Practica de laboratorio 2: Microprocesadores. Control de un motor paso a paso.

Arias Ríos Jessica Lorena; Gallego Sánchez Samantha;

Grupo: 4.

Valle del Cauca, Universidad Autónoma de occidente Cali, Colombia

jessica.arias@uao.edu.co

samantha.gallego@uao.edu.co

DESCRIPCIÓN DE LA PRÁCTICA.

La práctica de laboratorio consiste en el desarrollo de una aplicación de control de un motor paso a paso, para la cual se le solicita al usuario el numero de pasos a correr en el motor, valor entre 1 y 9, la dirección a la que se debe realizar el giro (derecha o izquierda) y la velocidad a la que se deben realizar los pasos contando con 4 velocidades, siendo 1 la velocidad mínima y 4 la velocidad máxima. La solicitud de los datos se debe realizar a través de la terminal virtual de Proteus, por los pines del USART del microcontrolador, y estos datos se deben proyectar en una pantalla LCD antes de proceder a controlar el motor.

Además, el programa debe servir para enviar más de un conjunto de instrucciones, de modo que después de que el motor paso a paso realice las instrucciones dadas por el usuario, este preparado para recibir nuevos datos, conservando el paso anterior como nuevo paso 0 de referencia.

Para la implementación de las soluciones se utiliza un microcontrolador ATMega2560, se realiza el código de programación de este en el programa Microchip Studio, a través del lenguaje c, después se procede a realizar la simulación en Proteus la solución. Se utiliza una frecuencia de 16MHz para el microcontrolador. Para la comunicación serial se utiliza una velocidad de 9600 baudios.

ANÁLISIS Y CÁLCULOS DE LA SOLUCIÓN PROPUESTA.

En primer lugar, se realizó una librería, para el control de la pantalla LCD, de forma que esta se pueda utilizar en proyectos futuros, esta librería cuenta con las funciones de inicializar, teniendo en cuenta que se utiliza una pantalla 16x2, conectando a esta los 8 bits de datos, al igual que con las funciones utilizadas para escribir caracteres en la pantalla, y por ultima la función que funciona para pasar a la primera posición de la segunda línea de la pantalla (enter).

Luego en el programa principal, se realiza primero el llamado a las librerías, tomando también la librería creada, y definiendo la frecuencia del microcontrolador de 16MHz, además de realizar el llamado de las librerías de las interrupciones y de los retrasos (delay).

Después se realizan las declaraciones iniciales, tanto de variables globales como de subprocesos a utilizar.

Ya dentro de la función principal, se inicializan las variables, se configuran los puertos A, B, C y D como puertos de salida, y se inicializan como 0, se llama al proceso inicializar LCD, para configurar la pantalla, y se configuran los registros necesarios para la comunicación serial.

La comunicación serial se realiza a una velocidad de 9600 baudios, para lo que se realiza el calculo de que valor debe tomar el registro UBRR0.

$$UBRR_n = \frac{fosc}{16BAU} - 1$$

$$UBRR_n = \frac{16MHz}{16 * 9600} - 1$$

$$UBRR_n = 103{,}166$$

Se utiliza entonces el valor de 103, que en hexadecimal viene dado por 67, por lo que se configura el registro como 067, pues los primeros cuatro bits de la parte alta del registro están no son modificables.

Luego se configura la comunicación serial para el tamaño de carácter, que sea de 8 bits, y se habilita la transmisión y recepción de datos. Por ultimo se habilita la interrupción producida por recepción de datos en la comunicación social.

Después se configura el modo “SLEEP”, para que funcione en modo “Idle”, de forma que se ahorre la energía consumida, pero no se vean afectados el rendimiento del microcontrolador, ni se borre la memoria RAM.

Se pasa a enviar a través de comunicación serial el mensaje “INGRESE NUMERO DE PASOS (1-9)” y por ultimo se habilita el registro global de interrupciones, de forma que cuando el usuario ingrese el numero de pasos, se produzca una interrupción. Como tarea de fondo se tiene SLEEP para el ahorro de energía mientras no se esté utilizando el microprocesador.

De entrada, en la interrupciones, como todas se habilitan de la misma forma, pero cada una debe realizar diferentes tareas, se utiliza un contador, para conocer si es la primera interacción del usuario o que numero de interacción es. La primera interacción se encarga de recibir el número de pasos, por lo que se valida el carácter ingresado por el usuario, y si se encuentra entre 1 y 9, se recibe el dato, se incrementa el contador de interrupciones, se envía el dato a la pantalla LCD y se pide el siguiente dato, en caso de que no se encuentre entre los valores correctos, se envía un mensaje de error y no se incrementa el contador de interrupción.

Para la segunda interrupción, se recibe la velocidad, y se realiza el mismo proceso anterior de validación, luego se pide la dirección, y por ultimo se pide un mensaje de 1 para verificar y continuar, lo que significaría usar esos datos para el control del motor, o 0 para pedir nuevamente los datos. Ya para la siguiente interrupción, se reinicia el contador para la llegada de nuevos datos.

Para el control del motor, una vez se tienen los datos de control, se definen cuales son el giro a la derecha y a la izquierda, para lo que se define el valor de bits de cada paso, en el orden correspondiente, para mayor precisión, se utiliza un paso medio, de forma se realiza no solo un paso entre bobina y bobina, si no que dos, uno en la bobina y otro en medio de dos bobinas adyacentes.

Una vez se conoce que dirección tendrá el giro, lo que se hace es buscar cual fue la ultima posición en la que quedo el motor paso a paso, esto sobre todo para los datos siguientes al inicial, para que se continúe a partir de la posición anterior, una vez se obtiene esta posición, se recibe cual es el numero de pasos, y se cuentan a partir de ahí, según la velocidad dada, la cual lo que modifica es el tiempo entre paso y paso.

DIAGRAMA DE FLUJO Y CIRCUITO.

Se desarrollaron los respectivos diagramas de flujo de la solución. En primera instancia, se desarrolla la librería para el control de la pantalla LCD para la visualización, inicializando primero el LCD, configurando los puertos utilizados por el LCD como puertos de salida y asignándoles los valores de 0, luego se limpia el LCD, luego se realiza un pulso para que el LCD reciba la instrucción, luego se activa el display, el cursor y se activa el parpadeo de este, y se realiza el mismo pulso. Por último, se configura a que el display sea a 2 líneas, que se le conecten 8 bits y que los caracteres visualizados sean de matriz 5x7.

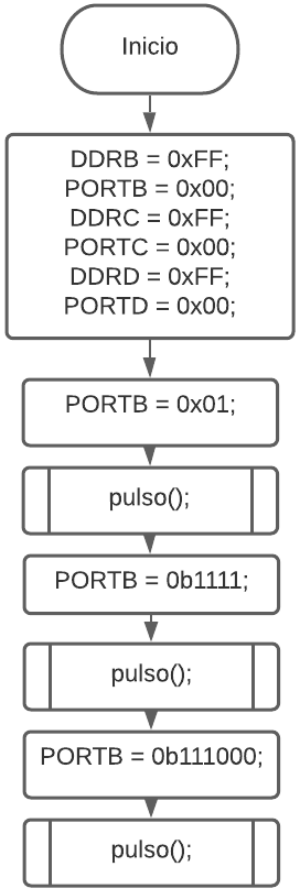

Figura 1. Diagrama de flujo Inicializar LCD.

El pulso se genera en el bit 0 del puerto C, que es el bit conectado al habilitador del LCD, y se genera, encendiendo y apagando este, dejando un tiempo entre encendido y apagado.

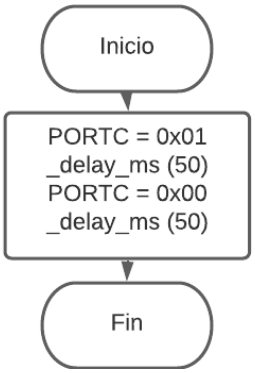

Figura 2. Diagrama de flujo Pulso.

Para escribir en el LCD, lo primero que se hace es encender el puerto D, que es el conectado a la entrada RS del LCD, entrada que, al encontrarse en 0, coloca al LCD en modo programa, para configurarlo, y al encontrarse en 1, el LCD se pone en modo visualización, para recibir el carácter que se desee escribir.

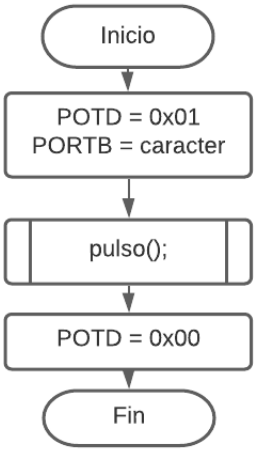

Figura 3. Diagrama de flujo Escribir LCD.

Para realizar el salto de línea, se tiene el LCD en modo programa, y se le manda el dato 0xC0 para que pase a la posición de 0x40, que es la primera posición de la segunda fila.

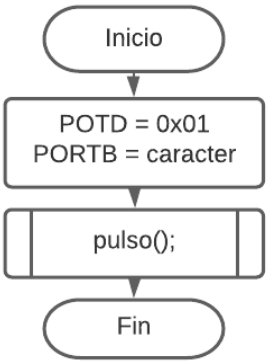

Figura 4. Diagrama de flujo Enter LCD.

Luego se pasa a desarrollar el programa principal, teniendo en cuenta las interrupciones provocadas por la recepción de datos de la comunicación serial, ocurrida cada que el usuario presiona un botón. Además de que se utilizan funciones de la librería creada.

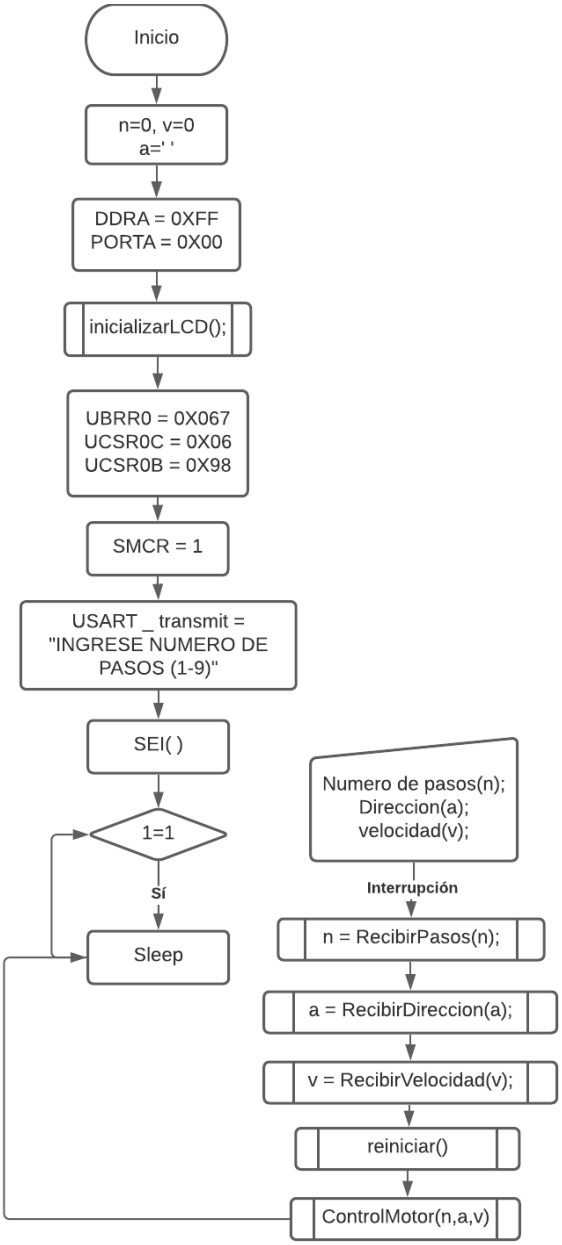

Figura 5. Diagrama de flujo Main.

A continuación, se muestra el flujograma de la subrutina de RecibirPasos, donde se realiza la validación, en caso de error se entra al la subrutina de verificar, y en caso de estar correcto el dato, se pide el siguiente dato, se envia el dato recibido al a pantalla LCD de visualización y se le resta 48 al numero recibido, esto teniendo en cuenta que en codigo ASCII (codigo recibido), los numeros estan a partir de la posición 48, por lo que esa es la convesión de variable tipo caracter a variable tipo entero y se incrementa el contador de interrupciones.

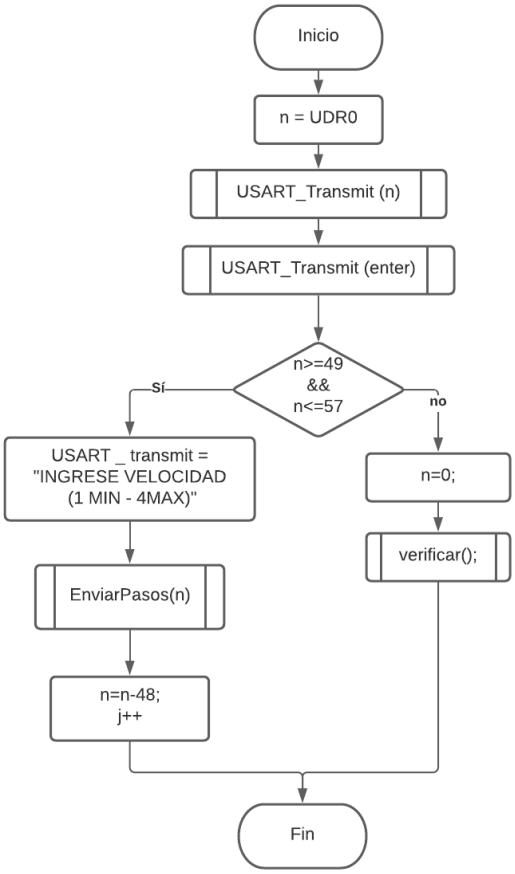

Figura 6. Diagrama de flujo Recibir pasos.

El proceso verificar, se encarga de enviar el mensaje de error al usuario.

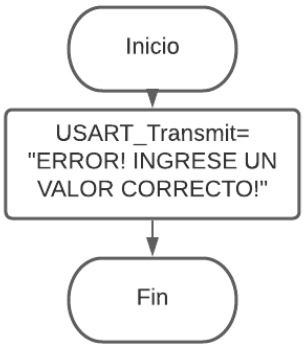

Figura 7. Diagrama de flujo Verificar.

La subrutina de enviar pasos se encarga de escribir en el LCD “PASOS:” seguido del numero de pasos ingresados por el usuario.

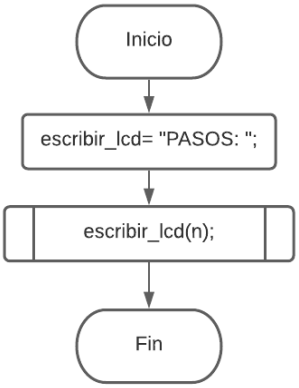

Figura 8. Diagrama de flujo Enviar pasos.

Luego se pasa a la subrutina de recibir velocidad, que se encarga de hacer lo mismo que el subproceso anterior.

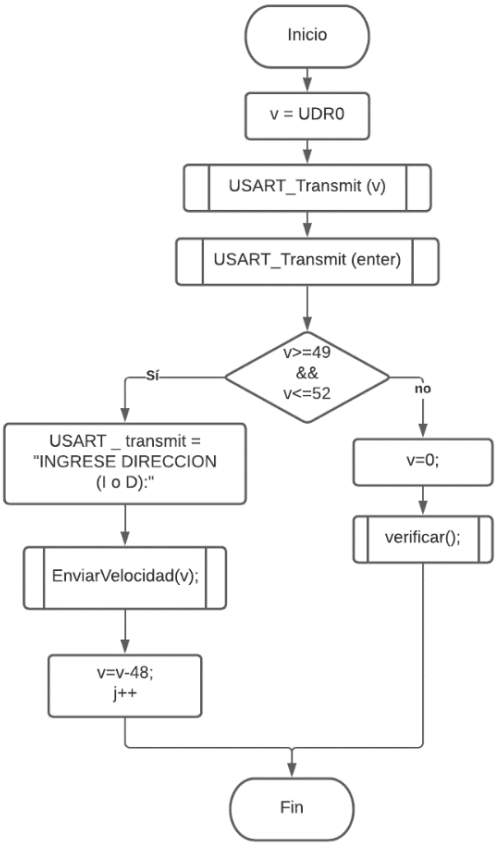

Figura 9. Diagrama de flujo Recibir velocidad.

De igual forma el proceso de enviar velocidad, se escribe en el LCD los datos recibidos.

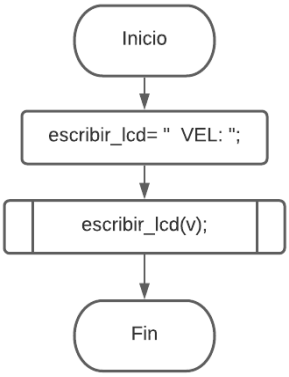

Figura 10. Diagrama de flujo Enviar velocidad.

Posteriormente se encuentra el proceso de recibir dirección, la cual puede ser izquierda o derecha, por lo que se valida que el carácter recibido sea una i (izquierda) o d (derecha), ya sean mayúscula o minúscula, luego se procede a realizar los mismos pasos de los procesos anteriores.

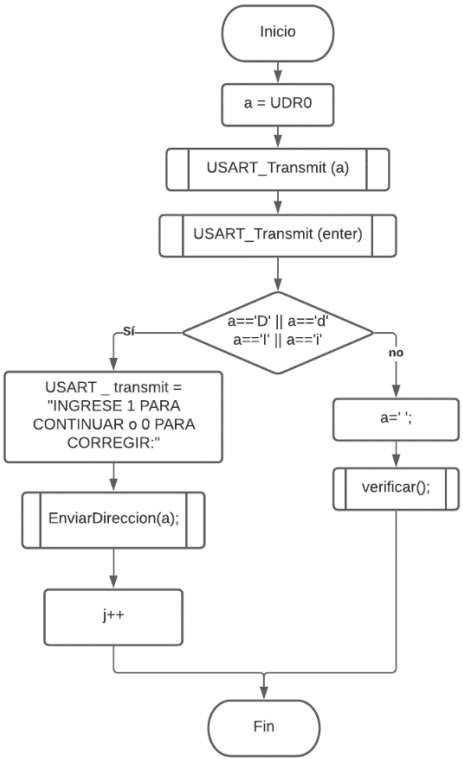

Figura 11. Diagrama de flujo Recibir Dirección.

Luego el proceso de enviar dirección se encarga de escribir en el LCD la dirección escogida por el usuario.

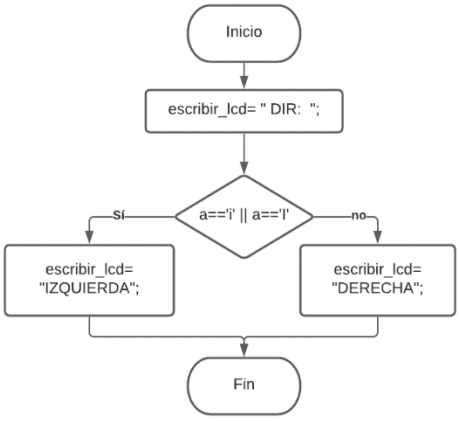

Figura 12. Diagrama de flujo Enviar Dirección.

A continuación, se muestra el subproceso de Continuar o Corregir, donde primero se valida que el carácter ingresado por el usuario sea 1 (continuar) o 0 (corregir), en caso de que no, se envía un mensaje de error, una vez validado, si es 0, se pasa a reiniciar, iniciando nuevamente a pedir los datos al usuario y borrando los datos antes. Si es 1, se guarda en la variable c, el valor de 1 y se realiza el Control del Motor.

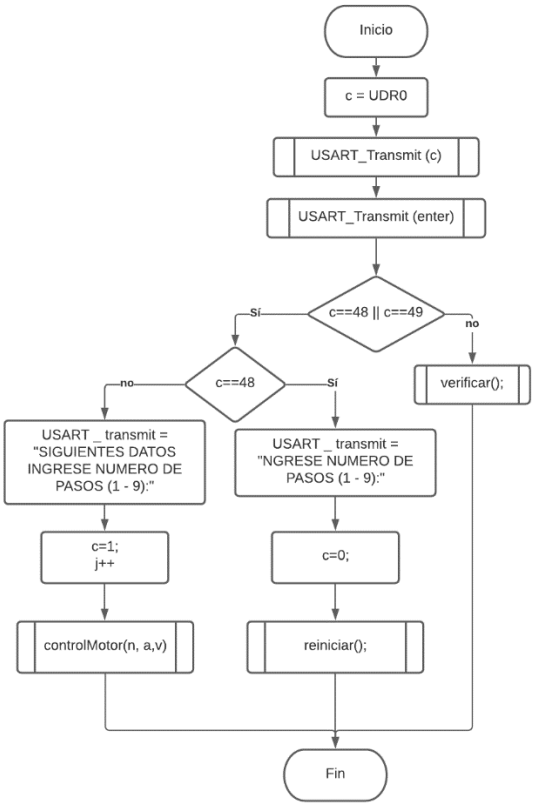

Figura 13. Diagrama de flujo Continuar o corregir.

El proceso de reiniciar se encarga de mandarle a los datos el valor inicial y de limpiar el LCD, volviéndolo a configurar, además de reiniciar el puerto A y de reiniciar el contador de interrupciones.

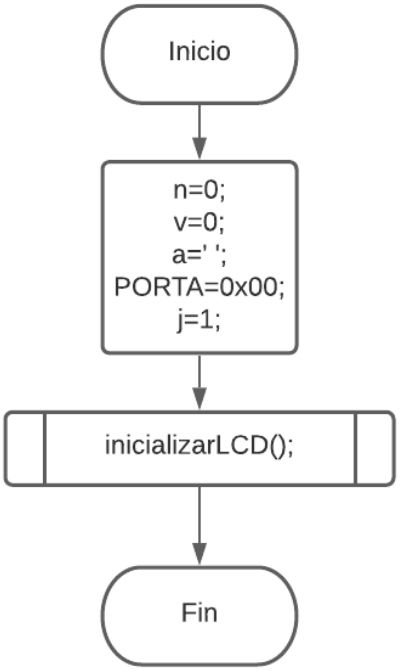

Figura 14. Diagrama de flujo Reiniciar.

Por último, en el flujograma del subproceso de control de motor, lo primero que se hace es definir el orden de las bobinas encendidas para tener las dos opciones de giro, horario (derecha) y antihorario (izquierda), teniendo ambas los mismos valores, pero en orden contrarios.

Después entra a comparar si se desea girar a la izquierda o derecha, entra a un ciclo para buscar cual fue la última posición del motor, esto dependiendo en qué sentido se vaya a girar, así mismo se busca en ese arreglo, sabiendo que si recién se empezó la aplicación (primeros datos), la posición es la última en cada uno de los arreglos. A esta posición se le suma 1, para que sea la primer posición del nuevo movimiento, como son arreglos definidos de tamaño 8, se compara, si esta posición es mayor o igual a 8, se empieza en 0, pues las posiciones son un ciclo consecutivo.

Luego se entra en un ciclo, donde se cuenta el número de pasos, según los datos del usuario, para lo que se entra a ver a que velocidad se desea el giro, para conocer que tiempo habrá entre paso y paso, luego se pasa a la siguiente posición, haciendo antes la comprobación de que no sea mayor o igual a 8, hasta que se complete el numero de pasos dados por el usuario.

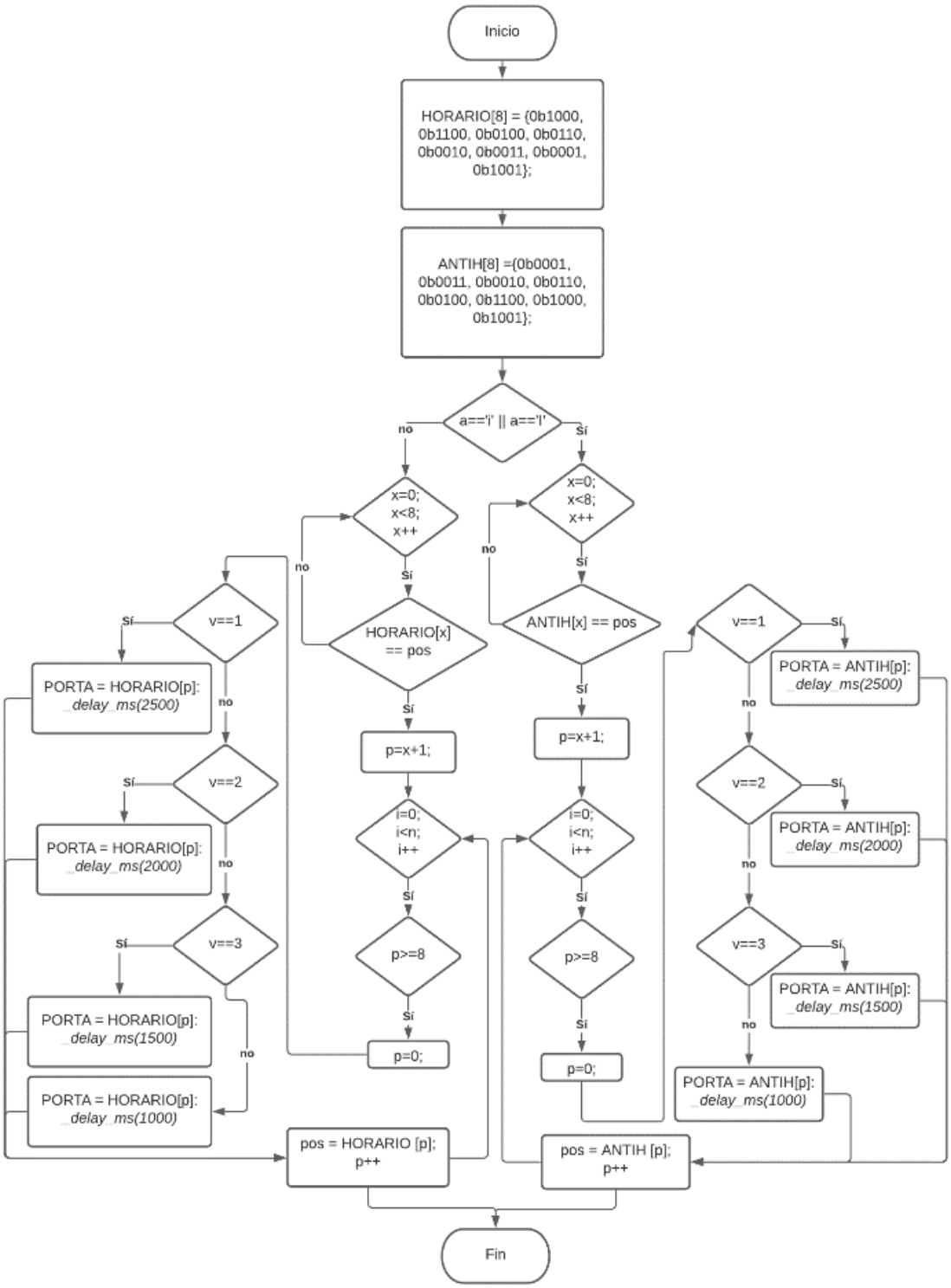

Figura 15. Diagrama de flujo Control motor.

Por último, a continuación, se muestra la figura del circuito en Proteus. Para la conexión del motor paso a paso, fue necesario utilizar un puente H, por la corriente necesaria para hacer girar el motor, para este puente H, se utilizaron 4 transistores tipo Darlington.

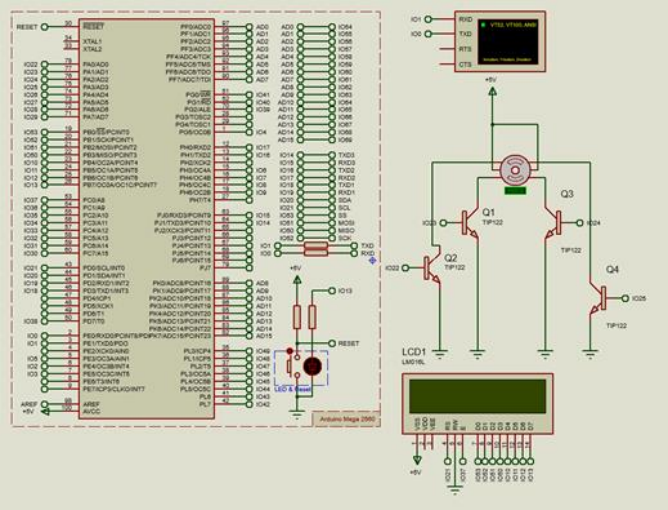

Figura 16. Circuito.

RESULTADOS DE SIMULACIÓN Y ANÁLISIS DE ESTOS.

Se probó el programa en Proteus, enviando un total de 5 pasos, siendo cada paso de 20°, hacia la derecha a una velocidad máxima (4), se ve en la simulación que el giro lo realiza de manera adecuada, además de que se imprime en la pantalla los datos del usuario (figura 17).

Después se envía el carácter g, el cual no se encuentra entre los datos válido, y aparece el mensaje de error en la terminal virtual, por lo que el siguiente conjunto de instrucciones es 2 pasos a la izquierda a una velocidad de 2, por lo que se devuelve 2 pasos, a partir de los 5 dados anteriormente, además en la animación se ve que estos los hace de manera más pausada, al ser su velocidad menor (figura 18).

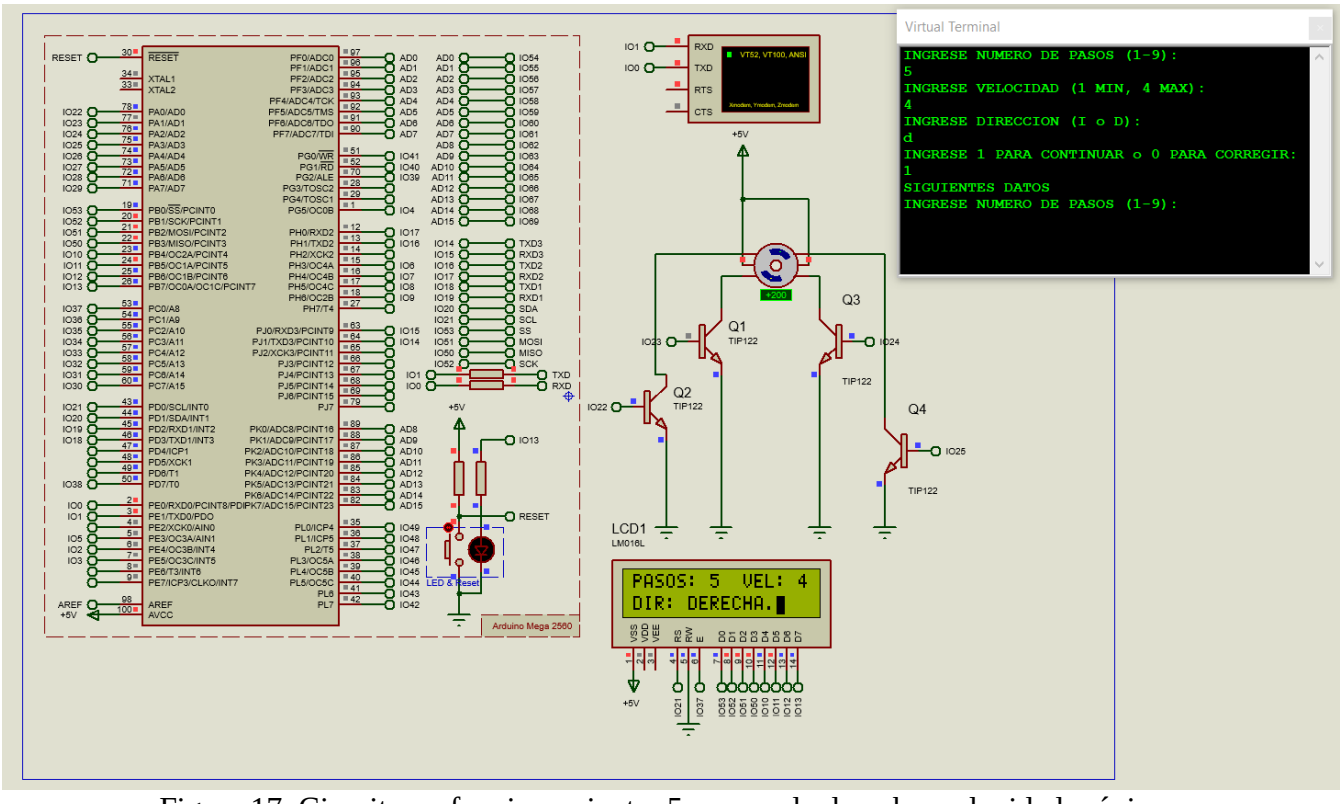

Figura 17. Circuito en funcionamiento, 5 pasos a la derecha, velocidad máxima.

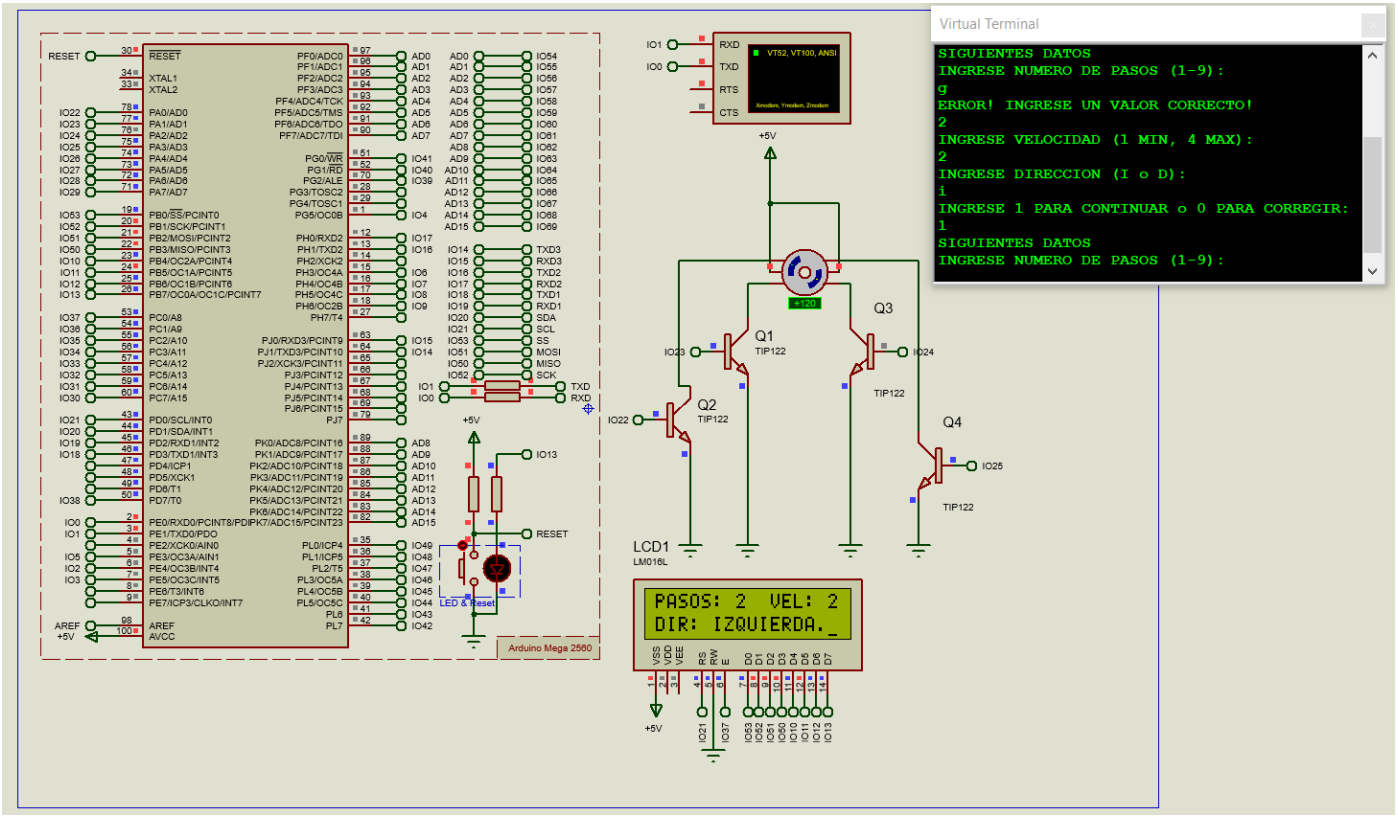

Figura 18. Circuito en funcionamiento, segunda acción, 2 pasos a la izquierda, velocidad media (2).

Por lo que se ve que el circuito funciona de la manera esperada y no se observan errores en su simulación.

CÓDIGO DEL PROGRAMA.

A continuación, se presenta el código de la librería utilizada “LCD” y creada:

```
char caracter;
void escribir_lcd(char caracter);
void inicializarLCD();
void enter_lcd();

void inicializarLCD()
{

	asm("LDI R16,0xFF") ;	//DDRB = 0xFF;(0x04)		//Puerto B salida
	asm("OUT 0x04,R16");

	asm("LDI R16,0x00");	//PORTB = 0x00;  (0x05)	//puerto B en 0
	asm("OUT 0x05, R16");

	asm("LDI R16,0xFF");//DDRC = 0xFF;
	(0x07)		//Puerto C salida
	asm("OUT 0x07,R16");

	asm("LDI R16,0x00");	//PORTC = 0x00;(0x08)		//puerto C en 0
	asm("OUT 0x08, R16");

	asm("LDI R16,0xFF");//DDRD = 0xFF;
	(0x0A)		//Puerto D es salida
	asm("OUT 0x0A,R16");

	asm("LDI R16,0x00");	//PORTD = 0x00;
	(0x0B)		//Se inicia en 0
	asm("OUT 0x0B, R16");


	asm("LDI R16,0x01");//PORTB = 0x01;
		//Clear display
	asm("OUT 0x05, R16");
	pulso();

	asm("LDI R16,0b1111");		//PORTB = 0b1111;		//Activacion display, cursor en on, y blinking activado
	asm("OUT 0x05, R16");
	pulso();

	asm("LDI R16,0b111000");		//PORTB = 0b111000;		//Display configurado a 2 linea, 8 bits , matriz 5x7
	asm("OUT 0x05, R16");
	pulso();

}

void pulso()
{
	asm("LDI R16,0x01");	//PORTC = 0X01;		//Se activa el enable (bit0 PORTC)
	asm("OUT 0x08, R16");
	_delay_ms(50);
	asm("LDI R16,0x00");	//PORTC = 0X00;		//Se baja el enable
	asm("OUT 0x08, R16");
	_delay_ms(50);
}
```

```
void escribir_lcd(char caracter)
{
	asm("LDI R16,0x01");	//PORTD =
0X01;			//Se activa pin
conectado a RS PARA ACTIVAR MODO VISUALIZACION
	asm("OUT 0x0B, R16");
	PORTB = caracter;		//Se
configura puerto B con el valor del caracter
	pulso();
	_delay_ms(50);
	asm("LDI R16,0x00");	//PORTD =
0X00;
	asm("OUT 0x0B, R16");
}

void enter_lcd()
{
	asm("LDI R16,0xc0");	//PORTB=0xC0;
	//POSICION 1 FILA 2
	asm("OUT 0x05, R16");
	pulso();
}
```

Y también, se presenta el código del programa principal.

```
#include <avr/io.h>
#define F_CPU 16000000UL
#include <util/delay.h>
#include <avr/interrupt.h>
#include "LCD.c"


int n, v, j=1, p=0;
char a, c, pos=0b1001;
char enter=13;


int RecibirPasos(int n);
void EnviarPasos (int n);

int RecibirVelocidad (int v);
void EnviarVelocidad (int v);

char RecibirDireccion (char a);
void EnviarDireccion (char a);

void verificar();
char ContinuarCorregir();
void reiniciar();

void controlMotor (int n,char a,int v);

int main(void)
{
	n=0;
	a=' ';
	v=0;

	asm("LDI R16,0xFF");	//DDRA =
0xFF;(0x01)		//Puerto A es salida
	asm("OUT 0x01,R16");

	asm("LDI R16,0x00");//PORTA = 0x00;
	(0x02)		//Se inicia en 0
	asm("OUT 0x02, R16");


	inicializarLCD();

	asm("LDI R16,0x00");
//UBRR0=0x067;UBRR0H(0xC5)	// BAUDIOS=9600
	asm("STS 0xC5,R16");

	asm("LDI R16,0x67");//UBRR0L(0xC4)
	asm("STS 0xC4,R16");


	asm("LDI R16,0x06");
	//UCSR0C=0x06;(0xC2)//TAMAÑO CARACTER=8
BIT (UCSZN)
	asm("STS 0xC2,R16");

	asm("LDI R16,0x98");
	//UCSR0B=0x98;(0xC1)//RXCIE
(INTERRUPCION RX- BIT 7)
	asm("STS 0xC1,R16");	//RXEN, TXEN
(ENABLE RX Y TX BIT 4,3)


	asm("LDI R16,0x01");
	//SMCR=1;(0x33) //HABILITAR SLEEP
	asm("OUT 0x33,R16");

	USART_Transmit('I');
	USART_Transmit('N');
	USART_Transmit('G');
	USART_Transmit('R');
	USART_Transmit('E');
	USART_Transmit('S');
	USART_Transmit('E');
	USART_Transmit(' ');
	USART_Transmit('N');
	USART_Transmit('U');
	USART_Transmit('M');
	USART_Transmit('E');
	USART_Transmit('R');
	USART_Transmit('O');
	USART_Transmit(' ');
	USART_Transmit('D');
	USART_Transmit('E');
	USART_Transmit(' ');
	USART_Transmit('P');
	USART_Transmit('A');
	USART_Transmit('S');
	USART_Transmit('O');
	USART_Transmit('S');
	USART_Transmit(' ');
	USART_Transmit('(');
	USART_Transmit('1');
	USART_Transmit('-');
	USART_Transmit('9');
	USART_Transmit(')');
	USART_Transmit(':');
	USART_Transmit(enter);

	sei();

	while (1){
		asm("SLEEP"); //TAREA DE FONDO
		asm("NOP");
		asm("NOP");
	}
```

```
}

ISR(USART0_RX_vect){
	if (j==1)
	{
		n=RecibirPasos(n);
	}
	else if (j==2)
	{
		v=RecibirVelocidad(v);

	}else if (j==3)
	{
		a=RecibirDireccion(a);
	}else if (j==4)
	{
		c=ContinuarCorregir( c);
		if (c==1)
		{
			controlMotor(n, a, v);
		}

	}else if(j==5)
	{

		reiniciar();
	}
}

void controlMotor (int n,char a,int v){

	char const HORARIO[8] = {  0b1000,
0b1100, 0b0100,

	0b0110, 0b0010, 0b0011,

	0b0001, 0b1001};

	char const ANTIH[8] ={  0b0001, 0b0011,

	0b0010, 0b0110, 0b0100,

	0b1100, 0b1000, 0b1001};


	if (a=='i'|| a=='I')
	{

		for (int x = 0; x < 8; x++) {
			if (ANTIH[x] == pos){
			p=x+1;
			}

		}

		for (int i=0; i<n;i++)
		{
			if (p>=8)
			{
				p=0;
			}
			if (v==1)
			{
				PORTA=ANTIH[p];
				_delay_ms(2500);
			}
			else if (v==2){
				PORTA=ANTIH[p];
				_delay_ms(2000);
			}
			else if (v==3)
			{
				PORTA=ANTIH[p];
				_delay_ms(1500);
			}
			else {

				PORTA=ANTIH[p];
				_delay_ms(1000);

			}

			pos=ANTIH[p];

			p++;
		}
		}else {


			for (int x = 0; x < 8; x++)
{
			if (HORARIO[x] == pos){
					p=x+1;
				}

			}

		for (int i=0; i<n;i++)
		{

			if (p>=8)
			{
				p=0;
			}
			if (v==1)
			{
				PORTA=HORARIO[p];
				_delay_ms(2500);
			}
			else if (v==2){
				PORTA=HORARIO[p];
				_delay_ms(2000);
			}
			else if (v==3)
			{
				PORTA=HORARIO[p];
				_delay_ms(1500);
			}
			else {

				PORTA=HORARIO[p];
				_delay_ms(1000);

			}

			pos=HORARIO[p];

			p++;
		}

	}

}

void USART_Transmit( unsigned char data )
{
	/* Wait for empty transmit buffer */
```

```
	while ( !( UCSR0A & (1<<UDRE0)) )
	;
	/* Put data into buffer, sends the data
*/
	UDR0 = data;
}

int RecibirPasos(int n){

	n=UDR0;
	USART_Transmit(n);
	USART_Transmit(enter);

	if (n>=49 && n<=57)
	{

		USART_Transmit('I');
		USART_Transmit('N');
		USART_Transmit('G');
		USART_Transmit('R');
		USART_Transmit('E');
		USART_Transmit('S');
		USART_Transmit('E');
		USART_Transmit(' ');
		USART_Transmit('V');
		USART_Transmit('E');
		USART_Transmit('L');
		USART_Transmit('O');
		USART_Transmit('C');
		USART_Transmit('I');
		USART_Transmit('D');
		USART_Transmit('A');
		USART_Transmit('D');
		USART_Transmit(' ');
		USART_Transmit('(');
		USART_Transmit('1');
		USART_Transmit(' ');
		USART_Transmit('M');
		USART_Transmit('I');
		USART_Transmit('N');
		USART_Transmit(',');
		USART_Transmit(' ');
		USART_Transmit('4');
		USART_Transmit(' ');
		USART_Transmit('M');
		USART_Transmit('A');
		USART_Transmit('X');
		USART_Transmit(')');
		USART_Transmit(':');
		USART_Transmit(enter);

		j++;

		EnviarPasos(n);

		n=n-48;

		}else{

		n=0;
		verificar();
	}

	return n;
}

int RecibirVelocidad (int v){

	v=UDR0;
	USART_Transmit(v);
	USART_Transmit(enter);

	if (v>=49 && v<=52)
	{

		USART_Transmit('I');
		USART_Transmit('N');
		USART_Transmit('G');
		USART_Transmit('R');
		USART_Transmit('E');
		USART_Transmit('S');
		USART_Transmit('E');
		USART_Transmit(' ');
		USART_Transmit('D');
		USART_Transmit('I');
		USART_Transmit('R');
		USART_Transmit('E');
		USART_Transmit('C');
		USART_Transmit('C');
		USART_Transmit('I');
		USART_Transmit('O');
		USART_Transmit('N');
		USART_Transmit(' ');
		USART_Transmit('(');
		USART_Transmit('I');
		USART_Transmit(' ');
		USART_Transmit('o');
		USART_Transmit(' ');
		USART_Transmit('D');
		USART_Transmit(')');
		USART_Transmit(':');
		USART_Transmit(enter);

		j++;

		EnviarVelocidad(v);

		v=v-48;

		}else{

		v=0;
		verificar();

	}
	return v;
}

char RecibirDireccion (char a){

	a=UDR0;
	USART_Transmit(a);
	USART_Transmit(enter);

	if (a=='D' || a=='I' || a=='d' || a=='i')
	{
		USART_Transmit('I');
		USART_Transmit('N');
		USART_Transmit('G');
		USART_Transmit('R');
		USART_Transmit('E');
		USART_Transmit('S');
		USART_Transmit('E');
		USART_Transmit(' ');
		USART_Transmit('1');
		USART_Transmit(' ');
		USART_Transmit('P');
		USART_Transmit('A');
```

```
                USART_Transmit('R');
                USART_Transmit('A');
                USART_Transmit(' ');
                USART_Transmit('C');
                USART_Transmit('O');
                USART_Transmit('N');
                USART_Transmit('T');
                USART_Transmit('I');
                USART_Transmit('N');
                USART_Transmit('U');
                USART_Transmit('A');
                USART_Transmit('R');
                USART_Transmit(' ');
                USART_Transmit('o');
                USART_Transmit(' ');
                USART_Transmit('0');
                USART_Transmit(' ');
                USART_Transmit('P');
                USART_Transmit('A');
                USART_Transmit('R');
                USART_Transmit('A');
                USART_Transmit(' ');
                USART_Transmit('C');
                USART_Transmit('O');
                USART_Transmit('R');
                USART_Transmit('R');
                USART_Transmit('E');
                USART_Transmit('G');
                USART_Transmit('I');
                USART_Transmit('R');
                USART_Transmit(':');
                USART_Transmit(enter);

                EnviarDireccion(a);
                j++;
        }
        else
        {

                a=' ';
                verificar();

        }
        return a;
}

void verificar(){

        USART_Transmit('E');
        USART_Transmit('R');
        USART_Transmit('R');
        USART_Transmit('O');
        USART_Transmit('R');
        USART_Transmit('!');
        USART_Transmit(' ');
        USART_Transmit('I');
        USART_Transmit('N');
        USART_Transmit('G');
        USART_Transmit('R');
        USART_Transmit('E');
        USART_Transmit('S');
        USART_Transmit('E');
        USART_Transmit(' ');
        USART_Transmit('U');
        USART_Transmit('N');
        USART_Transmit(' ');
        USART_Transmit('V');
        USART_Transmit('A');
        USART_Transmit('L');
        USART_Transmit('O');
        USART_Transmit('R');
        USART_Transmit(' ');
        USART_Transmit('C');
        USART_Transmit('O');
        USART_Transmit('O');
        USART_Transmit('R');
        USART_Transmit('R');
        USART_Transmit('E');
        USART_Transmit('C');
        USART_Transmit('T');
        USART_Transmit('O');
        USART_Transmit('!');
        USART_Transmit(enter);
}

char ContinuarCorregir (){

        c=UDR0;
        USART_Transmit(c);
        USART_Transmit(enter);

        if (c==48 || c==49)
        {

                if (c==48)
                {
                        c=0;
                        reiniciar();
                        USART_Transmit('I');
                        USART_Transmit('N');
                        USART_Transmit('G');
                        USART_Transmit('R');
                        USART_Transmit('E');
                        USART_Transmit('S');
                        USART_Transmit('E');
                        USART_Transmit(' ');
                        USART_Transmit('N');
                        USART_Transmit('U');
                        USART_Transmit('M');
                        USART_Transmit('E');
                        USART_Transmit('R');
                        USART_Transmit('O');
                        USART_Transmit(' ');
                        USART_Transmit('D');
                        USART_Transmit('E');
                        USART_Transmit(' ');
                        USART_Transmit('P');
                        USART_Transmit('A');
                        USART_Transmit('S');
                        USART_Transmit('O');
                        USART_Transmit('S');
                        USART_Transmit(' ');
                        USART_Transmit('(');
                        USART_Transmit('1');
                        USART_Transmit('-');
                        USART_Transmit('9');
                        USART_Transmit(')');
                        USART_Transmit(':');
                        USART_Transmit(enter);
                }
                else
                {
                        c=1;
                        j++;


                        USART_Transmit('S');
                        USART_Transmit('I');
                        USART_Transmit('G');
```

```
                        USART_Transmit('U');
                        USART_Transmit('I');
                        USART_Transmit('E');
                        USART_Transmit('N');
                        USART_Transmit('T');
                        USART_Transmit('E');
                        USART_Transmit('S');
                        USART_Transmit(' ');
                        USART_Transmit('D');
                        USART_Transmit('A');
                        USART_Transmit('T');
                        USART_Transmit('O');
                        USART_Transmit('S');
                        USART_Transmit(enter);
                        USART_Transmit('I');
                        USART_Transmit('N');
                        USART_Transmit('G');
                        USART_Transmit('R');
                        USART_Transmit('E');
                        USART_Transmit('S');
                        USART_Transmit('E');
                        USART_Transmit(' ');
                        USART_Transmit('N');
                        USART_Transmit('U');
                        USART_Transmit('M');
                        USART_Transmit('E');
                        USART_Transmit('R');
                        USART_Transmit('O');
                        USART_Transmit(' ');
                        USART_Transmit('D');
                        USART_Transmit('E');
                        USART_Transmit(' ');
                        USART_Transmit('P');
                        USART_Transmit('A');
                        USART_Transmit('S');
                        USART_Transmit('O');
                        USART_Transmit('S');
                        USART_Transmit(' ');
                        USART_Transmit('(');
                        USART_Transmit('1');
                        USART_Transmit('-');
                        USART_Transmit('9');
                        USART_Transmit(')');
                        USART_Transmit(':');
                        USART_Transmit(enter);
                }

        }
        else
        {

                verificar();

        }

        return c;
}

void reiniciar(){
        n=0;
        v=0;
        a=' ';
        PORTA=0x00;



        inicializarLCD();
                j=1;
        }

        void EnviarPasos(n){

                escribir_lcd('P');
                escribir_lcd('A');
                escribir_lcd('S');
                escribir_lcd('O');
                escribir_lcd('S');
                escribir_lcd(':');
                escribir_lcd(' ');

                escribir_lcd(n);

        }
        void EnviarVelocidad(v){

                escribir_lcd(' ');
                escribir_lcd(' ');
                escribir_lcd('V');
                escribir_lcd('E');
                escribir_lcd('L');
                escribir_lcd(':');
                escribir_lcd(' ');

                escribir_lcd(v);
                enter_lcd();

        }

        void EnviarDireccion(char a){

                escribir_lcd('D');
                escribir_lcd('I');
                escribir_lcd('R');
                escribir_lcd(':');
                escribir_lcd(' ');

                if (a=='i'||a=='I')
                {
                        escribir_lcd('I');
                        escribir_lcd('Z');
                        escribir_lcd('Q');
                        escribir_lcd('U');
                        escribir_lcd('I');
                        escribir_lcd('E');
                        escribir_lcd('R');
                        escribir_lcd('D');
                        escribir_lcd('A');
                        escribir_lcd('.');

                }
                else
                {
                        escribir_lcd('D');
                        escribir_lcd('E');
                        escribir_lcd('R');
                        escribir_lcd('E');
                        escribir_lcd('C');
                        escribir_lcd('H');
                        escribir_lcd('A');
                        escribir_lcd('.');
                }

        }
```

CONCLUSIONES.

Del desarrollo de la práctica de laboratorio se puede concluir que:

- La programación en un lenguaje de mayor nivel requiere menos líneas de código para el desarrollo de soluciones.
- El control de un dispositivo que requiera una alta corriente o voltaje de funcionamiento mayor al entregado por un microcontrolador (aproximadamente 5V), se puede realizar con el uso de transistores conectados al microcontrolador.
- El uso de comunicación serial es más económica de desarrollar, además de que es menos propensa a errores.
- El uso de las terminales virtual permite una comunicación más eficaz entre el programa y el usuario.