#include <Servo.h>

#define servoBase 3 // Porta Digital do Servo da Base

#define servoAlturaBraco 4 // Porta Digital do Servo da Altura do BraÃ§o

#define servoAnguloBraco 5 // Porta Digital do Servo do Ã‚ngulo do BraÃ§o

#define servoGarra 6 // Porta Digital do Servo da Garra

#define potBase A0 // Porta AnalÃ³gica do Potenciometro para Controle da Base

#define potAlturaBraco A1 // Porta AnalÃ³gica do Potenciometro para Controle da Altura do BraÃ§o

#define potAnguloBraco A2 // Porta AnalÃ³gica do Potenciometro para Controle do Ã‚ngulo do BraÃ§o

#define potGarra A3 // Porta AnalÃ³gica do Potenciometro para Controle da Garra.

#define botaoCongela 8 // Porta Digital do Botao Congela

#define botaoDescongela 9 // Porta Digital do Botao Descongela

//InstanciaÃ§Ã£o dos Objetos de Controle dos Servos

Servo base;

Servo altura;

Servo angulo;

Servo garra;

//Variaveis para cÃ¡lculo dos angulos

int joyAX = A0;

int joyAY = A2;

int joyBX = A1;

int joyBY = A3;

int AX, AY, BX, BY;

//Variaveis para controle dos botÃµes

int estadoBotaoC, estadoBotaoD;

int estadoAntBotaoC = HIGH;

int estadoAntBotaoD = HIGH;

boolean congelado = false;

void setup(){

pinMode(8, INPUT\_PULLUP);

pinMode(9, INPUT\_PULLUP);

//Usa o LED do Arduino para informar estado de congelado

pinMode(13, OUTPUT);

digitalWrite(13, LOW);

//ConfiguraÃ§Ã£o das Portas dos Servos

base.attach(3);

altura.attach(4);

angulo.attach(5);

garra.attach(6);

}

void loop() {

if (!congelado) {

AX = joyAX;

AY = joyAY;

BX = joyBX;

BY = joyBY;

AX = analogRead(joyAX); //Leitura do Angulo do Potenciometro

AX = map(analogRead(joyAX), 0, 1023, 900, 2100); //ConversÃ£o do valor do potenciometro (de 0 atÃ© 1024) para o angulo (de 0 atÃ© 180)

base.write(AX); //Envio do angulo para o Servo

delay(15);

}

//Se o botao de congelar foi apertado

estadoBotaoC = digitalRead(8);

if (estadoBotaoC != estadoAntBotaoC) {

congelado = true;

digitalWrite(13, HIGH); //Informa o estado de congelado no LED interno do Arduino

}

estadoAntBotaoC = estadoBotaoC;

//Se o botao de descongelar foi apertado

estadoBotaoD = digitalRead(9);

if (estadoBotaoD != estadoAntBotaoD) {

congelado = false;

digitalWrite(13, LOW); //Informa o estado de congelado no LED interno do Arduino

}

estadoAntBotaoD = estadoBotaoD;

}

// Código para teste de funcionamento do Braço Robótico para Arduino;

// Controle de movimento realizado por Joysticks;

// Usinainfo - www.usinainfo.com.br

#include <VarSpeedServo.h> // Inclui a Biblioteca VarSpeedServo.h

VarSpeedServo servo\_sobe; //Cria objeto para controlar o servo sobe

VarSpeedServo servo\_frente; //Cria objeto para controlar o servo frente

VarSpeedServo servo\_garra; //Cria objeto para controlar o servo garra

VarSpeedServo servo\_corpo; //Cria objeto para controlar o servo corpo

int pino\_x = A0; //Inicializa o pino analógico ao eixo X do joystick

int pino\_y = A1; //Inicializa o pino analógico ao eixo Y do joystick

int pino\_z = A3; //Inicializa o pino analógico ao eixo Z do joystick

int pino\_w = A4; //Inicializa o pino analógico ao eixo W do joystick

int val\_x; //Armazena o valor lido pelo eixo X do joystick

int val\_y; //Armazena o valor lido pelo eixo Y do joystick

int val\_z; //Armazena o valor lido pelo eixo Z do joystick

int val\_w; //Armazena o valor lido pelo eixo W do joystick

void setup() {

servo\_sobe.attach(3, 1, 180); //Define que o servo está conectado a porta 5 do Arduino

servo\_frente.attach(4, 1, 180); //Define que o servo está conectado a porta 3 do Arduino

servo\_garra.attach(5, 1, 180); //Define que o servo está conectado a porta 10 do Arduino

servo\_corpo.attach(6, 1, 180); //Define que o servo está conectado a porta 11 do Arduino

}

void loop() {

val\_x = analogRead(pino\_x); //Recebe o valor lido pelo eixo X do joystick

val\_x = map(val\_x, 0, 1023, 1, 180); //Converte o valor lido para um valor em graus (1 a 180º)

servo\_sobe.slowmove(val\_x, 60); //Movimenta o servo até a posição definida pelo eixo X

val\_y = analogRead(pino\_y); //Recebe o valor lido pelo eixo Y do joystick

val\_y = map(val\_y, 0, 1023, 1, 180); //Converte o valor lido para um valor em graus (1 a 180º)

servo\_frente.slowmove(val\_y, 60); //Movimenta o servo até a posição definida pelo eixo Y

val\_z = analogRead(pino\_z); //Recebe o valor lido pelo eixo Z do joystick

val\_z = map(val\_z, 0, 1023, 1, 180); //Converte o valor lido para um valor em graus (1 a 180º)

servo\_garra.slowmove(val\_z, 60); //Movimenta o servo até a posição definida pelo eixo Z

val\_w = analogRead(pino\_w); //Recebe o valor lido pelo eixo W do joystick

val\_w = map(val\_w, 0, 1023, 1, 180); //Converte o valor lido para um valor em graus (1 a 180º)

servo\_corpo.slowmove(val\_w, 60); //Movimenta o servo até a posição definida pelo eixo W

delay(30);

}

#include <Servo.h>

#define servoBase 3 // Porta Digital do Servo da Base

#define servoAlturaBraco 4 // Porta Digital do Servo da Altura do Braço

#define servoAnguloBraco 5 // Porta Digital do Servo do Ângulo do Braço

#define servoGarra 6 // Porta Digital do Servo da Garra

#define potBase A0 // Porta Analógica do Potenciometro para Controle da Base

#define potAlturaBraco A1 // Porta Analógica do Potenciometro para Controle da Altura do Braço

#define potAnguloBraco A2 // Porta Analógica do Potenciometro para Controle do Ângulo do Braço

#define potGarra A3 // Porta Analógica do Potenciometro para Controle da Garra.

#define botaoCongela 8 // Porta Digital do Botao Congela

#define botaoDescongela 9 // Porta Digital do Botao Descongela

//Instanciação dos Objetos de Controle dos Servos

Servo base;

Servo altura;

Servo angulo;

Servo garra;

//Variaveis para cálculo dos angulos

int valorPotBase;

int valorPotAltura;

int valorPotAngulo;

int valorPotGarra;

int joya = A0;

int joyb = A1;

int joyc = A2;

int joyd = A3;

//Variaveis para controle dos botões

int estadoBotaoC, estadoBotaoD;

int estadoAntBotaoC = HIGH;

int estadoAntBotaoD = HIGH;

boolean congelado = false;

void setup(){

pinMode(8, INPUT\_PULLUP);

pinMode(9, INPUT\_PULLUP);

//Usa o LED do Arduino para informar estado de congelado

pinMode(13, OUTPUT);

digitalWrite(13, LOW);

//Configuração das Portas dos Servos

base.attach(base);

altura.attach(altura);

angulo.attach(angulo);

garra.attach(garra);

}

void loop() {

if (!congelado) {

valorPotBase = analogRead(joya); //Leitura do Angulo do Potenciometro

valorPotBase = map(valorPotBase, 0, 1023, 180, 1); //Conversão do valor do potenciometro (de 0 até 1024) para o angulo (de 0 até 180)

base.write(valorPotBase); //Envio do angulo para o Servo

valorPotAltura = analogRead(joyb); //Leitura do Angulo do Potenciometro

valorPotAltura = map(valorPotAltura, 0, 1023, 1, 180); //Conversão do valor do potenciometro (de 0 até 1024) para o angulo (de 0 até 180)

altura.write(valorPotAltura); //Envio do angulo para o Servo

valorPotAngulo = analogRead(joyc); //Leitura do Angulo do Potenciometro

valorPotAngulo = map(valorPotAngulo, 0, 1023, 1, 180); //Conversão do valor do potenciometro (de 0 até 1024) para o angulo (de 0 até 180)

angulo.write(valorPotAngulo);

valorPotGarra = analogRead(joyd); //Leitura do Angulo do Potenciometro

valorPotGarra = map(valorPotGarra, 0, 1023, 1, 180); //Conversão do valor do potenciometro (de 0 até 1024) para o angulo (de 0 até 180)

garra.write(valorPotGarra);

delay(30);

}

//Se o botao de congelar foi apertado

estadoBotaoC = digitalRead(8);

if (estadoBotaoC != estadoAntBotaoC) {

congelado = true;

digitalWrite(13, HIGH); //Informa o estado de congelado no LED interno do Arduino

}

estadoAntBotaoC = estadoBotaoC;

//Se o botao de descongelar foi apertado

estadoBotaoD = digitalRead(9);

if (estadoBotaoD != estadoAntBotaoD) {

congelado = false;

digitalWrite(13, LOW); //Informa o estado de congelado no LED interno do Arduino

}

estadoAntBotaoD = estadoBotaoD;