CAPITULO III

3. INGENIERÍA DE DETALLE

3.1. DIAGRAMA ELÉCTRICO E IMPLEMENTACIÓN

El diagrama eléctrico representa un concepto técnico y se vuelve complejo si no se está familiarizado con el área eléctrico-electrónica, en el mismo se representa de forma simbólica la combinación de cierto número de elementos unidos en puntos terminales que proporcionan al menos una ruta cerrada a través de la cual la carga puede fluir. El diagrama de la figura 3.1 representa el diseño de cómo funciona el sistema controlador electrónico automático en conjunto con el diagrama eléctrico completo de la máquina cortadora, líneas de alimentación, líneas de control, líneas de datos y sistema mecánico intervenido en el proceso.

La importancia del diagrama eléctrico radica en que gracias a él se conocen los componentes del sistema y con esta información se puede definir las pruebas aplicables al mismo. Cabe recalcar que la explicación de cada parte de este diagrama se detalla en los temas del diagrama de bloques.

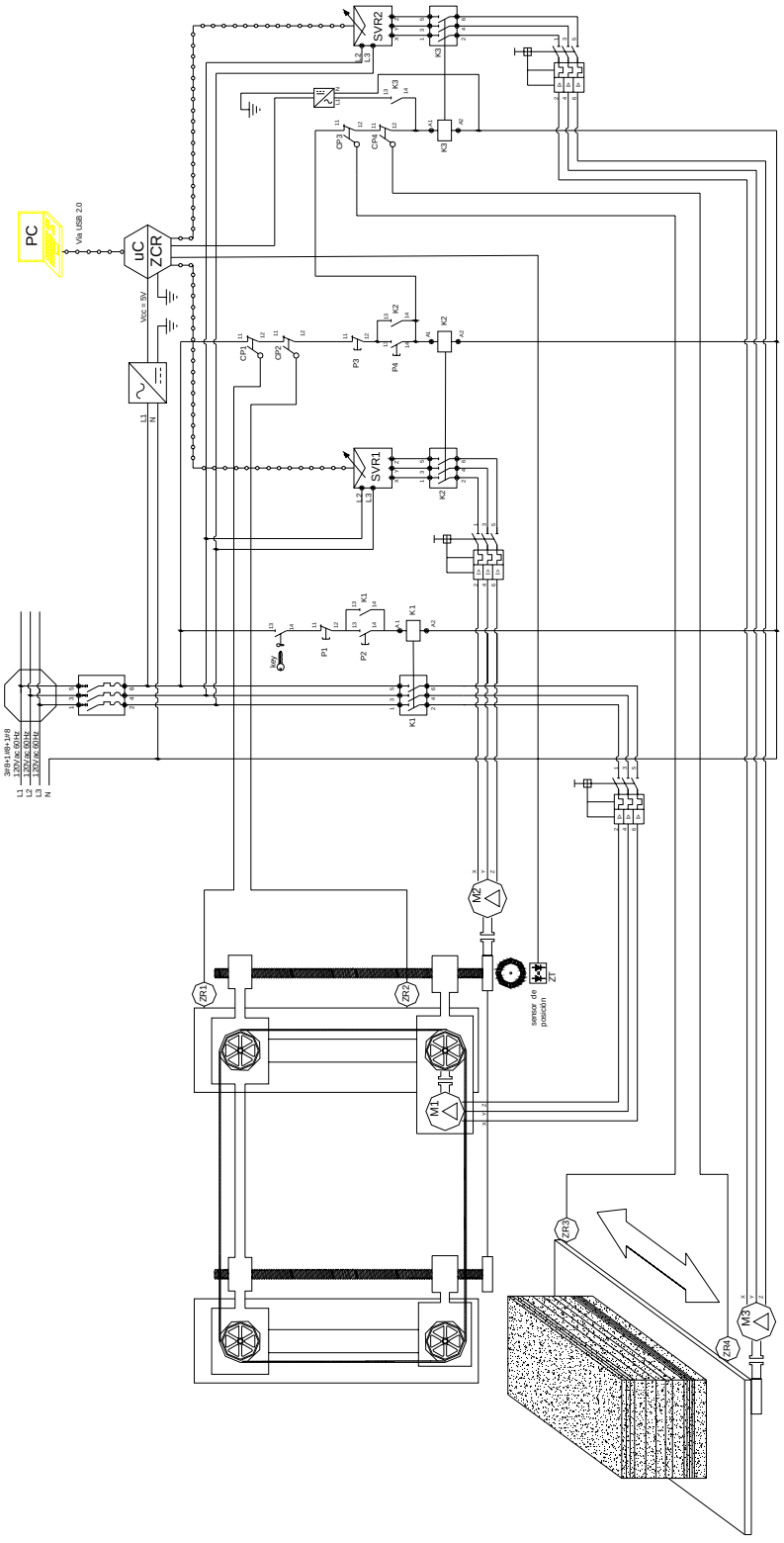


Figura 3.1. Diagrama eléctrico implementado

3.2. DIAGRAMA DE FUERZA

Dentro del diagrama eléctrico implementado, se encuentra el sistema de alimentación de los motores que viene a ser las líneas de fuerza que accionan los mismo, la siguiente figura 3.2, indica cómo se encuentra instalado el sistema eléctrico de las líneas de fuerza y sus respectivas protecciones desde la alimentación trifásica principal del sistema hasta las entradas a los bobinados de los motores, tanto para M1 que es el motor de la cuchilla y que se activa directamente, como para M2 y M3 que son los motores controlados por los variadores.

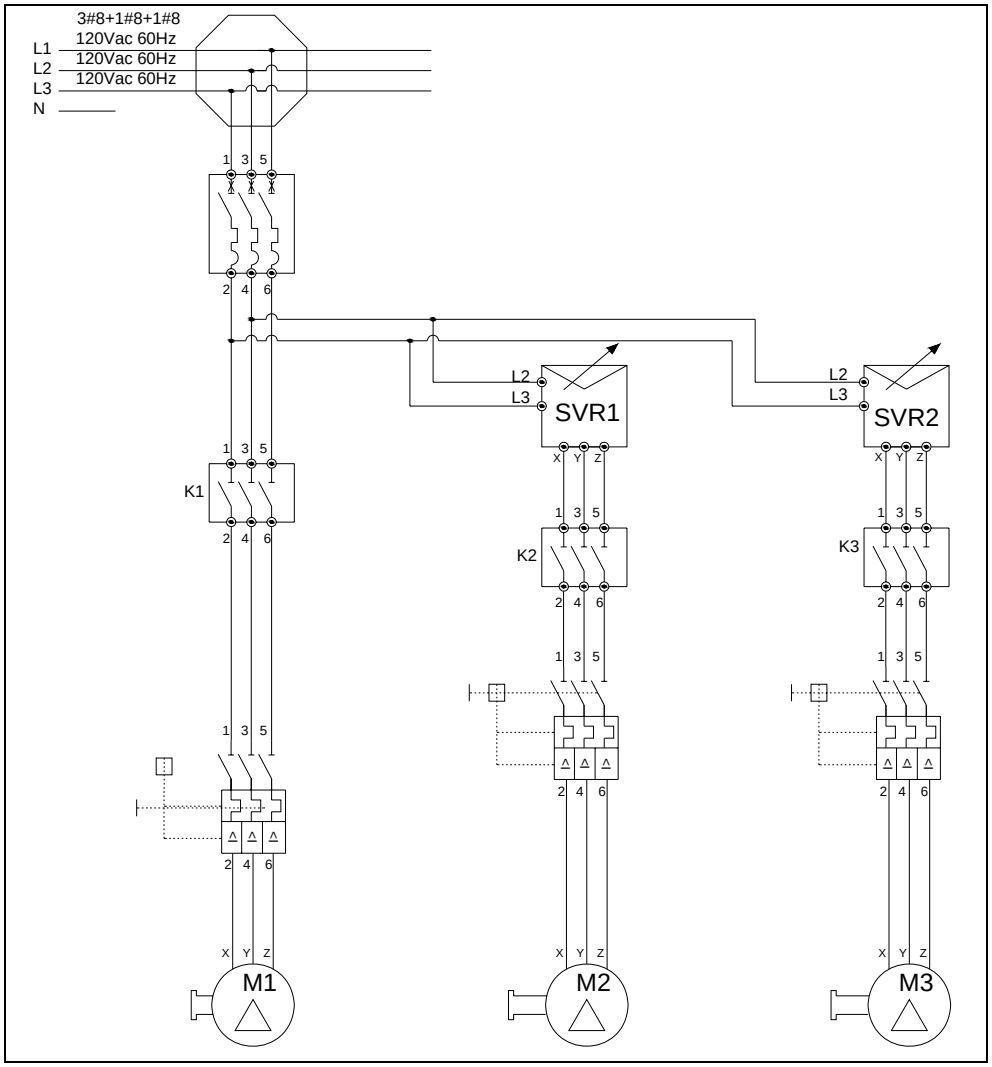


Figura 3.2. Diagrama de fuerza

3.2.1 Actuadores

Para dar energía a los variadores de frecuencia, la alimentación trifásica pasa por un breaker de tres líneas el cual viene a ser la primera protección del sistema en cuestiones eléctricas. De donde se toman dos fases y los variadores se conectan en paralelo a estas fases, dejando una de las tres libre para utilizarla en las líneas de control.

La alimentación eléctrica para los motores, proviene de la salida trifásica de los variadores de frecuencia, pasa por los contactores enclavados, y pasa a través de la protección de los disyuntores termomagnéticos.

Los motores M2 y M3 son considerados como actuadores, ya que están encargados de los movimientos vertical de la estructura portadora de la cuchilla y el movimiento horizontal de la plataforma portadora del bloque de esponja, respectivamente.

Control por frecuencia.- El método se fundamenta en que la velocidad sincrónica del campo magnético rotatorio de un motor asincrónico puede ser controlada por medio de la variación de la frecuencia de la línea, ya que:

ns = 120 f/P
Donde:
ns: Velocidad sincrónica, rpm.
f: Frecuencia de la línea, Hz).
P: Número de polos.

Pero a fin de mantener el desplazamiento en el eje Y aproximadamente constante y que no haya afectaciones en el momento que desarrolla el motor, la tensión de línea debe variarse también proporcionalmente a la frecuencia, es decir, U1 / f debe ser aproximadamente constante.

En este caso, al disminuir la tensión en mayor proporción que la frecuencia, se produce una reducción de velocidad de desplazamiento inicial y mejoran los indicadores energéticos del motor, al mismo tiempo que la disminución del momento máximo no es peligrosa desde el punto de vista de la capacidad de sobrecarga.

Cargabilidad del motor.- Cuando se utiliza un motor asincrónico con convertidor de frecuencia, en adición a los criterios generales de selección, se deben considerar los aspectos siguientes:

La tensión (y la corriente) con la cual el convertidor alimenta al motor no es puramente sinusoidal, lo cual, como resultado, incrementa las pérdidas, las vibraciones y el ruido de los motores. Distintos convertidores con diferentes frecuencias de corte y de modulación proporcionan comportamientos distintos para el mismo motor. Por esta razón, no resulta recomendable utilizar métodos empíricos generales para determinar la cargabilidad del motor.

Se debe hacer la selección a partir de las curvas de cargabilidad del motor, correspondientes a los distintos tipos específicos de convertidores de frecuencia que suministran los fabricantes.

En la figura 3.3 se aprecia una curva de arranque con un convertidor de frecuencia en base a datos adquiridos con un multímetro Fluke y una conexión a pc vía cable IR189 USB y software de adquisición de datos Flukeview. Este tipo de curva muestra el momento máximo continuo con respecto al momento nominal, para el motor M2, en función de la corriente. Operando en las condiciones que establece esta curva, no se sobrepasa el calentamiento nominal que alcanza el motor cuando trabaja alimentado de una red a frecuencia y tensión sinusoidal nominales, y a plena carga. Así como los momentos que pueden desarrollarse en el proceso de arranque y las sobrecargas de corto tiempo permisibles.

Amperios (A)	Tiempo (s)
0	0
0,11	19
0,22	8
0,33	2,65
0,44	2,92
0,55	3,23
0,66	3,45
0,77	4,12
0,88	4,34
0,99	4,5
1,1	4,93
1,21	5,14
1,32	5,34
1,43	5,56
1,54	5,98
1,65	6,23
1,76	6,54
1,87	6,98
1,98	7,2
2,09	7,58
2,2	7,83
2,31	7,92
2,42	7,89
2,53	7,64
2,64	7,77
2,75	7,68
2,86	7,71
2,97	7,74

Tabla 3.1. Datos adquiridos de arranque del motor M2

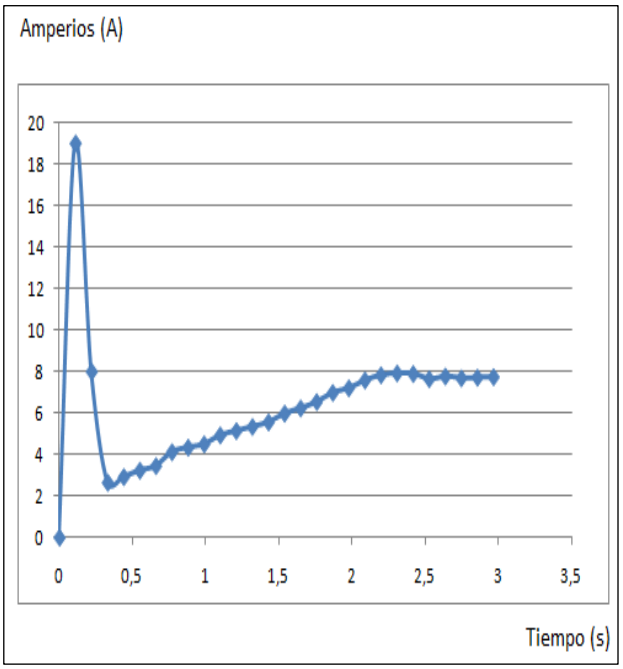


Fig. 3.3. Curva de arranque del motor M2

3.2.2. Transferencia de energía a los contactores

Cálculos para los breakers contactores y selección de cables:

$$P = V * I$$

$$P = \sqrt{3} * V * I * fp$$

$$V = 110Vac$$

$$P = 3000W$$

$$fp = 0.9$$

$$I = \frac{P}{\sqrt{3} * V * fp}$$

$$I = \frac{3000W}{\sqrt{3} * 110Vac * 0.9}$$

$$I = 17{,}45A$$

De acuerdo a la tabla 3.2. que representa la capacidad en corriente de los conductores, se eligió conductores calibre 12. Ya que el mismo soporta hasta 22A

Calibre AWG No	Resistencia Ω/100 m	Amperaje Máximo (A)* TIPO DE CABLE UF	USE,THW TW,THWN	NM	Dimensiones Diám. mm	Area cm2
4/0	0,01669	211	248		13,412	1,4129
3/0	0,02106	178	216		11,921	1,1161
2/0	0,02660	157	189		10,608	0,8839
1/0	0,03346	135	162		9,462	0,7032
2	0,05314	103	124		7,419	0,4322
4	0,08497	76	92		5,874	0,2710
6	0,1345	59	70		4,710	0,1742
8	0,2101	43	54		3,268	0,0839
10	0,3339	32	32	30	2,580	0,0523
12	0,5314	22	22	20	2,047	0,0329
14	0,8432	16	16	15	1,621	0,0206

Tabla 3.2 Calibres de cables conductores

Para la selección de los interruptores Termomagnéticos y la capacidad de conducción de los contactores se parametrizó según la sección de los conductores de ese circuito, es decir de 20A, que es la carga máxima admisible en el circuito, y está dada por la sección de los conductores implementados.

La tensión sale de los variadores de frecuencia, llega a los contactores que al estar previamente enclavados, se transmitirá a los motores, se manejó un margen de error de ±10% para la parametrización de los disyuntores termomagnéticos que se colocaron después de los contactores según lo recomendado por el fabricante para proteger el motor de un posible arranque brusco. Es importante indicar que los variadores funcionarán perfectamente cuando la tensión que ingrese en los variadores tenga una frecuencia de red de 47 a 63 Hz y un Cos phi ≥ 0,95.

3.2.3 Alimentación de voltaje del circuito de control

El sistema al constar de un microcontrolador, se sabe que por indicaciones en el datasheet del elemento, necesita un voltaje de 5Vdc de 200W con una estabilidad garantizada para lo que por recomendación y experiencia, se optó por utilizar una fuente de PC ATX de la figura 3.4 en vista de la reducción de costos.

Figura 3.4 Fuente de alimentación 5Vdc

La Fuente de Alimentación, es un montaje eléctrico/electrónico capaz de transformar la corriente de la red eléctrica en una corriente que el microcontrolador pueda soportar.
Para encender la fuente ATX, es necesario identificar los cables verde y negro que están en el conector ATX principal, El verde (pin 14) es designado como el poder en línea, mientras que el negro (Pin 13) es sólo una línea de tierra como se muestra en la figura 3.5.

Figura 3.5. Distribución de pines de una fuente ATX

Sobre estos cables, se va a soldar un pulsador que ira colocado en el tablero de control, mismo que enviara la señal de encendido a la fuente, luego se identifica en cualquier conector un cable rojo que es el de 5Vdc y un cable negro para referencia de 0v o tierra, una vez identificados los cables de encendido de la fuente de PC ATX, se puede conectar los cables de 5Vdc al circuito para que entre en funcionamiento.

Cabe indicar que todo el circuito electrónico del cerebro funciona con tecnología TTL de aquí la importancia de la alimentación con una gran estabilidad y correcto funcionamiento de la fuente.

3.3. DIAGRAMA DE BLOQUES DEL SISTEMA DE CONTROL

El siguiente diagrama de bloques es la representación gráfica del funcionamiento interno del sistema, que se hace mediante bloques y sus relaciones, y que, además, definen la organización de todo el proceso, sus entradas y sus salidas.

En el diagrama de bloques el proceso de análisis principal lo realiza básicamente un microcontrolador como se muestra en la figura 3.6 y lo que hace es recibir todas las instrucciones provenientes de: teclado y sensores previamente acondicionadas para procesarlas según el programa establecido en el microcontrolador, este generará señales de control las cuales son recibidas por los variadores de frecuencia para controlar los motores M2 y M3. También tiene comunicación directa con el L.C.D para la interacción con el usuario, y una interface USB para el monitoreo.

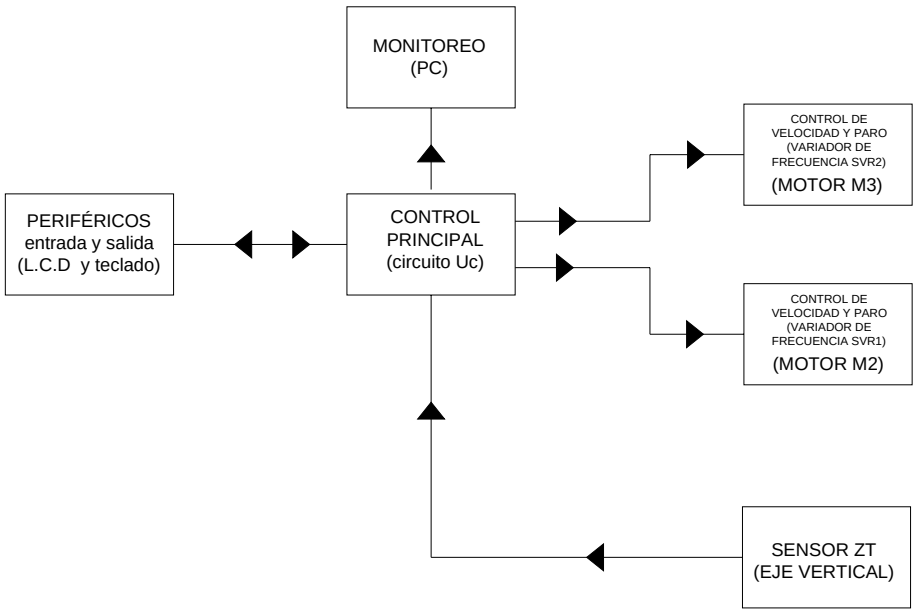


Figura 3.6 Diagrama de bloques del sistema

3.3.1. Control Principal (UC)

Este sistema de control es completamente diseñado en base a todas las necesidades, requerimientos y sugerencias del usuario y desarrollado aplicando teoría de microcontroladores, para lo cual se requiere de un conocimiento previo de un sistema basado en entradas y salidas lógicas programables, además de un correcto manejo de control a base de interrupciones del microcontrolador, conociendo tiempo de ejecuciones de las rutinas programadas y sistemas de acondicionamiento de señales tanto de entradas como de salidas al microcontrolador.

Toda su programación se realizó en lenguaje C para microntroladores bajo el programa PCW C Compiler IDE versión 3.227 y tanto su simulación como su placa base fueron desarrolladas bajo el software Proteus 7.0 de Labcenter Electronics © como se puede apreciar en las figuras 3.7 y 3.8.

Como características relevantes del uC, se presenta la funcionalidad de comunicación vía USB que genera facilidad de interconexión con PC's actuales, además de una amplia memoria de programación a comparación con otros modelos de uC, misma que mejora el rendimiento del mismo y no limita en opciones de programación, cabe nombrar que este modelo de uC, trabaja a 4Mhz, suficiente velocidad para mejorar el proceso de análisis de datos de entrada respecto a las señales de salida.

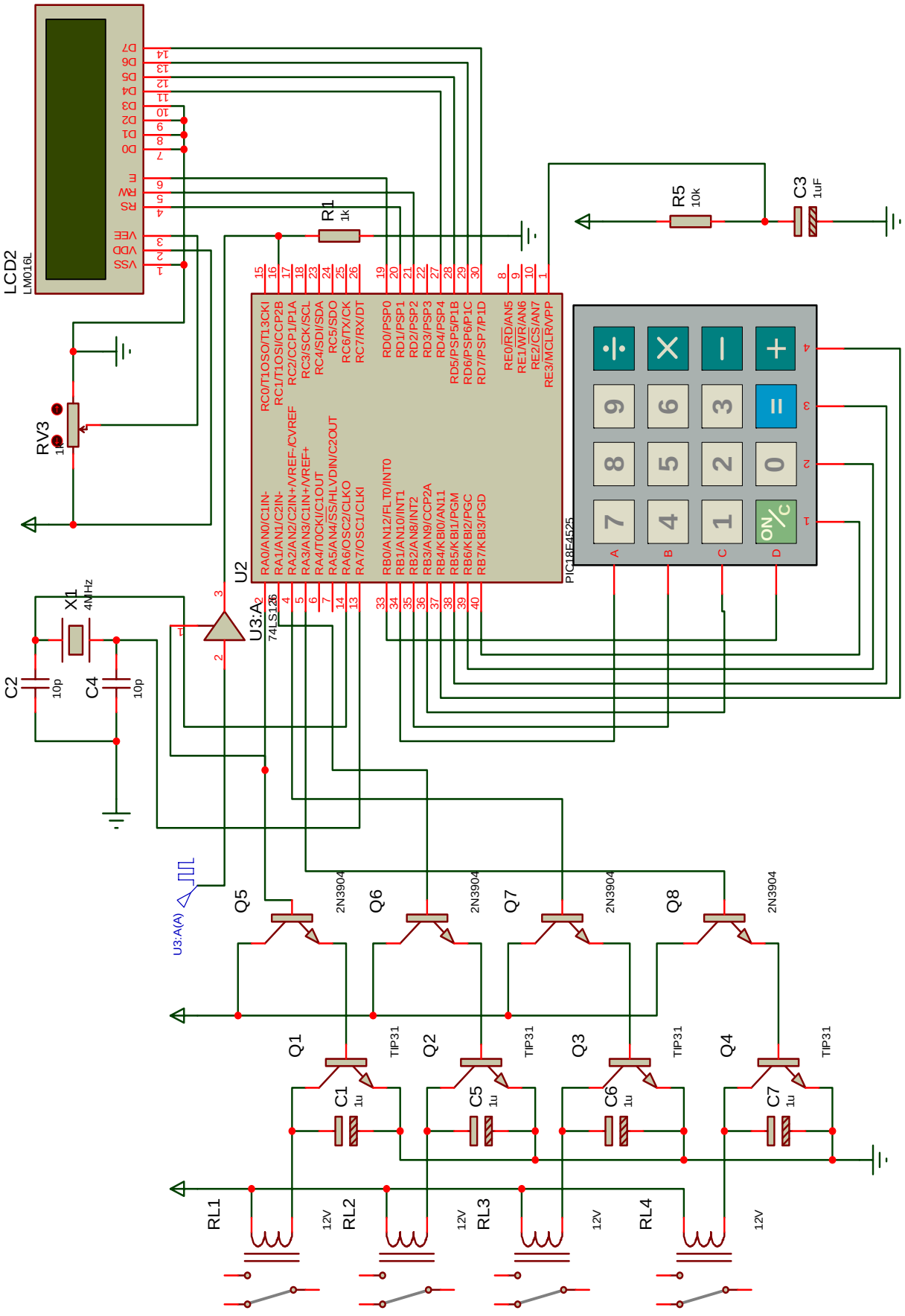


Figura 3.7 Diagrama circuito controlador principal uC

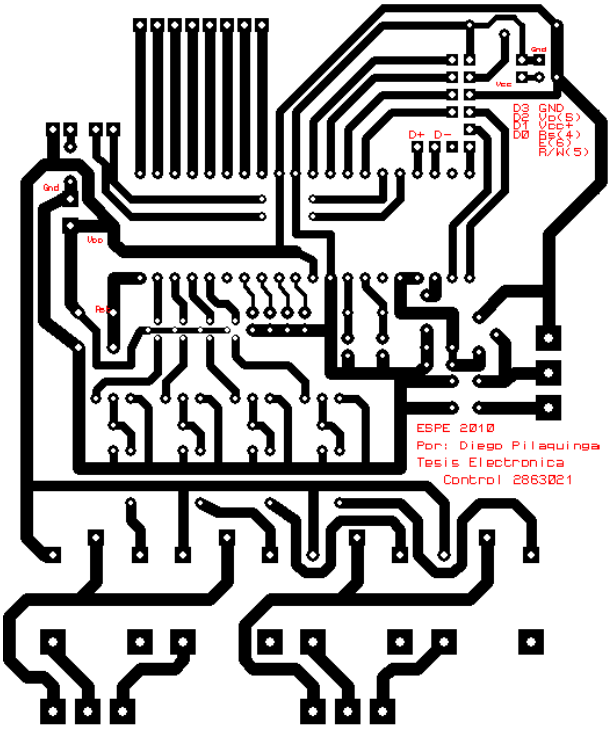


Figura 3.8 Diseño de la tarjeta principal para el uC

Para la comunicación de datos, usamos la conexión USB, hay un orden estándar de conexión, que es el que se puede ver en la figura 3.9 mismo que esta implementado sobre la tarjeta principal, de los cuales sale un cable USB directo a ser conectado a un puerto USB de cualquier PC.

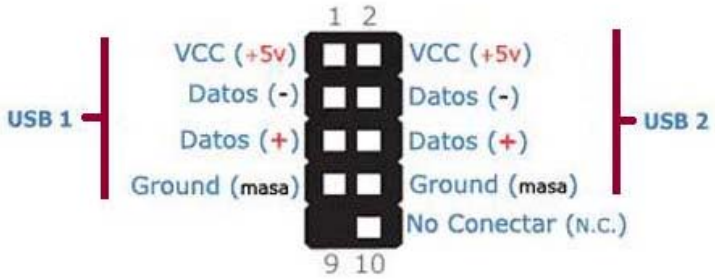


Figura 3.9. Distribución de pines USB

Los conectores USB son dobles (es decir, en cada conector se puede insertar dos puertos USB). Es muy importante que no se mezcle los cables de uno y otro, sobre todo los de datos (señalados como **-D** y **+D**).

El orden correcto de conexión es el siguiente:

VCC (5v)

- D

+ D

GROUND (GD, GR, MASA)

A un USB corresponden los pines pares (2, 4, 6 y 8) y a otro los impares (1, 3, 5 y 7), quedando el hueco correspondiente al pin 9 y el pin 10 como pin de control (no se conecta, aunque algunos conectores lo tienen, sobre todo los dobles).

Ahora bien, siempre se debe consultar el manual de la placa base para asegurarse de que el orden de los pines de esa placa en concreto se corresponde con el estándar.

En la figura 3.10 se ve un juego de conectores de un puerto USB externo diseñado para funcionar de acuerdo a nuestra placa de circuito impreso.

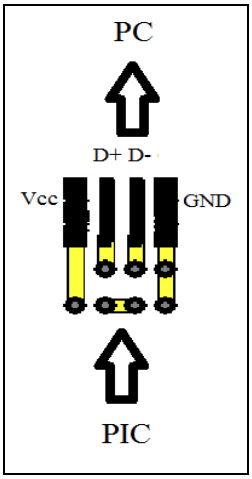


Figura 3.10 Distribución de pines en el cable USB.

En él se puede ver serigrafiados a que pin corresponde cada cable.

3.3.2. Periféricos

Los periféricos que tiene el sistema son:

El teclado, que se muestra en la figura 3.11, es la única forma de intervención del usuario con el control principal y sirve para asignar los parámetros que son ingresados secuencialmente e indicados por el L.C.D.

El teclado internamente consta de interruptores internos, se envía desde el uC una señal tipo barrido a través de las columnas que son las entradas del mismo y al presionar cualquier botón, se provoca una interrupción en los pines de salida que vienen a ser las filas del teclado, esta señal viaja a través del interruptor presionado y llega a las entradas RB# del uC, el uC identifica el pin donde se provocó la interrupción y llama a la librería kbd_getc() la misma que fue inicializada por medio de la subfunción principal de manejo de teclado kbd_init(); que viene dentro de las librerías funcionales que brinda PCW C Compiler, esta función me devuelve el dato en código anssi de la tecla presionada y lo almacenamos en la variable tecla mediante el código: tecla=kbd_getc();

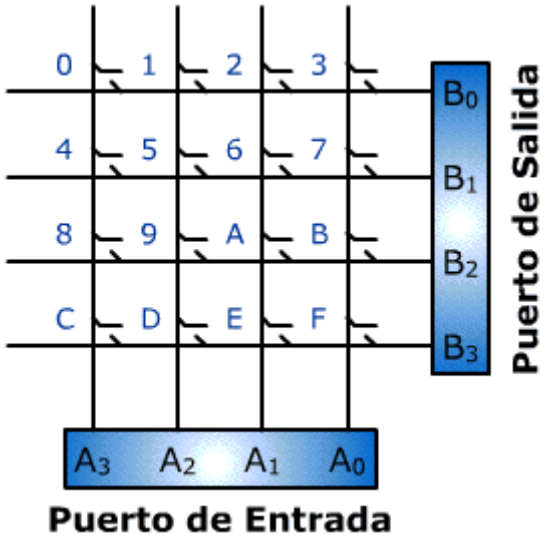


Figura 3.11. Teclado matricial 4x4

- El L.C.D que muestra en la figura 3.12 se encarga de mostrar todo lo que el usuario va a ingresar por teclado y además mensajes de errores como: que no están listas las condiciones iniciales de trabajo o que alguna protección está activada. También pide los datos que se necesita para realizar el corte.

  El L.C.D. es una pantalla de cristal líquido de 2 filas por 16 caracteres, suficiente para mostrar el ingreso de los datos y el proceso de la máquina, este teclado se inicializa con la librería lcd_init(); esta función viene dentro del programa PCW C Compiler y facilita el manejo de los datos que van hacia el L.C.D. ya que una vez inicializado, se le envía en forma de código anssi los datos desde el PIC a dicha pantalla de la siguiente manera:

  printf(lcd_putc,” MENSAJE A MOSTRAR “);

  Esta identifica y decodifica internamente los datos enviados por el uC y los muestra en la pantalla de acuerdo a la posición donde se lo haya indicado.

Figura 3.12. Lcd 2x16 caracteres implementado

3.3.3. Control de velocidad y paro

Se ha colocado sobre el diagrama de fuerza los variadores y sus conexiones ya que reciben una señal eléctrica de control de 24 V y mediante esa señal, controlan las líneas de alimentación eléctrica de 110V para dirigir el funcionamiento de los actuadores que en este caso son M2 y M3. Los pasos para su programación, calibración y puesta en marcha se detalla a continuación y es adecuada para la mayoría de las aplicaciones.

Las características de funcionamiento del variador se configuran mediante la asignación de valores a registros de configuración que tienen el formato PXXXX

Los registros son propios del software del variador en donde se ingresa los siguientes valores a través de un teclado desmontable BOP y en este orden:

1) Seleccionar puesta en servicio en P0010 de las siguientes opciones:
   0 Preparado
   1 Puesta en servicio
   30 Ajustes de fábrica

2) En P0100 se selecciona el Funcionamiento para Europa o Norteamérica, se tomó la opción tres por los datos de placa del motor :
   0 Potencia en kW; f por defecto 50 Hz
   1 Potencia en hp; f por defecto 60 Hz
   2 Potencia en kW; f por defecto 60 Hz

3) En P0304 se ingresa la Tensión nominal del motor es decir 220V.
   Rango de ajuste: 10 V - 2000 V
   Tensión nominal del motor (V) de la placa de Características

4) En P0305 se ingresa la Corriente nominal del motor
   Rango de ajuste: 0 - 2 x corriente nominal del convertidor (A)
   Corriente nominal del motor (A) de la placa de Características

5) En P0307 se ingresa la Potencia nominal del motor
   Rango de ajuste: 0,12 kW – 3,0 kW (0,16 hp –4,02 hp)
   Potencia nominal del motor (kW) de la placa de características. Si P0100 = 1, los valores serán en hp

6) En P0310 se ingresa la Frecuencia nominal del motor que es 60 Hz.
   Rango de ajuste: 12 Hz - 650 Hz
   Frecuencia nominal del motor (Hz) de la placa de Características

7) En P0311 se ingresa la Velocidad nominal del motor que es 1200 rpm
   Rango de ajuste: 0 - 40000 1/min
   Velocidad nominal del motor (rpm) de la placa de Características

8) En P0700 la selección de la fuente de comandos se escogió la opción dos ya que las señales de control del uC ingresan al variador por los bornes.
   1 BOP
   2 Bornes/entradas digitales
   5 USS (sólo variante USS)
9) En P1080 se ingresa la Frecuencia mínima del motor la cual es 0 Hz para el caso extremo de frenado.

   Ajusta la frecuencia mínima del motor (0-650Hz) a la que girará el motor con independencia de la consigna de frecuencia. El valor aquí ajustado es válido tanto para giro a derecha como a izquierda.

10) En P1082 se ingresa la Frecuencia máxima del motor la cual se ha dado como 60 Hz

Ajusta la frecuencia máxima del motor (0-650Hz) a la que girará el motor con independencia de la consigna de frecuencia. El valor aquí ajustado es válido tanto para giro a derechas como a izquierdas.

11) En P1120 se ingresa el Tiempo de aceleración el cual tiene un estimado de 2s

Rango de ajuste: 0 s - 650 s

Tiempo que tarda el motor para acelerar desde el estado de reposo hasta la frecuencia máxima del motor.

12) En P1121 se ingresa el Tiempo de deceleración par el cual se asigno un valor de 0.5 s ya que por el control se necesita un frenado inmediato casi instantáneo.

Rango de ajuste: 0 s - 650 s

Tiempo que tarda el motor para decelerar desde la máxima frecuencia del motor hasta el estado de reposo.

13) Desde la P0701 hasta la P0704 se dan las funciones de las entradas digitales de la 0 a la 3 respectivamente y correspondientes físicamente a los bornes 3, 4, 5 y 9. La función se da ingresando el valor deseado de la tabla 3.3.

Posibles ajustes:	
0	Entrada digital deshabilitada
1	ON / OFF1
2	ON inverso / OFF1
3	OFF2 - parada natural
4	OFF3 - deceleración rápida
9	Acuse de fallo
10	JOG derechas
11	JOG izquierda
12	Inversión
13	MOP subida (incremento frec.)
14	MOP bajada (decremento frec.)
15	Frec. fija (selección directa)
16	Frec. fija (sel. dir. + MARCHA)
21	Local/remoto
25	Act. freno inyecc.corr.continua
29	Fallo externo

Tabla 3.3 Configuración del Variador

Para lo que se asigno a:

Borne 3 el valor de 10 para que gire a la derecha

Borne 4 el valor de 11 para que gire a la izquierda

Borne 5 el valor de 4 para un frenado de emergencia.

14) En P3900 se debe finalizar la puesta en servicio escogiendo la opción tres al ingresar el número dos.

0= Sin puesta en servicio rápida sin cálculo del motor ni reajuste de fábrica.

1= Fin puesta en servicio rápida con cálculo del motor y reajuste de fábrica.

2= Fin puesta en servicio rápida con cálculo del motor y reajuste de E/S.

3= Fin puesta en servicio rápida con cálculo del motor pero sin reajuste de fábrica

Funcionamiento eléctrico-electrónico interno de los variadores.- Para una mejor comprensión de los variadores de velocidad Sinamics G110, se incluyo el diagrama de bloques del mismo en la figura 3.13 donde se muestra sus respectivas conexiones así como sus prestaciones de donde se orienta la programación y utilización de sus pines de control.

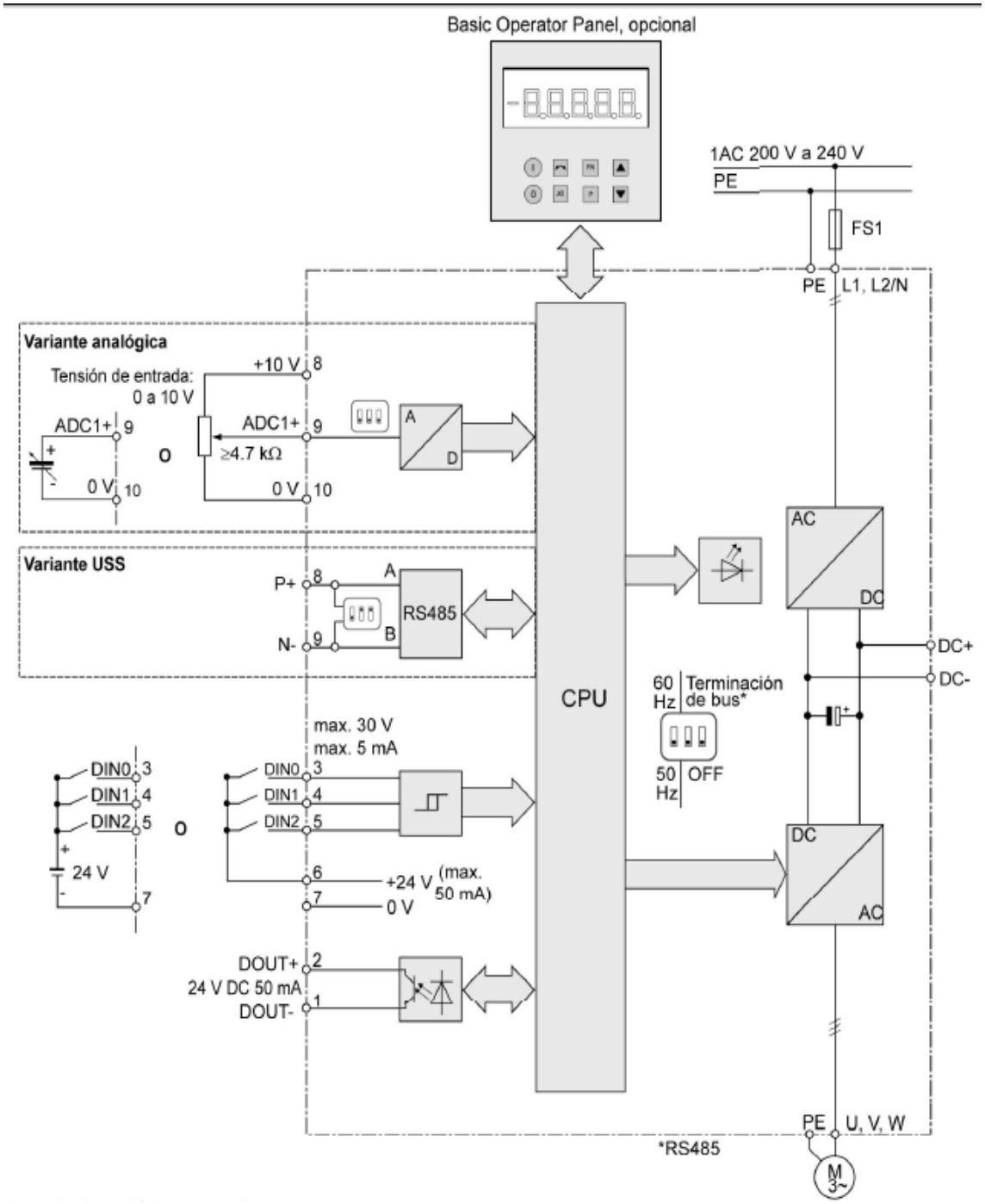
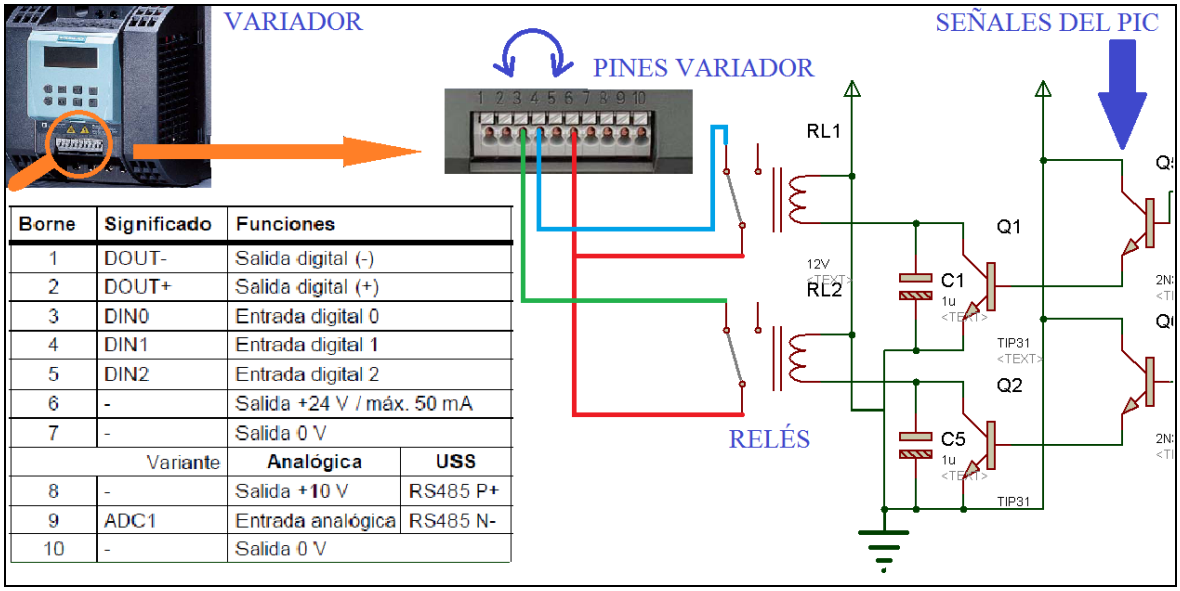


Figura 3.13. Diagrama de bloques del variador Sinamics G110

Sistema de control de los variadores.- Para transmitir la señal de control del uC a los bornes del variador se utilizó la conexión mostrada en la figura 3.14, donde el pin DIN0 es de giro a la derecha, el pin DIN1 giro a la izquierda y el pin DIN2 es parada de emergencia.

Se ocupó esta opción para asegurar la llegada de los datos por que las entradas digitales parametrizables toman valores mayores a 10V como uno lógico y menores a 8V como cero lógico, y teniendo en cuenta que soporta una tensión máxima de entrada de 30V.



Borne	Significado	Funciones	
1	DOUT-	Salida digital (-)	
2	DOUT+	Salida digital (+)	
3	DIN0	Entrada digital 0	
4	DIN1	Entrada digital 1	
5	DIN2	Entrada digital 2	
6	-	Salida +24 V / máx. 50 mA	
7	-	Salida 0 V	
	Variante	**Analógica**	**USS**
8	-	Salida +10 V	RS485 P+
9	ADC1	Entrada analógica	RS485 N-
10	-	Salida 0 V	

Figura 3.14 Conexiones de control del variador

3.3.4. Sensor ZT (Eje vertical)

La luz emitida por el diodo LED infrarrojo, se refleja en el área blanca de la polea que produce los pulsos como se muestra en la figura 3.15, incidiendo en el fototransistor (en el que la corriente del colector es proporcional a la intensidad luminosa que recibe la base del dispositivo), mientras la luz se refleja con menos intensidad en el área negra. Esta señal eléctrica a la salida del foto-transistor del sensor, es de 54uA la cual viene a ser muy débil para que la impedancia del Pic la detecte como un flanco ascendente, por esa razón, esta señal es llevada hacia la base de un transistor 2n3906 configurado como emisor común.

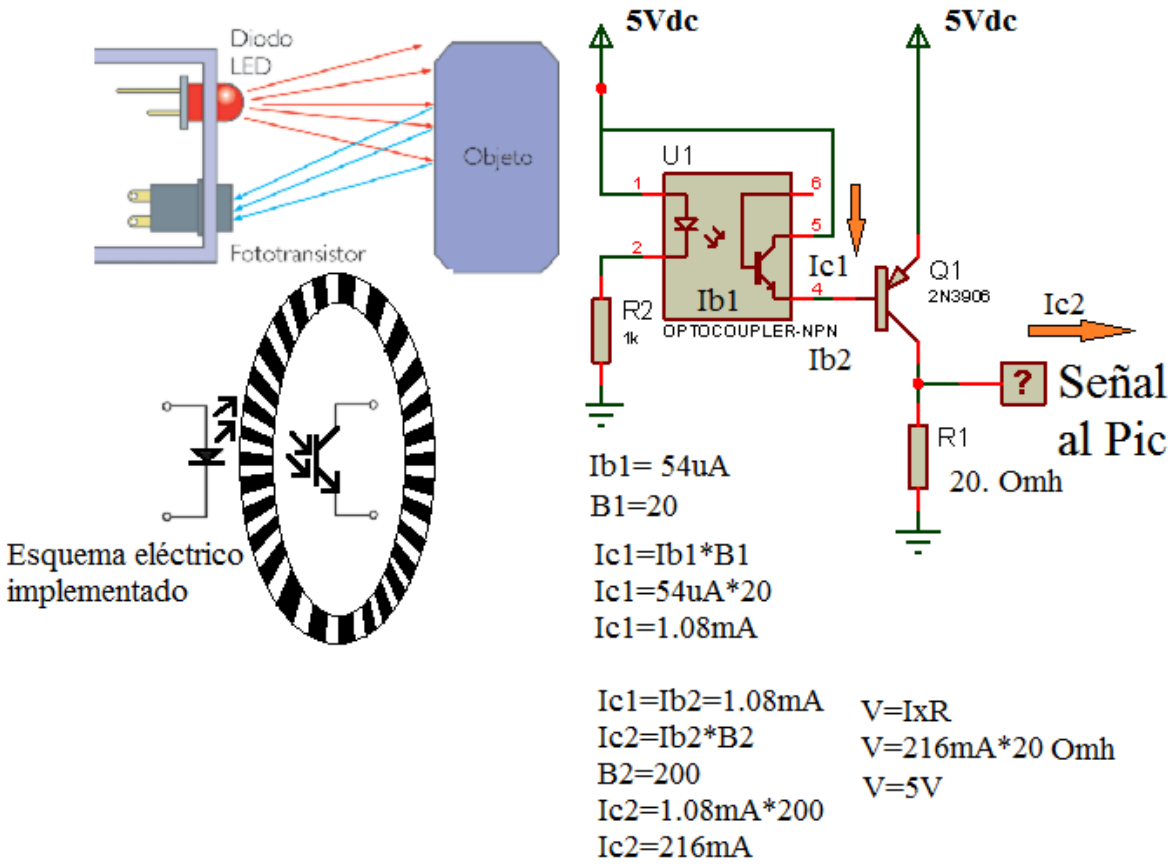


Figura 3.15 Acondicionamiento de señal del sensor

3.4 PROGRAMACIÓN Y DESARROLLO DE SOFTWARE DE CONTROL SOBRE EL MICROCONTROLADOR

3.4.1. Función principal

El sistema inicia con una función principal donde se declara todo los puertos que se van a utilizar como del mismo modo el tipo de interrupciones que se van a ejecutar y a la vez todo tipo de variables principales y auxiliares.

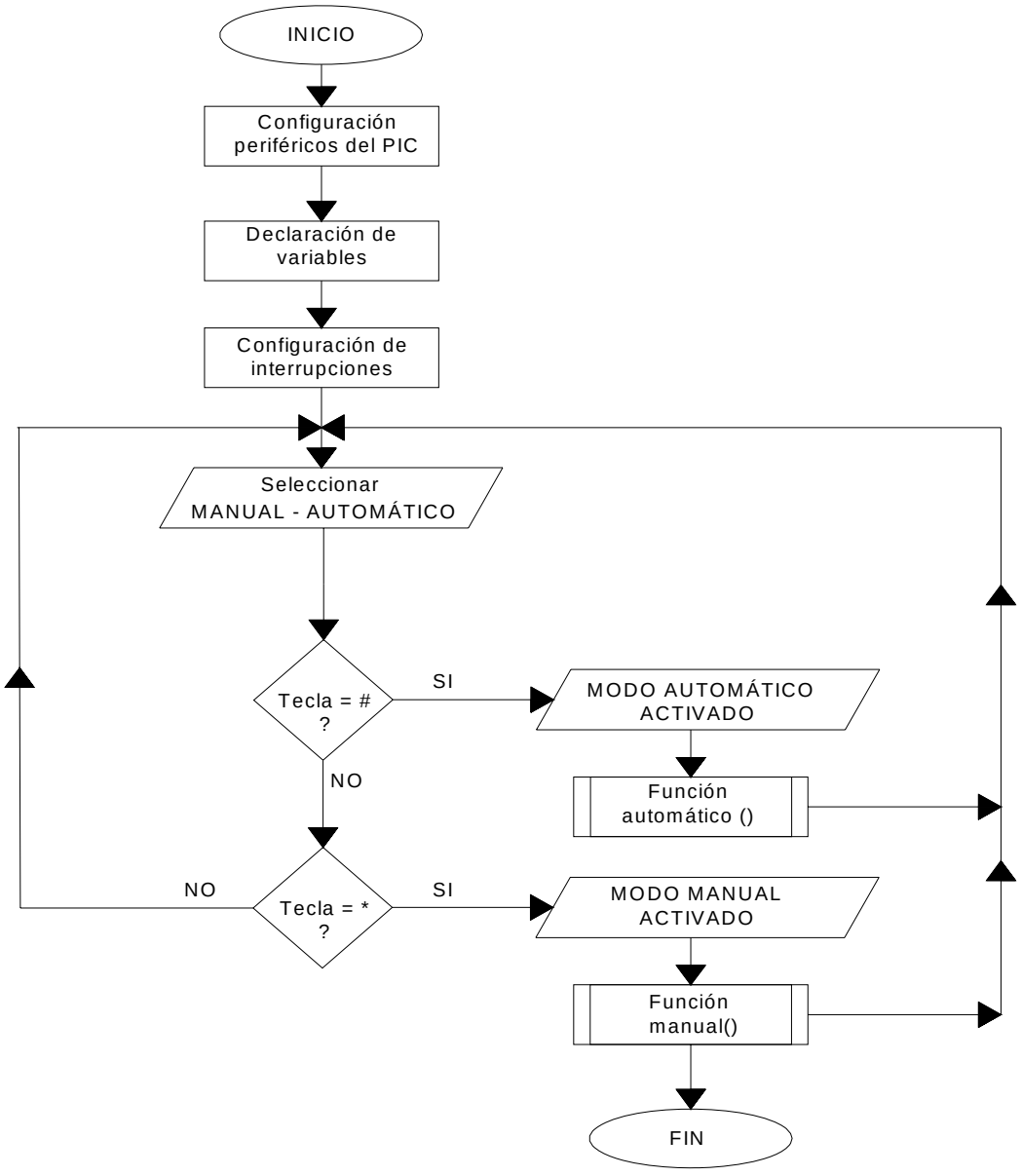


Figura 3.16. Diagrama de flujo de la función principal

```
#include <18f4550.h>
```

Se declara los fusible internos del pic los mismo que en el orden de la línea de código son: XT Cristal de 4Mhz, NOWDT no watch dog timer, Sin protección contra escritura, y no utilizamos voltajes de referencia

```
#fuses XT,NOWDT,NOPROTECT,NOLVP,NOBROWNOUT  // Fusibles
#use delay(clock=4000000)
//Librería para el manejo del lcd
#include “lcd.c”  // #include “KBD4.C”
#use standard_io(a)
#use standard_io(b)
#use standard_io(c)
#use standard_io(d)
```

Una de las partes más importantes en el programa, es la interrupción por flanco ascendente que se captura en el PIN CCP2 del Pic, la que se programa con la siguiente línea de código para utilizara en la parte automática.

```
#INT_CCP2
char  tecla;      //variable para manipular los valores ansii del teclado
int   dato;       //variable para controlar los valores decimales de las teclas
int   i;          //conteo progresivo de retardos y repeticiones
int   selec;      //variable bandera de control de selección automático/manual
```

```
float   pulsos;      //variable que cuenta los pulsos
float   distancia;        //valor de la distancia que se requiere recorrer ingresada por teclado
int   distancia2;   //valor de la distancia que se requiere recorrer comparada con la cantidad de pulsos
int   cont;       //Variable que cuenta la cantidad de pulsos que generan la interrupción por flanco ascendente en el pin CCP2
int   valor,valor1,valor2,valor3;  //Valores ingresados por teclado cada que se presiona una tecla
int avance;  //Variable tipo bandera que indica la dirección del motor el instante en que arranca

void main()

{
//Puerto A como salida ya que desde el mismo será donde se controla las señales que van
//hacia los variadores de velocidad de acuerdo a lo que el Pic determine como distancia
//recorrida necesaria y control de giro del motor.
    set_tris_a(0x00);
    set_tris_c(0xFF);


//Se utiliza los siguientes  pines del puerto A, A0, A1, A2, A3 que corresponden a las
//patillas 2, 3, 4 y 5 en el Pic.

    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
    output_bit(pin_a2,0);
    output_bit(pin_a3,0);
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//Con estas cuatro líneas siguientes de código, se manipula los bits del puerto A
//colocándolas en 0 lógico, de forma que se garantiza que ninguno de los variadores
//SVR1 O SVR2 pueda estar encendido por error, ya que con un 0 lógico en estos pines,
//la señal que va hacia los transistores que activan los relés que llevan la señal a los
//variadores, no están activados ya que las bases de los transistores 2n3904 tienen 0
//voltios lo que hace que a la salida, el emisor también tenga 0 Voltios, y por ende, los
//transistores tip31, no activan la bobina que lleva la señal a los variadores.
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//Para identificar la interrupción generada por los pulsos del sensor, se utiliza el modulo
//de captura CCP2 en modo interrupción por flanco ascendente.

setup_ccp2(CCP_CAPTURE_RE);        // Configura CCP2 para capturas en flanco ascendente.
setup_timer_1(T1_INTERNAL);
enable_interrupts(INT_CCP2);   // Habilita la interrupción por captura de CCP2 (PIN RC1)
enable_interrupts(GLOBAL);


//Iniciamos las librerías del Lcd y Teclado, que son las formas de ingresar y verificar los
//datos que ingresan hacia el Pic.

   lcd_init();
   kbd_init();


//Como estamos utilizando interrupciones por el Puerto B que es el de teclado, activamos
//las resistencias de pull-ups internas del pic para el teclado.


   port_b_pullups(TRUE);

   printf(lcd_putc,"   SELECCIONE");
   printf(lcd_putc,"\nMANUAL=*  AUTO=#");

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//Dentro de un bucle DO que se repite continuamente, con la instrucción
//tecla=kbd_getc(); el programa espera que se presione una tecla, una vez que se //presiona
una tecla cualquiera, el programa la identifica con la librería de teclado y por //medio de
funcione IF, compara las teclas para identificar dos en especial, la tecla de * //para manual
y la # para automático, caso contrario, si el programa no identifica esas //teclas, no realiza
ninguna acción y sigue en el bucle DO
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////


   do {
```

```
tecla=kbd_getc();
selec=0;
cont=0;
disable_interrupts(GLOBAL);   // desabilita la interrupción por captura de CCP2 (PIN RC1)

if(tecla==42)
  {
ansii_to_dec();
borra_lcd();
printf(lcd_putc," MODO MANUAL ");
printf(lcd_putc,"\n  ACTIVADO");

      for(i=0;i<10;i++)
        {
      delay_ms(100);
        }
selec=1;
manual();

  }
```

3.4.2. La función manual ()

La función manual() permite que el usuario pueda desplazar la cuchilla hacia arriba con la tecla A y hacia abajo con la tecla B, y la mesa con la combinación de teclas C + A para desplazar a la derecha y la combinación C + B para desplazar a la izquierda, aplicando la combinación mencionada de teclas, se garantiza que los dos motores no se muevan al mismo tiempo, ya que un movimiento como tal, causaría un corte erróneo en el bloque de esponja, provocando que se desperdicie de esa forma la materia prima de los colchones ya que la maquina siempre nos debe dar láminas planas uniformes.

Al entrar a la función manual, se muestra el mensaje MODO MANUAL ACTIVADO, durante 1 segundo, luego del mismo se muestra un mensaje donde se indica las teclas que controlan la cuchilla a través del eje Y, y la combinación de teclas que desplazan la mesa a través del eje X.

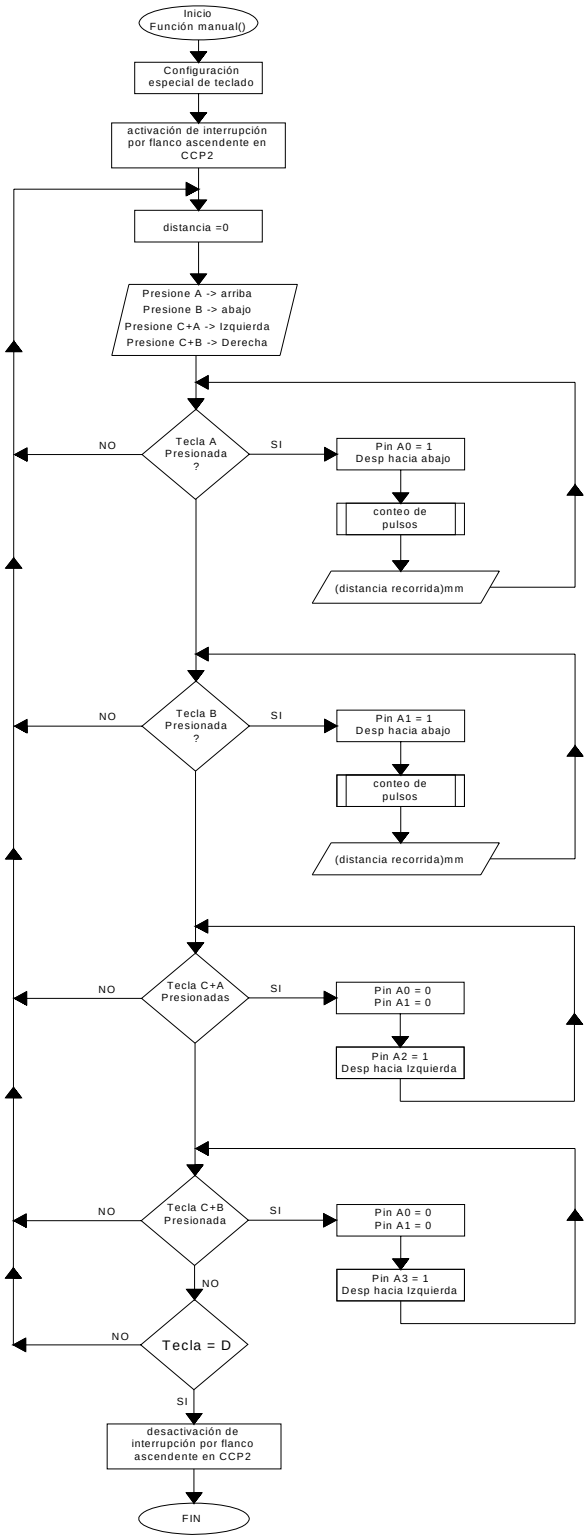


Figura 3.17. Diagrama de flujo de la función manual

```
void manual()
{
   borra_lcd();  printf(lcd_putc,"Pres:A%c      C+A   %c",0b01011110,0b01111110);
   printf(lcd_putc,"\n   Bv  C+B %c",0b01111111);
```

Control manual de desplazamiento en el eje Y.- Todo el sistema manual esta dentro de un bucle del tipo DO, el mismo que se repite mientras la bandera selec se mantenga con 1 como se la declaro al momento de presionar la tecla que ingresaba a la función manual.

Como están activadas las resistencias PULL-UP internas del Pic, se manipula los pines más significativos del puerto B para enviar ya sea un 0 lógico o un 1 lógico, mediante el teclado hacia los pines menos significativos, dentro del bucle DO enviamos un 0 por el pin B4 del pic, y nos fijamos un 1 en el pin B5.

```
   do
      {
   output_bit(pin_b4,0);
   output_bit(pin_b5,1);

//De este modo, se puede escoger entre 4 posibles interrupciones por flanco descendente
//en los 4 pines menos significativos, así al presionar la tecla A, entra al bucle IF que
//identifica un 0 en el pin B3 del pic.
   if(input(pin_b3)==0)
            {
//A la vez, activamos el pin A0 que controla el variador SVR1 en sentido de bajada de la
//cuchilla, y mientras se mantenga presionada.
```

```
output_bit(pin_a0,1);
output_bit(pin_a1,0);
output_bit(pin_a2,0);
output_bit(pin_a3,0);
```

//Se presenta en pantalla el avance progresivo de la distancia de la cuchilla.

```
lcd_gotoxy(1,2);
printf(lcd_putc,"%u",avance);
        }
```

//De igual manera, al presionar la tecla B, enviamos un 0 lógico al pin B2 en el Pic, //mismo que identifica la instrucción, como desplazamiento hacia arriba, para lo cual, //desactivamos el desplazamiento que se estaba realizando hacia abajo y la instrucción //que va al variador SVR1 invierte el giro del motor.

```
if(input(pin_b2)==0)
        {
output_bit(pin_a0,0);
output_bit(pin_a1,1);
output_bit(pin_a2,0);
output_bit(pin_a3,0);
        }
```

Combinación de teclas que controlan el movimiento en el eje X.-Al presionar la tecla C, enviamos un 0 lógico al pin B1, y entramos a una función IF donde a la vez, entramos en un bucle DO que se repetirá mientras se tenga presionada la tecla C.

```
if(input(pin_b1)==0)
        {
```

```
        do{
//Mientras la tecla C permanezca presionada, se envía un 0 lógico adicional por el pin //B5.
    output_bit(pin_b4,0);
    output_bit(pin_b5,0);

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//Ahora la condición para desplazar el bloque de esponja en el eje X, depende de una
//doble condición que se cumple al presionar primero la letra C y a la vez la letra A para
//que un 0 lógico vaya al pin B3, de este modo, el bloque de esponja se desplaza hacia la
//izquierda al enviar un 1 lógico por el pin A2 que envía la señal al SRV2, y al presionar
//la combinación de la tecla C y la tecla B, el molde realizara el movimiento hacia la
//derecha enviando ahora un 1 lógico por el pin A3 hacia el variador SRV2.
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

    if(input(pin_b3)==0)
                        {
    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
    output_bit(pin_a2,1);
    output_bit(pin_a3,0);
                        }

     if(input(pin_b2)==0)
                        {
    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
    output_bit(pin_a2,0);
    output_bit(pin_a3,1);
                        }

    if(input(pin_b3)==1 && input(pin_b2)==1)
                        {
```

```
output_bit(pin_a0,0);
output_bit(pin_a1,0);
output_bit(pin_a2,0);
output_bit(pin_a3,0);
                }

 }while(input(pin_b1)==0);
        }
```

//Al no tener presionada ninguna tecla, los pines B2 y B3, están en 1 lógico, al estar en //esta condición, los pines de control del puerto A, todos se encuentran desactivados con //lo que los variadores mantienen a los motores M2 Y M3 detenidos.

```
if(input(pin_b3)==1 && input(pin_b2)==1)
        {
output_bit(pin_a0,0);
output_bit(pin_a1,0);
output_bit(pin_a2,0);
output_bit(pin_a3,0);
avance=0;
pulsos=0;
lcd_gotoxy(1,2);
printf(lcd_putc,"\n     Bv  C+B %c",0b01111111);
        }
```

//Si se presiona la tecla D, el 0 lógico va hacia el pin B0, la siguiente instrucción //identifica esta instrucción y cambia la bandera de selec de 1 a 0, con lo que //abandonamos el bucle que nos permitía estar en modo manual y nuevamente nos //encontramos en el menú principal.

```
if(input(pin_b0)==0)
        {
    selec=0;
```

```
            }

    }while(selec==1);

    tecla=0;
    borra_lcd();
    printf(lcd_putc,"   SELECCIONE");
    printf(lcd_putc,"\nMANUAL=*  AUTO=#");

}
```

3.4.3. La función automático ()

Dentro del bucle Do de la función Void Main(), para seleccionar la función automática, se debe presionar la tecla #, esta teca devuelve el valor de condigo anssi que es identificado por el Pic como 35 en decimal, al tener la variable tecla el valor de 35, entra dentro de una función IF, dentro de esta función, se muestran dos líneas de mensajes donde se muestra durante 1 segundo "MODO AUTOMATICO ACTIVADO"

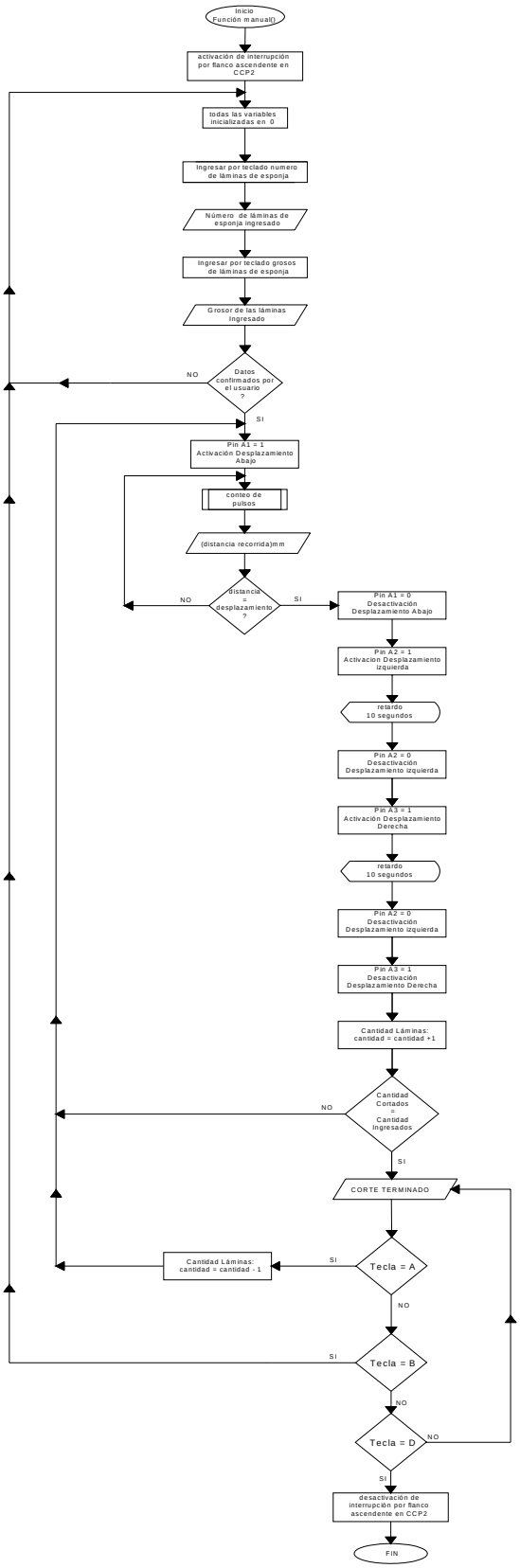


Figura 3.18. Diagrama de flujo de la función automático

```
    if(tecla==35)
      {
    ansii_to_dec();
    borra_lcd();
    printf(lcd_putc,"MODO AUTOMATICO");
    printf(lcd_putc,"\n   ACTIVADO  ");
    for(i=0;i<100;i++)
          {
        delay_ms(10);
          }
```

//La bandera de control selec toma el valor de 2 y se llama a la función automatico().

```
    selec=2;
    automatico();
      }

     }while(true);

}
```

//

//Al hacer el llamado a la función automatico(), esta activa todo tipo de interrupciones //globales, pero prestando principal atención a la interrupción por flanco ascendente que //se va a tomar en el modulo CCP2, luego de activar las interrupciones, declaramos las //variables que vamos a manipular con sus valores en 0 porque estamos iniciando el //conteo de cantidad de colchones a cortarse, así como el espesor que debe tener cada //uno, de igual manera, como el grosor del colchón lo determina la cantidad de pulsos //que genera el sensor colocado para determinar distancia angular de la rotación del eje //principal, este contador de pulsos inicia desde 0 cada vez que se desplaza la cuchilla //para cortar un nuevo colchón.

```
//La variable tecla y la variable dato, también las colocamos en 0  para poder con las
//mismas seguir manipulando la máquina en modo automático y en especial para un
//correcto ingreso de datos sin oportunidad a errores que se pueden producir por datos
//anteriores guardados en estas variables.
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void automatico()
{
     enable_interrupts(GLOBAL);
    distancia=0;
    pulsos=0;
    cont=0;
    tecla=0;
    dato=0;

    do
          {
          selec=3;

//La función cantidad() devuelve a la función automatico(), la cantidad de láminas de
//esponja que se deben cortar.

    cantidad();

//Mientras la función recorre, en su lugar, devuelve la cantidad de pulsos que se deben
//contar con respecto al giro de eje principal, para llegar con precisión milimétrica al
//grosor de lámina de esponja que el usuario desea.

     recorre();
reingreso:
    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
```

```
borra_lcd();
printf(lcd_putc,”Cortando %u lam”,cant);
printf(lcd_putc,”\n%f \t\t\t\tPres A”,pulsos);

//La función automatico() ingresa en una función IF la misma que mantiene en 1 lógico el
//pin A0 del Pic, que es el que controla el giro del motor que desplaza la cuchilla hacia
//abajo, mientras los otros pines se encuentran en 0 lógico evitando activaciones
//innecesarias del variador SVR2 como confusión de instrucciones al variador SVR1, a la
//vez muestra la cantidad en milímetros que la cuchilla está desplazando
if(cant<numero)
        {
pulsos=0;
do{
   output_bit(pin_a0,1);
   output_bit(pin_a1,0);
   output_bit(pin_a2,0);
   output_bit(pin_a3,0);

   lcd_gotoxy(1,1);
   printf(lcd_putc,”Desplazando ....”);
   lcd_gotoxy(1,2);
   printf(lcd_putc,”Autoavance=%f”,pulsos);

  }while(pulsos<distancia);
selec=2;
tecla=0;
```

Control de desplazamiento en el eje X.- Luego de desplazada la cantidad de pulsos de acuerdo al grosor de la lámina de esponja, se llama a la función deshoriz() que controla el desplazamiento del bloque de esponja a través de la cuchilla, tanto un tiempo de ida como el mismo tiempo de regreso, con un inicio lento al ingresar la cuchilla en la esponja, misma

velocidad que está programada en el variador, para evitar que la cuchilla dañe el borde de corte de la lámina mientras inicia el corte.

```
    deshoriz();
//La función deshoriz() desactiva los pines de control del variador SVR1 colocando los
//pines A0 y A1 en 0 lógico, a la vez que desactiva el pin A2 que es el que controla el
//desplazamiento del bloque a través de la cuchilla durante un tiempo programado
//suficiente para que la cuchilla atraviese todo el bloque y desactiva el pin A3 que
//controla el retorno del bloque de esponja
void deshoriz()
{
    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
    output_bit(pin_a2,1);
    output_bit(pin_a3,0);
    delay_ms(1000);

//Al terminar el corte, el bloque vuelve a atravesar la cuchilla pero esta vez solo para
//volver al origen, desactivando el pin A2 y activando el pin A3 durante un tiempo
//programado, igual al tiempo necesario para atravesar la cuchilla con el bloque al ser
//cortado.
    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
    output_bit(pin_a2,0);
    output_bit(pin_a3,1);
    delay_ms(1000);

//Una vez que el bloque fue cortado creando una nueva lámina, desactivamos los pines de
//control de desplazamiento del bloque y regresamos a la función automatico(). A seguir
//con la siguiente instrucción.
```

```
    output_bit(pin_a0,0);
    output_bit(pin_a1,0);
    output_bit(pin_a2,0);
    output_bit(pin_a3,0);

}

//Al retornar a la función automatico() la variable de contador de láminas cortadas cont
//aumenta en 1 y con la instrucción goto reingreso; se vuelve a cortar la siguiente lámina
//hasta cortar la cantidad ingresada


    cant=cant+1;
    goto reingreso;

            }
//Terminado el corte, la función automatico() ingresa en un bloque DO que se repetirá
//mientras no se presione ninguna tecla indicando una nueva instrucción,
    do
            {
        tecla=kbd_getc();
     }while(tecla==0);
```

Laminas adicionales.- El sistema tiene la posibilidad de cortar láminas adicionales a las solicitadas con anterioridad con solo presionar la tecla A, Al presionar esta tecla, el sistema la identifica y dentro de la función IF, resta uno de la cantidad de láminas cortadas, lo que hace que el sistema repita nuevamente un solo corte de forma automática.

```
    if(tecla==65)
            {
```

```
    cant=numero-1;
    goto reingreso;
                }

//Al presionar la tecla D, colocamos la bandera selec en 0 y mostramos en //pantalla
//nuevamente las opciones de MANUAL Y AUTOMATICO, volviendo a la //función
//main() principal.
    if(tecla==68)
                {
    selec=0;
    borra_lcd();
    distancia=0;
    printf(lcd_putc,"   SELECCIONE");
    printf(lcd_putc,"\nMANUAL=*  AUTO=#");
                }

     }while(selec==2);
     tecla=0;

}

//La función cantidad(), inicia con el contador cont de cantidad de láminas en 0, solicita se
//ingrese la cantidad de láminas que se desea cortar, el usuario determina con un mínimo de
//1 lámina y un máximo de  99 láminas.


void cantidad()
{

    cont=0;
    borra_lcd();
    printf(lcd_putc,"Ingrese Cantidad");
    printf(lcd_putc,"\nde Laminas");
```

```
//Dentro de la función DO, se programo funciones IF que verifican que las teclas que se
//estén presionando sean las correctas, es decir del 1 al 9 para determinar cantidad de
//láminas y que en el instante en que se está ingresando los datos, si se presiona una tecla
//incorrecta, saldrá en pantalla un mensaje indicando que la tecla presionada no
//corresponde a una tecla para ingresar un valor correcto.

do{
    tecla=kbd_getc();
    if(tecla>47 && tecla<58 && tecla!=68)
        {
            cont++;
            ansii_to_dec();
            valor=dato;
            if(cont==1){valor1=valor;}
            if(cont==2)
               {
                    valor2=valor;
                    cont=3;
               }

            if(cont==3)
               {
                    valor2=valor;
                    numero=valor1*10+valor2;

    do
                  {
lcd_gotoxy(1,1);
printf(lcd_putc,"Num Laminas=%u",numero);
printf(lcd_putc,"\nA-Acep  B-Reingr");
do{
```

```
tecla=kbd_getc();
}while(tecla==0);
if(tecla==65)
                        {
borra_lcd();
printf(lcd_putc,"VALOR INGRESADO");
printf(lcd_putc,"DAT0=%ulaminas",numero);
delay_ms(1000);
selec=4;
cont=0;
tecla=0;
pulsos=0;
cant=0;
                        }
```

//Si se presiona la tecla B, nos lleva dentro del bucle que permite un reingreso de datos si no se está de acuerdo o se cometió un error al ingresar el dato.

```
    if(tecla==66)
                        {
borra_lcd();
printf(lcd_putc,"INGRESE NUEVO");
printf(lcd_putc,"\nDATO");
delay_ms(1000);
valor1=0;
valor2=0;
valor3=0;
valor=0;
selec=3;
cont=0;
tecla=0;
                        }
```

```
//Al presionar la tecla B, el sistema borra todos los valores anteriormente de teclas
//presionadas y de contador de número de veces que se presiona tales teclas, al igual que
//el valor que este almacenado en la variable tecla.

        if(tecla!=65 || tecla!=66)
                                {
        cont==4;
                                }
        }while(cont==3);


                        }
//Se imprime nuevamente en el LCD los valores de decenas y unidades de esponja que se
//ingreso y se vuelve a la función automatico() que llamara a una nueva subfunción para
//ingresar el valor en cantidad de pulsos por milímetro que se desea comprobar.

    borra_lcd();
    lcd_gotoxy(1,2);
    printf(lcd_putc,"Laminas=%u%u",valor1,valor2);
    tecla=0;
                }

        if(tecla==66  || tecla==67  || tecla==35  || tecla==42)
                {
    borra_lcd();
    printf(lcd_putc,"TECLA INCORRECTA");
    printf(lcd_putc,"\nPres 0-9  A-acep");
    tecla=0;
                }

//Nuevamente, dentro de la función para ingresar valores, con la tecla D que corresponde
//en código anssi al valor de 68, la función IF nos permite salir de la función y regresar al
//menú principal.
```

```
        if(tecla==68)
          {
   selec=0;
   borra_lcd();
   distancia=0;
   printf(lcd_putc,"   SELECCIONE");
   printf(lcd_putc,"\nMANUAL=*  AUTO=#");
   tecla=1;
          }

    }while(selec==3);
}
```

Distancia recorrida.- Similar a la función cantidad, la función recorre inicia con los valores correspondientes al ingreso de datos de grosor de laminas en 0, el grosor de láminas se ingresa en milímetros, siendo la menor de 1 milímetro y la lámina de mayor grosor se la puede medir de hasta 199 milímetros, es decir 19,9 centímetros, ya que a recomendación del fabricante de colchones, no se ingresa valores superiores a los 20 centímetros.

```
void recorre()
{
   valor=0;
   valor1=0;
   valor2=0;
   valor3=0;
   cont=0;
   borra_lcd();
   printf(lcd_putc,"Ingrese distancia");
```

```
 do{
      tecla=kbd_getc();
      if(tecla>47 && tecla<58 && tecla!=68)
           {

//Nuevamente, con cada tecla presionada, el contador cont que determina cuantas
//teclas se ha presionado, aumenta en uno con cada tecla que se presione y el mismo
//nos ayuda a determinar si se está ingresando unidades, decenas o centenas.

cont++;
ansii_to_dec();
valor=dato;
if(cont==1 && tecla==49){valor1=valor;}
if(cont==1 && tecla!=49){valor1=0;}
if(cont==2){valor2=valor;}
if(cont==3)
                {
    valor3=valor;
    cont=4;
                }

      if(cont==4)
                {
valor3=valor;
//La variable distancia2 es la que almacena en forma decimal el valor en milímetros
//del dato que se ingreso como grosor de lámina, y como cada milímetro esta
//subdividido en 30 pulsos generados por el sensor, para saber que hemos recorrido un
//milímetro, usamos la ecuación distancia=distancia2/0.03316, de donde se obtiene el
//dato que se muestra en pantalla ayuda a saber a qué distancia exacta estamos cortando.

distancia2=valor1*100+valor2*10+valor3;
```

```
 distancia=distancia2/0.03316;
      do
                        {
lcd_gotoxy(1,1);
printf(lcd_putc,”Distancia=%u mm”,distancia2);
printf(lcd_putc,”\nA-Acep  B-Reingr”);
do{
tecla=kbd_getc();
}while(tecla==0);
if(tecla==65)
                                {
borra_lcd();
printf(lcd_putc,”VALOR INGRESADO”);
 printf(lcd_putc,”DAT0=%u%u.%u cm”,valor1,valor2,valor3);
delay_ms(1000);
selec=2;
cont=0;
tecla=0;
pulsos=0;
                                }

      if(tecla==66)
                                {
borra_lcd();
printf(lcd_putc,”INGRESE NUEVO”);
printf(lcd_putc,”\nDATO”);
delay_ms(1000);
valor1=0;
valor2=0;
valor3=0;
```

```
valor=0;
selec=4;
cont=0;
tecla=0;
                        }

      if(tecla!=65 || tecla!=66)
                        {
      cont==4;
                        }

      }while(cont==4);

            }
      borra_lcd()
      ;
```

distancia=(valor1*100+valor2*10+valor3)/0.03316; //aqui es donde se determina que la distancia minima es de 30 pulsos por milimetro

```
lcd_gotoxy(1,2);
printf(lcd_putc,"DAT0=%u%u.%u cm",valor1,valor2,valor3);
tecla=0;
        }

      if(tecla==66  || tecla==67  || tecla==35  || tecla==42)
            {
borra_lcd();
printf(lcd_putc,"TECLA INCORRECTA");
printf(lcd_putc,"\nPres 0-9  A-acep");
tecla=0;
            }

      if(tecla==68)
```

```
        {
    selec=0;
    borra_lcd();
    distancia=0;
    printf(lcd_putc,"   SELECCIONE");
    printf(lcd_putc,"\nMANUAL=*  AUTO=#");
    tecla=1;
        }

     }while(selec==4);

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//La función generada por interrupción de flanco ascendente, corresponde a una de las
//más esenciales funciones en la programación del Pic, ya que esta interrupción es donde
//contamos la cantidad de pulsos generados por el sensor del eje principal hacia el Pic,
//que luego compararemos con el valor ingresado por teclado de grosos de lámina.

//Cada vez que se produce un flanco ascendente en la patilla ccp2 del Pic detect una
//interrupción que nos lleva a aumentar el contador de pulsos en 1 y muestra el dato en //la
variable avance que se muestra en pantalla y que resulta de dividir la cantidad de //pulsos
recorridos para la distancia en cantidad de pulsos que representa un milímetro.

//En este caso y como se nombro antes, un milímetro está representado por 30 pulsos //del
sensor del eje principal.
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
#int_ccp2
void isr()
{
    pulsos = pulsos+1;
    avance=pulsos/30;
}
```

3.4.4 Funciones auxiliares

Función para borrar el LCD, enviando el siguiente código al LCD, el mismo reconoce la instrucción como código para eliminar todos los caracteres que se encuentren en pantalla.

```
void borra_lcd()
{
    lcd_putc("\f");
}
```

La siguiente función void ansii_to_dec() nos permite convertir valores hexadecimales de la variable tecla que adquiere del teclado, incluido símbolos, en datos decimales y los almacena en la variable dato.

```
void ansii_to_dec()     //función para cambiar los datos anssi del teclado por su equivalente decimal
{
        if(tecla==48){dato=0;}     //0
        if(tecla==49){dato=1;}     //1
        if(tecla==50){dato=2;}     //2
        if(tecla==51){dato=3;}     //3

        if(tecla==52){dato=4;}     //4
        if(tecla==53){dato=5;}     //5
        if(tecla==54){dato=6;}     //6
        if(tecla==55){dato=7;}     //7

        if(tecla==56){dato=8;}     //8
        if(tecla==57){dato=9;}     //9
```

```
        if(tecla==65){dato=10;}    //A
        if(tecla==66){dato=11;}    //B

        if(tecla==67){dato=12;}    //C
        if(tecla==68){dato=13;}    //D
        if(tecla==42){dato=14;}    //*
        if(tecla==35){dato=15;}    //#
}
```

3.4.5. Monitoreo (PC)

El monitoreo se basa en un HMI realizado en LAVIEW que se detalla a continuación.

El panel de control se muestra en la figura 3.19, el panel dispone de indicación del proceso de corte de la esponja, que consta de un indicador para verificar la posición de la cuchilla en su respectivo eje vertical y el desplazamiento del bloque a través de la esponja, el programa muestra también la cantidad de láminas que se está cortando.

Incluye una alarma que se activa si se detecta un desfase en el desplazamiento vertical o un deslizamiento horizontal fuera de margen, además de un panel de activación y desactivación del sistema.
En el panel se muestran luces indicadoras que se activan en sincronismo con el proceso que esté realizando la máquina

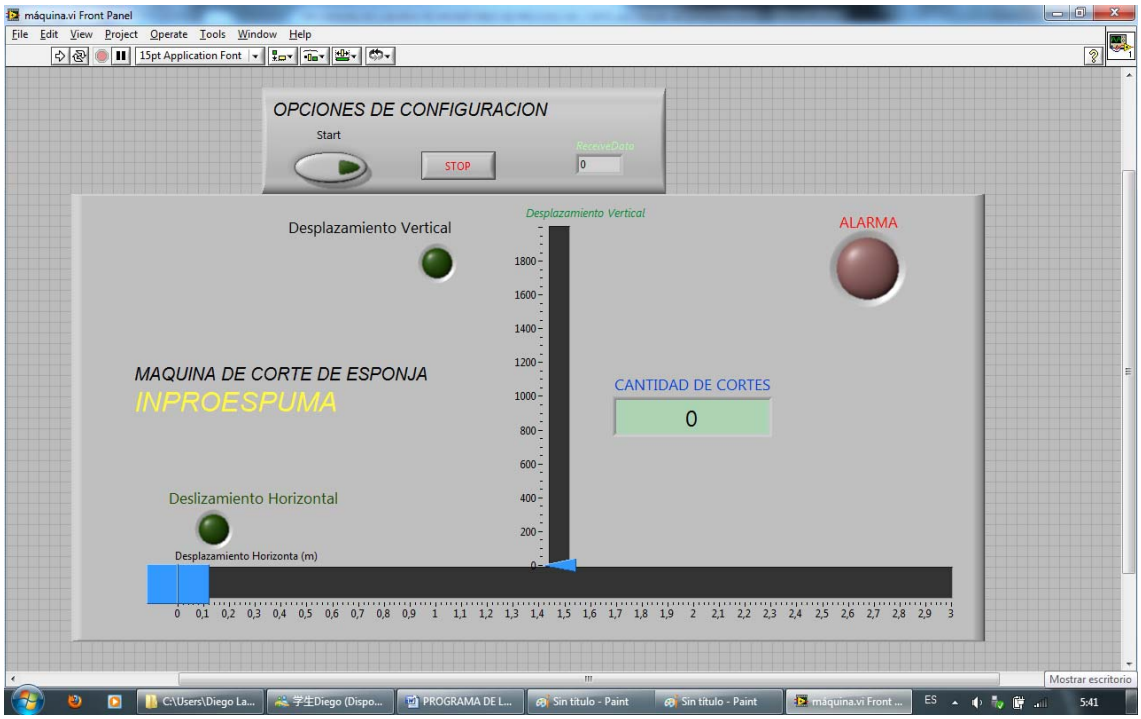


Figura 3.19 Programación en LabView

El envío de datos de la Pc al uC, requiere de los siguientes procesos.

El programa de envío de datos de la figura 3.20, se lo hace a través del bloque comando USB WRITE, el mismo que envía al uC una serie de líneas de comando que permiten la configuración e identificación del pic con la PC y en especial con el programa de monitoreo de LabView.

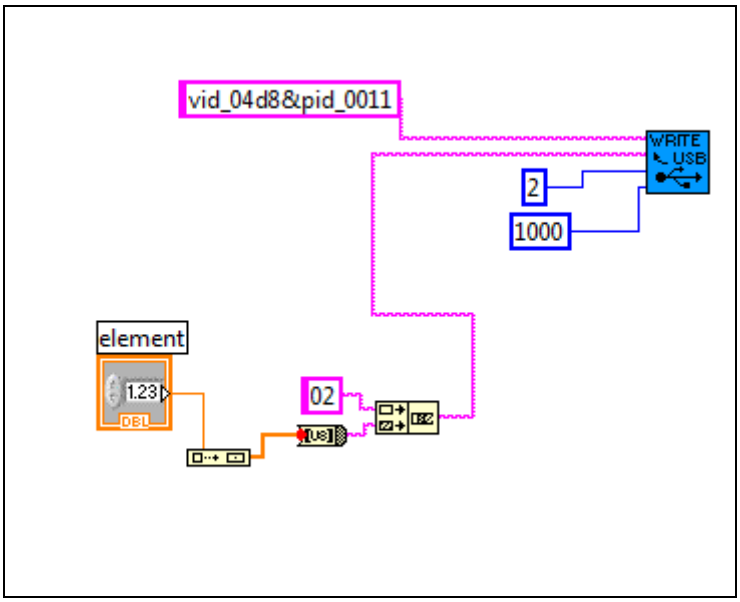


Figura 3.20.a. Envío de datos

Para el envío de datos, el programa creado en LabView, consta de una matriz de datos tipo carácter, estos datos son los primeros en transferirse del programa de la PC al pic, e indican al pic el código con el que se relaciona el programa, en nuestro caso el código es: vid_04d8&pid_0011 en donde vid_04d8 indica que el programa tiene relación directa con un pic de la familia Microchip y el segundo código pid_0011 es un valor único que identifica en este caso a nuestra tarjeta para que el envío de datos no cree confusiones con otros elementos USB que tenga la PC que realiza el monitoreo.

Luego de enviar este código de identificación de la tarjeta, el PC envía otra matriz de datos tipo carácter para comprobar que la conexión es correcta y para sincronizar el uC con la PC, esta vez previo al envío de datos, se creó una matriz tiene dos valores, estos valores pueden ser modificados de acuerdo a la necesidad del programa, principalmente, este valor enviado es 01, el uC adquiere este dato, lo procesa e identifica que viene de la PC, inmediatamente entra en sincronismo.

Las siguientes líneas de código muestran la programación del uC para una configuración de sincronismo entre el uC y la PC.

```
#define USB_HID_DEVICE     FALSE            //deshabilitamos el uso de las directivas HID
#define USB_EP1_TX_ENABLE  USB_ENABLE_BULK   //turn on EP1(EndPoint1) puntero de la matriz de transmición de datos.
#define USB_EP1_RX_ENABLE  USB_ENABLE_BULK   //turn on EP1(EndPoint1) puntero de la matriz de recepción de datos.
#define USB_EP1_TX_SIZE    4                 //matriz de transmisión de 4 datos a la PC
#define USB_EP1_RX_SIZE    2                 //matriz de recepción de 2 datos de la PC

#include <pic18_usb.h>     //Microchip PIC18Fxx5x Hardware librería que llama a las funciones especificas del uso de la transmición USB
#include <PicUSB.h>      //Configuración del USB y los descriptores para este dispositivo
#include <usb.c>                 //funciones especiales de la librería de pics ubs
```

Luego de la configuración del uC, para realizar la captura de datos que envía el PC, se utiliza

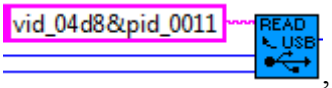
```
usb_get_packet(1, dato_de_pc, 2);  // tomamos el paquete de tamaño 2 bytes de la PC y almacenamos en dato_de_pc
```

Para realizar la adquisición de datos del uC, a la PC, el uC utiliza las siguientes líneas para realizar el envío de datos.

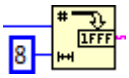
```
usb_put_packet (1, dato_a_pc, 4, USB_DTS_TOGGLE);
```

Donde queda especificado que se está enviando un solo paquete de datos, es decir, una matriz, y con el 4, se indica que la matriz está formada de 4 elementos, es importante nombrar que en la intercomunicación de estos datos, se está procesando datos de tipo carácter y los mismos son de 8bits o 1 byte.

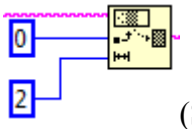
El programa realizado en lenguaje de bloques del software LabView, se muestra en la figura 3.21, el programa adquiere los datos enviados por el uC vía USB 2.0, estos datos son

capturados en la PC con el bloque comando USB READ 

, los valores son analizados y procesados para mostrar el proceso que realiza la máquina.

El comando 

(number to hexadecimal string) convierte los datos que el pic envía, en una matriz de 8 números tipo hexadecimal, es decir, divide cada byte en 2 grupos de 4 bits para formar números hexadecimales.

Seguido por la instrucción 

(String subset) que identifica un puntero de matriz y la cantidad de números a capturarse, de este modo, se selecciona grupos pares de la matriz de 8 números hexadecimales y se captura 2 números hexadecimales seguidos, que son los mismos que el uC envía pero transformados de cadena de caracteres en cadenas de un byte a la vez, se lo identifica una vez separado.

El comando 

(Hexadecimal String to number) nos permite volver a convertir el valor hexadecimal enviado por el uC y capturado en el programa, en un valor decimal que se muestra luego en el panel de control principal como dato de posición de la cuchilla, desplazamiento del bloque de esponja, cantidad de láminas cortadas o falla en el sistema, ya que para analizar estos 4 datos, se necesita utilizar la matriz de 4 bytes que el uC envía.

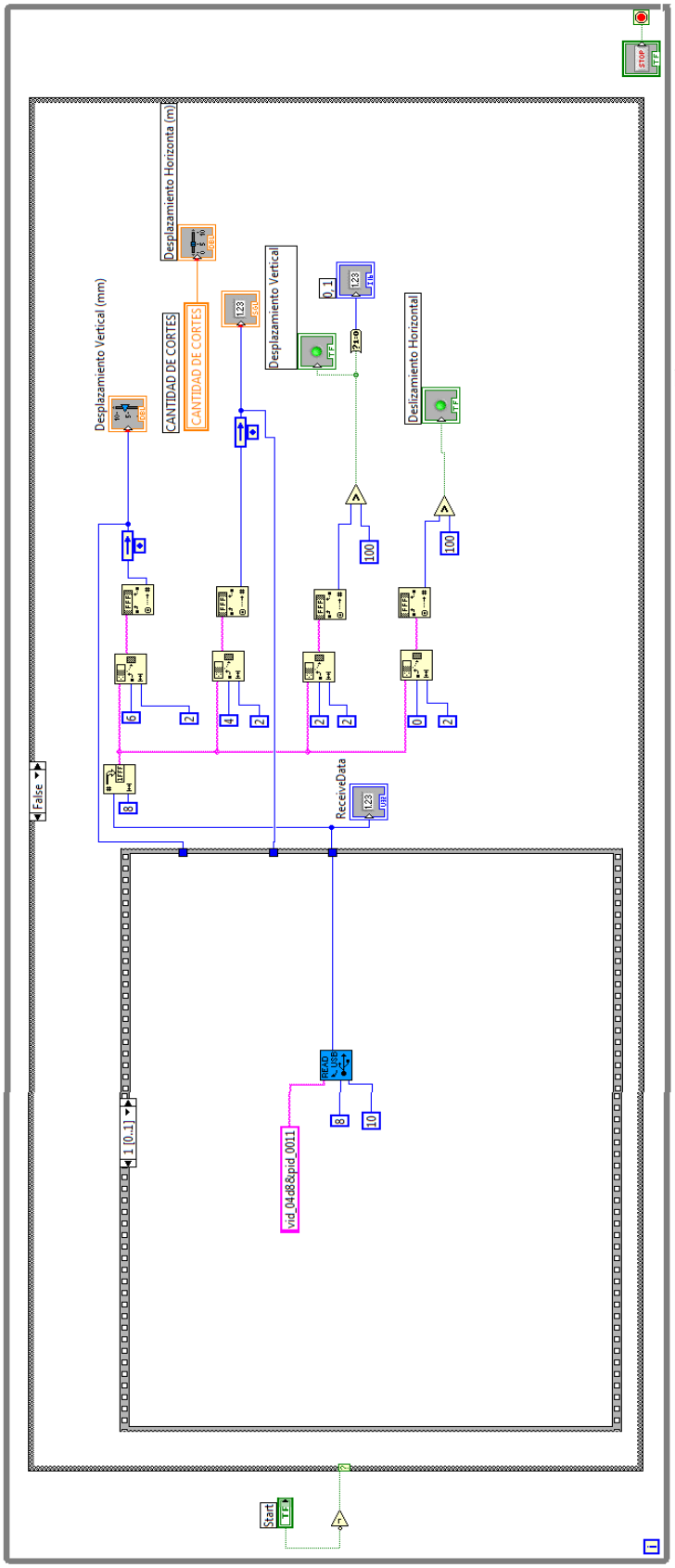


Figura 3.20.b. Adquisición de datos

3.5. INSTALACIONES ELÉCTRICAS Y MONTAJE

3.5.1. Descripción del armario de control y automatización

La gama de armarios de control del tipo SR representa una oferta completa de envolventes monobloque diseñados para construir equipos de conmutación eléctricos pequeños y medios, para: automatización, funcionamiento, control y distribución, cumpliendo con todas las necesidades de nuestro sistema, adecuado para la creación de tableros de control eléctricos tipo pared o suelo, con la posibilidad de ajustar la posición de la placa de montaje interna.

El armario modular fue diseñado con las siguientes dimensiones, 100cm x 20cm x 70cm (ancho, alto y profundidad) donde se permite la construcción de equipos de conmutación de distribución secundarios gracias a la posibilidad de instalar toda la gama de aparatos e interruptores automáticos modulares en rieles DIN o dentro de ellos, combinados con paneles frontales modulares especiales previamente perforados y articulados del tipo doble fondo, permitiendo así la construcción de equipos de conmutación eléctricos del tipo AS/ANS, cumpliendo las normas EN 60439-1 (CEI 17/13-1), donde se especifica que fue chapas de acero de 2.5mm espesor en la puerta y 2mm de espesor en el techo y paredes, pintado a color gris perla anticorrosivo de calidad ISO 9001 como se muestra su diseño en la figura 3.21.

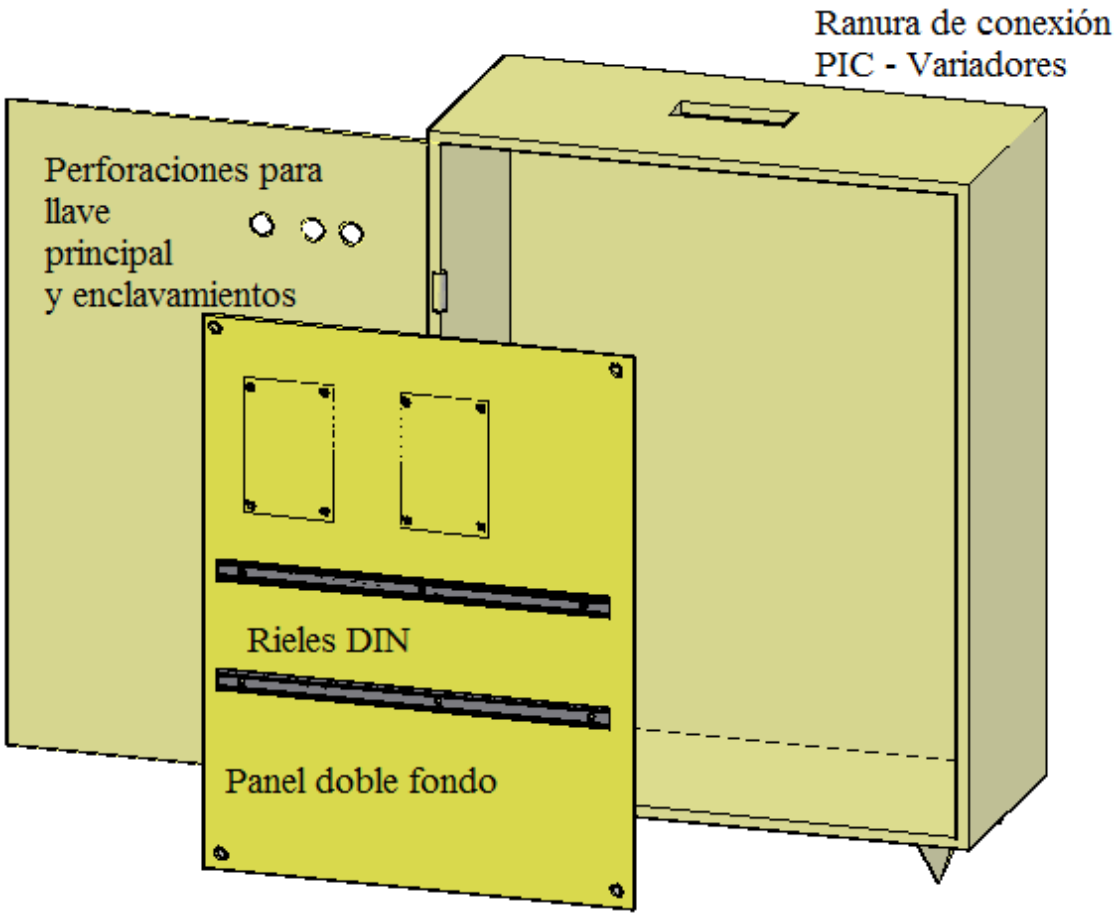


Figura 3.21. Armario de control

Al armario se le acopló un panel de control superior como se muestra en la figura 3.22 donde va colocada la tarjeta diseñada con el microcontrolador a la vez que contiene la pantalla LCD y el teclado para que el operador pueda acceder al control de la máquina.

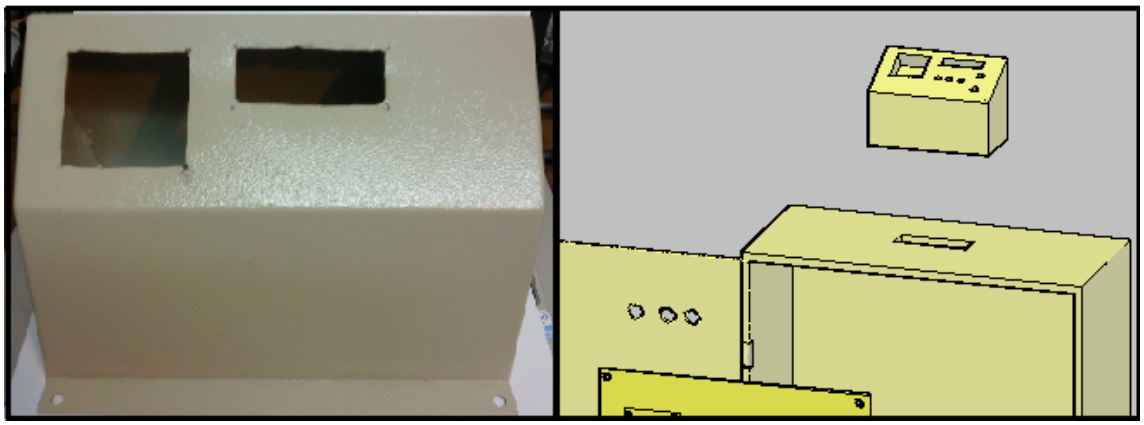


Figura 3.22 Panel de control

El armario una vez armado con todos los elementos, se colocó contra la pared cercana a la máquina para evitar que el operador innecesariamente entre en contacto con alguna parte mecánica de la máquina, el armario queda cerrado con un seguro de llave triangular y a la puerta se le colocó un empaque de caucho para sellarla y así evitar el ingreso de partículas de esponja que pudiesen poner en riesgo el funcionamiento normal del nuevo equipo, con el panel y el armario pegados a la pared, el operador puede manipular la máquina a una distancia previa como se aprecia en la figura 3.23.

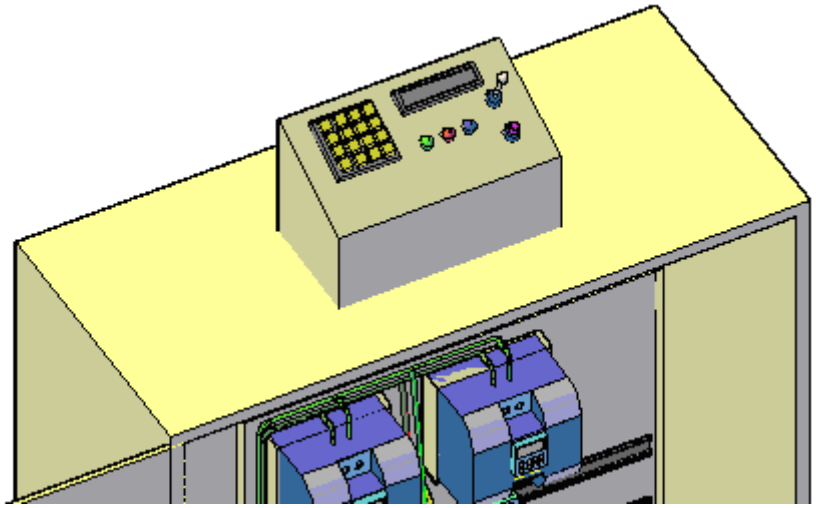


Figura 3.23. Panel y armario de control

3.5.2. Diseño del circuito impreso de la tarjeta principal

Aunque cada caso requiere un tratamiento especial y cada Empresa tendrá sus propias normas, se deben de tener en cuenta unas reglas básicas que podrían considerarse comunes y que nos ayudan a una correcta funcionalidad de la tarjeta.

Se diseñó sobre una hoja cuadriculada, de modo que se hiso coincidir las pistas con las líneas de la cuadrícula o formando un ángulo de 45º con éstas, y los puntos de soldadura con las intersecciones de las líneas (Fig. 3.24.).

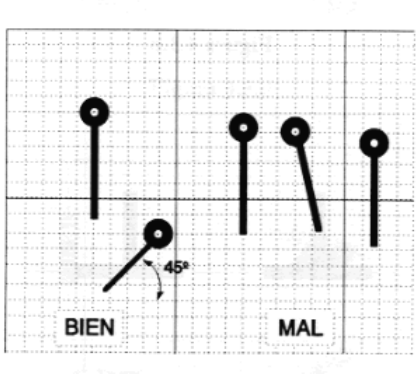


Figura 3.24. Líneas de cobre sobre la placa

Se trató de realizar un diseño lo más sencillo posible; cuanto más cortas sean las pistas y más simple la distribución de componentes, mejor resultará el diseño.

No se realizó pistas con ángulos de 90º; cuando sea preciso efectuar un giro en una pista, se hiso con dos ángulos de 135º (Fig. 3.25)

Los puntos de soldadura son círculos cuyo diámetro es al menos el doble del ancho de la pista que en él termina.

El ancho de las pistas depende de la intensidad de corriente que va a circular por ellas. Se tuvo en cuenta un ancho de 0,8 mm para 2 amperios; 2 mm para 5 amperios; y 4,5 mm para 10 amperios. En general, se realizaron pistas de 2 mm.

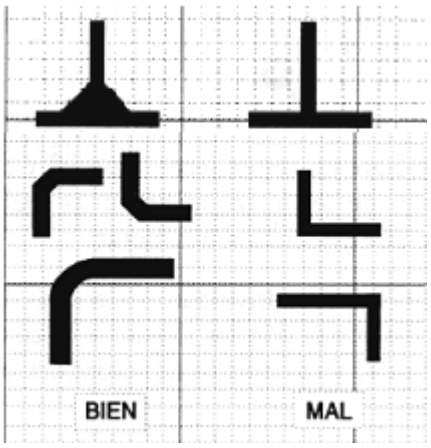


Figura 3.25. Curvas y bifurcaciones entre pistas del circuito

Entre pistas próximas y entre puntos de soldadura, se observa una distancia que dependerá de la tensión eléctrica que se prevea existirá entre ellas; como norma general, se deja una distancia mínima de 0,8 mm.; en casos de diseños complejos, se puede disminuir los 0,8 mm hasta 0,4 mm.

La distancia mínima entre pistas y los bordes de la placa es de dos décimas de pulgada, aproximadamente unos 5 mm, donde todos los componentes se colocaron paralelos a los bordes de la placa, además de no colocar pistas entre los bordes de la placa y los puntos de soldadura de terminales de entrada, salida o alimentación, exceptuando la pista de masa, se cuidó de que no se pase pistas entre dos terminales de componentes activos (transistores, tiristores, etc.). donde al final se obtuvo una placa como se muestra en la figura 3.26.

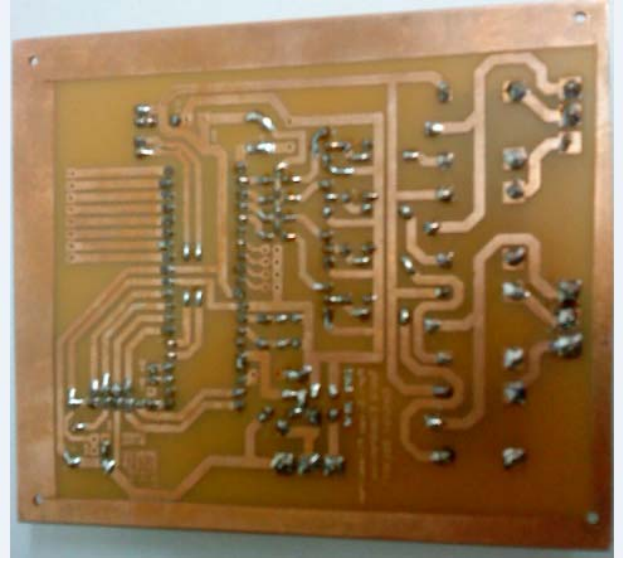


Figura 3.26 Placa finalizada del circuito

Se debe prever la sujeción de la placa a un chasis o caja; para ello se dispone de los tornillos colocados en el panel de control para que la placa vaya sujeta y de la misma salgan los conductores hacia el teclado y la pantalla LCD.

Como norma general, se debe dejar, una o dos décimas de pulgada de patilla entre el cuerpo de los componentes y el punto de soldadura correspondiente (Fig. 3.27).

Figura 3.27. Interior del tablero de control y tarjeta principal a colocarse

3.5.3. Montaje de los variadores sinamics G110

La hoja técnica de los variadores, muestra especificaciones de montaje, razón por la cual, el tablero se lo adecúa para cumplir con esta normativa, conservando una distancia mínima de 30 mm entre los variadores y una distancia mínima de 15mm entre: los variadores, la pared y techo del armario, además de que fueron sujetados con pernos de aluminio de ¼" pulgada contra el doble fondo del armario, y a la vez este doble fondo está correctamente aterrizado para disipar las sobrecargas que se pueden generar en el variador, la figura 3.28. muestra la correcta colocación y el espacio suficiente de separación entre los variadores para ayudar a disipar el calor. Además de las líneas de alimentación correspondientes entrantes a cada variador, como se aprecia en la figura 3.29. hacia los variadores se ingresan dos de las tres fases de 110Vac, de donde salen nuevamente tres fases las cuales alimentan y controlan los motores.

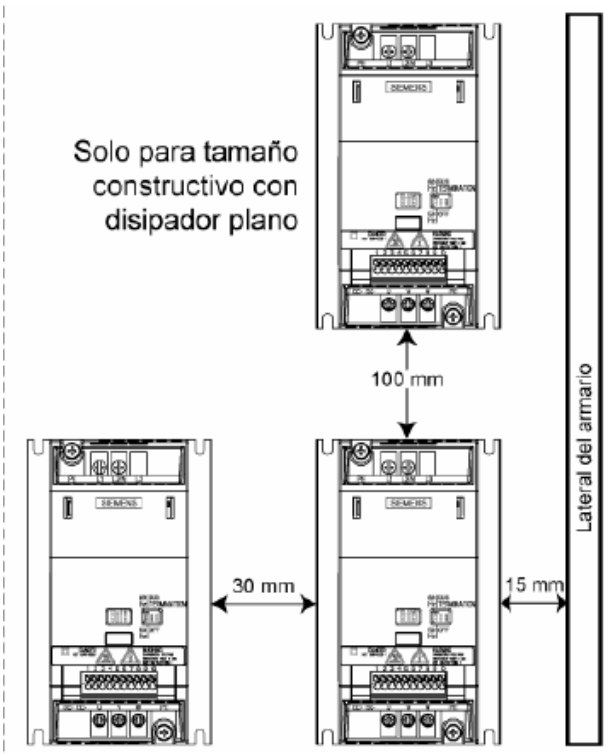


Figura 3.28. Montaje de los variadores Sinamics G110 sobre el panel

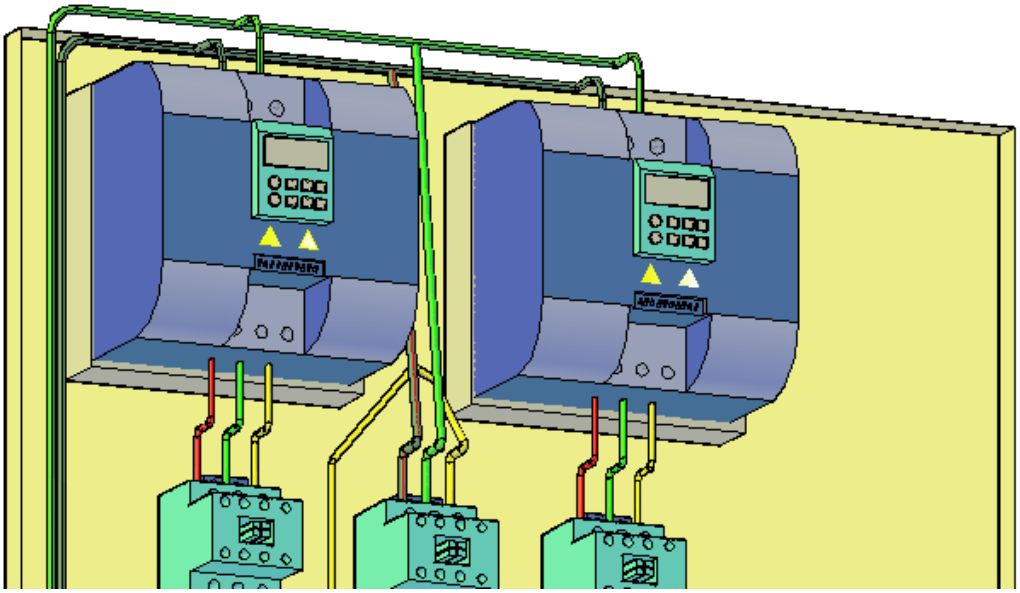


Figura 3.29. Líneas de alimentación de entrada a los variadores y salida a los motores

3.5.4. Montaje de los contactores

Para la colocación de contactores y disyuntores como circuitos de protección, se instaló dos rieles DIN de 80cm cada una, dejando una separación entre las mismas que permita una correcta instalación del cableado, y cumpliendo lo diseñado en el diagrama de control se colocó a la salida de cada uno de los variadores, el correspondiente contactor que permite el paso de la corriente de alimentación de los motores y a la vez entre el cada contactor y motor, se colocó su respectivo disyuntor como se muestra en la figura 3.30.

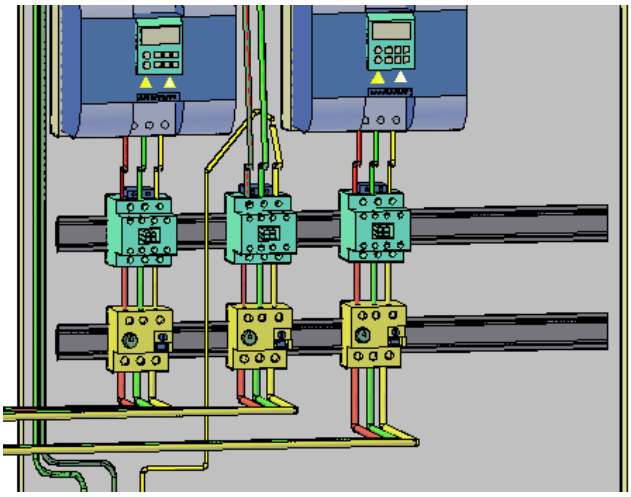


Figura 3.30. Contactores y disyuntores

El panel de doble fondo desmontable, debe ser removido para colocar todos los equipos que intervienen en la automatización, una vez colocados los elementos sobre la riel DIN y los variadores sobre los espacios designados sobre el panel, este panel debe volver a ser colocado dentro del armario y asegurado con sus respectivos tornillos, luego se coloca el panel de control con la tarjeta madre en la parte superior y de igual manera se sujeta con sus correspondientes tornillos hacia la parte superior del techo del armario de modo que todo el equipo de automatización quedara montado como se aprecia en la figura 3.31.

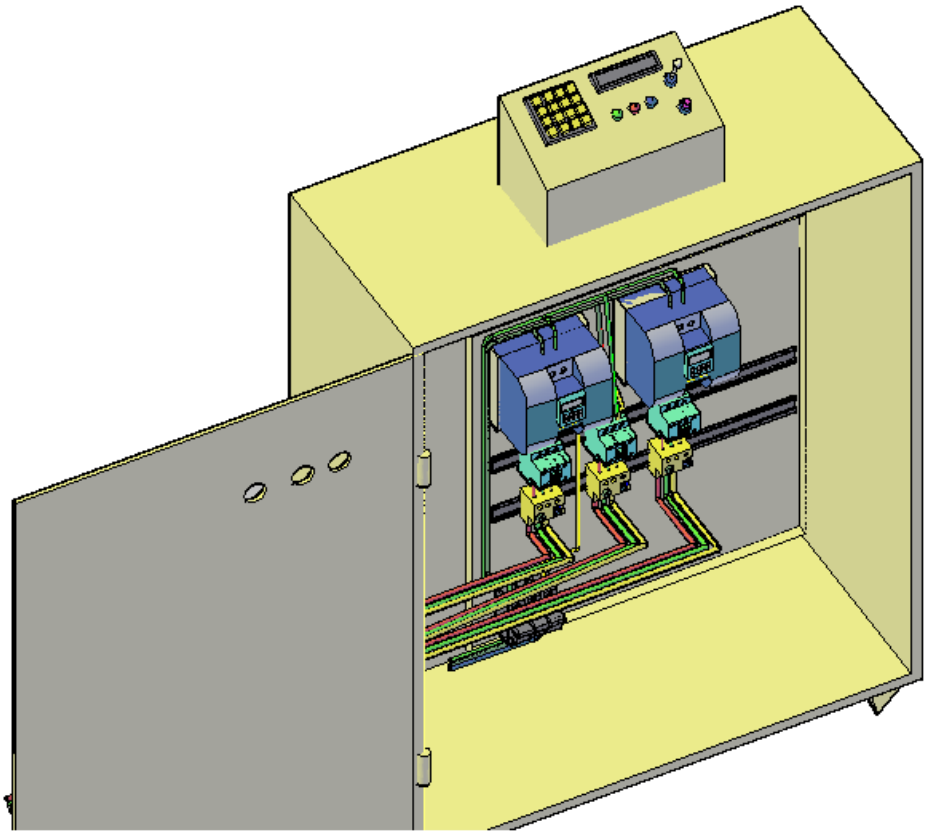


Figura 3.31. Armario con diseño de equipamiento completo

Figura 3.32. Armario y panel con equipamiento instalado junto a la máquina

3.5.5. Conexión de motores

La conexión de los motores M1, M2 y M3 se realizó directamente de las salidas de los disyuntores, el motor debe estar conectado como lo muestra la placa para el voltaje que se le aplica, en este caso, 3 fases de 110Vac, 60Hz a lo que la placa del motor indica en la figura 3.33. se lo debe conectar en modo delta.

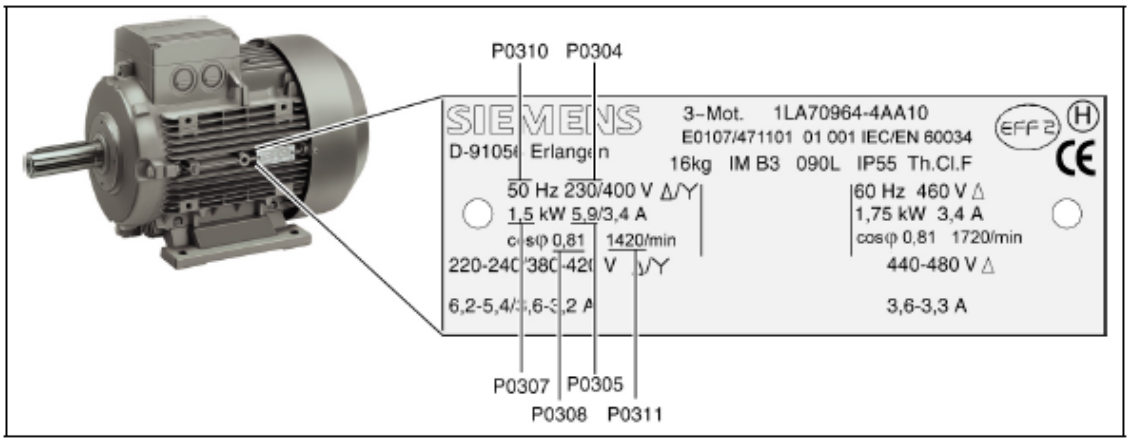
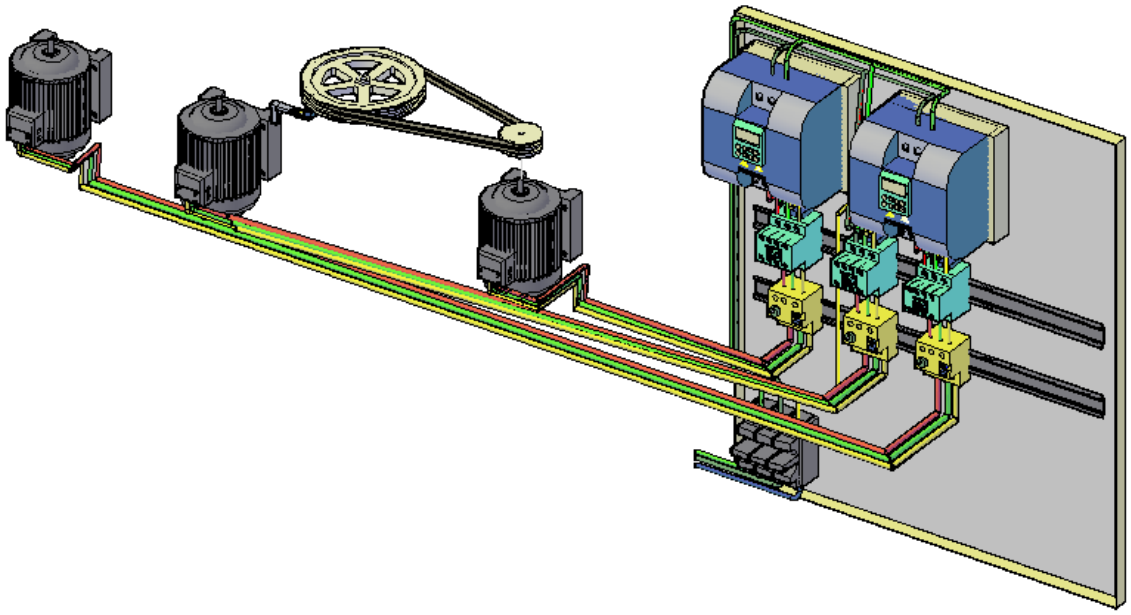


Figura 3.33. Datos del motor M1

Y de igual forma, los motores restantes M2 y M3, que comparten las mismas características, su conexión es delta aunque la misma esta echa por dentro de la tapa de conexiones del motor, misma que durante el proceso de automatización ha sido comprobado su eficiencia, mas no se ha manipulado sus conexiones, lo que finalmente nos deja con un esquema de conexión de los motores como lo muestra la figura 3.34.



3.34. Sistema automático implementado a los motores.

3.5.6. Acople del sensor óptico ZT

El acople del sensor óptico ZT se colocó por debajo del eje principal que traslada el movimiento del motor M2 hacia el perno sin fin (figura 3.35.), este sensor óptico posee un sócalo ya que el mismo tiene un tiempo de vida útil como todo equipo electrónico, o bien puede llegar a llenarse de polvo o esponja, razón por la cual se lo diseñó con la opción de que sea desmontable y se lo pueda cambiar por uno nuevo, se garantiza su funcionalidad a largo plazo ya que el mismo está diseñado e implementado bajo ecuaciones de cálculos de voltaje de alimentación y salida como se explicaba en el subtema anterior, además, el mismo está sujeto con un tornillo que permite regular su altura para hacer más efectiva la calibración de la señal de luz emitida hacia la distancia del eje principal para no perder pulsos.

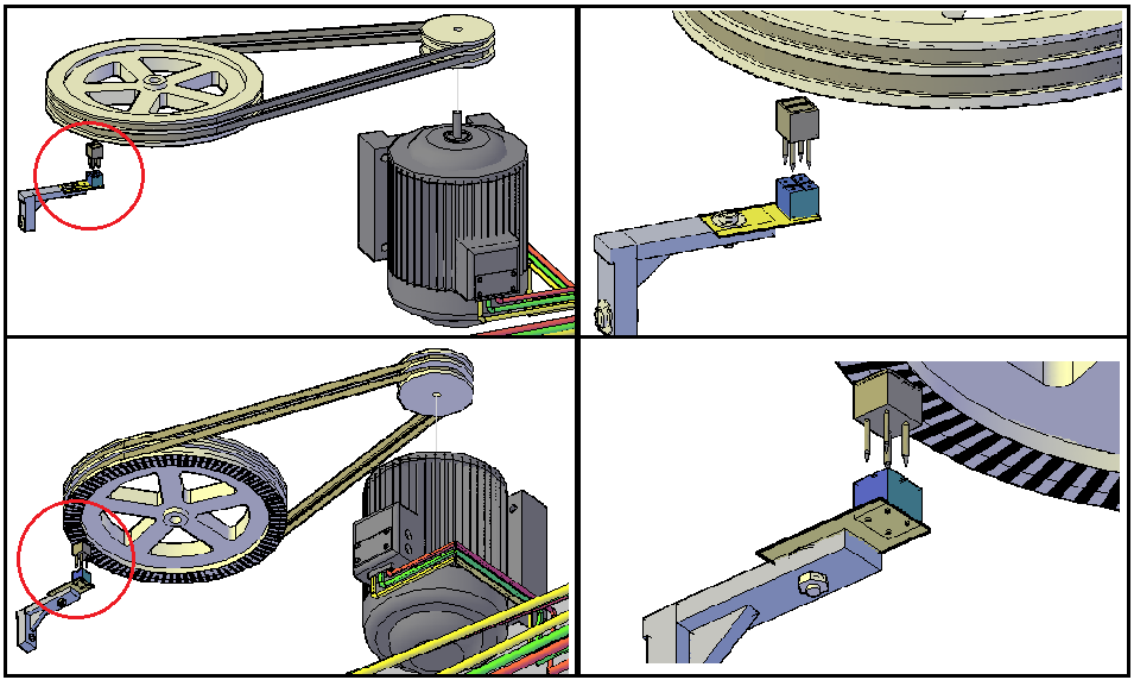


Figura 3.35. Acople del sensor al sistema

3.5.7. Sensores de posición (finales de carrera)

Los aquí nombrados sensores de posición ZR1 y ZR2, son interruptores de tipo final de carrera colocados sobre la mecánica vertical de la máquina de corte, estos sensores son normalmente cerrados (NC) el voltaje que enclava el contactor K2 pasa a través de estos, y al desplazarse la cuchilla fuera del rango vertical de trabajo, topa uno de estos sensores, los mismos que en ese instante pasan de N.C. a abiertos, momento en el cual, el voltaje que mantenía enclavado el contactor K2 deja de circular hacia la bobina y el contactor se desenclava, al desenclavarse el contactor K2, ya no pasa más voltaje del variador SRV1 hacia el motor M2 y la cuchilla deja de desplazarse. La figura 3.36. muestra la posición del sensor ZR1, el sensor ZR2 está colocado de igual manera en la parte inferior límite de la cuchilla.

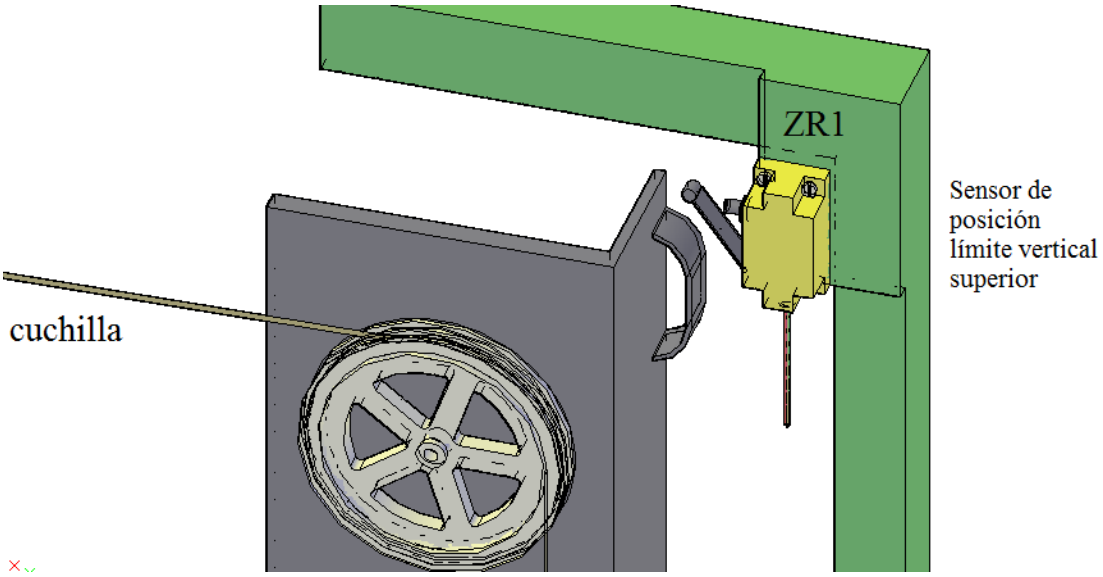


Figura 3.36. Sensor de posición vertical ZR1

Nuevamente nombrando los sensores de posición, esta vez ZR3 y ZR4, que también son interruptores de tipo final de carrera colocados debajo de la mesa que desplaza el bloque de esponja máquina a través de la cuchilla, estos sensores son normalmente cerrados (NC) el voltaje que enclava el contactor K3 pasa a través de estos, y al desplazarse la mesa mas allá de los limites, topa uno de estos sensores, los mismos que en ese instante pasan de N.C. a abiertos, en ese instante el voltaje que mantenía enclavado el contactor K3 deja de circular hacia la bobina y el contactor se desenclava y un contacto auxilia envía una señal hacia el PIC deteniendo el proceso por fallo, al desenclavarse el contactor K3, ya no pasa más voltaje del variador SRV2 hacia el motor M3 y la mesa con el bloque deja de desplazarse. La figura 3.37. muestra la posición del sensor ZR4, el sensor ZR3 está colocado de igual manera en la parte inferior límite de desplazamiento horizontal de la mesa.

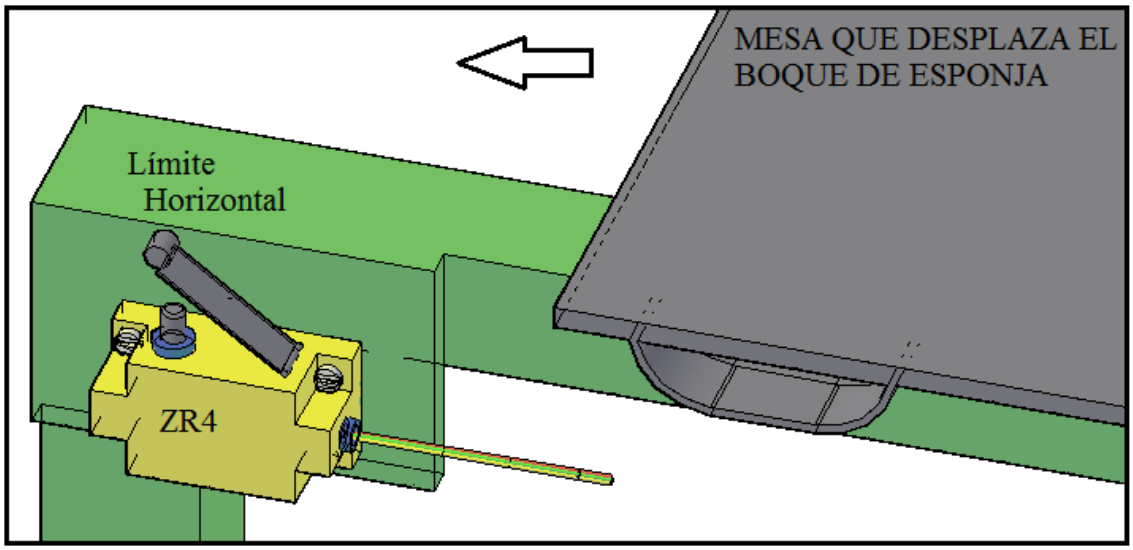


Figura 3.37. Sensor de posición vertical ZR1