); break ;

case SAMSUNG: Serial.print("SAMSUNG"); break ;

case LG: Serial.print("LG"); break ;

case WHYNTER: Serial.print("WHYNTER"); break ;

case AIWA\_RC\_T501: Serial.print("AIWA\_RC\_T501"); break ;

case PANASONIC: Serial.print("PANASONIC"); break ;

case DENON: Serial.print("DENON"); break ;

default:

case UNKNOWN: Serial.print("UNKNOWN"); break ;

}

Serial.print(": ");

if (results.decode\_type == NEC && results.value == REPEAT) {

Serial.println("repeat");

} else {

Serial.println(results.value, HEX);

}

irrecv.resume();

switch (results.value) {

case 0xFFAA55: ir\_key = Forward; by = 10; break;

case 0xFF629D: ir\_key = Reverse; by = display.height()-10; break;

case 0xFF6A95: ir\_key = Stop; break;

case 0xFF5AA5: ir\_key = LeftSpin; bx = 10; break;

case 0xFF4AB5: ir\_key = RightSpin; bx = display.width()-10; break;

case 0xFF9A65: ir\_key = LeftForward; bx = 10; by = 10; break;

case 0xFF42BD: ir\_key = RightForward; bx = display.width()-10; by = 10; break;

case 0xFF52AD: ir\_key = LeftReverse; bx = 10; by = display.height()-10; break;

case 0xFF8A75: ir\_key = RightReverse; bx = display.width()-10; by = display.height()-10; break;

case 0xA1B79867: ir\_key = Record; break;

case 0xFF12ED: ir\_key = Replay1; break;

case 0xFF32CD: ir\_key = Replay2; break;

case 0xFF22DD: ir\_key = Replay3; break;

case 0xFF02FD: ir\_key = Replay4; break;

default: ir\_key = Undefined; break;

}

if (ir\_key != Undefined) {

// blink led

digitalWrite(6,HIGH);

delay(10);

digitalWrite(6,LOW);

#if 0

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(30,20);

int len=strlen(banner);

for (int idx=0; idx<len; idx++) {

display.write(banner[idx]);

}

//display.print(banner);

display.display();

display.clearDisplay();

#else

// draw a white circle, 10 pixel radius

display.fillCircle(bx, by, 10, WHITE);

display.display();

display.clearDisplay();

#endif

ir\_action(ir\_key);

}

}

}

void ir\_action(enum ir\_command ir\_key)

{

// speedLeft, speedRight ranges: Backward Linear Stop Linear Forward

// -200 -100......0......100 200

if (ir\_key == Forward) {

servoLeft.writeMicroseconds(1700);

servoRight.writeMicroseconds(1300);

} else if (ir\_key == Reverse) {

servoLeft.writeMicroseconds(1300);

servoRight.writeMicroseconds(1700);

} else if (ir\_key == Stop) {

servoLeft.writeMicroseconds(1500);

servoRight.writeMicroseconds(1500);

} else if (ir\_key == LeftSpin) {

servoLeft.writeMicroseconds(1300);

servoRight.writeMicroseconds(1300);

} else if (ir\_key == RightSpin) {

servoLeft.writeMicroseconds(1700);

servoRight.writeMicroseconds(1700);

} else if (ir\_key == LeftForward) {

servoLeft.writeMicroseconds(1550);

servoRight.writeMicroseconds(1300);

} else if (ir\_key == RightForward) {

servoLeft.writeMicroseconds(1700);

servoRight.writeMicroseconds(1450);

} else if (ir\_key == LeftReverse) {

servoLeft.writeMicroseconds(1450);

servoRight.writeMicroseconds(1700);

} else if (ir\_key == RightReverse) {

servoLeft.writeMicroseconds(1300);

servoRight.writeMicroseconds(1550);

}

}