Universidad de Sevilla
Centro Nacional de Aceleradores
Grupo de Física Nuclear Básica

Diseño de un controlador para el manipualdor 612-CLD50-C16

Andrés Gómez Ramos

agomez30@us.es

20 Septiembre 2016

Contribución

Éste es un proyecto que no está basado en un proyecto anterior.

El objetivo principal es utilizar un manipulador motorizado para utilizarlo en la camra intermedia de la línea FNB del acelerador tandem

Mi objetivo es el diseño del controlador del motor que el manipulador lleva incorporado.

Sumario

El propósito principal de este documento es mostrar el diseño paso a paso del controlador y enseñar como utilizarlo.

Tabla de contenidos

1 Introducción

El grupo FNB ha comprado un manipulador de la Empresa Allectra, tiene una precisión de 5 um. Posee un tornillo sin fin que, al dar una vuelta completa, avanza 1 mm. Para poder controlar el movimiento de dicho dispositivo de manera remota se ha comprado con un motor incorporado que permite controlarlo con un ordendor.

2 El problema ingenieril

El elemento principal es el controlador del manipulador. Se debe seleccionar un controlador preciso, barato e intuitivo.

3 Requerimientos

El controlador debe permitir al manipulador moverse con una precisión de 0,5 mm en ambos sentidos.

4 Posibles soluciones

4.1 Solución 1

El controlador indicado para controlar el manipulador es el modelo "SMCI33", permite controlar el movimiento del motor y detectar cuando llega al final de carrera.

Este dispositivo es conectado al ordenador mediante una conexión RS-232 y es conectado al manipulador con cables mediante una conexión RS-232.

Es necesario instalar un software incorporado a este controlador.

El precio del controlador es 300€.

4.2 Solución 2

Se ha buscado un driver de un motor paso a paso parecido al que se encuentra el manipulador. El modelo seleccionado es "DBP-0 3-2 0A-12-36V-1-128Mp-FM-B".

Este driver necesita recibir señales de paso, dirección y activación para poder funcionar.También será necesario una fuente de alimentación para dar potencia al motor a traves del driver.

Para controlar el motor se ha seleccionado una placa Arduino uno encargada de mandar las señales necesarias al driver. El precio del controlador ha sido de 45 € y ya se disponía de la placa arduino y de la fuente de alimentación.

4.3 Solución 3

Control del motor utilizando el driver EasyDriver 4.1 controlado mediante un Arduino utilizando la interfaz Lavbiew.

5 Análisis ingenieril de las soluciones planteadas

5.1 Análisis de la Solución 1

Esta solución fué la sugerida por allectra. Permite utilizar un driver que se sabe es compatible con el motor del manipulador. Además el software del control del motor es bastante intuitivo y fácil de utilizar.

Las principales desventajas de esta solución es la necesidad de utilizar un controlador bastante caro además de tener que instalar un software que no puede ser instalado en cualquier sistema operativo.

5.2 Análisis de la Solución 2

Esta solución es la que ofrece la mayor libertad de programación ya que se debe alterar tanto el controlador como el software que controlará el Arduino que mandará las instrucciones al controlador del motor.

Para controlar el motor se ha utilizado un código Arduino creado desde cero.

Esta solución será la menos intuitiva ya que, para controlar el manipulador, será necesario modificar el código de arduino.

Sin embargo es una solución que cuesta menos de 50€ y permite saber exactamente el número de pasos que da el motor y, por lo tanto, el movimiento del manipulador. Además permite introducir mejoras en el sistema.

5.3 Análisis de la solución 3

Esta solución permite utilizar un sistema intuitivo y programable de controlar el manipulador.

Para utilizar Arduino con Labview es necesario instalar la extensión “Labview Interface For Arduino” y el software de Arduino. Esta extensión incluye de serie un controlador para un motor paso a paso utilizando una placa Arduino y EasyDriver.

La principal desventaja de esta solución es que es necesario tener instalado Labview en el ordenador para lo que es necesario conseguir una licencia del CSIC y tener Windows como sistema operativo.

6 Conclusión

Se ha seleccionado la solución 2 (controlador DBP-0 3-2 0A-12-36V-1-128Mp-FM-B + Arduino) debido a una combinación de precio y manejabilidad de los elementos.

Se ajusta el número de pasos a 400 por vuelta bajando los interruptores SW6 y SW7 en el controlador. También se ajusta la intensidad de entrada al motor a 1 A bajando los interrupotres SW1 y SW2. El motor puede manejar de forma nominal hasta 2 A y de forma puntual 3 A. Aun así se ha seleccionado una corriente más baja para alargar la vida útil del motor.

El código creado se puede observar en el anexo A.

Además se ha creado un código que se utilizará como Limit Switch digital si aquel integrado en el manipulador no se activase. Puede observarse el código completo en el anexo B.

La placa Arduino es conectada al ordenador mediante un cable usb. La placa es conectada al controlador mediante cables tal y como se muestra en la tabla en el anexo C.

El controlador es conectado al manipulador mediante un conector RS-232 hembra como puede observarse en la tabla del anexo C.

La tabla del anexo C también muestra como conectar la placa Arduino UNO al manipulador para utilizar el Limit Switch y como conectar el controlador a la fuente de alimentación.

El anexo D muestra el código completo de movimiento y posicionamiento del manipulador.

Para su correcto funcionamiento, debe introducirse primero el valor “0” en “distancia” con lo que se activa la función de posicionamiento en el origen, el manipulador se moverá hasta el límite inferior y, desde ahí, alcanzará la posición de origen marcada como la posición de 52 mm en las marcas situadas en el manipulador. Desde este momento se mostrará en el “Monitor Serie” de la aplicación de Arduino la posición de nuestro manipulador.

Para poder mostrar la posición correcta por el "Monitor Serie" es necesario no modificar de forma manual la posición del manipulador ya que el sistema solo es capaz de recibir la posición del manipulador cuando alcanza alguno de los dos Limit Switch.

Se han diseñado dos sistemas de seguridad: un sistema analógico a partir de los Limit Switch que impide al manipulador superar ese límite y lo manda a la posición de origen y un sistema digital que, a partir de su posición actual, indica si la distancia que debe recorrer el manipulador supera los límites especificados y, si fuese así, no manda la orden al motor para que se mueva.

La placa Arduino almacena la última orden que se le ha mandado, por lo tanto puede dar problemas al conectar y desconectar el aparato debido a que no se puede ver por pantalla qué orden se le ha mandado al motor. Por este motivo, en el Anexo E, se muestra el protocolo que debe seguirse a la hora de conectar el controlador.

Anexo A

Código Arduino sin Limit Switch

un límite de valor de 32767 really basic code for the 'MakeBlock 2H Microstep Driver'

```
//seledccionamos la configuracion  SW6 = ON  SW7 = ON
//pasos = 200 da una vuelta -> una vuelta = 1 mm
//paso mínimo 200/200=2 -> 2=10um
//recorrido 128 mm,
//step controller
int dirPin = 2;
int pulPin = 3;
int enblPin = 4;
int pasos = 0;
float distancia = -1; //mm  <<-- introducir dostancia (>0 baja, <0 sube)
float aux = 0;
int a = 0;
//int sentido = 0; //izquierda  <<--
//int sentido = 1; //derecha  <<--
char s1 = 'L';
char s2 = 'R';



void setup(){


pasos = abs(distancia*200);
Serial.begin(9600);
pinMode(dirPin, OUTPUT);
```

```
pinMode(pulPin, OUTPUT);
pinMode(enblPin, OUTPUT);
digitalWrite(enblPin, HIGH);

}

void loop(){
        if (distancia < aux)
       {
        s1 = 'R';
        s2 = 'L';
        }
//if (sentido == 1)
//{
  //s1 = 'R';
  //s2 = 'L';
//}
//if (pasos > 18000)
//{
  //pasos = 18000;
//}
slide('L', pasos, 4000); //go Left for 500 steps at speed 500, (the lower the number is the faster the motor will go)
delay(2000);

slide('R', pasos, 4000); //go Right for 500 steps at speed 500
```

```
delay(2000);


}

void slide(int dir, int steps, int sspeed){

while (a == 0){
  Serial.println("hola");
if (sspeed < 50) sspeed = 10; //keeps the speed above 10

if (dir == s1) digitalWrite(dirPin, HIGH);
else if (dir == s2) digitalWrite(dirPin, LOW);

while (steps > 0){
digitalWrite(pulPin, HIGH);
delayMicroseconds(10);
digitalWrite(pulPin, LOW);
delayMicroseconds(10);
delayMicroseconds(sspeed);
steps--;
a = 1;
}
digitalWrite(enblPin, LOW);
}
}
```

Anexo B

Código Arduino con Limit Switch

```
//tiene un límite de valor de 32767 really basic code for the 'MakeBlock 2H Microstep Driver'
//seledccionamos la configuracion  SW6 = ON  SW7 = ON
//pasos = 400 da una vuelta -> una vuelta = 1 mm
//paso mínimo 400/200=2 -> 2=5um
//recorrido 64 mm, por encima de las especificaciones tiene un límite digital a 45 mm
//step controller



float distancia = -100; //mm  <<--
```

```
int dirPin = 2;
int pulPin = 3;
int enblPin = 4;
int com = 8;
int backno = 9;
int backnc = 10;
int frontno = 11;
int frontnc = 12;
int pasos = 0;
int val = 0;
float aux = 0;
int a = 0;
//int sentido = 0; //izquierda  <<--
```

```
    //int sentido = 1; //derecha  <<--
    char s1 = 'L';
    char s2 = 'R';



    void setup(){


    pasos = abs(distancia*200);
    Serial.begin(9600);
    pinMode(dirPin, OUTPUT);
    pinMode(pulPin, OUTPUT);
    pinMode(enblPin, OUTPUT);
    pinMode(com, OUTPUT);
    pinMode(backno, INPUT);
    pinMode(backnc, INPUT);
    pinMode(frontno, INPUT);
    pinMode(frontnc, INPUT);
    digitalWrite(enblPin, HIGH);
    digitalWrite(com, HIGH);
}


    void loop(){
            if (distancia < aux)
           {
            s1 = 'R';
            s2 = 'L';
```

```
        }

    slide('L', pasos, 4000); //go Left for 500 steps at speed 500, (the lower the number is the faster the motor will go)
    delay(2000);

    slide('R', pasos, 4000); //go Right for 500 steps at speed 500
    delay(2000);

    }

    void slide(int dir, int steps, int sspeed){

    while (a == 0){
      Serial.println("hola");
    if (sspeed < 50) sspeed = 10; //keeps the speed above 10

    if (dir == s1) digitalWrite(dirPin, HIGH);
    else if (dir == s2) digitalWrite(dirPin, LOW);

    while (steps > 0){
    digitalWrite(pulPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pulPin, LOW);
    delayMicroseconds(10);
    delayMicroseconds(sspeed);
```

```
  steps--;
  a = 1;
  int buttonState = digitalRead(backnc);
// print out the state of the button:
  Serial.println(buttonState);
  if (distancia < aux)
  {
    //if (digitalRead(backnc) == HIGH)
    //{digitalWrite(enblPin, HIGH);}
    if (digitalRead(backnc) == LOW)
    {digitalWrite(enblPin, LOW);}
   }
  if (distancia > aux)
  {
    if (digitalRead(frontnc) == LOW)
    {digitalWrite(enblPin, LOW);}
   }
  }
  }
  digitalWrite(enblPin, LOW);
  }
```

Anexo C

Tabla de conexión

ARDUINO	CONTROLADOR
DIGITAL 2	DIR
DIGITAL 3	PUL
POWER 5V	VCC
CONTROLADOR	MANIPULADOR R232
A+	CABLE ROJO
A-	CABLE NEGRO
B+	CABLE VERDE
B-	CABLE GRIS
ARDUINO	MANIPULADOR R232
DIGITAL 8	CABLE AZUL CABLE AZUL
DIGITAL 9	CABLE AZUL Y NEGRO CABLE NARANJA
DIGITAL 10	CABLE GRIS Y NEGRO CABLE VERDE
DIGITAL 11	CABLE AZUL Y ROJO CABLE ROJO
DIGITAL 12	CABLE GRIS Y ROJO CABLE GRIS
FUENTE DE ALIMENTACION	CONTROLADOR
V+	VDC+ CABLE ROJO
V-	GND- CABLE NEGRO

Anexo D

Código arduino de posicionamiento

```
//  ^ Press there to run
// To watch te position press "Ctrl+Mayus+M" but launch the last algorithm compiled
   //seledccionamos la configuracion  SW5 = ON SW6 = ON  SW7 = ON
    //pasos = 200 da una vuelta -> una vuelta = 1 mm
    //paso mínimo 200/200=1 -> 1=5um
    //recorrido 128 mm, por encima de las especificaciones tiene un límite digital a 45 mm

#include <EEPROM.h>
// El 100 es la posición más alta y el 0 la más baja//
// Para llevarlo a la posición de origen de 52 mm
// se debe introducir un valor en "distancia" de 0 //

        float distancia = 80; //mm  <<--

// introduce un valor negativo en "distancia" para bajar (<0) //
// introduce un valor positivo en "distancia" para subir (>0) //
    float origin = 52;
```

```
float limit = 100;
float i;
float d;
int posintegrer = 1;
int posdecimal = 2;
int dirPin = 2;
int pulPin = 3;
int enblPin = 4;
int com = 8;
int backno = 9;
int backnc = 10;
int frontno = 11;
```

```
    int frontnc = 12;
    int pasos = 0;
    int pasos2 = 0;
    int pasos3 = 0;
    int val = 0;
    float aux = 0;
    int a = 0;
//int sentido = 0; //izquierda  <<--
    //int sentido = 1; //derecha  <<--
char s1 = 'L';
    char s2 = 'R';
char s12 = 'R';
    char s22 = 'L';
char s13 = 'L';
    char s23 = 'R';

void setup(){
    Serial.begin(9600);
    Serial.println("Print distancia");
    Serial.println(distancia);
    distancia = -distancia;
    aux = origin - 10.36;
    pasos = abs(distancia*200);
    pasos2 = abs(aux*200);
    pasos3 = abs(47*200);
    aux = 0;
```

```
    pinMode(dirPin, OUTPUT);
    pinMode(pulPin, OUTPUT);
    pinMode(enblPin, OUTPUT);
    pinMode(com, OUTPUT);
    pinMode(backno, INPUT);
    pinMode(backnc, INPUT);
    pinMode(frontno, INPUT);
    pinMode(frontnc, INPUT);
    digitalWrite(enblPin, HIGH);
    digitalWrite(com, HIGH);
    Serial.println("Print origin");
    Serial.println(origin);
    Serial.println("Print limit");
    Serial.println(limit);
    i = EEPROM.read(posintegrer);
    d = EEPROM.read(posdecimal);
    i = i + (d/100); //contains the value of the previous position//
    Serial.println("Print previous position");
    Serial.println(i);
}

    void loop(){

////////// origin command ////////////////////////////

      if (distancia == aux){
```

```
      Serial.println("Wait 2 minutes to reach the origin");
      pasos = 20000;
      distancia = limit;

      slide('L', pasos, 4000); //go Left for 500 steps at speed 500, (the lower the number is the faster the motor will go)
      delay(2000);

      slide('R', pasos, 4000); //go Right for 500 steps at speed 500
      delay(2000);



      }

////////////// selecting direction //////////

             if (distancia < aux)
            {
             s1 = 'R';
             s2 = 'L';
             }

 ///////// checking the distance //////////////

    i = EEPROM.read(posintegrer);
    d = EEPROM.read(posdecimal);
```

```
    i = i + (d/100); //contains the value of the previous position//
    if (a == 0){
    if (distancia > aux)
    {
      d = distancia + i;
      if(d > limit)
      {
        Serial.println("The distance selected reaches the limit of the motorknob");
        a = 2;
        }
    }
    if (distancia < aux)
    {
      d = i + distancia;
      if(d < aux)
      {
        Serial.println("The distance selected reaches the limit of the motorknob");
        a = 2;
        }
      }
```

////////////////////////// Moving to position between the limits ////////////////////////////////////

```
    if (a == 0){
      slide('L', pasos, 4000); //go Left for 500 steps at speed 500, (the lower the number is
```

the faster the motor will go)

```
      delay(2000);


      slide('R', pasos, 4000); //go Right for 500 steps at speed 500
      delay(2000);


      actualposition(distancia, posintegrer, posdecimal);
    }
    }

////////////////////////// Reaching lower limit and returning to origin position
//////////////////////////////////////////////////////////////

    if (a == 1){
    Serial.println("Limit Switch reached -> returning to origin position");


    slide2('L', pasos2, 4000); //go Left for 500 steps at speed 500, (the lower the number is the faster the motor will go)
    delay(2000);


    slide2('R', pasos2, 4000); //go Right for 500 steps at speed 500
    delay(2000);


    originposition(origin, posintegrer, posdecimal);
    }


////////////////////// Reaching to upper limit and returning to origin position
//////////////////////////////////////////////////////////////

    if ( a == 3){
```

```
    Serial.println("Limit Switch reached -> returning to origin position");

    slide3('L', pasos3, 4000); //go Left for 500 steps at speed 500, (the lower the number is the faster the motor will go)
    delay(2000);

    slide3('R', pasos3, 4000); //go Right for 500 steps at speed 500
    delay(2000);

    originposition(origin, posintegrer, posdecimal);
    }
    }
////////////////// Moving to selected position function ////////////////////////////////////////////
    void slide (int dir, int steps, int sspeed){

    while (a == 0){
    if (sspeed < 50) sspeed = 10; //keeps the speed above 10

    if (dir == s1) digitalWrite(dirPin, HIGH);
    else if (dir == s2) digitalWrite(dirPin, LOW);

    while (steps > 0){
    digitalWrite(pulPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pulPin, LOW);
    delayMicroseconds(10);
```

```
delayMicroseconds(sspeed);
steps--;
if (distancia < aux)
{
  //if (digitalRead(backnc) == HIGH)
  //{digitalWrite(enblPin, HIGH);}
  if (digitalRead(backnc) == LOW)
    {digitalWrite(enblPin, LOW);
     steps = 1;
     a = 3;}
 }
 else {a==2;}
if (distancia > aux)
{
  if (digitalRead(frontnc) == LOW)
    {digitalWrite(enblPin, LOW);
     steps = 1;
     a = 1;}
     else {a==2;}
 }
}
if(a == 0) a = 2;
}
  digitalWrite(enblPin, LOW);
}
```

```
//////////////////// Moving to origin when lower limit is reached function
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////


   void slide2(int dir, int steps, int sspeed){


   digitalWrite(enblPin, HIGH);
   while (a == 1){
   if (sspeed < 50) sspeed = 10; //keeps the speed above 10


   if (dir == s12) digitalWrite(dirPin, HIGH);
   else if (dir == s22) digitalWrite(dirPin, LOW);


   while (steps > 0){
   digitalWrite(pulPin, HIGH);
   delayMicroseconds(10);
   digitalWrite(pulPin, LOW);
   delayMicroseconds(10);
delayMicroseconds(sspeed);
   steps--;
   a = 2;



     //if (digitalRead(backnc) == HIGH)
     //{digitalWrite(enblPin, HIGH);}
```

```
    if (digitalRead(backnc) == LOW)
      {digitalWrite(enblPin, LOW);
       a = 2;}

  }
  }
digitalWrite(enblPin, LOW);
  }



//////////////////// Moving to origin when upper limit is reached function
////////////////////////////////////////////////////////////

  void slide3(int dir, int steps, int sspeed){

  while (a == 3){
    digitalWrite(enblPin, HIGH);
  if (sspeed < 50) sspeed = 10; //keeps the speed above 10

  if (dir == s13) digitalWrite(dirPin, HIGH);
  else if (dir == s23) digitalWrite(dirPin, LOW);

  while (steps > 0){
  digitalWrite(pulPin, HIGH);
  delayMicroseconds(10);
```

```
    digitalWrite(pulPin, LOW);
    delayMicroseconds(10);
delayMicroseconds(sspeed);
    steps--;
    a = 2;



      if (digitalRead(frontnc) == LOW)
        {digitalWrite(enblPin, LOW);
         a = 2;}


    }
    }
digitalWrite(enblPin, LOW);
    }

////////////////////// Printing origin position function ////////////////////////////////////////////////////////

void originposition(float origin, int posintegrer, int posdecimal)
{
    Serial.println("Print origin");
    Serial.println(origin);
    EEPROM.write(posintegrer, origin);
    i = EEPROM.read(posintegrer);
    d = origin - i;
    d = d*100;
```

```
    EEPROM.write(posdecimal, d);
    d = EEPROM.read(posdecimal);
    i = i + (d/100);
    Serial.println("Print saved origin reached");
    Serial.println(i);
  }

//////////////////// Printing actual position function ////////////////////////////////////////////////////////

void actualposition(float distancia, int posintegrer, int posdecimal)
{
    i = EEPROM.read(posintegrer);
    d = EEPROM.read(posdecimal);
    i = i + (d/100); //contains the value of the previous position//
    Serial.println("Print previous position");
    Serial.println(i);
    d = i - distancia; //contains the value of the new position//
    Serial.println("Print new position reached");
    Serial.println(d);
    EEPROM.write(posintegrer, d);
    i = EEPROM.read(posintegrer);
    d = d - i;
    d = d*100;
    EEPROM.write(posdecimal, d);
  }
```

Anexo E

Protocolo de instalación del controlador

CONTROLLER INSTALLATION

1. connect the controller to the Arduino board, the R232(sub D9) adapter and the power supply as indicated in the Anex C of the document.

2. Install Arduino on the OS.

3. Turn on the power supply.

4. Connect the Arduino board to the computer. DO NOT connect the R232 adapter to the motorknob.

5. Launch the arduino program "motorknobcontroller612CLD50C16".

6. Connect the Arduino program to the Arduino board (Tools -> Board(Placa) -> Arduino UNO; Tools -> Port(Puerto) -> COM).

7. Launch Serie Monitor (Tools -> Monitor Serie or "Ctrl+Mayus+M).

8.Wait 1 minute, when you open the Monitor serie, the last program compiled in the Arduino board is launched. We do not want this program to reach the motorknob.

9. Connect the R232 adapter to the motorknob.

10. Now you can use the motorknob, congratulations.