ESCUELA POLITÉCNICA DEL EJÉRCITO

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA

ESTUDIO DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR (HARDWARE Y SOFTWARE) PARA TRES GRADOS DE LIBERTAD DEL MANIPULADOR ROBÓTICO CRS-A255

DANIEL A. MALDONADO QUINTEROS
ANDREA E. SÁNCHEZ ANDRADE

SANGOLQUÍ - ECUADOR

2013

Declaración de Responsabilidad

ESCUELA POLITÉCNICA DEL EJÉRCITO
INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

DECLARACIÓN DE RESPONSABILIDAD

ANDREA ELIZABETH SÁNCHEZ ANDRADE

DANIEL ALBERTO MALDONADO QUINTEROS

DECLARAMOS QUE:

El proyecto de grado denominado “ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR (HARDWARE Y SOFTWARE) PARA TRES GRADOS DE LIBERTAD DEL MANIPULADOR ROBÓTICO CRS A255”, ha sido desarrollado con base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie, de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de nuestra autoría.

En virtud de esta declaración, nos responsabilizamos del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, 21 de febrero de 2013

_____ _____

Andrea Elizabeth Sánchez Andrade Daniel Alberto Maldonado Quinteros

Autorización de publicación

ESCUELA POLITÉCNICA DEL EJÉRCITO
INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORIZACIÓN

Nosotros, Andrea Elizabeth Sánchez Andrade y Daniel Alberto Maldonado Quinteros

Autorizamos a la Escuela Politécnica del Ejército la publicación, en la biblioteca virtual de la Institución del trabajo “ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR (HARDWARE Y SOFTWARE) PARA TRES GRADOS DE LIBERTAD DEL MANIPULADOR ROBÓTICO CRS A255”, cuyo contenido, ideas y criterios son de nuestra exclusiva responsabilidad y autoría.

Sangolquí, 21 de febrero de 2013

_____ _____

Andrea Elizabeth Sánchez Andrade Daniel Alberto Maldonado Quinteros

Certificado de tutoría

ESCUELA POLITÉCNICA DEL EJÉRCITO
INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICADO

Ing. Alexander Ibarra

Ing. Edgar Tipán

CERTIFICAN

Que el trabajo titulado "ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR (HARDWARE Y SOFTWARE) PARA TRES GRADOS DE LIBERTAD DEL MANIPULADOR ROBÓTICO CRS A255", realizado por Andrea Elizabeth Sánchez Andrade y Daniel Alberto Maldonado Quinteros, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la ESPE, en el Reglamento de Estudiantes de la Escuela Politécnica del Ejército.

Debido a que se trata de un trabajo de investigación recomiendan su publicación. El mencionado trabajo consta de un documento empastado y un disco compacto el cual contiene los archivos en formato portátil de Acrobat (pdf). Autorizan a Andrea Elizabeth Sánchez Andrade y a Daniel Maldonado Quinteros que lo entregue al Ingeniero Víctor Proaño, en su calidad de Coordinador de la Carrera.

Sangolquí, 21 de febrero de 2013

Ing. Alexander Ibarra DIRECTOR	Ing. Edgar Tipán CODIRECTOR

ESCUELA POLITÉCNICA DEL EJÉRCITO

INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICADO

Ing. Alexander Ibarra

Ing. Edgar Tipán

CERTIFICAN

Que el proyecto de grado titulado “ESTUDIO DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR (HARDWARE Y SOFTWARE) PARA TRES GRADOS DE LIBERTAD DEL MANIPULADOR ROBÓTICO CRS-A255” ha sido desarrollado en su totalidad por la Srta. Andrea Elizabeth Sánchez Andrade y el Sr. Daniel Alberto Maldonado Quinteros.

Sangolquí, 21 de febrero de 2013

_____	_____
Ing. Alexander Ibarra DIRECTOR	Ing. Edgar Tipán CODIRECTOR

AGRADECIMIENTO

Quiero agradecer a Dios, porque gracias a él estoy aquí y voy dejando mi huella en el mundo. A mis padres, quienes siempre se han preocupado por mí y me han brindado su apoyo incondicional, sin ellos no sería posible culminar esta etapa en mi vida, ya que han sabido darme ánimos para no desmayar y seguir adelante.

A mis hermanos, mi cuñada y mis sobrinos por compartir conmigo todos los momentos, ya sean buenos o malos, por saberme disculpar en muchas ocasiones que no pude compartir con ellos por motivos de estudio, quiero que sepan que los amo mucho.

A Daniel, mi compañero incondicional quien siempre me ha brindado su amor, consejos, paciencia.

A mis amigas Gabi y Majo, muchas gracias por su apoyo y por su amistad.

Al Ing. Alexander Ibarra e Ing. Edgar Tipán por su tiempo, orientación y consejo en este proyecto.

A Karel Espinoza, quien nos ayudó incondicionalmente durante este proyecto, muchas gracias por compartir sus conocimientos con nosotros.

Andrea

AGRADECIMIENTO

En primer lugar quiero agradecer a Dios quien me ha guiado por este largo camino, quien ha puesto personas en mi vida que me han hecho ver las cosas de la mejor manera.

A mis padres, hermano y familia por darme siempre su apoyo, tiempo, cariño, que son las cosas primordiales en mi vida.

A Andrea, por compartir conmigo este logro, y ser mi amiga y compañera incondicional en todo momento.

A mis profesores, que han compartido sus conocimientos y experiencias para que logremos ser lo que deseamos ser.

A nuestro amigo Karel, por su inmensa colaboración en este proyecto.

Por útlimo, quiero agradecer a todas las personas que han hecho posible culminar esta etapa en mi camino con recuerdos increíbles que guardaré en mi corazón.

Daniel

DEDICATORIA

Este proyecto fruto de mi esfuerzo se lo dedico a mis padres porque gracias ellos he logrado culminar una etapa más de mi vida. A mis hermanos, mi cuñada por brindarme su apoyo. A mis sobrinos por ser las personitas que ponen esa chispa de alegría en mi vida. A Daniel por ser la persona maravillosa que es, por sus consejos, por siempre ser incondicional para mí. A toda mi familia, por estar presente en mi vida y brindarme sus consejos, por todo el cariño que siempre me han manifestado.

Andrea

DEDICATORIA

Este logro en mi vida se lo quiero dedicar en primer lugar a mis padres, quienes con su amor, guía, apoyo incondicional y palabras de aliento me han permitido llegar a donde me encuentro hoy.

A mi hermano, quien ha formado parte de cada una de las etapas de mi vida y ha sido un amigo en quien confiar.

A mi novia Andre, quien me ha brindado su cariño y apoyo en todo mis proyectos.

A mis abuelos, primos, tíos y demás miembros de la familia que tanto quiero y admiro.

A mis amigos, con quienes hemos compartido muchos momentos que hoy forman parte de mi memoria y vivirán ahí por siempre.

Daniel

PRÓLOGO

Considerando que los controladores C500 que actualmente se encuentran en la Escuela Politécnica del Ejército (ESPE) se encuentran en decadencia, se consideró implementar un controlador (*Hardware y Software*) para tres grados de libertad del manipulador robótico CRS-A255 para que de esta manera los estudiantes de Ingeniería Electrónica en Automatización y Control tengan acceso a trabajar con el manipulador robótico.

Antes de proceder al diseño del controlador, se realizó un análisis del controlador C500 existente así como del manipulador robótico en donde se obtuvieron parámetros que sirvieron de base para el diseño del nuevo controlador. Como características del sistema a implementar se tiene un manipulador robótico que cuenta con cinco grados de libertad, con motores DC como actuadores, con frenos en todas las articulaciones con excepción de la primera y con *encoders* incrementales para conocer su posición. Un controlador que consta de tres etapas para su funcionamiento: alimentación, control y potencia. Además se tiene dos formas de manejo: mediante un *teach pendant* ó por computador.

Para el diseño del sistema del controlador se consideró seguir conservando la estructura por etapas, es decir, se realizó una etapa de alimentación la misma que será la encargada de suministrar el voltaje y corriente necesaria al sistema. La etapa de potencia fue diseñada para proporcionar la energía necesaria para accionar los motores y frenos de las articulaciones. La etapa de control es la encargada de gestionar las acciones necesarias para efectuar el control sobre el manipulador robótico; de igual manera existen dos formas de mover el brazo robótico, con el *teach pendant* y

mediante el computador, para este último se realizó una interfaz en un *software* abierto. Cabe recalcar que este proyecto es la fase inicial de un proyecto que abarca varias etapas, por lo que los diseños tanto en *hardware* como en *software* son escalables.

ÍNDICE DE CONTENIDOS

CAPÍTULO I

INTRODUCCIÓN 14

1.1 ANTECEDENTES 14

1.2 JUSTIFICACIÓN E IMPORTANCIA 15

1.3 ALCANCE 16

1.4 OBJETIVOS 16

1.4.1 Objetivo General 16

1.4.2 Objetivo Específicos 17

CAPÍTULO II

CARACTERÍSTICAS DEL SISTEMA A IMPLEMENTAR 18

2.1 CARACTERÍSTICAS DEL MANIPULADOR ROBÓTICO CRS A255 18

2.1.1 Volumen de Trabajo 20

2.1.2 Descripción Mecánica 21

2.1.3 Sistemas de Accionamiento y Transmisión 22

2.1.4 Descripción de los conectores 26

2.2 ESTUDIO Y ANÁLISIS DEL TIPO DE CONTROLADOR A IMPLEMENTAR 34

2.2.1 Estudio y Análisis del Controlador C500 38

2.2.2 Estudio y Análisis del Controlador a implementar 44

2.3 ESTUDIO Y ANÁLISIS DEL *SOFTWARE* A DESARROLLAR 47

CAPÍTULO III

DISEÑO DEL SISTEMA DEL CONTROLADOR 49

3.1 DISEÑO DEL *HARDWARE* 49

3.1.1 Diseño de la Etapa de Potencia 49

3.1.2 Diseño de la Etapa de Control 60
3.1.3 Diseño de la Etapa de Alimentación 98
3.2 DISEÑO DEL *SOFTWARE* 101
3.2.1 Comunicación Computador - Controlador 101
3.2.2 Programa desarrollado en NETBEANS 7.2 108

CAPÍTULO IV

IMPLEMENTACIÓN DEL SISTEMA DEL CONTROLADOR 111
4.1 IMPLEMENTACIÓN DEL *HARDWARE* 111
4.1.1 Consideraciones para el diseño de circuitos impresos 111
4.1.2 Tarjeta de Control 114
4.1.3 Tarjeta de Potencia 116
4.1.4 Etapa de Alimentación 120
4.1.5 Estructura externa del controlador (Caja) 121
4.1.6 Teach Pendant 123

CAPÍTULO V

PRUEBAS Y RESULTADOS 123
5.1 PRUEBAS 123

CAPÍTULO VI

CONCLUSIONES Y RECOMENDACIONES 127
6.1 CONCLUSIONES 127
6.2 RECOMENDACIONES 128

CAPITULO I

INTRODUCCIÓN

1.1. ANTECEDENTES

Durante los últimos años la robótica ha ido incrementando su aplicación a nivel industrial debido a la complejidad de los procesos en las fábricas, ya sea por su precisión, fuerza o para liberar a las personas de tareas que pueden ser peligrosas o agotadoras. Los manipuladores robóticos industriales, específicamente en la configuración revoluta, se caracterizan por la facilidad de realizar trayectorias complejas y accesibilidad a zonas con obstáculos, por lo que son utilizados en la industria en tareas de pintura, soldadura, entre otras.

En el laboratorio de Robótica del Departamento de Eléctrica y Electrónica de la ESPE se encuentran tres manipuladores robóticos CRS A255, los cuales han sido una herramienta muy útil en el aprendizaje y capacitación de los alumnos de la Escuela, ya que estos como robots de entrenamiento, brindan la posibilidad de conocer y entender los principios de funcionamiento y programación de un robot industrial.

Debido al tiempo de vida de los manipuladores robóticos se han ido presentando inconvenientes en los controladores, quedando así, inhabilitados dos de los tres robots en mención, y consecuentemente se ve afectado el aprendizaje de los estudiantes del Departamento ya que las clases prácticas no se pueden llevar a cabo con la frecuencia apropiada.

1.2. JUSTIFICACIÓN E IMPORTANCIA

La realización del estudio, diseño e implementación de un controlador para el manipulador robótico CRS A255 permitirá al Departamento de Eléctrica y Electrónica obtener un sistema abierto tanto en hardware como en software, ya que actualmente la parte física de los controladores existentes está en decadencia porque los dispositivos constitutivos del mismo se encuentran descontinuados, razón por la cual ya no existen repuestos que permitan volver a tener operativos a los robots en caso de averías. Al mismo tiempo, no se dispone del código fuente del software de las memorias que ocupan dichos controladores, por lo que actualmente se está extrayendo este código de las memorias en mención como hexadecimal, pero no se conoce lo que estos caracteres alfa-numéricos representan en línea de código. El presente proyecto proporcionará información detallada sobre todos los aspectos (hardware y software) y sentará bases para próximos proyectos de investigación en el ámbito del desarrollo del software del controlador para aplicaciones de la robótica como posicionamiento, generación de trayectorias, control inteligente, etc.

Desde el punto de vista educativo permitirá a los estudiantes tener un laboratorio con prototipos funcionales en donde se puedan realizar prácticas de la asignatura de robótica con la frecuencia apropiada, ya que para la realización de clases prácticas solo se dispone de un Manipulador Robótico CRS-A255 funcional y dos inactivos, por la falta de repuestos antes mencionada.

1.3. ALCANCE

El presente proyecto de tesis abarca el estudio, diseño e implementación de hardware y software de un controlador para tres grados de libertad del manipulador robótico CRS-A255, para lo cual se verificará el buen estado de la parte mecánica del manipulador robótico, incluyendo los sensores y actuadores de los tres ejes que se desea operar.

Se realizará el acondicionamiento de señales provenientes de los sensores del manipulador robótico, así como el diseño de la etapa de potencia para el control de los motores de las articulaciones del manipulador, en el cual se incluye un *"teach pendant"* que permita manipular el robot sin necesidad de un computador.

Adicionalmente el proyecto incluye la programación del microcontrolador encargado del movimiento del robot, así como también la creación de un programa en software libre que permitirá el control del robot desde un computador.

1.4. OBJETIVOS

1.4.1. Objetivo General

- Realizar el estudio, diseño e implementación de un controlador (*hardware* y *software*) para tres grados de libertad del manipulador robótico CRS-A255.

1.4.2. Objetivos Específicos

- Verificar el funcionamiento correcto de la mecánica, sensores y actuadores del manipulador robótico.

- Diseñar e implementar los circuitos de acondicionamiento de señales de los sensores y los circuitos de potencia necesarios para el movimiento del robot.

- Crear un programa en *software* libre que permita la manipulación del robot desde un computador.

- Verificar el adecuado funcionamiento, tanto del *software* y *hardware*, del controlador elaborado para tres grados de libertad del manipulador robótico CRS-A255.

CAPITULO II

CARACTERÍSTICAS DEL SISTEMA A IMPLEMENTAR

2.1. CARACTERÍSTICAS DEL MANIPULADOR ROBÓTICO CRS A255

La constitución física del manipulador robótico tiene una gran similitud con la anatomía de las extremidades superiores del cuerpo humano, es por esto que para hacer referencia a las partes del robot se utilizan términos como cadera, hombro, etc.

El manipulador CRS A255 es un robot de configuración Angular o Antropomórfica que posee 5 grados de libertad: cadera, hombro, codo, muñeca y su rotador como se muestra en la Figura 2.1. Consta de una placa de montaje que asegura el brazo sobre una plataforma fija.

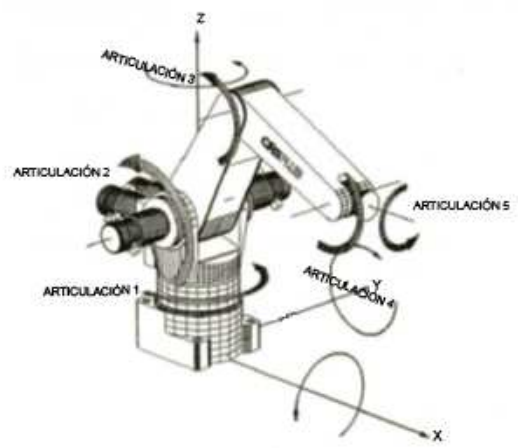

Figura 2.1 Descripción de Articulaciones del Manipulador CRS A255

En este robot se pueden colocar diversos efectores finales como por ejemplo: pinzas, dispensadores, etc. Para el presente proyecto el efector final es un *gripper* neumático. En la Tabla 2.1 se detallan las características del manipulador:

Tabla 2.1 Especificaciones físicas y ambientales del manipulador robótico CRS A255 [1]

Especificaciones Físicas	
Grados de libertad	5
Peso del brazo	19 Kg
Montaje	Vertical o invertida
Carga útil nominal	1 Kg
Alcance	660 mm (1 eje común de los dedos del *gripper*)
Repetibilidad	± 0.05 mm
Sistema de Accionamiento	Motores DC electromecánicos *Encoders* incrementales en cada articulación
Transmisión	*Drives* de armónicos
Frenos	Frenos en todas las articulaciones, excepto la Articulación 1
Modos de movimiento	Dirigido / Automático
Conexiones al final del brazo	Servo *gripper* o conector de aire en la muñeca Conector DB-9 en la muñeca

[1] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide*

Especificaciones Ambientales	
Temperatura	10^{o} a 40^{o}C [50 a 104 F]
Humedad	Mantener bajo el 80% de humedad
Vibración	No clasificado para vibraciones o golpes excesivos
Interferencia Electromagnética	No exponerlo a ruido eléctrico excesivo o plasma

2.1.1. Volumen de Trabajo

El volumen de trabajo hace referencia al espacio que puede ser barrido por el extremo de la muñeca sin tomar en cuenta el efector final, debido a que los efectores finales pueden ser de diferentes tamaños y formas. En las figuras 2.2 y 2.3 se puede apreciar el volumen de trabajo del robot.

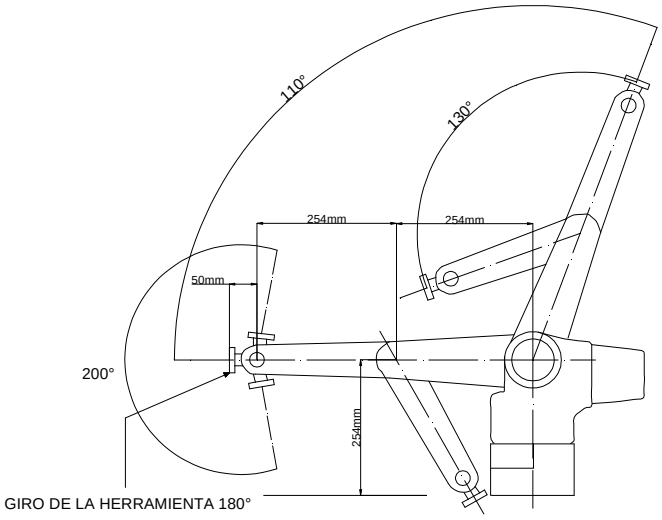

Figura 2.2 Vista lateral del volumen de trabajo del manipulador robótico CRS A255

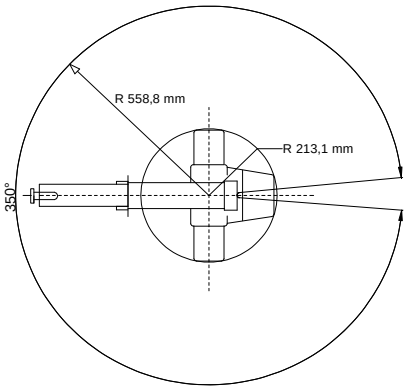

Figura 2.3 Vista superior del volumen de trabajo del manipulador robótico CRS A255

Con el conocimiento de este volumen de trabajo, se puede limitar el acceso y crear espacios seguros para los operarios, o estudiantes que manipulen el robot.

2.1.2. Descripción Mecánica

El brazo robótico está constituido por materiales que hacen liviana su estructura, dando como beneficios, la capacidad de moverse rápidamente y con gran precisión. En la Tabla 2.2 se detalla las características de movimiento y velocidad de las articulaciones del brazo.

Tabla 2.2 Descripción de movimiento y velocidad de las articulaciones del robot [2]

Articulación	Eje	Rango de Movimiento	Velocidad máxima	Aceleración por defecto
1	Cadera	$\pm 175^{o}$	$210^{o}/s$	$500^{o}/s^{2}$
2	Hombro	0^{o} a 110^{o}	$210^{o}/s$	$500^{o}/s^{2}$
3	Codo	-130^{o} a 0^{o}	$210^{o}/s$	$500^{o}/s^{2}$
4	Muñeca	$\pm 110^{o}$	$551^{o}/s$	$1836^{o}/s^{2}$

[2] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide*

5	Rotador	$\pm 180^{o}$	$1102^{o}/s$	$3673^{o}/s^{2}$

2.1.3. Sistemas de Accionamiento y Transmisión

- **Motores**

Para el movimiento de cada articulación existe un motor DC, cuyas características se indican en la Tabla 2.3.

Tabla 2.3 Descripción del Motor DC [3]

	Descripción
Tipo	Motor DC de imán permanente
Rodamientos	ABEC clase 1 – 0.375" ID
Voltaje Máximo	± 35 V
Corriente Máxima	2 A
Máxima velocidad a 35V	3600 rpm
Par máximo	100 oz – in

Los datos que se pueden observar en las placas de los motores son los que se indican a continuación en la Tabla 2.4

[3] García Saquicela, M. R. (2010). *DISEÑO DEL SISTEMA DE CONTROL DEL BRAZO ROBÓTICO CRS A255 UTILIZANDO LA PLATAFORMA KINETIX DE ALLEN BRADLEY.* Escuela Politécnica del Ejército, Quito.

Tabla 2.4 Motores de las articulaciones 2, 3, 4 y 5

	Motor Art. 2	Motor Art. 3	Motor Art. 4	Motor Art. 5
	PM Field Servo Motor	PM Field Servo Motor	PM Field Servo Motor	PM Field Servo Motor
Modelo	ME2110-057E 21-22950	ME2110-057E 21-22950	ME2110-057E 49-22733	ME2110-057E 49-22733
Código	9426	9426	-	-
Serie	1073944	1073935	1083817	1083778
Cust.	R-MTR-ML2110	R-MTR-ML2110	R-MTR-ML2110	R-MTR-ML2110

Debido a la construcción del robot, sólo se puede observar físicamente cuatro de los cinco motores, correspondientes a las articulaciones 2,3,4 y 5 (hombro, codo, muñeca y rotador), que se pueden apreciar en la Figura 2.4.

Figura 2.4 Ubicación de los motores de las articulaciones 2, 3, 4 y 5

- **Encoder**

Cada motor cuenta con un *encoder* de tipo incremental, es decir, que solo

puede realizar cuentas ascendentes o descendentes a partir de la posición actual de la articulación, por lo que siempre al encender el robot será necesario un proceso de inicialización manual, estos sensores nos permiten conocer la posición del manipulador robótico dentro del volumen de trabajo.

Los datos que se pueden observar en las placas de los *encoders* son los que se indican a continuación en la Tabla 2.5

Tabla 2.5 *Encoders* de las articulaciones 2, 3, 4 y 5

	Encoder Art. 2	*Encoder* Art. 3	*Encoder* Art. 4	*Encoder* Art. 5
Modelo	D1DM-1000-SL37AS	D1DM-1000-SL37AS	1DM-1000-SL37AS	1DM-1000-SL37AS
Código	9427	9427	9437	9437
Serie	1075192	-	-	-

Los *encoders* se alimentan con 5 V y tienen una resolución de 1000 pulsos por revolución con tres canales de salida (A,B y Z) que entregan una señal TTL como la que se muestra en la Figura 2.5. Siendo el canal A y B desfasados en una porción de ciclo, mientras que el canal Z indica cada vuelta que da el motor.

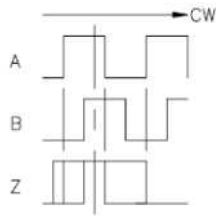

Figura 2.5 Forma de onda de la salida del *encoder* en rotación horaria

- **Frenos**

Con el objetivo de evitar que el brazo robótico caiga por efectos de la

gravedad cuando éste se encuentra desenergizado tiene frenos de falla en las articulaciones. La única articulación que no tiene un freno de este tipo es la cadera (Articulación 1).

Para permitir el movimiento de las articulaciones que disponen de este freno se debe energizar un solenoide magnético que descarga la abrazadera de sujeción, la misma que se activa con una señal de 35 VDC.

- **Transmisiones y Reducciones**

Las transmisiones dentro del brazo robótico son las encargadas de trasladar el movimiento desde los motores hasta las articulaciones. En el caso particular del brazo robótico se tiene cadenas, cuya principal ventaja es la transmisión de gran torque.

En el motor de todas las articulaciones se tienen reducciones para disminuir la velocidad e incrementar el torque. Para las articulaciones que soportan mayor peso, es decir, las articulaciones 1, 2 y 3 se tienen reducciones de tipo cicloidales, mientras que las articulaciones 4 y 5 tienen reducciones por engranajes. Las relaciones de reducción se detallan a continuación en la Tabla 2.6

Tabla 2.6 Relación de reducción de engranajes [4]

Articulación	Relación de engranajes
1	72:1
2	72:1
3	72:1
4	19.6:1
5	9.8:1

2.1.4. Descripción de los Conectores

Tan importante como las características físicas del robot, son las conexiones tanto eléctricas como neumáticas (en el caso de que la herramienta final sea un *gripper* neumático) que se tienen en el brazo robótico. En la Figura 2.6 se puede apreciar la distribución de dichas conexiones, necesarias para el funcionamiento del robot, tales como:

- Conector de Alimentación de Potencia de los motores
- Conector de Realimentación de las señales de los sensores, colocados en los motores
- Puerto de Entrada de Aire
- Punto de conexión a tierra (GND)

[4] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

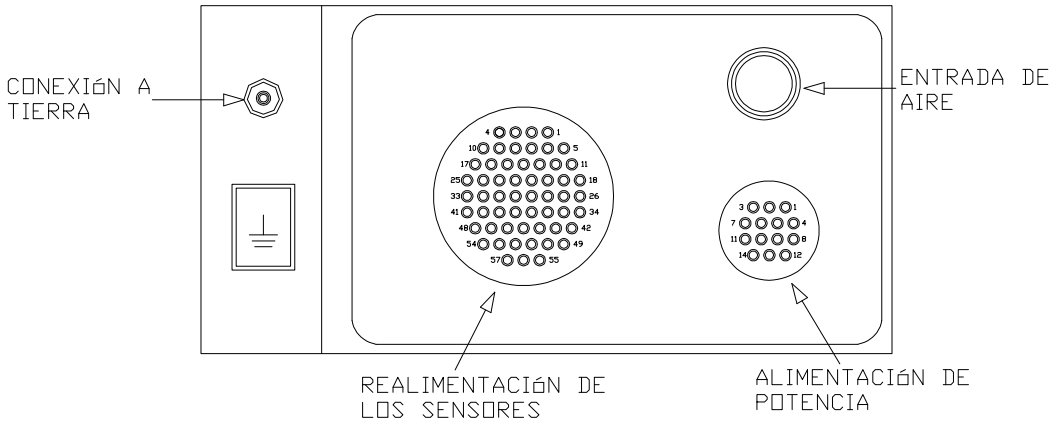

Figura 2.6 Disposición de conectores en la parte posterior del brazo robótico

Para conectar el brazo robótico con el controlador existen dos cables umbilicales, los cuales son los encargados de transmitir tanto las señales de potencia, necesarias para accionar los motores; como las señales que emiten los *encoders.*

Se debe aclarar que estas conexiones son las que se tienen de fábrica, necesarias para la comunicación con el controlador C500, las cuales van a ser utilizadas en el nuevo controlador, según la necesidad que se tenga de las mismas.

- **Conector de Alimentación de Potencia para los motores**

Este es un conector tipo hembra de 14 puntos de conexión, de los cuales están utilizados 10 puntos, conectados a cada uno de los motores encargados del movimiento, como se indica en la Figura 2.7

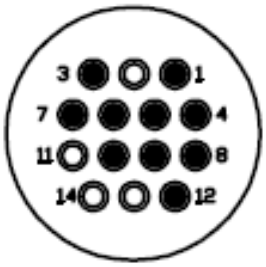

Figura 2.7 Conector para Alimentación de potencia de los motores [5]

El cable umbilical encargado de llevar la alimentación de potencia a cada uno de los motores, tiene un conector tipo macho para conectarse con el brazo robótico y un conector tipo hembra para conectarse con el controlador como se muestra en la Figura 2.8

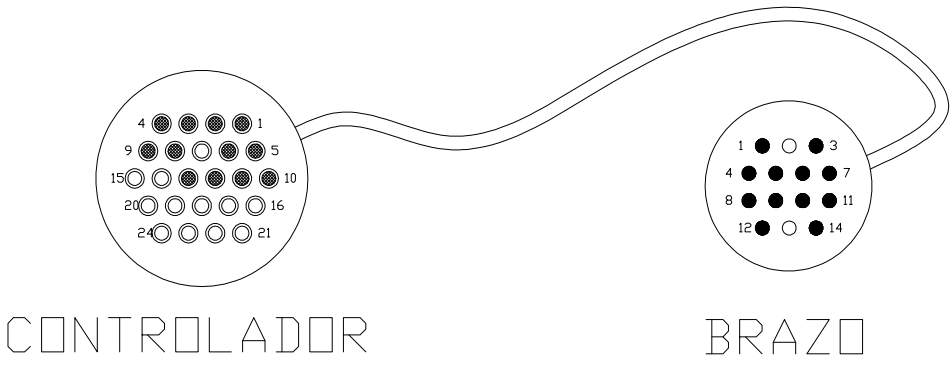

Figura 2.8 Cable umbilical de Alimentación de Potencia de los motores

La distribución de cada extremo del cable se detalla a continuación en la Tabla 2.7.

[5] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

Tabla 2.7 Detalle de conectores del cable umbilical de alimentación de potencia de los motores [6]

Detalle del Cable de Potencia			
# Pin en Brazo	# Pin en Controlador	Señal	Descripción
1	1	J1_MtrPwr+	Articulación 1. Alimentación del motor +/- 30-35 VDC a 2A max
4	2	J1_MtrPwr-	Articulación 1. Alimentación del motor (referencia)
3	3	J2_MtrPwr+	Articulación 2. Alimentación del motor +/- 30-35 VDC a 2A max
7	4	J2_MtrPwr-	Articulación 2. Alimentación del motor (referencia)
5	5	J3_MtrPwr+	Articulación 3. Alimentación del motor +/- 30-35 VDC a 2A max
9	6	J3_MtrPwr-	Articulación 3. Alimentación del motor (referencia)
6	8	J4_MtrPwr+	Articulación 4. Alimentación del motor +/- 30-35 VDC a 2A max
10	9	J4_MtrPwr-	Articulación 4. Alimentación del motor (referencia)
8	10	J5_MtrPwr+	Articulación 5. Alimentación del motor +/- 30-35 VDC a 2A max
12	11	J5_MtrPwr-	Articulación 5. Alimentación del motor (referencia)

[6] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

- **Conector de Realimentación de los Encoders**

Este es un conector tipo hembra de 57 puntos de conexión, de los cuales están utilizados 30 puntos, conectados a cada uno de los *encoders* de los servomotores y a la solenoide, como se indica en la Figura 2.9.

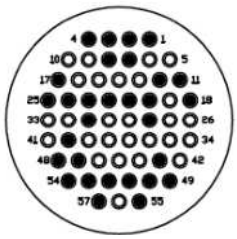

Figura 2.9 Conector para Realimentación de los *encoders*

Este cable umbilical es el encargado de llevar las señales provenientes de los *encoders*, proporcionar la señal de referencia para los frenos de cada articulación, así como también la alimentación para la solenoide, tiene un conector tipo macho en ambos extremos para conectarse con el brazo robótico y el controlador como se muestra en la Figura 2.10

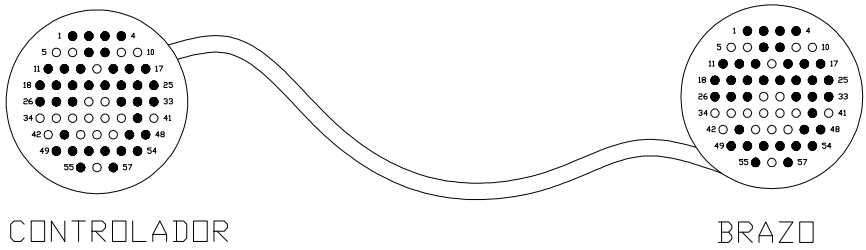

Figura 2.10 Cable umbilical de Realimentación

La distribución de cada extremo del cable se detalla a continuación en la Tabla 2.8

Tabla 2.8 Detalle de conectores del cable umbilical de realimentación de los *encoders* y solenoide [7]

Detalle de Cable de Realimentación		
# Pin	Señal	Descripción
1	Canal A, Encoder Articulación 1	0 – 5 VDC, RS422
2	Canal B, Encoder Articulación 1	0 – 5 VDC, RS422
3	Canal A, Encoder Articulación 2	0 – 5 VDC, RS422
4	Canal B, Encoder Articulación 2	0 – 5 VDC, RS422
7	Canal Z, Encoder Articulación 1	0 – 5 VDC, RS422
8	Canal Z, Encoder Articulación 2	0 – 5 VDC, RS422
12	Alimentación del Motor para Servo gripper	+/- 12 VDC a 300mA
17	VCC Encoder, Articulación 1	+5VDC a 80mA
20	Canal A, Encoder Articulación 3	0 – 5 VDC, RS422
21	Canal B, Encoder Articulación 3	0 – 5 VDC, RS422
22	Canal A, Encoder Articulación 4	0 – 5 VDC, RS422
23	Canal B, Encoder Articulación 4	0 – 5 VDC, RS422
24	Referencia de tierra, Encoder 1	
25	Referencia de tierra, Encoder 1	
28	Canal Z, Encoder Articulación 3	0 – 5 VDC, RS422
31	Canal Z, Encoder Articulación 4	0 – 5 VDC, RS422
40	Señal de referencia para Encoder, Articulaciones 3 y 5	
43	Canal A, Encoder Articulación 5	0 – 5 VDC, RS422
48	Realimentación de posición del Servo	0 – 5 VDC a 2.5 mA

[7] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

	Gripper	
49	Canal B, Encoder Articulación 5	0 – 5 VDC, RS422
50	Canal Z, Encoder Articulación 5	0 – 5 VDC, RS422
51	VCC del Encoder, Articulaciones 2 y 4	+ 5 VDC a 80 mA
52	VCC del Encoder, Articulaciones 3 y 5	+ 5 VDC a 80 mA
53	Alimentación +12 VDC	+12 VDC a 200mA para: • Vcc del potenciómetro del Servo Gripper • Alimentación de la solenoide • Verificación de Conexión de cable
54	Referencia de tierra para el potenciometro y válvula de aire	Referencia para solenoide Referencia para el potenciómetro del Servo Gripper
55	Señal de referencia para Encoder, Articulaciones 2 y 4	
57	Red resistiva de alimentación del Servo Gripper	

- **Conector para Servo gripper**

Se encuentra ubicado entre el codo y la muñeca, es un conector tipo hembra de 12 puntos de conexión, de los cuales están utilizados 7 puntos, los mismos que están conectados al puerto de realimentación del brazo robótico, la configuración se puede ver en la Figura 2.11.

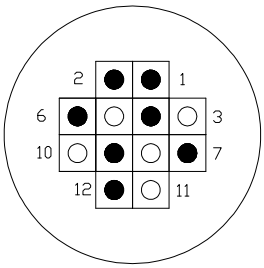

Figura 2.11 Conector para Servogripper

A continuación en la Tabla 2.9 se detalla la distribución del conector.

Tabla 2.9 Detalle del conector para *Servo Gripper* [8]

Detalle de Conector de Servogripper		
# Pin	Señal	Descripción
1	VCC del *gripper*	+ 12 VDC
2	Red resistiva de alimentación del Servo Gripper	
4	Alimentación del Motor para Servo gripper	+/- 12 VDC a 300mA
6	Alimentación de retorno del servo *gripper*	
9	Realimentación del Servo Gripper	0 – 5 VDC a 2.5 mA

[8] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

12	Tierra del chasis	GND

- **Conector para Gripper Neumático**

Este es un conector neumático de 2 puntos como se indica en la Figura 2.12, se encuentra ubicado entre el codo y la muñeca del brazo robótico.

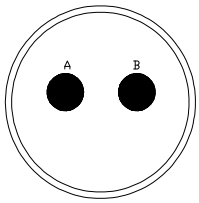

Figura 2.12 Conector para *Gripper Neumático* [9]

2.2. ESTUDIO Y ANÁLISIS DEL TIPO DE CONTROLADOR A IMPLEMENTAR

Para empezar con el análisis del tipo de controlador que se desea implementar es indispensable conocer la organización funcional de un robot, en la cual se puede apreciar las partes que conforman un sistema robótico. A continuación en la Figura 2.13 se puede ver el esquema de la organización funcional básica de un robot.

[9] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

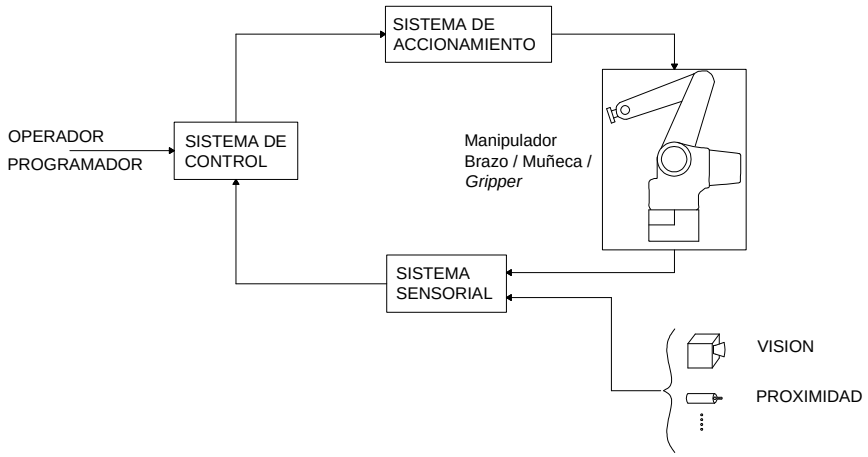

Figura 2.13 Organización Funcional de un robot

- **Brazo Robótico**

Esta es una estructura mecánica cuya función es la de soportar y conducir el efector final hacía una posición deseada dentro del volumen de trabajo.

Para el caso particular de este proyecto el brazo robótico a utilizar es el CRS A255, el mismo que está descrito anteriormente en este capítulo.

- **Sistema de Accionamiento**

El sistema de accionamiento eléctrico está compuesto por todos los motores encargados del movimiento de cada una de las articulaciones y del efector final.

Para el presente proyecto se cuenta con cinco motores para cada una de las articulaciones, y un solenoide encargado de la apertura/cierre del efector final.

- **Sistema Sensorial**

Tanto para poder conocer la posición del brazo robótico como para poder conocer el entorno en el que se desenvuelve el robot se requiere sensores de posición, sensores de proximidad, cámaras externas, etc. Todos estos elementos son los que forman parte del sistema sensorial del robot.

Para este proyecto se cuenta con sensores internos como son los *encoders incrementales* presentes en cada uno de los motores, que nos permiten conocer la posición en la que se encuentra el robot dentro del volumen de trabajo.

- **Sistema de Control**

Este sistema es el encargado de gobernar los actuadores en función de la información que recibe, tanto de la posición en la que se encuentra el brazo robótico, como del entorno.

Uno de los puntos a tomar en cuenta es que el objetivo principal de este proyecto es dejar implementada la parte física del controlador, es decir, la etapa de potencia. Se debe considerar para próximos proyectos la implementación de un control de cuatro cuadrantes ya que para las articulaciones superiores, es decir, hombro, codo, muñeca y rotador, se debe tomar en cuenta la gravedad como factor influyente en el movimiento de estos motores. Para el presente proyecto se implementará un control proporcional para la articulación 1 (cadera), mientras que para las articulaciones 2 y 3 (hombro y codo) cuando se desea subir se mantiene el concepto del control proporcional, pero cuando se desea bajar se analiza si debe frenar o acelerar para mantener la velocidad en la articulación.

2.2.1. ESTUDIO Y ANÁLISIS DEL CONTROLADOR C500

El controlador C500 (Figura 2.14) consta de tres etapas para su funcionamiento:

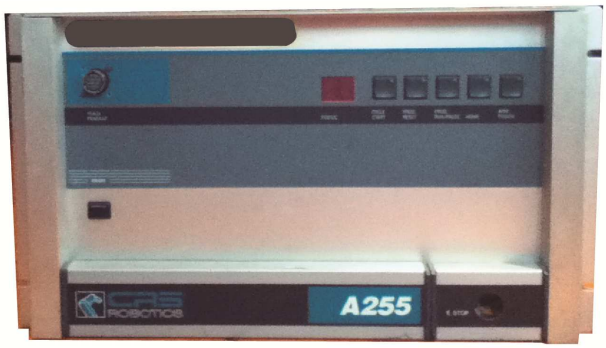

Figura 2.14 Controlador C500

- **Etapa de Transformación (Fuente de Alimentación)**

La etapa de transformación es la encargada de suministrar el voltaje y amperaje necesarios para el funcionamiento del sistema, tomando la alimentación proveniente de la red eléctrica y acoplándola a los requerimientos del sistema.

Esta etapa es una fuente de alimentación basada en el esquema que se puede apreciar en la Figura 2.15

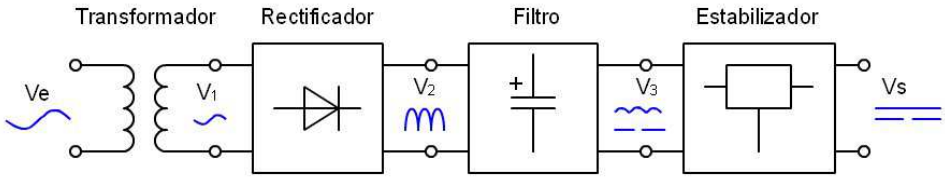

Figura 2.15 Esquema básico de una fuente de alimentación

Se puede observar esta fuente en la Figura 2.16 con cada uno de sus

componentes.

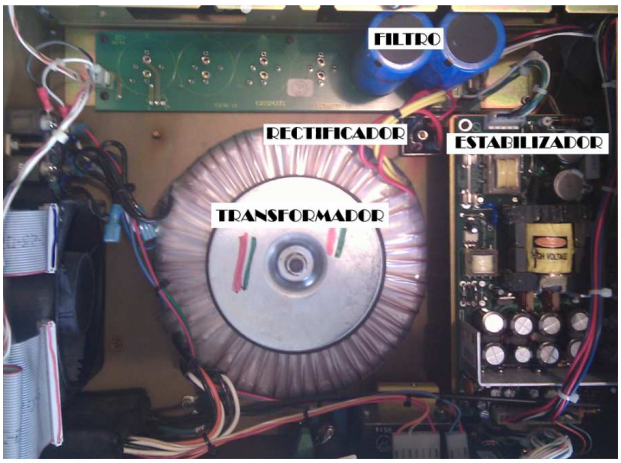

Figura 2.16 Etapa de Alimentación del controlador C500

Por ser un controlador industrial diseñado para trabajar a nivel mundial, en la entrada de alimentación de la red eléctrica cuenta con un módulo de conexión porta fusibles en el que se puede seleccionar el voltaje de alimentación de la red, y dos fusibles de 4 A. Este módulo se muestra en la Figura 2.17 a continuación:

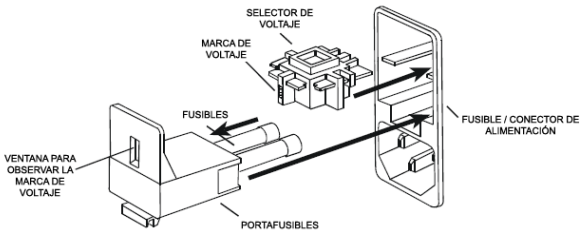

Figura 2.17 Modulo de entrada de alimentación del controlador C500 [10]

El transformador de la siguiente fase tiene las siguientes características

Tabla 2.10 Características del transformador

[10] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

TRANSFORMADOR		
Primario	100/115/230 V	50/60 Hz
Secundario	115 V	0.94 A
	50 VCT	6.6 A
	45 V	13.2 A

Se cuenta en este controlador con dos fuentes bien definidas, una fuente no regulada, que es la encargada de suministrar +/- 35VDC para el movimiento de los motores y activación de los frenos, y las fuentes reguladas de 5VDC y +/- 12VDC, que son las encargadas de la alimentación de los circuitos integrados y solenoide respectivamente.

- **Etapa de Aislamiento y Potencia**

La etapa de aislamiento se puede apreciar en la Figura 2.17, esta es la encargada de separar tanto las entradas como las salidas de los puertos *GPIO* (General Purpose Input Output) y *SYSIO* (System Input Output) a través de optoacopladores, relés y transistores para las salidas. Todo esto para prevenir daños en los circuitos integrados de control, contra posibles corrientes parásitas.

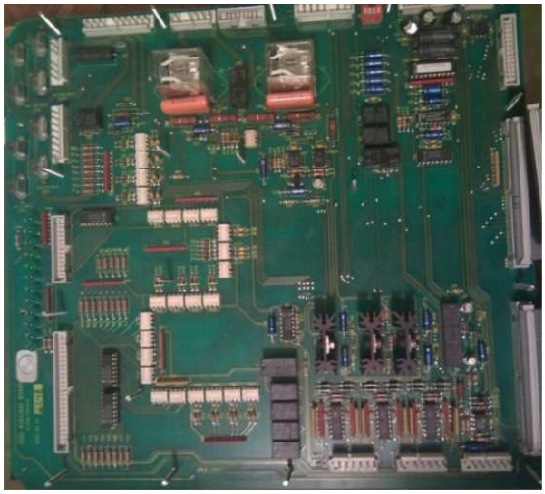

Figura 2.18 Tarjeta de Aislamiento del controlador C500

En la Figura 2.18 adicionalmente se puede observar transistores de potencia que son los encargados de alimentar a los motores de cada una de las articulaciones del robot, así como también la activación de los efectores finales, para el caso del presente proyecto, el solenoide que activa el *gripper* neumático. Para completar esta etapa de potencia se tiene dos tarjetas para el acoplamiento de los motores a un costado dentro del controlador, cada una controla 3 motores, esta tarjeta se puede ver en la Figura 2.19

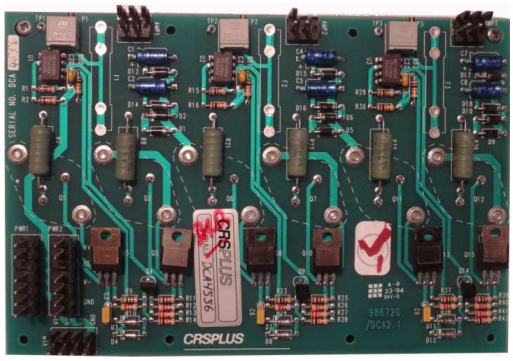

Figura 2.19 Tarjeta de acoplamiento de los motores y el controlador C500

- **Etapa de Control**

En el controlador C500 se tiene una tarjeta de control la cual se puede apreciar en la Figura 2.20. Está formada por microprocesadores, FPGA, memorias y otros circuitos integrados.

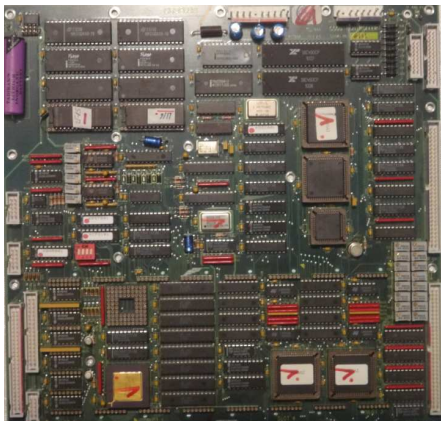

Figura 2.20 Tarjeta de control

En la Tabla 2.11 se puede apreciar algunas de las características de la etapa de control del sistema.

Tabla 2.11 Características del controlador C500 [11]

Controlador C500		
Procesadores	133 MHz i486DX	Procesador del sistema
	60 MHz TMS320C31 DSP	Control de movimiento
Memoria	4 MB	Memoria RAM de usuario
	512 KB	Almacenamiento de Aplicaciones NVRAM
	1 MB	Memoria FLASH para *firmware* del sistema

[11] Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

Esta etapa es la encargada del control total del manipulador y comunicación con el usuario, ya sea mediante un computador, *teach pendant,* o panel de control frontal.

Una de las condiciones iniciales para poder trabajar con el manipulador robótico es inicializarlo, para lo cual este cuenta con marcas de cada una de las articulaciones. Esto es debido a que los sensores con los que cuenta el manipulador son *enconders* de tipo incremental como se mencionó anteriormente en este capítulo. El controlador tiene diferentes modos de trabajo:

El primer modo de trabajo es mediante el *teach pendant* (Figura 2.21), con el que se puede manipular el robot manualmente, ya sea moviéndolo articulación por articulación, o realizando movimientos en los ejes X, Y, Z. También se pueden modificar algunas características de movimiento del manipulador como la velocidad. Este tipo de trabajo es indispensable para realizar la inicialización del manipulador (*"POSICIÓN HOME"*).

Figura 2.21 *Teach Pendant*

El segundo modo de trabajo es mediante la introducción de comandos en el computador, esto se puede lograr mediante el programa *ROBCOMM 3* (Figura 2.22). En este modo el computador envía al controlador la instrucción deseada, el controlador ejecuta la instrucción y envía una señal de aviso al computador. Mientras no se cumpla con la instrucción asignada no se puede ingresar una nueva instrucción en modo terminal.

Figura 2.22 Captura de pantalla inicial del software *ROBCOMM3*

2.2.2. ESTUDIO Y ANÁLISIS DEL CONTROLADOR A IMPLEMENTAR

Una vez descrito el controlador actual del manipulador robótico CRS A255 tenemos las bases para poder implementar el nuevo controlador.

Para el nuevo controlador se requiere una etapa de alimentación, una etapa de potencia y una etapa de control, cada una de estas se detallan a continuación.

- **Etapa de Alimentación**

La primera etapa del controlador va a ser la encargada de suministrar los voltajes y corrientes de alimentación necesarios para el funcionamiento del mismo. El esquema de la fuente de alimentación es el mismo que se tiene en el controlador actual, en la Figura 2.14 se puede apreciar el esquema básico de la fuente.

Las principales características que se debe tomar son los requerimientos de potencia que se necesitan, tanto para los sensores y actuadores que se tienen en el sistema, como para la etapa de control.

A continuación en la Tabla 2.12 se detalla los voltajes y corrientes

necesarios para el funcionamiento del manipulador.

Tabla 2.12 Requerimientos de energía para el controlador

Cantidad	Elementos	Voltaje	Corriente Individual
5	Motores	35 Vdc	2 A
4	Frenos	35 Vdc	400 mA
5	*Encoders*	5 Vdc	80 mA
1	Solenoide	12 Vdc	300 mA

Por lo que se requiere una fuente de alimentación que entregue, +35 Vdc / 11 A, +12 VDC / 2 A y +5 VDC / 2A.

- **Etapa de Potencia**

Una vez definidos los voltajes de operación se requiere acondicionar las señales para poder accionar los motores y el solenoide desde el controlador.

Como se puede apreciar después del análisis del controlador C500, para mover cada uno de los motores se cuenta con un arreglo de transistores de potencia y amplificadores operacionales que funcionan con voltajes de +35 VDC y -35 VDC.

De igual manera se realizó un estudio y un análisis de cómo funciona el controlador al momento de activar y desactivar los frenos que impiden que el manipulador caiga por efectos de la gravedad cuando está desconectado. Cuando uno desea encender y mover el manipulador robótico, lo primero que se debe hacer es energizar el brazo mediante un botón que se encuentra en la parte frontal del controlador llamado *“ARM POWER”*, el cual es el encargado de entregar el voltaje necesario para el funcionamiento de los motores, así como

también de activar las bobinas que controlan a los frenos magnéticos que sostienen el manipulador robótico. En este proyecto, la idea de energizar el brazo y desactivar los frenos no se va a mantener, lo que se va a realizar el desactivar el freno de la articulación que se desea mover; es en esta parte del controlador donde se va a colocar el paro de emergencia, necesario en cualquier aplicación de tipo industrial.

La última parte de la etapa de potencia, es la encargada de mover el *gripper.* En el presente proyecto se va a implementar la etapa de potencia necesaria para manejar un *gripper* neumático.

- **Etapa de Control**

La etapa de control es la más importante del proyecto, por esto se debe tomar muy en cuenta todos los aspectos necesarios para su desarrollo.

En primer lugar los elementos encargados del control de los movimientos y análisis del manipulador robótico y su entorno serán microcontroladores por su capacidad de comunicación, tanto entre circuitos integrados de este tipo como memorias, etc.; así como también la capacidad de comunicación con un computador y a través de este al operador, como dispositivos controladores se escogió trabajar con los PIC16F877A que cumplen con las características de comunicaciones necesarias, así como también los puertos de entrada / salida. La comunicación del controlador C500 al computador es mediante un cable serial, comunicación que se va a mantener con el nuevo controlador.

Con el objetivo de controlar los motores independientemente, se va a crear una red de microcontroladores en la configuración MAESTRO-ESCLAVO. Donde cada uno de los esclavos será encargado de controlar 1 o 2 articulaciones. Para crear esta red se eligió la comunicación I2C, ya que así como permite comunicar elementos del mismo tipo (microcontroladores),

también se puede extender la red hacia memorias y otro tipo de circuitos integrados que manejan también esta comunicación y posteriormente pueden ser utilizados para almacenamiento de puntos en el espacio, generación de trayectorias, etc. Esta comunicación se explicará más detalladamente en el capítulo siguiente.

2.3. ESTUDIO Y ANÁLISIS DEL SOFTWARE A DESARROLLAR

Como se puede apreciar en la organización funcional de un sistema robótico, una de las partes necesarias para el control del mismo es un medio de comunicación con el operador. En la actualidad a nivel industrial todos los sistemas de control que se utilizan tienen una interfaz de usuario en un computador, la cual provee al operador información del proceso, entorno y estado del manipulador robótico, para que este pueda tomar decisiones y ejecutar alguna acción.

El software *ROBCOMM 3* es un software propietario, completo en cuanto a todas las aplicaciones del manipulador robótico se refiere. Sin embargo, no se tiene el código fuente del mismo lo que nos dificulta la posibilidad de adaptar el nuevo controlador a este entorno de trabajo y programación.

En este proyecto se va a desarrollar una interfaz gráfica sencilla, marcando el inicio para que en las siguientes etapas del controlador se desarrolle un control cinemático y un *software* que emule de mejor manera el software ROBCOMM. Es por este motivo que el entorno gráfico que se debe desarrollar durante el proyecto tiene ciertas características básicas:

- El software debe ser desarrollado en un lenguaje de programación abierto o libre. El objetivo de este punto es que en un futuro se puedan tomar como base el código fuente del programa para poder crear mejoras al mismo. La plataforma Java es de conocimiento de todos los

estudiantes de la carrera, por lo que el *software* se desarrollará en un programa que trabaje sobre esta.

- Un punto muy importante al tomar la decisión sobre que lenguaje de programación es el que se debe utilizar, es la capacidad de comunicación que este tiene, es decir, si cuenta con comunicación serial, comunicación USB, etc. El entorno de desarrollo *NETBEANS* trabaja sobre la plataforma Java y permite el manejo de los puertos de comunicación del computador, por lo que se seleccionó este entorno como la base para el desarrollo del *software.*

- Este programa debe contar con las funciones básicas como movimiento y cambio de velocidad.

CAPITULO III

DISEÑO DEL SISTEMA DEL CONTROLADOR

3.1. DISEÑO DEL HARDWARE

Una vez que se tiene una idea clara del controlador que se desea implementar, se puede proceder con el diseño del mismo. Al igual que en el estudio y análisis, la fase de diseño se llevará a cabo dividiéndola en varias etapas siendo estas:

- Etapa de Control.
- Etapa de Potencia
- Etapa de Alimentación.

La etapa de alimentación será la última en diseñarse, esto debido a que para poder realizarla se necesita conocer los requerimientos de Voltaje y Consumo de Corriente que van a ser utilizados por cada una de las etapas siguientes.

3.1.1. DISEÑO DE LA ETAPA DE POTENCIA

En esta etapa se van a detallar los dispositivos usados para el accionamiento de los motores de cada articulación, para su selección se debe tomar en cuenta los requerimientos eléctricos de cada motor que son: 35 VDC, 2 A máximo. Los dispositivos considerados son: Transistores Bipolares (BJT) y

Transistores de efecto de campo de metal semiconductor (MOSFET). A continuación en la Tabla 3.1 se indican las características de los mismos, con las cuales se podrá elegir el dispositivo más adecuado para este proyecto.

Tabla 3.1 Tabla comparativa BJT vs MOSFET

	BJT	***MOSFET***
Método de activación	Por corriente	Por voltaje
Circuito de activación	Complejo	Simple
Impedancia de entrada	Baja	Alta
Velocidad de Conmutación	Baja (us)	Alta (ns)
Frecuencia de operación	Baja (menor a 100KHz)	Alta (menor a 1MHz)
Voltaje de saturación	Bajo	Alto
Fenómeno de "Segunda Ruptura"	Si	No
Aspecto Térmico	Inestable	Estable
Ruido	Alto	Bajo

Considerando que para realizar el control de los motores se va a trabajar con Modulación de Ancho de Pulso (PWM) a una frecuencia fuera del rango audible, es decir, a 20 [KHz] teniendo un periodo de 50 [µs] y que el mínimo ciclo de trabajo que se va a manejar es del 5% aproximadamente, es decir, de 2.5 [µs], se tiene una frecuencia de conmutación de 400 [KHz]. Por esta razón, se eligió como dispositivo de potencia al *MOSFET*.

Para elegir el *MOSFET* adecuado para la aplicación, se eligieron tres elementos a ser analizados, a continuación en la Tabla 3.2 se detalla el estudio respectivo:

Tabla 3.2 Tabla comparativa de MOSFET´S

	IRF530	IRF830	IRFZ44N
V_{DS} [VDC]	100	500	55
V_{GS} [VDC]	±20	±20	±20
I_D [A]	14	4.5	49
P_{tot} [W]	88	74	94
Q_g [nC]	26	38	63
$t_{d(on)}$ [ns]	10	8.2	12
$t_{d(off)}$ [ns]	23	42	44
$I_g = \frac{Q_g}{t_d}$ [A]	$I_{min} = 1.13$ $I_{máx} = 2.6$	$I_{min} = 0.90$ $I_{máx} = 4.63$	$I_{min} = 1.43$ $I_{máx} = 5.25$

Según los datos presentados en la Tabla 3.2, el elemento que a más de cumplir con los requerimientos de voltaje, corriente y potencia de disipación requerida, presenta una corriente instantánea de *gate* pequeña es el MOSFET IRF530, por lo que va a ser este dispositivo el indicado para activar el motor de cada articulación, en configuración de “Puente H” como se indica a continuación:

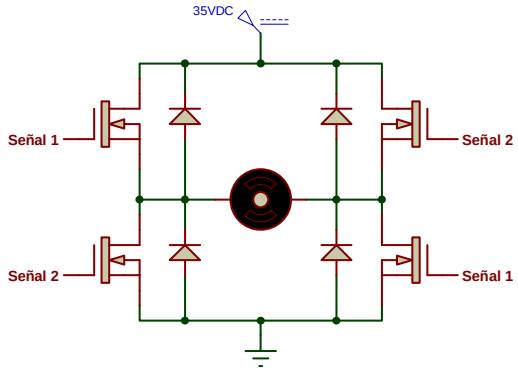

Figura 3.1 Configuración Puente H

Como se mencionó anteriormente, los *MOSFET* son dispositivos que se accionan por diferencia de voltaje existente entre *Gate* y *Source*, por lo que para asegurar esta diferencia de voltaje en los *MOSFET* de la parte superior del Puente H, se va a usar un driver, el mismo que será el encargado de generar la diferencia de voltaje y así mantener activo el *Gate* del *MOSFET*. Para el accionamiento de los *MOSFET* de la parte inferior del Puente H se utilizará únicamente opto-acopladores.

- **Driver IR2110**

Este es un driver para dispositivos de alto voltaje a una frecuencia alta, tales como *MOSFET* e *IGBT*; posee manejo independiente del lado alto como bajo. En la Tabla 3.3 se detallan las características de este driver:

Tabla 3.3 Características del driver IR2110

Símbolo	Definición	Valor
V_S	Voltaje de *offset* del lado alto	500 [$V_{máx.}$]
V_B	Voltaje absoluto flotante del lado alto	V_S + 20 [VDC]
V_{HO}	Voltaje flotante de salida del lado alto	Min V_S Máx V_B
I_O	Corriente de cortocircuito del lado alto	2.5 [A]

En la figura 3.2 se indica la conexión típica del circuito:

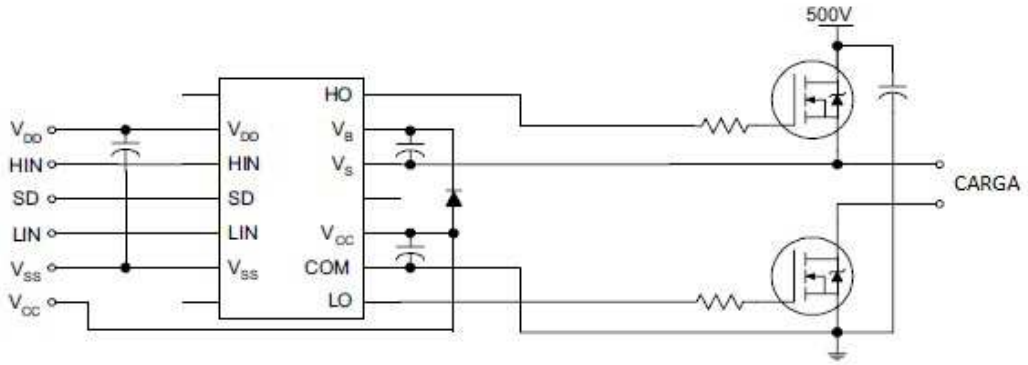

Figura 3.2 Conexión Driver IR2110 [12]

La descripción de los pines se detalla en la Tabla 3.4.

Tabla 3.4 Características del driver IR2110

Símbolo	Descripción
V_{DD}	Voltaje de alimentación del driver
HIN	Entrada lógica de activación del lado alto
SD	Entrada lógica de *shutdown*
LIN	Entrada lógica de activación del lado bajo
V_{SS}	Señal de GND
V_B	Voltaje flotante del lado alto
HO	Salida para el *gate* del lado alto
V_S	Retorno del voltaje flotante del lado alto
V_{CC}	Voltaje de alimentación que se entrega para la activación del *gate*
LO	Salida para el *gate* del lado bajo

[12] International Rectifier. (s.f.). *International Rectifier (IRF).* Obtenido de AN-978 HV Floating MOS-Gate Driver ICs: http://www.irf.com/technical-info/appnotes/an-978.pdf

COM	Retorno del lado bajo

Para el diseño del puente H se debe tener presente que el *drain de* los dispositivos del lado alto se encuentra conectado al voltaje de alimentación de la carga, y se debe asegurar que la tensión en el *gate* sea de 10 a 15 [VDC] superior al voltaje presente en el *source.*

Hay que tomar en cuenta ciertos criterios para la selección de los componentes de arranque, diodo y capacitores, ya que son los únicos componentes externos necesarios para trabajar en una aplicación con PWM en este *driver*. Para el caso del diodo, éste debe ser de recuperación ultra-rápida para ayudar a la descarga de los capacitores. Para los capacitores hay que tomar en cuenta que el que va conectado entre los pines 6 y 5 debe ser diez veces menor que el que va conectado entre los pines 3 y 2 del *driver*. [13]

- **Opto-acoplador 4n33**

Se seleccionó este elemento ya que cumple con los requerimientos para la activación del MOSFET, es decir, soporta el voltaje colector emisor a ser suministrado y la frecuencia de trabajo que se maneja.

Va a ser el encargado de activar el *gate* de los MOSFET de la parte baja del puente, es decir, proporcionará los 12 [VDC] necesarios para su activación. El diagrama esquemático es el siguiente:

[13] International Rectifier. (s.f.). *International Rectifier (IRF).* Obtenido de AN-978 HV Floating MOS-Gate Driver ICs: http://www.irf.com/technical-info/appnotes/an-978.pdf

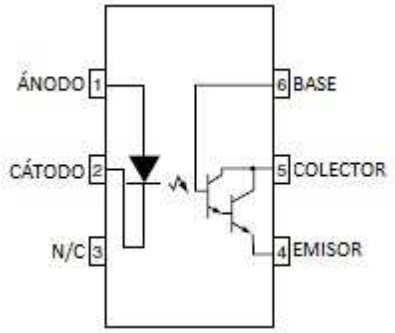

Figura 3.3 Opto-acoplador 4n33

Teniendo la señal de activación en el ánodo (PIN 1), el voltaje a ser suministrado en el colector (PIN 5) y la salida en el emisor (PIN 4).

- **Activación de los frenos**

Al realizar el estudio del robot, se detectó que la configuración actual para la desactivación de los frenos no va a ser apta para la aplicación que se está desarrollando, ya que, como se puede apreciar en la Figura 3.4(a), anteriormente para activar el freno era necesario enviar el voltaje de activación debido a que la señal de referencia (GND) era la misma del motor.

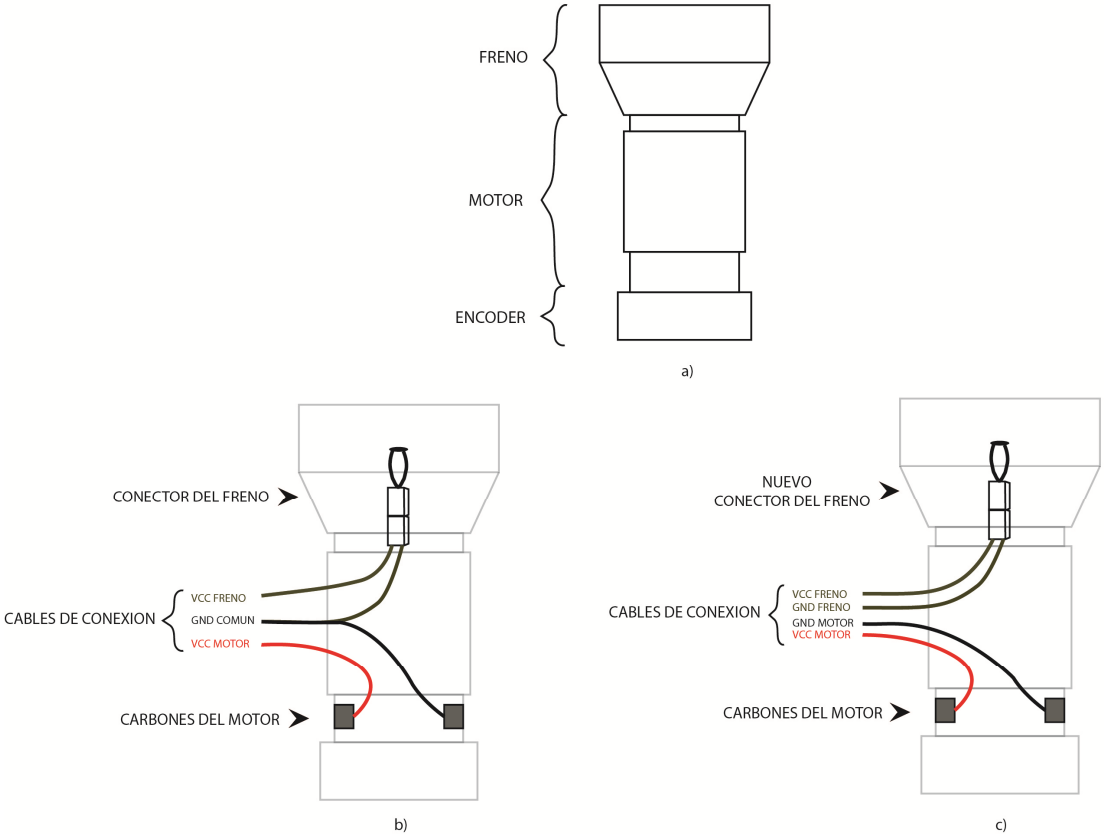

Figura 3.4 (a) Estructura de los motores del robot CRS A255
(b) Configuración de cableado de motor y freno con el controlador C500 (anterior)
(c) Configuración de cableado de motor y freno con el controlador actual

Para elaborar el controlador actual se consideró desactivar el freno de la articulación que se desea mover, y ejercer el control del motor por PWM, estas son las razones por las que se debió cambiar la configuración de activación de los frenos. Para su actual desactivación se usa un transistor NPN (C1162), como se muestra en la Figura 3.5. Los elementos usados son:

- 1 Transistor C1162, cuyas características se detallan en la Tabla 3.5
- 1 Diodo ultra-rápido FR104

Tabla 3.5 Características del transistor C1162

Símbolo	Definición	Valor
V_{CEO}	Voltaje de colector – emisor	35 [V]
V_{EBO}	Voltaje de base – emisor	5 [V]
Ic	Corriente de colector	2.5 [A]

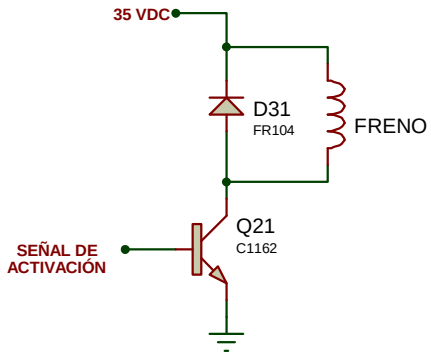

Figura 3.5 Circuito de potencia para un freno

En la Figura 3.5 se puede apreciar las conexiones realizadas para el accionamiento de un motor. Los elementos usados son:

- 2 Driver para MOSFET IR2110
- 2 Opto-acopladores 4N33
- 4 MOSFET IRF530
- 10 Diodos ultra-rápidos FR104
- 2 Capacitores electrolíticos de 0.22 [uF] de 50 [V]
- 2 Capacitores electrolíticos de 2.2 [uF] de 50 [V]
- 4 Resistencias de 10 [Ω] de 0.5 [W]
- 2 Resistencias de 1 [KΩ] de 0.5 [W]

También se puede apreciar las señales y voltajes requeridos para el accionamiento del motor, estos parámetros se describen en la Tabla 3.6 que se

detalla a continuación:

Tabla 3.6 Señales requeridas para activación del motor

Símbolo	Descripción
CCW_1	Señal de 5 [VDC] para activar el transistor del opto-acoplador (U2)
CW_1	Señal de 5 [VDC] para activar el transistor del opto-acoplador (U1)
CW_2	Señal de PWM para activar el lado alto del driver para MOSFET (U4)
CCW_2	Señal de PWM para activar el lado alto del driver para MOSFET (U3)
35 VDC	Voltaje requerido para activar el motor
12 VDC	Voltaje requerido para asegurar la activación del *gate* de los MOSFET del lado alto

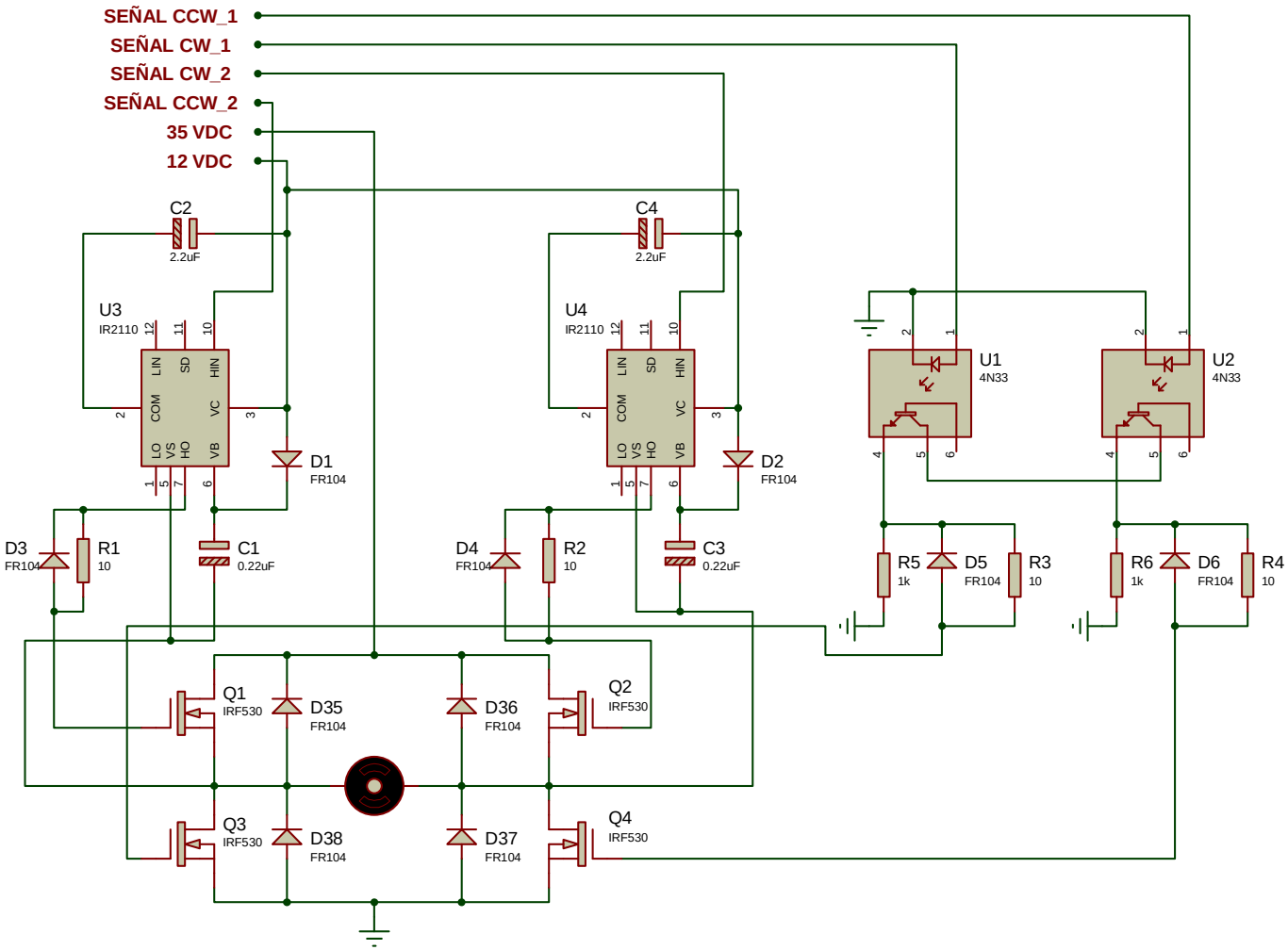

Figura 3.6 Circuito de potencia para un motor

3.1.2. DISEÑO DE LA ETAPA DE CONTROL

Para el presente proyecto, el alcance hace referencia al control y manipulación de tres de los cinco grados de libertad que posee el manipulador robótico CRS A255, sin embargo, la parte física (*Hardware*) quedará implementada para poder controlar todas sus articulaciones y un *Gripper* que funcione en base a un solenoide.

Tomando en cuenta las necesidades del controlador, mencionadas anteriormente, el sistema de control que se plantea como solución tiene una configuración “Maestro-Esclavo”, formado por:

- Un maestro que será el encargado de recibir las señales provenientes del *teach pendant*, estado de la alimentación hacia el robot, comunicación bidireccional con el computador, presentación de datos en la pantalla del controlador y envío de información a cada uno de los esclavos.

- Tres esclavos que serán los encargados de proveer las señales necesarias para el accionamiento de los motores. Cada esclavo controlará el movimiento de dos articulaciones, con excepción del que maneja la primera articulación (cadera), debido a que este será el encargado de accionar el *gripper*.

Se eligió trabajar con un microcontrolador de la familia PIC 16, el PIC 16F877A, tomando en cuenta todas las características necesarias para la implementación de este proyecto, dentro de las cuales se incluyen la comunicación I2C (interfaz de comunicación entre circuitos), RS232 (interfaz de comunicación con el computador), requerimientos de entradas y salidas necesarias, tanto para el maestro como para los esclavos.

- **Interfaz Inter-Circuitos (I^2C)** [14] [15]

Dado a que se va a manejar la configuración Maestro-Esclavo, es necesario establecer un protocolo de comunicación entre los microcontroladores, para lo que se seleccionó la Interfaz Inter-Circuitos (I^2C).

Esta comunicación se realiza a través de dos hilos que son líneas de colector abierto, sus características son:

- SCL: es la señal de reloj, pin RC3 del PIC
- SDA: es la línea de datos, pin RC4 del PIC

Se deben colocar resistencias externas o de *pull-up* para de esta manera asegurar un nivel alto cuando no existan dispositivos conectados al bus, la fórmula de cálculo es la siguiente:

$$R_P \geq \frac{V_{DD} - V_{OL}}{I_{OL}}$$

Para el microcontrolador que se usa en el presente proyecto se tiene:

- V_{DD} = 5 V
- V_{OL} = 0.6 V
- I_{OL} = 1.6 mA

$$R_P \geq 2750\ \Omega$$

Cada dispositivo conectado al bus tiene una dirección. El número de dispositivos conectados y la longitud están limitados por la capacidad de direccionamiento (7 a 10 bits) y por la máxima carga del bus (400pF). La velocidad estándar máxima es de hasta 100Kbps, la rápida hasta 400 Kbps y la

[14] García Breijo, E. (2008). *Compilador C CCS y Simulador PROTEUS Para Microcontroladores PIC* (Primera Edición ed.). México: Alfaomega Grupo Editor.
[15] *Foro: Todo PIC*. (s.f.). Obtenido de www.todopic.com.ar/foros/index.php?topic=23978.0

alta hasta 3.4 Mbps.

Existen dos tipos de configuración: Maestro–Esclavo y Multimaestro, como se mencionó anteriormente se va a utilizar la primera configuración, teniendo un maestro y tres esclavos.

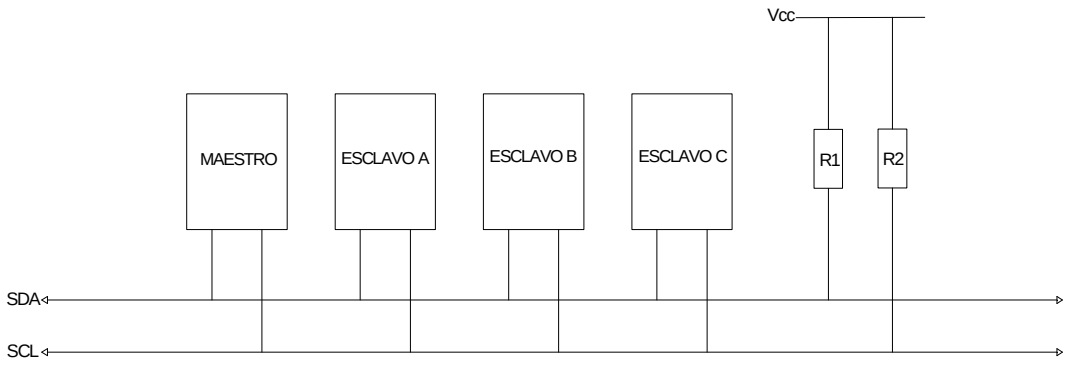

Figura 3.7 Configuración Maestro-Esclavo

La comunicación inicia con el bit de inicio ó *START*, esta condición se establece con una transición de alto a bajo en la línea de datos ó SDA cuando la línea de reloj ó SCL se encuentra en nivel alto; termina con el bit de finalización ó *STOP*, esta condición se establece con una transición de bajo a alto en la línea de datos o SDA cuando la línea de reloj ó SCL se encuentra en nivel alto. De este modo los datos en la línea SDA solo se modifican cuando la línea SCL se encuentra en estado bajo.

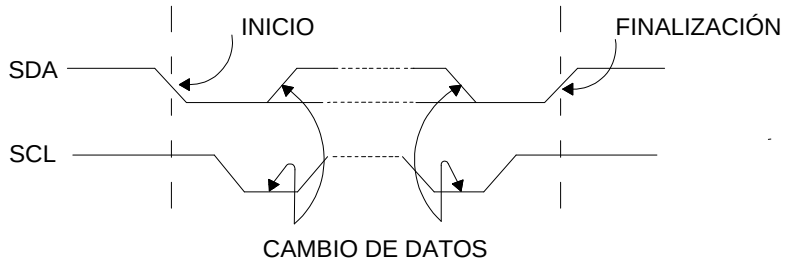

Figura 3.8 Condiciones de Inicio y Finalización de Comunicación I_2C

El maestro es el que manda en la comunicación, ya que:

- Genera la señal de reloj
- Inicia la comunicación
- Envía la dirección del dispositivo esclavo que desea establecer comunicación, esta puede ser de 7 o 10 bits, por lo que se manejan 1 o 2 bytes respectivamente
- El último bit que se envía junto con la dirección es el que indica la transmisión que se va a realizar, es decir, lectura ó escritura
- Finaliza la comunicación

Cuando el Maestro envía la dirección y datos, el esclavo genera un bit de reconocimiento de la información (ACK), si esto no se produce inmediatamente se interrumpe la comunicación generando la señal de finalización o *STOP.* Este bit de reconocimiento también es generado por el maestro cuando éste es el que recibe datos desde el esclavo, menos en el último dato ya que para ello el esclavo libera la línea de datos SDA, y el maestro genera la señal de finalización o *STOP*.

En ocasiones el maestro en lugar de abandonar el bus de comunicación, genera una señal de *START,* llamada *START* REPETIDA (Sr), similar a la primera pero generada luego de un pulso de reconocimiento (ACK).

Para el presente proyecto la programación se realizó en PICC, por lo que se utilizó la directiva de configuración así como las funciones existentes para establecer la comunicación. En la Tabla 3.7 se detalla las opciones para el uso de la directiva de configuración:

Tabla 3.7 Opciones para la Directiva de la Comunicación I_2C

Opciones	Descripción
MULTI_MASTER	Establece el modo Multimaestro
MASTER	Establece el modo Maestro
SLAVE	Establece el modo Esclavo
SCL = pin	Especifica el pin SCL ó señal de reloj
SDA = pin	Especifica el pin SDA ó de datos
ADDRESS = 0XNN	Especifica la dirección en modo esclavo
FAST	Configura velocidad alta
SLOW	Configura velocidad baja
RESTART_WDT	Borra el *Watch Dog* (WDT) mientras se encuentra en lectura
FORCE_HW	Utiliza la funciones I2C *hardware*
NOFLOAT-HIGH	No permite señales flotantes
SMBUS	Utiliza el bus en formato SMBUS
STREAM = id	Asocia un *stream*

En la siguiente tabla se detallan las funciones utilizadas con sus características:

Tabla 3.8 Funciones usadas para la Comunicación I_2C

Función	Descripción
I2C_START();	Inicializa la transmisión, luego de esta función el reloj es colocado en bajo hasta que se realice la escritura con la función *I2C_WRITE()*. Cuando se coloca esta función antes de llamar a la función *I2C_STOP()* se está generando una señal (Sr). Esta función depende de la respuesta del esclavo.
I2C_STOP();	Finaliza la comunicación
I2C_WRITE(dato);	Dato es un entero de 8 bits. Tanto en modo maestro como en modo esclavo la señal de reloj que marca la velocidad de transmisión es generada por el maestro. Cuando la transmisión termina se envía el bit de reconocimiento: 0 –> ACK, 1 –> NO ACK y 2 –> colisión en modo multimaestro. El bit menos significativo del primer dato enviado indica el sentido de la comunicación, si es 0 la transmisión será de maestro a esclavo y si es 1 la transmisión será de esclavo a maestro.
dato=I2C_READ([ACK]);	Dato es un entero de 8 bits. Como se había indicado anteriormente la señal de reloj es generada por el maestro; para evitar bloqueos se utiliza la función I2C_POLL(). Si se desea

	se puede incluir un ACK en donde: 1 –> ACK, y 0 –> NO ACK.
valor=I2C_POLL();	Sólo se puede utilizar si el PIC posee módulo SSP. Valor = 1 si se ha recibido un dato en el *buffer*, en este caso la función *I2C_READ()* guarda el dato. Valor = 0 se no se ha recibido el dato.
estado = I2C_ISR_STATE();	Sólo se puede utilizar si el PIC posee módulo SSP. Devuelve el estado del bus en modo esclavo luego de producirse una interrupción. El dato recibido es de 8 bits: 0 -> Dirección coincidente con *R/W* a cero 1 – 0x7F -> El maestro a escrito un dato, se utiliza la función *I2C_READ()*. 0x80 -> Dirección coincidente con *R/W* a uno, responder con la función *I2C_WRITE().* 0x81 – 0xFF -> Transmisión terminada y reconocida, responder con la función *I2C_WRITE().*

A continuación se aprecia la manera de configuración de la directiva así como el procedimiento para ejecutar la comunicación.

```
//MAESTRO
//=========

//Configuración de la directiva en el Maestro
      #use I2C(Master, SDA=PIN_C4, FAST, SCL=PIN_C3)

//Escritura

      i2c_start();                  //Comienzo de la comunicación I2C
      i2c_write(Dir);         //Escritura de la dirección del PIC esclavo
      i2c_write(dato);        //Escritura del dato
      i2c_stop();                   //Fin de la comunicación

//Lectura

      i2c_start();                  // Comienzo de la comunicación I2C
      i2c_write(dir+1);             // Escritura de la dirección del PIC esclavo, último
                                    bit en 1
      MSB=i2c_read();               // Lectura del dato
      LSB=i2c_read();               // Lectura del dato
      i2c_stop();                         // Fin de la comunicación


//ESCLAVO
//========

//Configuración de la directiva en el Esclavo
      #use I2C(Slave, SDA=PIN_C4, FAST, SCL=PIN_C3, ADDRESS=0xb0)

//Interrupción que detecta actividad en la comunicación I2C
        #INT_SSP
      void ssp_interupt (){
        state = i2c_isr_state();          //Estado del bus tras la interrupción
        if (state>=0 & state<0x80){       //Se debe realizar la lectura
         if(i2c_poll()) {                 //Se comprueba que exista dato en buffer
                dato=i2c_read();          //Se lee el dato
         }
         }
      if(state==0x80){                    //Transmisión exitosa, responder al
maestro
          i2c_write(buffer[0x00]);        //Envío parte MSB
       }
      if(state==0x81){
          i2c_write(buffer[0x01]);        //Envío parte LSB
      }
                    }
```

- **División del Conector de Realimentación**

Para el nuevo controlador, se va a ser uso tanto del conector y cable de potencia, como del conector y cable de realimentación; cabe recalcar que el conector de realimentación se encuentra dividido en tres conectores como se muestra en la Figura 3.9.

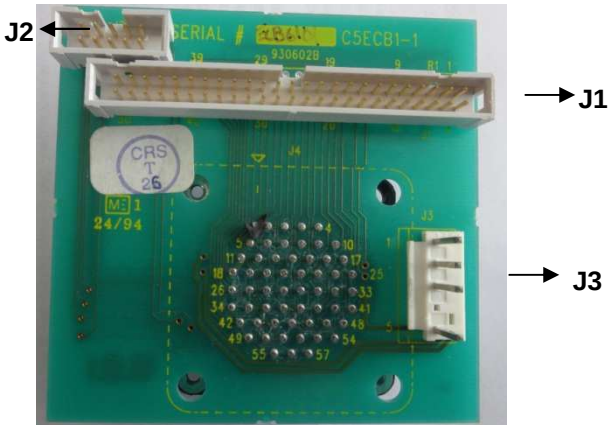

Figura 3.9 Conectores en lo que está dividido el conector de realimentación

A continuación en la Tabla 3.9 se detalla las señales que se encuentran en cada conector:

Tabla 3.9 Distribución de señales del Conector de Realimentación

Conector de Realimentación	Conector J1	Conector J2	Conector J3	Descripción
1	1			Canal A, Encoder Art. 1
2	3			Canal B, Encoder Art. 1

3	7			Canal A, Encoder Art. 2
4	9			Canal B, Encoder Art. 2
7	5			Canal Z, Encoder Art. 1
8	11			Canal Z, Encoder Art. 2
15, 16, 17		1 – 5		VCC de los *encoder* de todas las articulaciones
20	13			Canal A, Encoder Art. 3
21	15			Canal B, Encoder Art. 3
22	19			Canal A, Encoder Art. 4
23	21			Canal B, Encoder Art. 4
24, 25		6,7		Referencia de tierra de todos los *encoder*
28	17			Canal Z, Encoder Art. 3
31	23			Canal Z, Encoder Art. 4

43	25			Canal A, Encoder Art. 5
49	27			Canal B, Encoder Art. 5
50	29			Canal Z, Encoder Art. 5
51, 52		1 – 5		VCC de los *encoder* de todas las articulaciones
53	47,49			+12V para solenoide
54			2	Referencia de tierra del Solenoide

- **Acoplamiento de señales del *encoder***

Como se mencionó anteriormente, los *encoders* tienen una resolución de 1000 pulsos por revolución, poseen tres canales de salida (A, B y Z) que entregan una señal TTL, la misma que sebe ser acondicionada para posteriormente ser leída por el esclavo correspondiente.

Para realizar el acondicionamiento de señal se usó el integrado 74LS14, el cual es un inversor con *Schmitt Trigger*, cada entrada tiene histéresis lo que ayuda a reducir el ruido y transforma una señal de cambio lento en una rápida, es decir, mejora la forma de onda de la misma. Adicional a esto, en la entrada de cada circuito inversor se utilizó un filtro RC para eliminar interferencias (ruido en la señal). En la Figura 3.10 se describe las conexiones realizadas para el acondicionamiento del *encoder.*

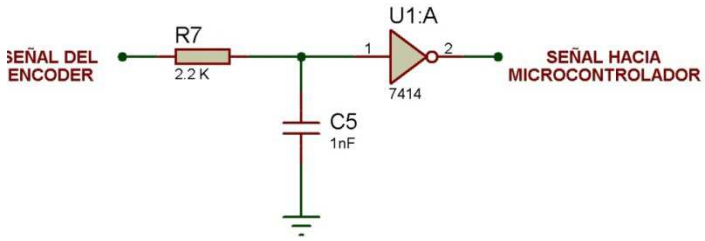

Figura 3.10 Circuito de acoplamiento

- **Circuito de enclavamiento**

En el proyecto se consideró, que para suministrar el voltaje necesario para el movimiento de los motores y accionamientos de los frenos, se debe tener un circuito electro-mecánico que permita en caso de alguna emergencia o evento inesperado detener e impedir el movimiento del manipulador robótico. Por este motivo, se diseñó un circuito de enclavamiento, como se muestra en la Figura 3.11

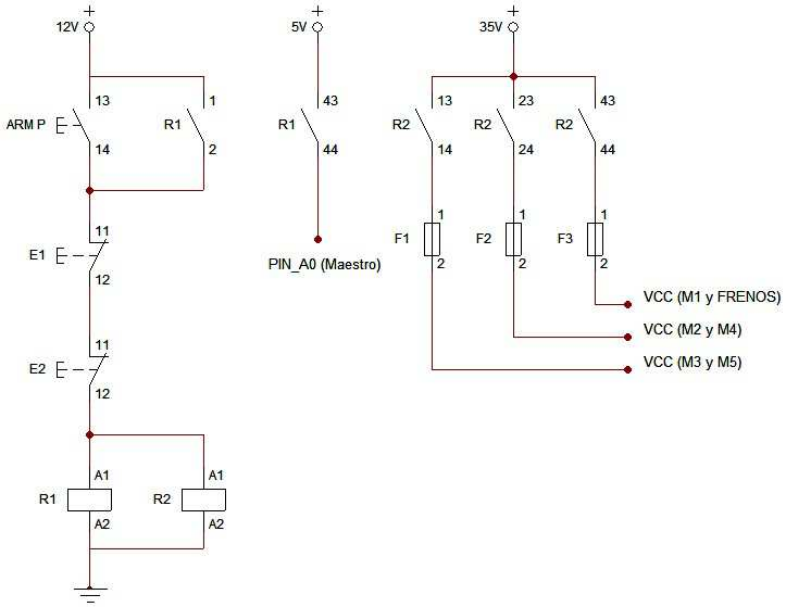

Figura 3.11 Circuito de enclavamiento

Los elementos usados para este circuito son:

- 2 Relés
- 1 Pulsador normalmente abierto, utilizado para "*Arm Power"*
- 1 Botón de paro de emergencia tipo hongo ubicado en la caja del Controlador *"E1"*
- 1 Pulsador normalmente cerrado, utilizado para generar un paro de Emergencia desde el *Teach Pendant* "*E2*"
- 3 Fusibles

El pulsador "*Arm Power"*, va a ser el encargado de activar las bobinas de los relés, de esta manera se energizan los circuitos de potencia encargados del movimiento de los motores y se envía una señal al microcontrolador maestro indicando que los circuitos anteriormente mencionados están energizados y listos para ser utilizados.

Los dispositivos de paro de emergencia son netamente mecánicos, es decir, cuando se presiona cualquiera de ellos, se desenergizarán los circuitos de potencia encargados de los motores y frenos, el operador del controlador deberá revisar la falla por la que fue necesario presionar un paro de emergencia y cuando desee continuar trabajando con el brazo robótico se debe presionar nuevamente el botón *"Arm Power".*

- **Inicialización**

Antes de comenzar a trabajar con el brazo robótico es necesario realizar un proceso de inicialización, *"Home"*, para el cual solo se podrá hacer uso del *Teach Pendant,* este proceso consiste en alinear cada articulación con las marcas que existen en el manipulador robótico. Luego de haber realizado este procedimiento, el usuario podrá seleccionar desde que dispositivo manejar el brazo robótico, *Teach Pendant* ó el computador, dicha selección será leída por el microcontrolador maestro para que este active las comunicaciones pertinentes.

- **Teach pendant**

Para el diseño del *Teach Pendant* se elaboró el circuito que se aprecia en la Figura 3.12, básicamente está compuesto por pulsadores normalmente abiertos, un pulsador normalmente cerrado y un LCD.

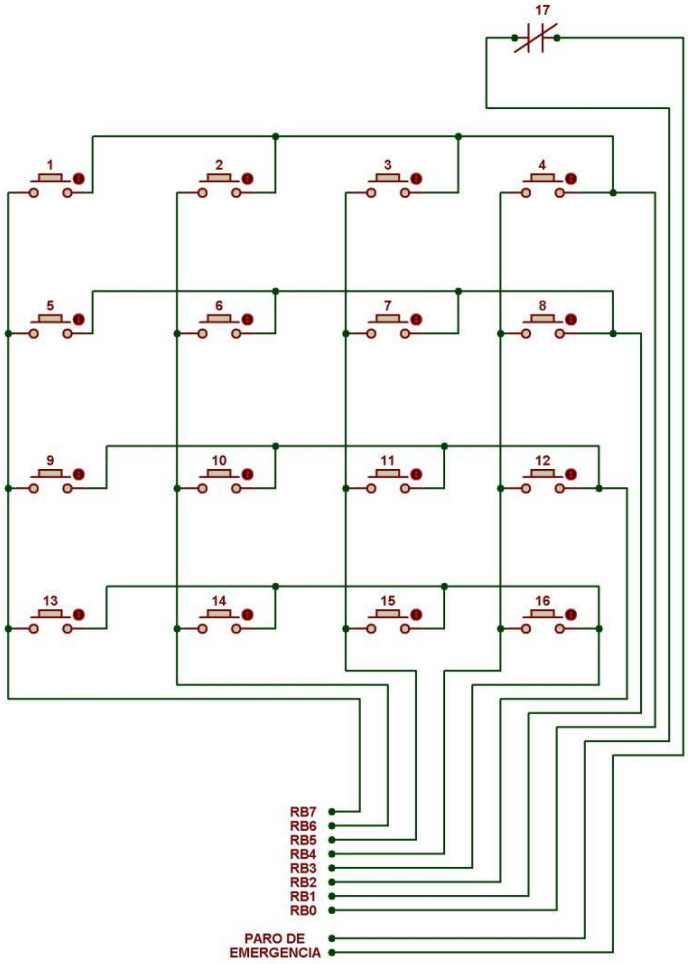

Figura 3.12 Circuito para el *Teach Pendant*

En la Tabla 3.10 se detalla las funciones de cada pulsador que conforma el *Teach Pendant.*

Tabla 3.10 *Teach Pendant*

Pulsador	Descripción
1	Activa el movimiento de la articulación 1 (cadera) en sentido CW
2	Activa el movimiento de la articulación 1 (cadera) en sentido CCW
3	Activa el movimiento de la articulación 5 (rotador) en sentido CW
4	Activa el movimiento de la articulación 5 (rotador) en sentido CCW
5	Activa el movimiento de la articulación 2 (hombro) en sentido CW
6	Activa el movimiento de la articulación 2 (hombro) en sentido CCW
7	Abre el *gripper*
8	Cierra el *gripper*
9	Activa el movimiento de la articulación 3 (codo) en sentido CW
10	Activa el movimiento de la articulación 3 (codo) en sentido CCW
11	Extra 1 (Botón libre)
12	Incrementa la velocidad
13	Activa el movimiento de la articulación 4 (muñeca) en sentido CW
14	Activa el movimiento de la articulación 4 (muñeca) en sentido CCW
15	Indica posición *Home* del manipulador robótico
16	Disminuye la velocidad
17	Paro de emergencia, desenergiza el circuito de potencia para los motores

- **Maestro**

Como su nombre lo indica este microcontrolador va a ser el encargado de gestionar todas las acciones necesarias para ejecutar el control del brazo robótico, tales como:

- Gestionar las comunicaciones, es decir, la comunicación I_2C usada para interactuar con los microcontroladores esclavos, y la comunicación RS232 usada para interactuar con la aplicación que va a ser ejecutada en el computador.

- Recibir la señal de activación del brazo robótico *"ARM POWER",* esta señal indica que los circuitos de potencia encargados del movimiento de los motores están energizados y listos para ser utilizados.

- Recibir las señales provenientes del *teach pendant*, es decir, señal de *home,* articulación que se desea mover, variación de velocidad y paro de emergencia.

- Manejo de los LCD, ubicados en el *teach pendant* y en el panel frontal del controlador.

Para efectuar la comunicación I2C, como primer punto, se establecieron las direcciones de los microcontroladores esclavos como se indica en la Tabla 3.11, así como las articulaciones que van a ser controladas por cada uno.

Tabla 3.11 Dirección de los esclavos

	Esclavo 1	Esclavo 2	Esclavo 3
Dirección	0X08	0X02	0X04
Articulación a controlar	Cadera y *Gripper*	Hombro y Muñeca	Codo y Rotador

Se debe configurar la directiva para la comunicación I_2C definiendo ciertos parámetros como el tipo de dispositivo en la comunicación (Maestro o Esclavo), definir los pines que serán usados, en este caso para datos se usará el pin 23 (SDA), para la señal de reloj el pin 18 (SCL), la velocidad de transmisión de datos, como se muestra a continuación:

#use I2C(Master, SDA=PIN_C4, FAST, SCL=PIN_C3)

El esquema utilizado para la comunicación se muestra en la Figura 3.13

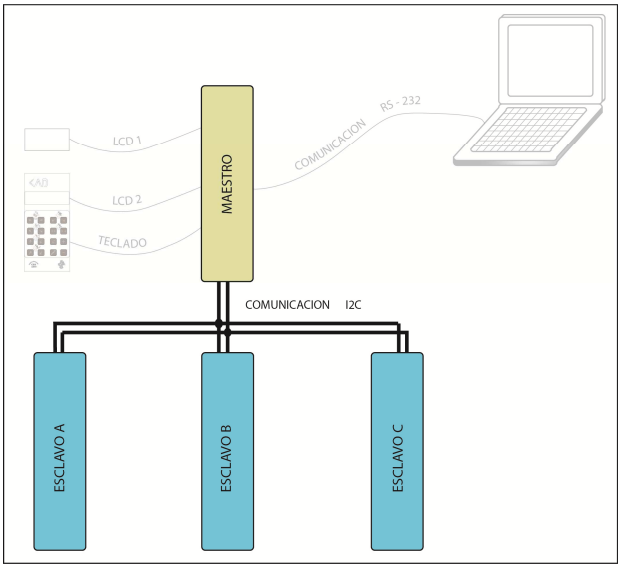

Figura 3.13 Esquema de la comunicación I2C

En el programa del MAESTRO se creó dos funciones para poder interactuar con los microcontroladores esclavos:

- La función **envio_I2C**, que recibe como argumentos la dirección del esclavo y el dato que se desea enviar y está estructurada de la siguiente manera:

```
void envio_I2C (int Dir,int dato){
    i2c_start();          //Comienzo de la comunicación I2C
    i2c_write(Dir);       //Envío de la dirección del PIC esclavo
    i2c_write(dato);      //Envío del dato
    i2c_stop();           //Finalización de la transmisión
    delay_ms(100);        //Espera de 100 milisegundos
}
```

- La función **recibo_I2C**, recibe como argumento la dirección del esclavo, se puede observar que para poder entrar en modo lectura es necesario poner en alto el bit menos significativo de la dirección, está estructurada de la siguiente manera:

```
void recibo_I2C (byte Dir){
    i2c_start();          //Comienzo de la comunicación I2C
    i2c_write(Dir+1); //Envío de la dirección del PIC esclavo con último bit en 1
    MSB=i2c_read();//Lectura del encoder, byte más significativo
    LSB=i2c_read();  //Lectura del encoder, byte menos significativo
    i2c_stop();           //Finalización de la transmisión
    delay_ms(100);   //Espera de 100 milisegundos
}
```

Esta función se ejecuta con la interrupción del Timer 1 (INT_TIMER1) como se indica a continuación:

```
#int_TIMER1                          //Interrupción del Timer 1

void  TIMER1_isr(void) {
  cont=cont+1;                       //Variable que demora el tiempo de
ejecución
  if(cont>=2){
    recibo_I2C(direccion);           //Envío de la dirección a la función
recibo_I2C
    encoder=(MSB*256)+LSB;           //Variable encoder de 16 bits
    posicion=(0.005*encoder)-162.5;  //Cálculo de la posición del manipulador
    lcd_gotoxy(6,2);                       || Escritura en el LCD
```

```
        printf(lcd_putc," %.2f  ",posicion);          ||
          }

    }
```

Como se mencionó anteriormente para el manejo del manipulador robótico se diseñó un *Teach Pendant*, el mismo que está compuesto por un lcd y un teclado, el mismo que es conectado hacia el controlador como se muestra en la Figura 3.14.

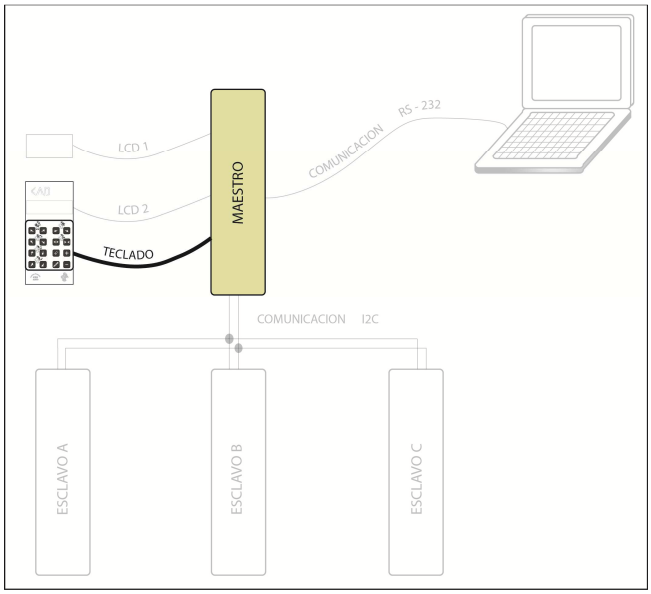

Figura 3.14 Esquema de conexión al *Teach Pendant*

Para poder realizar el uso del *Teach Pendant*, se utilizó la librería para teclado matricial de 4x4, la misma que fue modificada para que proporcione la información de la tecla que está siendo presionada, ya que originalmente devuelve el dato del botón cuando se deja de presionar el mismo.

Para proceder a ejecutar las acciones pertinentes de acuerdo al botón presionado se implementó la función lectura, en la que se define la dirección del esclavo que va a ejecutar el control y el dato a ser enviado, a continuación en la Tabla 3.12 se detalla el valor leído en el teclado, para proceder a enviar el dato correspondiente al microcontrolador esclavo.

Tabla 3.12 Tabla de lectura de datos desde el *Teach Pendant*

Teclado	Acción	Valor leído del teclado	Dato que se envía
1 (Cadera)	CW	A	dato=0X10+vel
	CCW	3	dato=0X20+vel
2 (Hombro)	CW	B	dato=0X10+vel
	CCW	6	dato=0X20+vel
3 (Brazo)	CW	C	dato=0X10+vel
	CCW	9	dato=0X20+vel
4 (Muñeca)	CW	D	dato=0X40+vel
	CCW	#	dato=0X80+vel
5 (Rotador)	CW	2	dato=0X40+vel
	CCW	1	dato=0X80+vel
Gripper	ABRIR	5	dato=0X40+vel
	CERRAR	4	dato=0X80+vel
Velocidad	Aumenta	7	Variable *vel* incrementa en 1
	Disminuye	*	Variable *vel* decrementa en 1
Home		0	dato=0X08
	Paro		dato=0X0f

La función lectura, recibe la variable *pb* proveniente de la función del teclado (kbd_lib), esta variable sirve para seleccionar la información que va a ser enviada mediante el uso de un *switch.*

```
void lectura(int d1){                    //Función lectura, recibe pb
  switch (d1) {                          //Sentencia switch
```

```
case ´q´:                       //pb=0
    art=0;                      //No existe movimiento de ninguna art.
    envio_I2C(0xa0,0x0f);       ||
    envio_I2C(0xb0,0x0f)|        >>Envío de señal de paro a los
    envio_I2C(0xc0,0x0f);       ||  esclavos
    disable_interrupts(INT_TIMER1); //Se deshabilita interrupción
    lcd_gotoxy(6,4);                    ||
    printf(lcd_putc,"   ");             ||
    if(!HOME){                           >> Escritura en el LCD
    lcd_gotoxy(7,3);                    ||
      printf(lcd_putc,"      ");       ||
    }                                   ||
  break;
case ´3´:                               //pb=0X41
    art=1;                              //Articulacion 1, movimiento CCW
    direccion=0xa0;                     //Dirección del esclavo 1
    dato=0x20+vel;                      //dato a enviar
    envio_I2C(direccion,dato);          //envío de dirección y dato a la
                                        //función envio_I2C
    lcd_gotoxy(7,3);                    ||
    printf(lcd_putc,"  1  ");           ||
    lcd_gotoxy(6,4);                     >> Escritura en el LCD
    printf(lcd_putc,"CCW");             ||
  break;

case ´7´:                               //pb=0X14
    disable_interrupts(INT_TIMER1); //Se deshabilita interrupción
    if(HOME){                           //Condición de inicialización
      vel=vel+1;                        //Incremento de velocidad en 1
      lcd_gotoxy(14,4);                     ||
      if(vel>=2){                           ||
        vel=2;  }                            >> Escritura en el LCD
        printf(lcd_putc,"%u  ",vel+1);  }   ||
  break;
```

➢ Esclavo A

Este microcontrolador es el encargado de controlar el movimiento de la articulación 1 y de efectuar la apertura y cierre del *gripper*, como se muestra en la Figura 3.15.

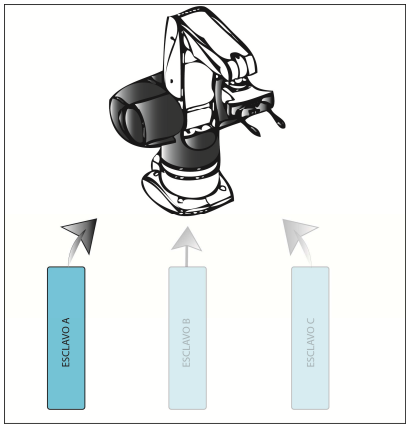

Figura 3.15 Articulaciones que controla el Esclavo A

A continuación se presenta un esquema de las conexiones de entradas / salidas con las que cuenta el microcontrolador (Figura 3.16).

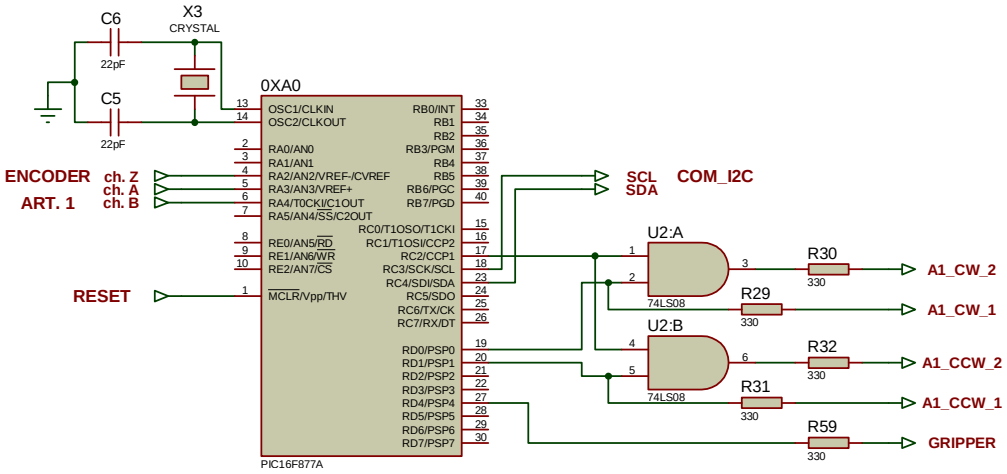

Figura 3.16 Conexiones del Esclavo A

Tabla 3.13 Descripción de los pines del microcontrolador

Pin	Entrada / Salida	Descripción
1	Entrada	Señal de *reset* general
13,14		Oscilador de 20 [MHz]
18,23		Comunicación I_2C
4	Entrada	Señal acondicionada proveniente del canal Z del *encoder* de la articulación 1.
5	Entrada	Señal acondicionada proveniente del canal A del *encoder* de la articulación 1.
6	Entrada	Señal acondicionada proveniente del canal B del *encoder* de la articulación 1, este pin funciona como contador externo para el *Timer 0.*
17	Salida	Señal de *PWM_1*
19	Salida	Señal para ejecutar el movimiento en sentido CW
20	Salida	Señal para ejecutar el movimiento en sentido CCW
27	Salida	Señal para activar el *gripper*

Tomando en cuenta que el microcontrolador PIC16F877A maneja dos señales de *PWM*, y se desea controlar dos motores con este, se consideró utilizar una compuerta *AND* (7408). Es por esta razón que en la Figura 3.16 se puede apreciar que la señal de *PWM* (Pin 17) está conectada a la entrada de dos compuertas, y a su vez la señal de activación para generar cada uno de los movimientos es conectada a la otra entrada disponible en la compuerta.

Para establecer la comunicación I_2C se definen ciertos parámetros, como el tipo de dispositivo en la comunicación (Maestro o Esclavo), definir los pines que serán usados, en este caso para datos se usará el pin 23 (SDA), para la

señal de reloj el pin 18 (SCL), la velocidad de transmisión de datos, la dirección correspondiente como se muestra a continuación:

```
#use I2C(Slave, SDA=PIN_C4, FAST, SCL=PIN_C3, ADDRESS=0x08)
```

Como se mencionó anteriormente, para poder realizar la comunicación es necesario trabajar con la interrupción SSP (Puerto Serie Síncrono), a continuación se detalla el código:

```
#INT_SSP                    //Interrupción Puerto Serie Síncrono
void ssp_interupt (){
  state = i2c_isr_state();      //Se verifica el estado del bus tras la interrupción
 if (state>=0 & state<0x80){ //Si se está recibiendo un dato:
  if(i2c_poll()) {        //Se comprueba la existencia de un dato en el buffer
    dato=i2c_read();              //Lee el dato
    velocidad=dato&0x03;          //Toma el valor de la velocidad de los bits
                                  menos significativos del dato

     if(dato==0x0f){                          ||
       output_low(pin_D0);                    ||
       output_low(pin_D1);                    || Si el dato recibido indica un paro
       sp=0;                                  >> de los motores se deshabilita las
      sum=0;                                  || interrupciones y envía 0 a la salida
       disable_interrupts(INT_TIMER1);        || del microcontrolador
       set_pwm1_duty(0);                      ||
     }                                        ||

     else if(dato==0x08){     ||
      a1=0;                   >> Si el dato indica la posición HOME, se inicializa
     }                        ||   el contadores de pulsos

     else if((dato&0xf0)>0){                  ||
       switch(velocidad){                     ||
         case 0:  aux=90;                     ||
                  set_pwm1_duty(aux);         ||
                  sp=100;                     ||
          break;                              ||Si el dato enviado se refiere a un
         case 1:  aux=90;                     ||movimiento de la articulación se
                set_pwm1_duty(aux);           >>define el SP en pulsos por cada
                  sp=250;                     ||52 [ms] y se da un impulso inicial
           break;                             ||para el movimiento de la
         case 2:  aux=90;                     ||articulación
                  set_pwm1_duty(aux);         ||
                  sp=400;                     ||
           break;                             ||
       }                                      ||
       enable_interrupts(INT_TIMER1);     //Se habilita interrupción del Timer 1
```

```
            if((dato&0x10)>0)     {              //Articulación 1 CW
              output_high(pin_D0);
              sum=1;
            }
            else if((dato&0x20)>0) {             //Articulación 1 CCW
              output_high(pin_D1);
              sum=2;
            }
            else if((dato&0x40)>0) {             //Cerrar Gripper
              output_low(pin_D4);
              sum=8;
            }
            else if((dato&0x80)>0) {             //Abrir Gripper
              output_high(pin_D4);
              sum=4;
            }
          }
        }
    if(state==0x80){                             ||Cuando el maestro solicita los
            i2c_write(buffer[0x00]);             ||datos, se envía los valores
       }if(state==0x81){                          >>colocados en el buffer
            i2c_write(buffer[0x01]);             ||anteriormente
       }                                         ||
  }
```

El lazo de control que se crea en este sistema se muestra a continuación en la Figura 3.17

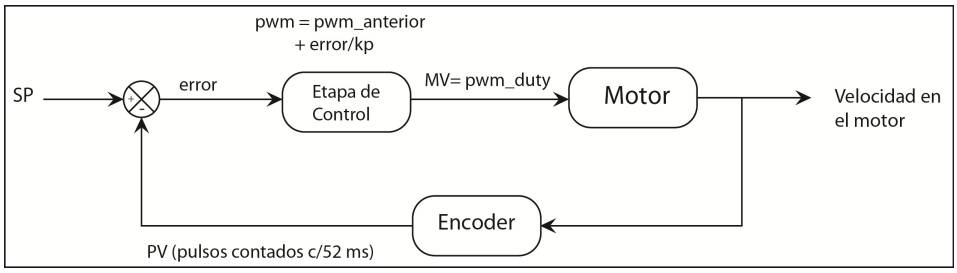

Figura 3.17 Lazo de Control

Para efectuar este lazo de control se utiliza la interrupción del *Timer1 (INT_TIMER1)*, la cual está configurada para activarse cada 52 [ms], ya que con este tiempo se puede manejar velocidades relativamente bajas en las articulaciones, teniendo una cuenta de pulsos considerable, en el gráfico de la Figura 3.18 se aprecian ejemplos de velocidad [rpm] comparado con el número de pulsos contados en cada interrupción (52,4 ms), basados en la siguiente fórmula:

$$Pulsos = 52.4 * 10^{-3} * \frac{1000 * w_{rpm}}{60}$$

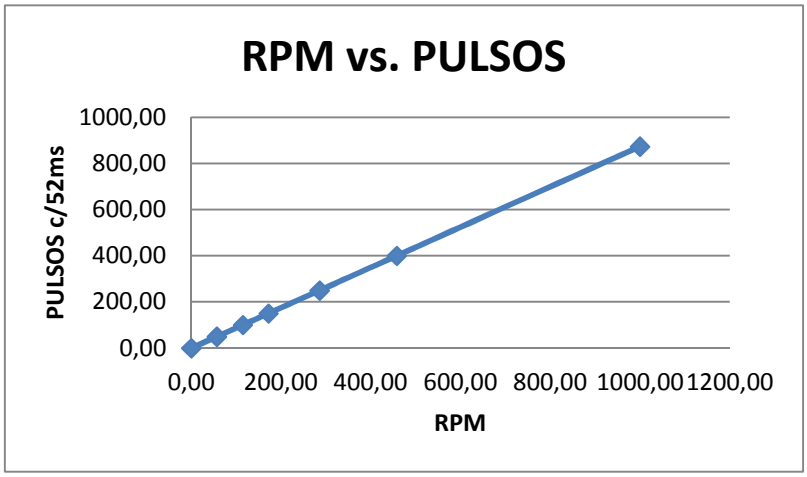

Figura 3.18 Comparación de velocidad del motor [rpm] vs. Pulsos contados cada 52.4 [ms]

Tomando en cuenta que los motores cuentan con reducción de tipo cicloidal la velocidad real de movimiento de las articulaciones corresponde a la fórmula que se presenta a continuación:

$$w_{[°/seg]} = \frac{\text{pulsos\_contados}}{52.4 * 10^{-3}} * \frac{360\ °}{1000\ \text{pulsos}} * relación\ de\ reduccion$$

De donde se puede obtener la velocidad máxima, con la configuración actual del contador del *Timer0*, para las articulaciones 1, 2 y 3, que tienen una

relación de reducción de 1:72 es de:

$$w_{max} = \frac{1000}{52.4 * 10^{-3}} * \frac{360}{1000} * {}^{1}/_{72} = 95\ [°/seg]$$

Una vez explicado el principio de funcionamiento del controlador que se desea implementar en el microcontrolador, se va a explicar el código desarrollado. Para contar el número de pulsos generados por el *encoder* se utilizó el *timer0* en modo contador. Con el objetivo de poder tener una cuenta de pulsos hasta 1024 se fijó con pre-escalador de 4, es por esto que en la interrupción del *timer1* se debe multiplicar el valor leído del contador por 4 para obtener el valor de pulsos real.

```
#int_TIMER1                 //Interrupción del Timer1
void  TIMER1_isr(void) {
  pv=get_timer0();          //Lectura del contador de pulsos
  pv=4*pv;                  //Valor real de pulsos contados

  error=sp-pv;              //Cálculo del error
  aux=aux+(error/kp);       //Sentencia de ajuste de pwm      ||
                                                              ||
  if(aux<=90){              ||                                ||
    aux=90;                 ||                                ||
  }                           >> Límites superior e inferior   >>Control
  else if(aux>=400){        || de ciclo de trabajo            ||
    aux=400;                ||                                ||
  }                         ||                                ||
                                                              ||
  set_pwm1_duty(aux);       //Variable manipulada


   Art=Art+pv;                      //Actualiza la posición de la articulación
   buffer[0x00]=(Art>>8)&0xff;      //Se coloca el valor de la articulación en
   buffer[0x01]=Art&0xff;           //un buffer para enviarse al maestro

   set_timer0(0);                   //Se encera el contador de pulsos
}
```

- **Esclavo B**

Este microcontrolador es el encargado de controlar el movimiento de las articulaciones 2 y 4 (hombro y muñeca), como se muestra en la Figura 3.19. En estas articulaciones como se había mencionado anteriormente en el análisis del controlador a implementar se debe tomar en cuenta que en el descenso no siempre se debe acelerar el movimiento del brazo, sino por el contrario ejercer un freno para poder alcanzar una velocidad deseada.

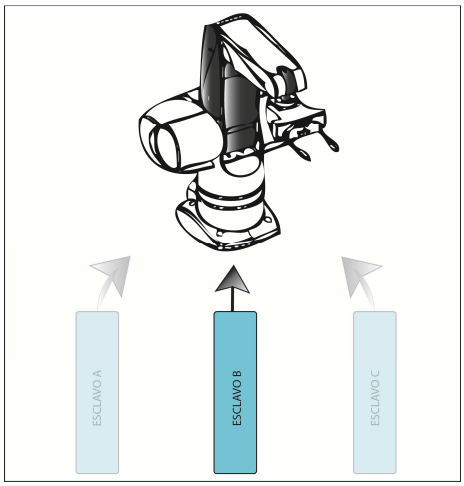

Figura 3.19 Articulaciones que controla el Esclavo B

A continuación se presenta un esquema de las conexiones de entradas / salidas con las que cuenta el microcontrolador (Figura 3.20).

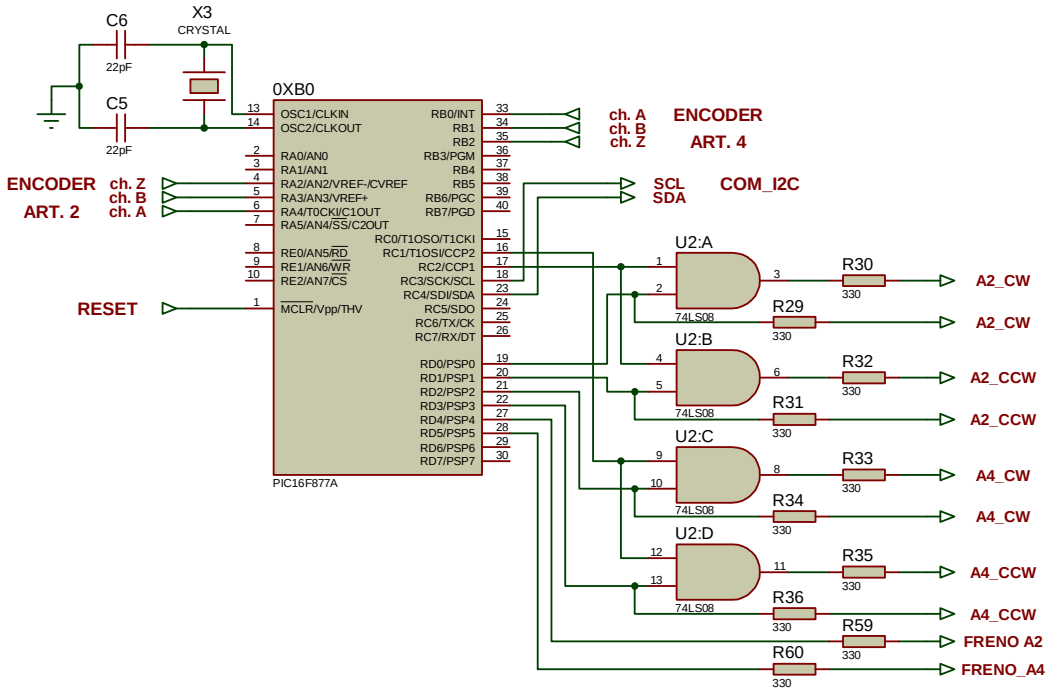

Figura 3.20 Conexiones del Esclavo B

Tabla 3.14 Descripción de los pines del Esclavo B

Pin	Entrada / Salida	Descripción
1	Entrada	Señal de *reset* general
13,14		Oscilador de 20 [MHz]
18,23		Comunicación I_2C
4	Entrada	Señal acondicionada proveniente del canal Z del *encoder* de la articulación 2.
5	Entrada	Señal acondicionada proveniente del canal B del *encoder* de la articulación 2.
6	Entrada	Señal acondicionada proveniente del canal A del *encoder* de la articulación 2, este pin funciona como contador externo para el *Timer 0.*
33	Entrada	Señal acondicionada proveniente del canal A del

		encoder de la articulación 4, este pin está listo para funcionar con la interrupción externa del puerto B (PIN_RB0).
34	Entrada	Señal acondicionada proveniente del canal A del *encoder* de la articulación 4.
35	Entrada	Señal acondicionada proveniente del canal Z del *encoder* de la articulación 4.
16	Salida	Señal de *PWM_2*, para articulación 4
17	Salida	Señal de *PWM_1*, para articulación 2
19	Salida	Señal para ejecutar el movimiento en sentido CW en articulación 2
20	Salida	Señal para ejecutar el movimiento en sentido CCW en articulación 2
21	Salida	Señal para ejecutar el movimiento en sentido CW en articulación 4
22	Salida	Señal para ejecutar el movimiento en sentido CCW en articulación 4
27	Salida	Señal para desactivar el freno de articulación 2
28	Salida	Señal para desactivar el freno de articulación 4

Como se explicó para el anterior esclavo lo primero que se debe hacer para trabajar con la comunicación I2C es colocar la directiva que indica la función del microcontrolador dentro de la red. La directiva mantiene los datos del esclavo A con la diferencia de la dirección asignada:

#use I2C(Slave, SDA=PIN_C4, FAST, SCL=PIN_C3, ADDRESS=0x02)

Al igual que en esclavo A se va a trabajar con la interrupción del puerto serie síncrono, pero en el caso de este existe una modificación ya que se toma

en cuenta el movimiento de bajada como un caso especial. Para la estructura de esta parte del programa, lo primero que se tomó en cuenta es que en la mayoría del volumen de trabajo la articulación 2 (hombro) la gravedad ejerce una fuerza que se debe controlar, es por esto que cuando se recibe la orden de bajar la articulación se determina un ciclo de trabajo de *PWM* bajo y se activa la señal de subida, con lo que se ejerce un freno inicial y posterior a esto el controlador decide si es necesario mantener el freno o acelerar el movimiento de bajada de esta articulación. Todo esto se puede apreciar en el código expuesto a continuación:

```
#INT_SSP
void ssp_interupt (){
  state = i2c_isr_state();         //Se verifica el estado del bus tras la interrupción
 if (state>=0 & state<0x80){    //Si se está recibiendo un dato:
    if(i2c_poll()) {         //Se comprueba la existencia de un dato en el buffer
      dato=i2c_read();              //Lee el dato
      velocidad=dato&0x03;          //Toma el valor de la velocidad de los bits
                                    menos significativos del dato
      if(dato==0x0f){                          ||
        sum=0;                                 ||   Si el dato recibido indica un paro de
        disable_interrupts(INT_TIMER1); >> los motores se deshabilita las
        output_d(0x00);                        ||  interrupciones y se pone 0  a la salida
        sp=0;                                  ||  del microcontrolador
      }
      else if(dato==0x08){
        a2=0;                    //Si el dato indica la posición HOME, se inicializa
        a4=0;                    //los contadores de pulsos
      }
      else if((dato&0xf0)>0){
        switch(velocidad){              ||
          case 0:  aux=220;             ||
                   set_pwm1_duty(aux);  ||
                   sp=50;               ||
                     break;             ||        Si el dato enviado se refiere a un
          case 1:  aux=220;             ||        movimiento de alguna Art. Se define
                   set_pwm1_duty(aux);   >>       el SP en pulsos por cada 52 [ms]
                   sp=150;              ||        y se da un impulso inicial para
                   break;               ||        el movimiento de la articulación
          case 2:  aux=250;             ||
                   set_pwm1_duty(aux);  ||
                   sp=250;              ||
                        break;          ||
        }
        enable_interrupts(INT_TIMER1);
        if((dato&0x10)>0)    {                  ||        Articulacion 2 Subir
          output_high(pin_D4);                  ||         Desactiva el freno
```

```
            kp=20;                          >>   Trabaja con el control similar
            output_d(0x11);                 ||   al movimiento de la Art. 1
            sum=1;                          ||
          }
          else if((dato&0x20)>0) {          ||   Articulacion 2 Bajar
            output_high(pin_D4);            ||   Desactiva el freno
            kp=15;                          ||
            aux=50;                          >>  Fija un ciclo de trabajo del 5 al 10%
            set_pwm1_duty(aux);             ||   del PWM, y active la señal de
            output_d(0x11);                 ||   subida
            sum=2;                          ||
          }
        }
      }
    }
    if(state==0x80){                        ||   Cuando el maestro solicita
            i2c_write(buffer[0x00]);        ||   los datos, se envía los valores
    }if(state==0x81){                        >>  colocados en el buffer
            i2c_write(buffer[0x01]);        ||   anteriormente
    }                                       ||
  }
```

En el código presentado, se puede observar la variable SUM, esta es la encargada de verificar que orden ha sido enviada por el maestro, y la misma va a ser utilizada en la siguiente parte que hace referencia al control implementado y las diferencias existentes entre la solicitud de subida o bajada de la articulación.

```
#int_TIMER1                 //Interrupción del Timer1
void  TIMER1_isr(void) {
  pv=get_timer0();          //Lectura del contador de pulsos
  pv=4*pv;                  //Valor real de pulsos contados

  error=sp-pv;              //Cálculo del error

  if(sum==1){                       //Subiendo la articulacion
     aux=aux+(error/kp);    //Sentencia de ajuste de pwm      ||
     if(aux<=180){          ||                              ||
       aux=180;             ||                              ||
     }                       >> Límites superior e inferior   >>Control de
     else if(aux>=400){     || de ciclo de trabajo          ||subida
       aux=400;             ||                              ||
     }                      ||                              ||
      output_d(0x11);                                       ||
  }

  else if(sum==2){          //Bajando la articulacion
```

```
 aux=aux-(error/kp);        //Sentencia de ajuste de pwm
if(aux<0){                  //Si el valor de ajuste es menor a 0 ||
   output_d(0x12);          //significa que necesita empujar      ||
   if(aux<-300){            //y se definió el límite superior a un ||
      aux=-300;             //30% aprox.de ciclo de trabajo       ||
   }       }                                                      ||
                                                                  ||
else if(aux>=150){          //Si el valor de ajuste es positivo,  ||
   aux=150;                 //se activa la señal de subida y se   >>Control de
   output_d(0x11);          // compara con el límite superior     ||bajada
}                           // de freno, es decir, aprox. Un      ||
else{                       //15% del ciclo de trabajo de PWM     ||
   output_d(0x11);          //                                    ||
}     }                                                          ||
set_pwm1_duty(abs(aux));// Se define el Nuevo ciclo de           ||
                            //trabajo                            ||
 Art=Art+pv;                        //Actualiza la posición de la articulación
buffer[0x00]=(Art>>8)&0xff;         //Se coloca el valor de la articulación en
buffer[0x01]=Art&0xff;              //un buffer para enviarse al maestro

set_timer0(0);        }                     //Se encera el contador de pulsos
```

- **Esclavo C**

Este microcontrolador es el encargado de controlar el movimiento de las articulaciones 3 y 5 (codo y rotador de la muñeca), como se muestra en la Figura 3.21. En estas articulaciones como se había mencionado anteriormente en el análisis del controlador a implementar se debe tomar en cuenta que en el descenso no siempre se debe acelerar el movimiento del brazo, sino por el contrario ejercer un freno para poder alcanzar una velocidad deseada.

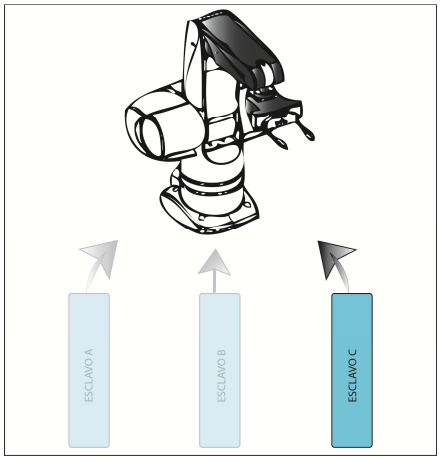

Figura 3.21 Articulaciones que controla el Esclavo B

A continuación se presenta un esquema de las conexiones de entradas / salidas con las que cuenta el microcontrolador (Figura 3.22).

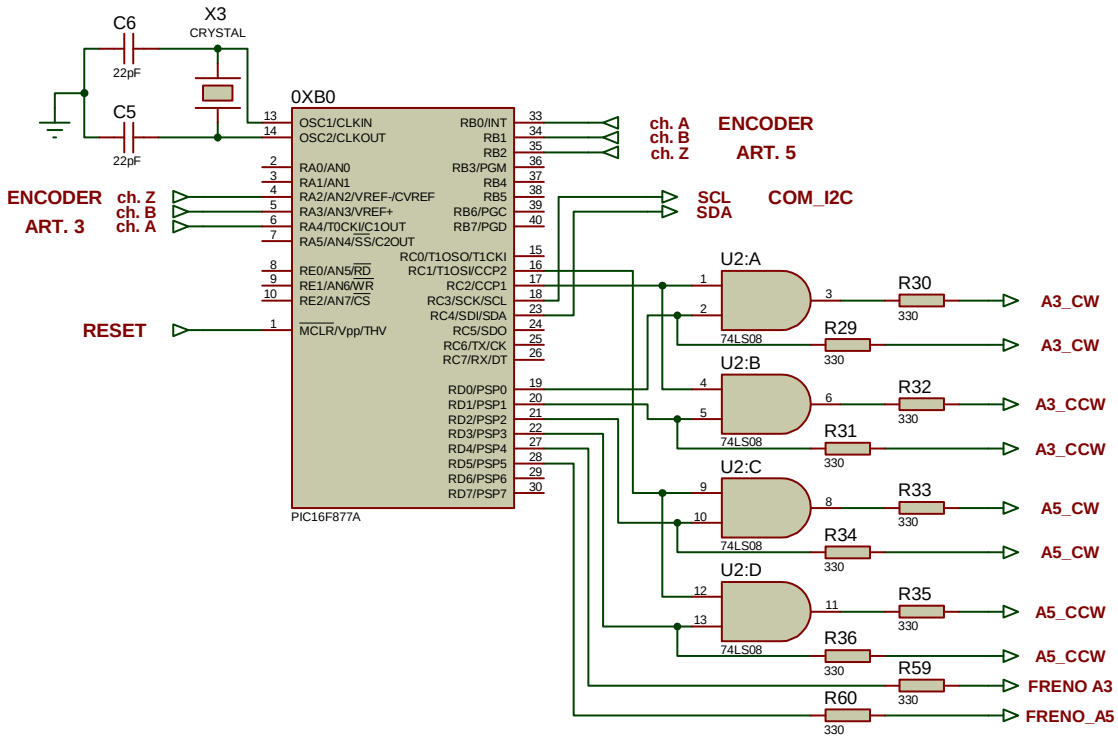

Figura 3.22 Conexiones del Esclavo C

Tabla 3.15 Descripción de los pines del Esclavo C

Pin	Entrada / Salida	Descripción
1	Entrada	Señal de *reset* general
13,14		Oscilador de 20 [MHz]
18,23		Comunicación I_2C
4	Entrada	Señal acondicionada proveniente del canal Z del *encoder* de la articulación 3.
5	Entrada	Señal acondicionada proveniente del canal B del *encoder* de la articulación 3.
6	Entrada	Señal acondicionada proveniente del canal A del *encoder* de la articulación 3, este pin funciona como contador externo para el *Timer 0.*
33	Entrada	Señal acondicionada proveniente del canal A del *encoder* de la articulación 5, este pin está listo para funcionar con la interrupción externa del puerto B (PIN_RB0).
34	Entrada	Señal acondicionada proveniente del canal A del *encoder* de la articulación 5.
35	Entrada	Señal acondicionada proveniente del canal Z del *encoder* de la articulación 5.
16	Salida	Señal de *PWM_2*, para articulación 5
17	Salida	Señal de *PWM_1*, para articulación 3
19	Salida	Señal para ejecutar el movimiento en sentido CW en articulación 3
20	Salida	Señal para ejecutar el movimiento en sentido CCW en articulación 3
21	Salida	Señal para ejecutar el movimiento en sentido CW

		en articulación 5
22	Salida	Señal para ejecutar el movimiento en sentido CCW en articulación 5
27	Salida	Señal para desactivar el freno de articulación 3
28	Salida	Señal para desactivar el freno de articulación 5

Como se explicó para el anterior esclavo lo primero que se debe hacer para trabajar con la comunicación I2C es colocar la directiva que indica la función del microcontrolador dentro de la red. La directiva mantiene los datos del esclavo A con la diferencia de la dirección asignada:

```
#use I2C(Slave, SDA=PIN_C4, FAST, SCL=PIN_C3, ADDRESS=0x04)
```

Al igual que en esclavo A se va a trabajar con la interrupción del puerto serie síncrono, pero en el caso de este existe una modificación ya que se toma en cuenta el movimiento de bajada como un caso especial. Para la estructura de esta parte del programa, lo primero que se tomó en cuenta es que en la mayoría del volumen de trabajo la articulación 3 (codo) la gravedad ejerce una fuerza que se debe controlar, es por esto que cuando se recibe la orden de bajar la articulación se determina un ciclo de trabajo de *PWM* bajo y se activa la señal de subida, con lo que se ejerce un freno inicial y posterior a esto el controlador decide si es necesario mantener el freno o acelerar el movimiento de bajada de esta articulación. Todo esto se puede apreciar en el código expuesto a continuación:

```
#INT_SSP
void ssp_interupt (){
  state = i2c_isr_state();          //Se verifica el estado del bus tras la interrupción
 if (state>=0 & state<0x80){    //Si se está recibiendo un dato:
    if(i2c_poll()) {         //Se comprueba la existencia de un dato en el buffer
      dato=i2c_read();           //Lee el dato
      velocidad=dato&0x03;          //Toma el valor de la velocidad de los bits
                                    menos significativos del dato
      if(dato==0x0f){                    ||
```

```
        sum=0;                          ||   Si el dato recibido indica un paro de los
        disable_interrupts(INT_TIMER1); >> motores se deshabilita las
interrupciones
        output_d(0x00);                 ||  y se pone 0  a la salida del
        sp=0;                           || microcontrolador
      }
      else if(dato==0x08){
        a3=0;                  //Si el dato indica la posición HOME, se inicializa
        a5=0;                  //los contadores de pulsos
      }
      else if((dato&0xf0)>0){
        switch(velocidad){              ||
          case 0:  aux=250;             ||
                set_pwm1_duty(aux);     ||
                sp=100;                 ||
                  break;                ||      Si el dato enviado se refiere a un
          case 1:  aux=250;             ||      movimiento de alguna Art. Se define
                set_pwm1_duty(aux);     >>      el SP en pulsos por cada 52 [ms]
                sp=250;                 ||      y se da un impulso inicial para
                break;                  ||      el movimiento de la articulación
          case 2:  aux=250;             ||
                set_pwm1_duty(aux);     ||
                sp=400;                 ||
                     break;             ||
        }
        enable_interrupts(INT_TIMER1);
        if((dato&0x10)>0)    {          ||      Articulacion 2 Subir
          output_high(pin_D4);          ||       Desactiva el freno
          kp=30;                        >>      Trabaja con el control similar al
          output_d(0x11);               ||      movimiento de la Art. 1
          sum=1;                        ||
        }
        else if((dato&0x20)>0)          ||      Articulacion 2 Bajar
          output_high(pin_D4);          ||      Desactiva el freno
          kp=30;                        ||
          aux=0;                        >>      Fija un ciclo de trabajo del 5 al 10%
          set_pwm1_duty(aux)            ||      del PWM, y active la señal de
          output_d(0x11);               ||      subida
          sum=2;                        ||
        }
      }
    }
  }
   if(state==0x80){                             ||   Cuando el maestro solicita los
            i2c_write(buffer[0x00]);            ||   datos, se envía los valores
  }if(state==0x81){                             >> colocados en el buffer
            i2c_write(buffer[0x01]);            ||   anteriormente
  }                                             ||
}
```

En el código presentado, se puede observar la variable SUM, esta es la

encargada de verificar que orden ha sido enviada por el maestro, y la misma va a ser utilizada en la siguiente parte que hace referencia al control implementado y las diferencias existentes entre la solicitud de subida o bajada de la articulación.

```
#int_TIMER1                 //Interrupción del Timer1
void  TIMER1_isr(void) {
  pv=get_timer0();          //Lectura del contador de pulsos
  pv=4*pv;                  //Valor real de pulsos contados

  error=sp-pv;              //Cálculo del error

  if(sum==1){               //Subiendo la articulacion
      aux=aux+(error/kp);   //Sentencia de ajuste de pwm      ||
      if(aux<=200){         ||                               ||
        aux=200;            ||                               ||
      }                       >> Límites superior e inferior  >>Control de
      else if(aux>=400){    || de ciclo de trabajo           ||subida
        aux=400;            ||                               ||
    }                       ||                               ||
       output_d(0x11);                                       ||
  }

  else if(sum==2){          //Bajando la articulacion
     aux=aux-(error/kp);    //Sentencia de ajuste de pwm     ||
     if(aux<0){             //Si el valor de ajuste es menor a 0 ||
       output_d(0x12);      //significa que necesita empujar ||
       if(aux<-300){        //y se definió el límite superior a un ||
         aux=-300;          //30% aprox.de ciclo de trabajo  ||
       }
     }                                                      ||
     else if(aux>=80){      //Si el valor de ajuste es positivo, ||
       aux=80;              //se activa la señal de subida y se >>Control de
       output_d(0x11);      // compara con el límite superior ||bajada
     }                      // de freno, es decir, aprox. Un  ||
     else{                  //15% del ciclo de trabajo de PWM ||
       output_d(0x11);      //                               ||
          }                                                 ||
      }                                                     ||
   set_pwm1_duty(abs(aux));// Se define el Nuevo ciclo de   ||
                           //trabajo                        ||
   Art=Art+pv;                  //Actualiza la posición de la articulación
   buffer[0x00]=(Art>>8)&0xff;  //Se coloca el valor de la articulación en
   buffer[0x01]=Art&0xff;       //un buffer para enviarse al maestro

   set_timer0(0);               //Se encera el contador de pulsos
}
```

3.1.3. DISEÑO DE LA ETAPA DE ALIMENTACIÓN

Una vez que conocemos que elementos y cuantos de cada tipo vamos a utilizar se puede proceder con el diseño de la etapa de alimentación de todo el proyecto.

Para esto debemos recordar que vamos a trabajar con varios voltajes de alimentación, los mismos que se detallan a continuación en la Tabla 3.16.

Tabla 3.16 Detalle de Alimentación

VOLTAJE	ELEMENTOS	CONSUMO DE CORRIENTE [mA]	CANTIDAD
5V	Microcontroladores (PIC16F877A)	250	4
	Compuertas AND (74HC08)	15	3
	Compuerta Inversora (74LS14)	15	3
	(MAX232)	15	1
	LCD 16x2	260	1
	LCD 16x4	260	1
	Driver para MOSFET (IR2110)	250	10
	Encoder	80	5
12V	Relé	250	2
	MOSFET (IRF530)	1	20
	Solenoide	130	1
35V	Motor	2000	5
	Freno	200	4

En base a la Tabla 3.16 el diseño de las fuentes de alimentación será de 5VDC a 6A, 12 VDC a 1.5 A, y 35VDC a 11A.

Para la fuente de alimentación de 35VDC se va a hacer una fuente de alimentación no regulada con el esquema que se presenta a continuación en la Figura 3.23.

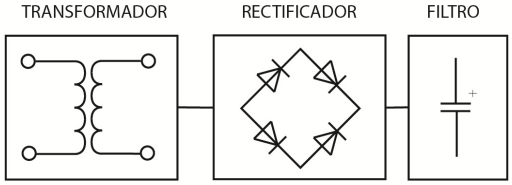

Figura 3.23 Esquema de fuente no regulada

En el esquema presentado se requiere de un transformador que entregue la corriente necesaria para el funcionamiento tanto de los motores como de los frenos para las articulaciones correspondientes, por lo que el transformador deberá entregar 11 A en el lado de 35VDC, en cuanto a los capacitores, se consiguió en el mercado capacitores de 15000uF a 60 V, por lo que para asegurar la potencia requerida por el controlador se va a colocar dos de estos elementos en paralelo.

Para las fuentes de 5VDC y 12VDC se va a hacer una fuente de alimentación regulada, es decir, en comparación al esquema anterior se aumenta la etapa de estabilización, como se muestra en la Figura 3.24.

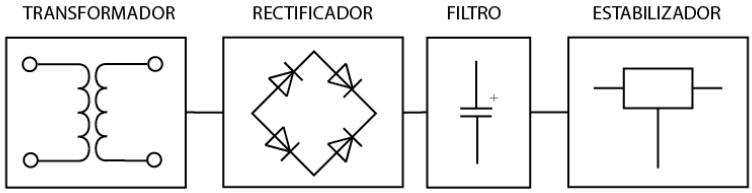

Figura 3.24 Esquema de fuente regulada

En el mercado local, se encontró elementos reguladores que son capaces de entregar hasta 3A, como es el caso del LM350K, y elementos capaces de entregar hasta 1.5 A como el LM317. Por esta razón se decidió hacer 2 fuentes de 5V a 3A, y una fuente de 12V a 1.5A. Para la implementación de las fuentes de alimentación se escogió trabajar con dos transformadores, uno que sea el encargado de suministrar los 35V y, 20V de donde se alimentará a una de las fuentes de 5V y a la fuente de 12 V. Para la otra fuente de 5V se utilizará un transformador diferente, con el objetivo de no sobrecargar al transformador. Estos esquemas se presentan en la Figura 3.25.

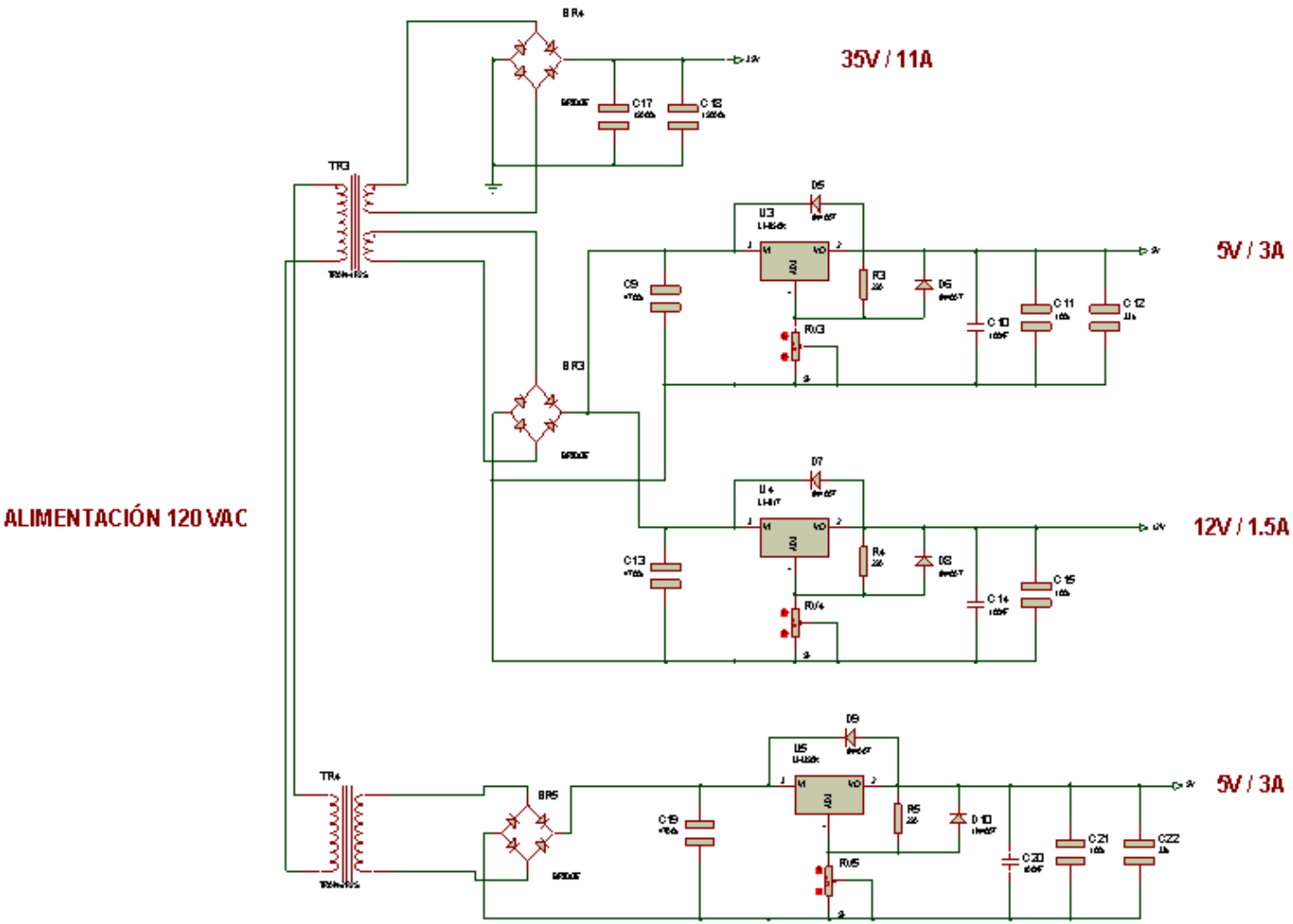

Figura 3.25 Fuentes de Alimentación de 5V, 12V y 35V

3.2. DISEÑO DEL SOFTWARE

Para el diseño del software se tomó en consideración la realización de un programa básico en el que se tenga la representación de un *teach pendant* para poder manejar al robot, así como también, recibir señales sobre el estado de ciertas variables de interés provenientes del controlador propiamente.

Otro de los aspectos a tomar en cuenta para el presente proyecto es que, las aplicaciones en computador sean desarrolladas en un software libre y de manejo común para los estudiantes de la carrera, por este motivo se decidió trabajar y desarrollar el programa en *NETBEANS 7.2,* el mismo que trabaja sobre la plataforma Java y permite comunicaciones con los puertos externos del computador.

3.2.1. Comunicación Computador - Controlador

Como se definió en el anterior punto de este capítulo se va a utilizar un microcontrolador de la familia PIC 16 (PIC16F877A), encargado de gestionar las comunicaciones tanto internas (configuración maestro-esclavo), como externas (comunicación con un computador). El PIC16F877A cuenta con comunicación RS232, y de igual manera, desde *NETBEANS* se puede manejar y controlar el puerto serial existente en el computador. En la figura 3.26 se muestra un esquema de la configuración a implementar. En caso de que los computadores en los que se utilice el programa no tengan puerto serial, se podrá manejar con un cable USB-SERIAL y de esta manera se podrá trabajar con un puerto serial normalmente.

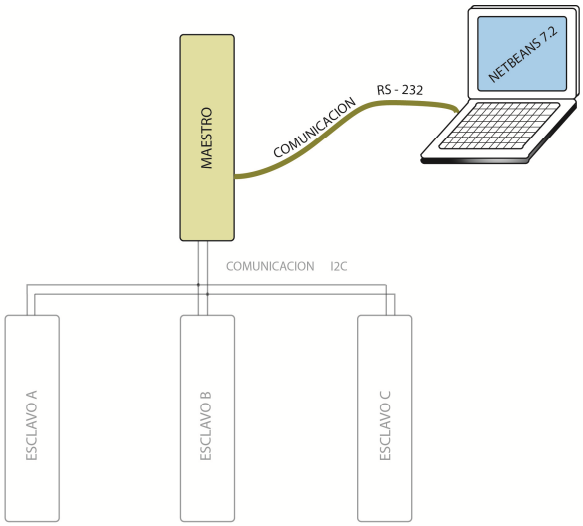

Figura 3.26 Comunicación Controlador - Computador

Como se explicó anteriormente, el microcontrolador *"Maestro"* es el encargado de manejar las señales provenientes desde el *teach pendant,* a través de una función que reconoce que tecla fue presionada y según esto ejecuta la acción correspondiente. Para el diseño del programa se tomó en cuenta esto, de tal manera que el programa trabaja como un control normal, es decir, se ejecuta la acción seleccionada mientras se esté presionando el botón, cuando se suelta el mismo se envía la señal de paro a los motores.

Para poder comunicar el microcontrolador con el computador se necesita de un circuito integrado que sea capaz de convertir las señales TTL a RS232 y viceversa, para el caso de este proyecto se utiliza el MAX232. Este circuito integrado para su correcto funcionamiento necesita de varios capacitores, los que se presentan en el esquema de la figura 3.27 a continuación:

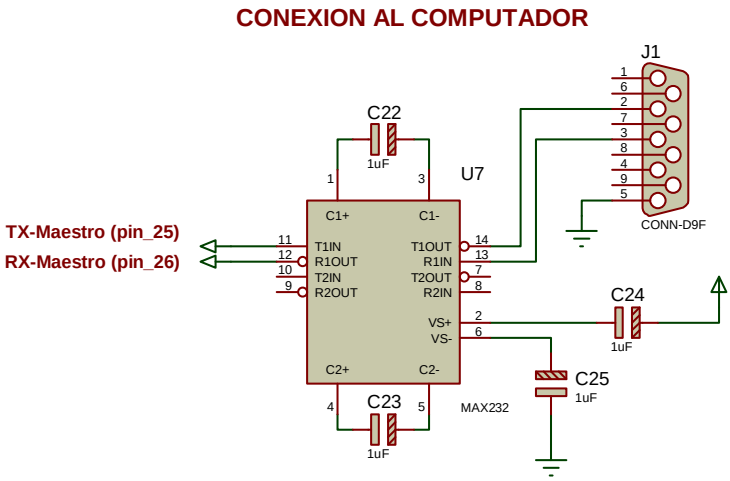

Figura 3.27 Conexión MAX232

Para el manejo de la comunicación serial, tanto desde el computador como desde el microcontrolador se va a explicar por separado la inicialización, la transferencia de datos computador – microcontrolador y microcontrolador - computador.

- **Inicialización de la comunicación**

Desde el microcontrolador solamente se necesita definir en un inicio del programa que se va a trabajar con la comunicación RS232, y cinco parámetros básicos como son la velocidad de transferencia, cantidad de bits por trama, bit de paridad, pin de transmisión y pin de recepción. Todo esto mediante la siguiente directiva:

#use rs232(baud=2400,bits=8,parity=N,xmit=PIN_C6,rcv=PIN_C7)

Una vez definida esta directiva en el programa del microcontrolador, este se encuentra listo para trabajar con la comunicación RS232.

Para trabajar con el puerto serial desde *NETBEANS* hay varios puntos que se debe tomar en cuenta, uno de ellos es que por defecto no vienen las librerías

necesarias para el manejo del mismo. Motivo por el cual se debe descargar estos archivos de internet y colocarlos en la carpeta de Java. Otro punto importante es el manejo del puerto propiamente, por facilidad, se crea una clase "SERIAL" que va a ser la encargada de esta función. A través de la función principal, se abre y configura el puerto:

```
public Serial(CommPortIdentifier portId) {

      //Si el puerto no está abierto se intenta abrir
      try {
         serialPort = (SerialPort) portId.open("TESIS_SERIALApp", 2000);
         System.out.println("\rPuerto Abierto\r");

      //Configuracion de buffer de entrada y salida
         inputStream = serialPort.getInputStream();
         outputStream = serialPort.getOutputStream();
         System.out.println("\rpaso 1\r");

      //Listener asociado al puerto
            serialPort.addEventListener(this);
         System.out.println("\rpaso 2\r");

        //Notifica cuando hay datos disponibles en el buffer de entrada
         serialPort.notifyOnDataAvailable(true);
         System.out.println("\rpaso 3\r");

      //Se configura parámetros de transmisión
       serialPort.setSerialPortParams(2400,SerialPort.DATABITS_8,SerialPort.STOPBI
       TS_1,SerialPort.PARITY_NONE);
         System.out.println("\rPuerto configurado\r");
      } catch (Exception e) {}

      //Se crea y activa el Thread, el cual se va a quedar esperando en el puerto hasta
que
      //existan datos
      readThread = new Thread(this);
      readThread.start();
}
```

Para completar la inicialización y definición del puerto desde la clase principal que va a ser la pantalla de presentación, al inicializar todos sus componentes se debe comprobar que el puerto con el que se va a trabajar esté disponible y crear un objeto de la clase "SERIAL" para el manejo del mismo como se muestra a continuación:

```
public Principal() {
       initComponents();

        //Se obtiene la lista de puertos disponibles en el computador
       portList = CommPortIdentifier.getPortIdentifiers();

      //Se verifica que exista el Puerto COM5 con el que se desea trabajar
       while (portList.hasMoreElements()) {
          portId = (CommPortIdentifier) portList.nextElement();
          if (portId.getPortType() == CommPortIdentifier.PORT_SERIAL) {

     //Si encuentra el Puerto requerido crea un objeto Serial
             if (portId.getName().equals("COM5")) {
                 reader = new Serial(portId);
                 System.out.println("Creando un "+portId.getName());
              }
          }
       }
       actualizar.start();
   }
```

- **Transferencia de datos Computador -> Controlador**

Para el envío de datos desde el computador hacia el controlador se creó una función dentro del programa principal, que permita transmitir el dato directamente como se muestra a continuación.

```
private void enviar(int dato_enviar) {
   try {
       reader.outputStream.write(dato_enviar);
   }   catch (IOException ex) {  }
 }
```

El recibo de los datos en el microcontrolador, se realiza a través de la interrupción que se genera cuando se tienen datos en el *buffer (INT_RDA),* después de leer el dato se llama a la función “lectura” que es la que reconoce cual tecla fue presionada y ejecuta la acción correspondiente. Dentro del programa principal del maestro, cuando se selecciona como opción de manejo el computador, se activa ésta interrupción.

```
#int_RDA
```

```
void  RDA_isr(void)
{
  recibo=getc();
  if(recibo==0) {
    disable_interrupts(INT_TIMER1);
  }
  else{
    enable_interrupts(INT_TIMER1);
  }
  lectura(recibo);
}
```

- **Transferencia de datos Controlador -> Computador**

Para el envío de datos desde el controlador hacia el computador se utiliza una codificación para identificar lo que significa el dato enviado:

Tabla 3.17 Comandos enviados desde el controlador hacia el computador

COMANDO	DESCRIPCIÓN	ENVIADO EN
Teach	Indicador de manejo desde el *teach pendant*	Cuando se activa la selección de manejo desde el *teach pendant*
Nada	Indicador de ningún método de manejo activo	Cuando se pone en posición neutra el selector del método de manejo
Pc	Indicador de manejo desde el computador	Cuando se activa la selección de manejo desde el computador
Axxxxx	Posición de la articulación 1, enviada en pulsos contados	Se envía cada 100 [ms] mediante interrupción de *Timer1*
Bxxxxx	Posición de la articulación 2, enviada en pulsos contados	Se envía cada 100 [ms] mediante interrupción de *Timer1*

Cxxxxx	Posición de la articulación 3, enviada en pulsos contados	Se envía cada 100 [ms] mediante interrupción de *Timer1*
Vx	Velocidad seleccionada	Cuando existe un cambio de velocidad

El código de programa que envía los datos desde el microcontrolador es:

printf ("cadena de caracteres")

El recibo de datos en el computador se realiza mediante un evento que se produce cuando el *buffer* de entrada contiene datos, este evento se lee desde la clase "SERIAL".

```
public void serialEvent(SerialPortEvent event) {
      switch(event.getEventType()) {
      case SerialPortEvent.DATA_AVAILABLE:
         String readBuffer_1 ="";
         BufferedReader B=new BufferedReader(new InputStreamReader
(inputStream));
         try {
            readBuffer_1= B.readLine();
            System.out.print(readBuffer_1+"\r");
            switch(readBuffer_1.charAt(0)){
               case 'A':   a1=readBuffer_1.substring(1);
                  break;
               case 'B':   a2=readBuffer_1.substring(1);
                  break;
               case 'C':   a3=readBuffer_1.substring(1);
                  break;
               case 'D':   a4=readBuffer_1.substring(1);
                  break;
               case 'E':   a5=readBuffer_1.substring(1);
                  break;
               case 'n':   indicador=readBuffer_1;
                  break;
               case 't':   indicador=readBuffer_1;
                  break;
               case 'p':   indicador=readBuffer_1;
                  break;
               case 'v':   vel=readBuffer_1.substring(1);
                  break;
```

```
                default:
                    break;
                }
                System.out.print(readBuffer_1.charAt(0)+"\t");
                System.out.print(readBuffer_1.substring(1) +"\r");
            } catch (IOException e) {}
            break;
        }
    }
```

Mediante un temporizador se actualizan las variables de interés en la pantalla principal del programa.

```
    private class Tiempo implements ActionListener{
        public void actionPerformed(ActionEvent C){

jTextField1.setText(String.valueOf(impdec.format((Float.parseFloat(reader.a1)*0.005)-
162.5)));
 jTextField2.setText(String.valueOf(impdec.format((Float.parseFloat(reader.a2)*0.005)-
162.5)));

jTextField3.setText(String.valueOf(impdec.format((Float.parseFloat(reader.a3)*0.005)-
162.5)));

jTextField4.setText(String.valueOf(impdec.format((Float.parseFloat(reader.a4)*0.005)-
162.5)));

jTextField5.setText(String.valueOf(impdec.format((Float.parseFloat(reader.a5)*0.005)-
162.5)));
            jLabel8.setText(reader.indicador);
        }
        public Tiempo() {
        }
    }
```

3.2.2. PROGRAMA DESARROLLADO EN NETBEANS 7.2

Como se explicó anteriormente, para el movimiento del robot se desea poner una representación del *teach pendant* en pantalla, que maneje los mismos gráficos y lenguaje. De esta manera se va a colocar botones para el movimiento de tres de las articulaciones, y, apertura y cierre del *gripper*, así como también para el incremento y disminución de la velocidad, indicadores de la posición actual del robot, indicadores de modo de manejo (*Teach Pendant* o Computador) y la velocidad.

Para el envío de información desde el computador hacia el controlador lo que se tomó en cuenta fue el funcionamiento del *teach pendant*, es decir, que se ejecute el movimiento de un motor a la vez y que este se produzca únicamente mientras se mantenga presionado el botón correspondiente. Para lograr esto en el programa se trabaja con los eventos *MousePressed* y *MouseReleased* de los botones existentes en el entorno de desarrollo.

- El evento *MousePressed* se produce el instante que se presiona el botón, en el que se envía el código de la tecla seleccionado hacia el controlador.

- El evento *MouseReleased* se produce el instante que se deja de presionar el botón, en el que se envía el código “0” indicando que ninguna tecla se encuentra presionada.

Para los indicadores se utilizan temporizadores que son los encargados de realizar la actualización de los campos que contienen estos datos informativos cada cierto tiempo.

A continuación se presenta la pantalla principal del programa implementado, el cual permite, al iniciar, buscar todos los puertos de tipo serial en el computador. Una vez iniciado el programa se debe seleccionar el puerto con el que se desea trabajar y ejecutar la conexión correspondiente, caso contrario aparecerá un error cuando se desea presionar alguno de los botones existentes en pantalla. Como se puede apreciar en la pantalla, solo está disponible la manipulación de tres de las cinco articulaciones existentes, la velocidad y el *gripper* neumático con el que cuenta el robot. Además de esto se monitorea el modo de operación que está seleccionado en el controlador (computador o *teach pendant*).

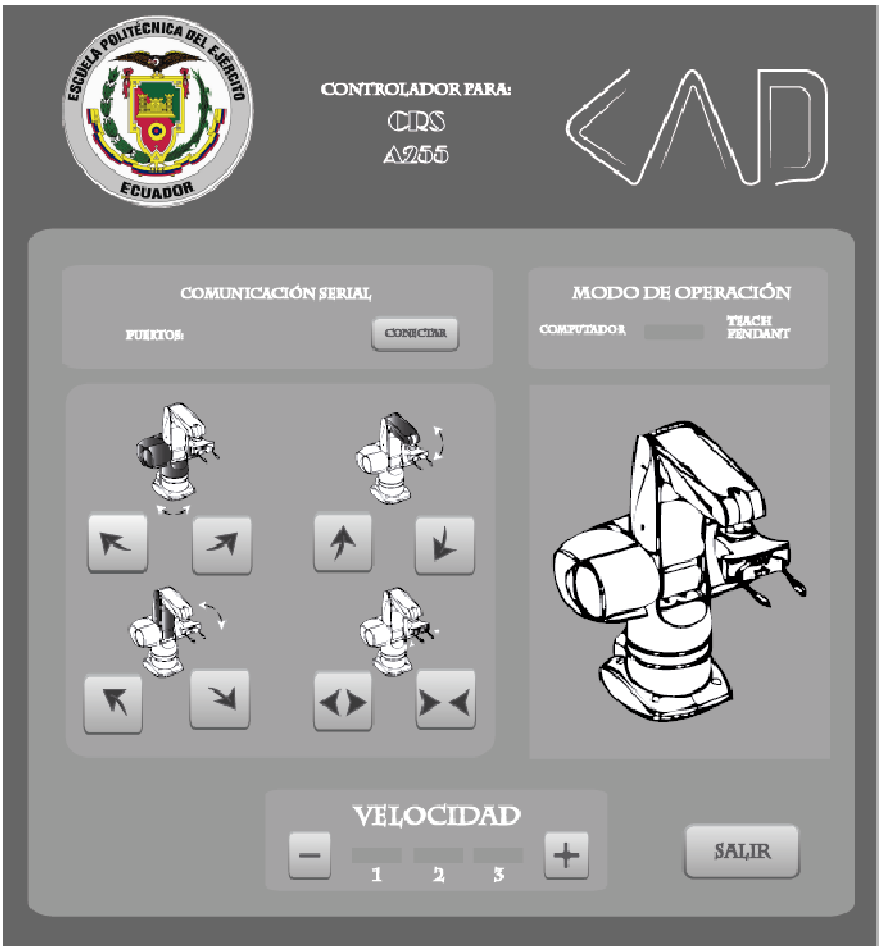

Figura 3.28 Pantalla de HMI

CAPITULO IV

IMPLEMENTACIÓN DEL SISTEMA DE CONTROL

4.1. IMPLEMENTACIÓN DEL HARDWARE

Una vez concluida con la etapa de diseño del controlador, se va a tratar sobre la implementación del sistema, es decir, la estructura física. Para lo cual se utilizó la herramienta ARES (Software de Edición y Ruteo Avanzado) la cual sirve para el diseño de placas de circuito impreso, es decir, permite la ubicación y edición de componentes, realizar el enrutamiento permitiendo editar generalmente, las capas superior *(Top Copper)*, e inferior (Bottom Copper).

Para la implementación lo primero que se tomó en cuenta es que este controlador es la primera fase de un proyecto que comprende varias etapas. Por esto se decidió separar el controlador en tarjeta de potencia, tarjeta de alimentación y tarjeta de control, de esta manera el controlador será de tipo modular. Se empezará por la descripción de las tarjetas impresas, para después explicar la estructura más grande, representada por la caja.

4.1.1. Consideraciones Para Diseño De Circuitos Impresos

Para realizar el circuito impreso hay que tener ciertas consideraciones de diseño, a fin de evitar que las interferencias electromagnéticas afecten el correcto funcionamiento del circuito, siendo estas:

- **Capacitor de Desacoplamiento [16]**

Un condensador de desacoplamiento es el encargado de suministrar la energía necesaria para absorber los picos de corriente a través de una pequeña impedancia de línea y al mismo tiempo se efectúa un cortocircuito a alta frecuencia para impedir la creación de señales parásitas indeseables de forma que, si existen, se derivan a masa.

Los valores ideales del capacitor de desacoplamiento se encuentran en valores entre 470 y 1000 ρF para circuitos integrados de mediana escala de integración (MSI), mientras que para los circuitos integrados de muy alta escala de integración (VLSI) como es el caso de microprocesadores y memorias RAM, los valores típicos son de 100 ρF.

Para los circuitos integrados que se utilizan en las tarjetas se trabaja con capacitores cerámicos de 100 ρF siguiendo las recomendaciones.

- **Consideraciones de las rutas (pistas) [17]**

Cuando se utilizan circuitos integrados lógicos, se debe cuidar que las pistas no tengan cambios abruptos de dirección ya que estos generan ondas estacionarias debido a las reflexiones, los ángulos rectos de las esquinas debe truncarse a 45°.

Para la separación que debe existir entre los espacios de soldadura de elementos (*PAD*), y las líneas de señales se utilizó la herramienta *Design Rule Manager* de ARES, esta es la encargada de verificar que existan las separaciones mínimas correspondientes para evitar interferencias entre líneas y *pad*.

[16] Balcells, J., Daura, F., Esparza, R., & Pallás, R. (1992). *INTERFERENCIAS ELECTROMAGNÉTICAS EN SISTEMAS ELECTRÓNICOS.* Alfaomega.
[17] Ott, H. W. (1988). *Noise Reduction Techniques in Electronic Systems.* John Wiley & Sons.

En las líneas se debe tomar en cuenta que por cada amperio que va a circular por la pista, esta debe tener 1mm de ancho. Y de igual manera, estas pistas deben mantener su ancho a través de todo el circuito impreso, para evitar inductancias indeseables que se pueden generar al cambiar el grosor de las líneas, especialmente al conectarse con los circuitos integrados.

- **Consideraciones de Alimentación** [18]

Con el objetivo de evitar acoplamiento por conducción en las tarjetas de circuitos impresos es recomendado utilizar tarjetas multicapa para establecer planos de masa y de alimentación. Otra de las recomendaciones indica que de ser posible se creen mallas de masa (tierra) para que las corrientes parásitas indeseables en los circuitos impresos puedan circular a través de esta malla, sin interferir en el funcionamiento del circuito.

Para la tarjeta de potencia presentada más adelante en este capítulo, se creó una malla de tierra para los circuitos encargados de los motores, frenos y solenoide, sin embargo, se separó la etapa del circuito de enclavamiento para evitar que las interferencias electromagnéticas que se puedan generar en los relés afecten el funcionamiento de los circuitos de potencia.

[18] Balcells, J., Daura, F., Esparza, R., & Pallás, R. (1992). *INTERFERENCIAS ELECTROMAGNÉTICAS EN SISTEMAS ELECTRÓNICOS.* Alfaomega.

4.1.2. Tarjeta De Control

La tarjeta de control es la que contiene todos los elementos encargados de las comunicaciones externas del controlador, manejo de los LCD utilizados, lectura del teclado, acondicionamiento de las señales de los *encoders* y envío de señales de control hacia la tarjeta de potencia. En la Figura 4.1 se presenta un gráfico con la ubicación de los elementos en la tarjeta.

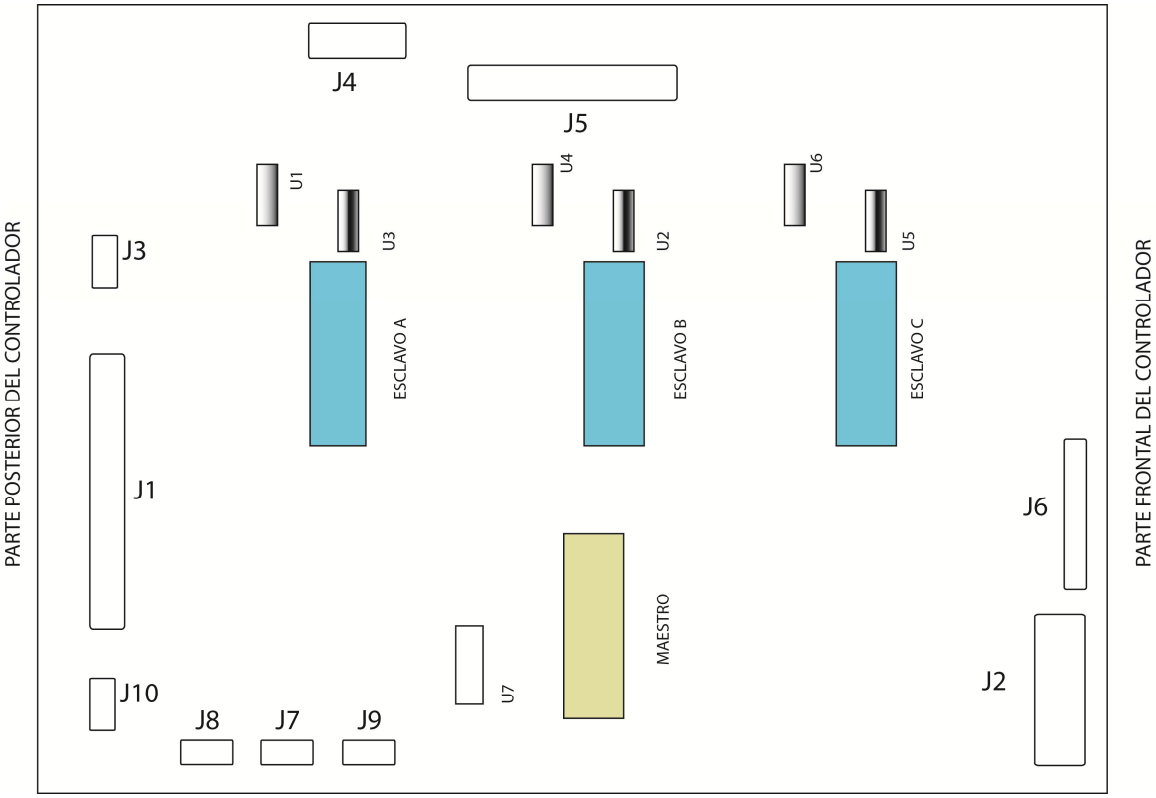

Figura 4.1 Ubicación de los elementos en la tarjeta de control

Tabla 4.1 Descripción de elementos en la tarjeta de control

Elemento	Descripción
J1	Señales provenientes del cable de realimentación, se utiliza para obtener las señales de los *encoders*.
J2	Conector DB25, utilizado para interactuar con el *teach pendant*
J3	Conectado al cable de realimentación, señal de referencia de tierra para la activación de la solenoide del *gripper*
J4	Entrada de alimentación, +5 VDC para el funcionamiento de los circuitos integrados
J5	Conector que envía las señales de control hacia la tarjeta de potencia
J6	Conector utilizado para el manejo del LCD ubicado en la caja del controlador
J7	Conector libre, tiene los puntos de conexión para la comunicación I_2C, +5 VDC y GND
J8	Conector utilizado para enviar las señales de activación de los frenos y para activar / desactivar el solenoide del *gripper*
J9	Conector utilizado para la comunicación serial, estas señales van a un conector RS232 ubicado en la caja del controlador
J10	Conector utilizado para energizar con +5 VDC y GND a los *encoders*
MAESTRO	Microcontrolador MAESTRO
ESCLAVOS A, B y C	Microcontroladores esclavos, encargados del control de movimiento de las articulaciones
U1, U4, U6	Compuertas inversoras con *Schmitt Trigger* (74LS14), utilizadas para el acondicionamiento de las señales de los

	encoders
U2, U3, U5	Compuertas lógicas AND (74HC08), utilizadas para enviar las señales de activación correspondientes a la tarjeta de potencia
U7	Circuito integrado MAX232, utilizado en la comunicación serial con el computador

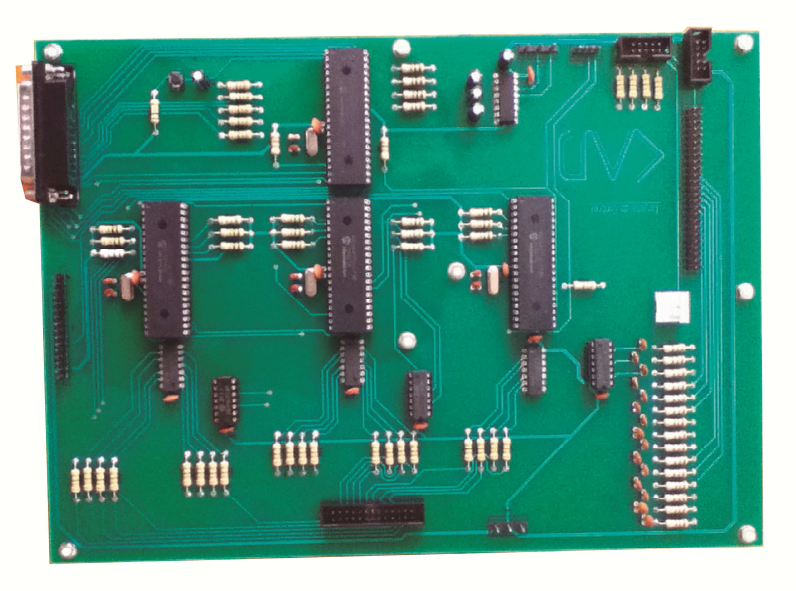

Figura 4.2 Tarjeta de Control

4.1.3. Tarjeta De Potencia

La tarjeta de potencia es la encargada de los movimientos de los motores y desactivación de los frenos, esta recibe las señales de control desde la correspondiente tarjeta y en función de esto activa los elementos necesarios para mover los motores. En la Figura 4.3 se presenta la ubicación de los elementos en la tarjeta.

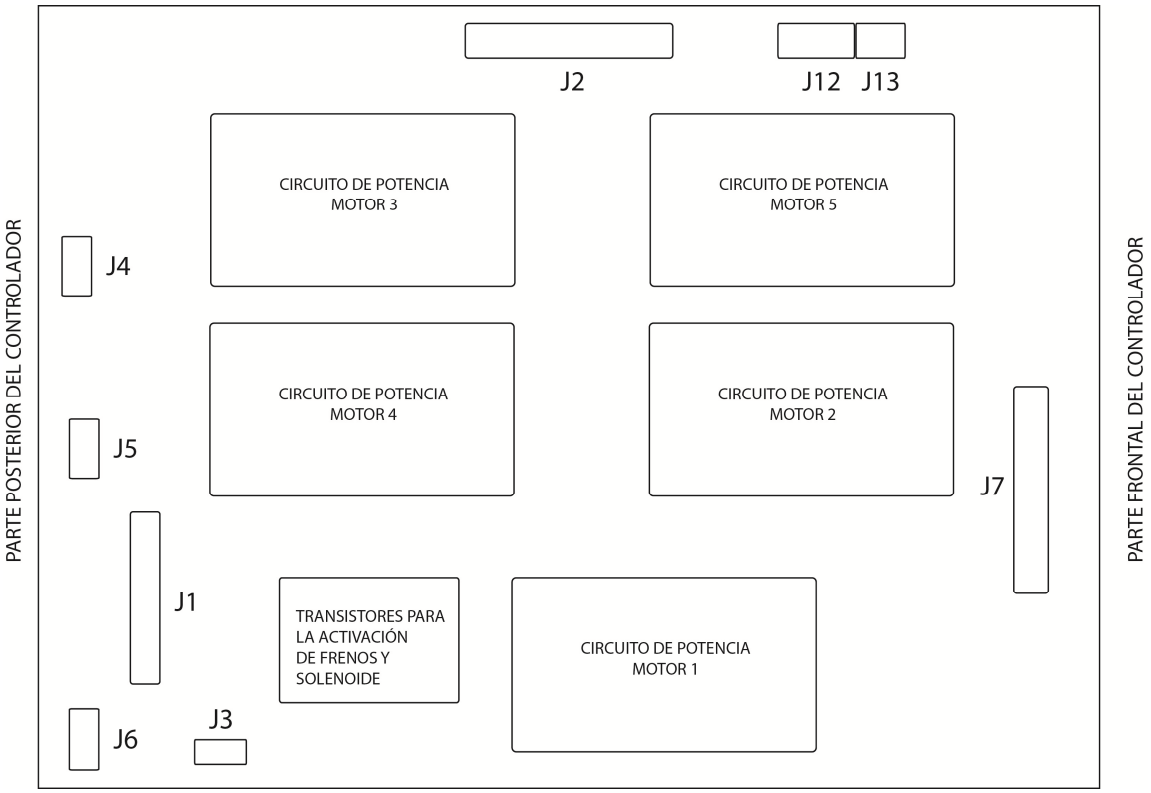

Figura 4.3 Ubicación de los elementos en la tarjeta de potencia

Tabla 4.2 Descripción de elementos en la tarjeta de potencia

Elemento	Descripción
J1	Conector utilizado para activar / desactivar los frenos
J2	Conector utilizado para recibir las señales de control desde la tarjeta correspondiente
J3	Conector utilizado para recibir las señales de activación de frenos y solenoide
J4	Conector utilizado para enviar la alimentación a los motores de las articulaciones 3 y 5
J5	Conector utilizado para enviar la alimentación a los motores de las

	articulaciones 2 y 4
J6	Conector utilizado para enviar la alimentación al motor de la articulación 1
J7	Conector utilizado para el circuito de enclavamiento
J12, J13	Conectores de alimentación, +5 VDC, +12 VDC, +35 VDC
CIRCUITOS DE POTENCIA	Circuitos que se detallan a continuación, encargados de formar un “PUENTE H” y la activación del mismo
TRANSISTORES PARA ACTIVACIÓN DE FRENOS Y SOLENOIDE	Se colocaron en grupo los transistores necesarios para la manipulación de frenos y *gripper*

En la Figura 4.3 se describe el circuito de potencia típico armado para el funcionamiento del “PUENTE H”, mientras que en la Figura 4.4 se detalla la ubicación de cada uno de los transistores y lo que controlan.

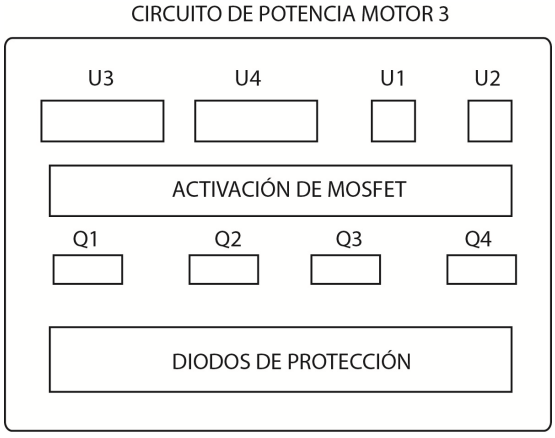

Figura 4.4 Ubicación de los elementos en el circuito de potencia

Tabla 4.3 Descripción de elementos en la tarjeta de potencia

Elemento	Descripción
U3	Driver para *MOSFET* del lado izquierdo superior del “PUENTE H”
U4	Driver para *MOSFET* del lado derecho superior del “PUENTE H”
U1	Optoacoplador para *MOSFET* del lado izquierdo inferior del “PUENTE H”
U2	Optoacoplador para *MOSFET* del lado derecho inferior del “PUENTE H”
Q1	MOSFET de lado izquierdo superior del “PUENTE H”
Q2	MOSFET de lado derecho superior del “PUENTE H”
Q3	MOSFET de lado izquierdo inferior del “PUENTE H”
Q4	MOSFET de lado derecho inferior del “PUENTE H”

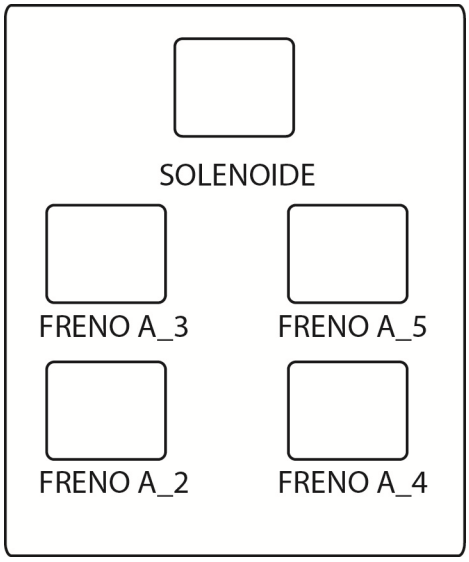

Figura 4.5 Ubicación de los elementos que controlan cada uno de los frenos y solenoide

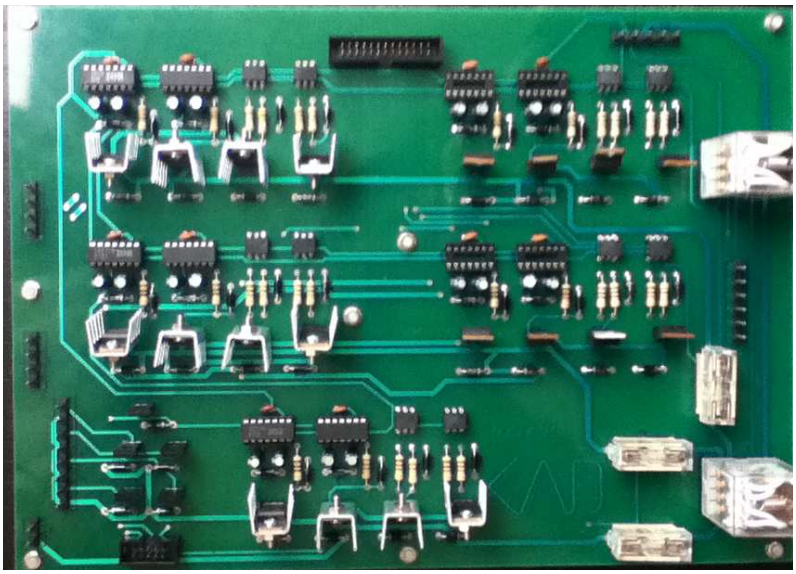

Figura 4.6 Tarjeta de Potencia

4.1.4. Etapa De Alimentación

En la etapa de alimentación se tienen varios elementos en los que se incluyen, la tarjeta correspondiente, el transformador y puentes de diodos. Los cuales se encuentran ubicados en la caja del controlador, en la Figura 4.7 se aprecia la tarjeta de alimentación.

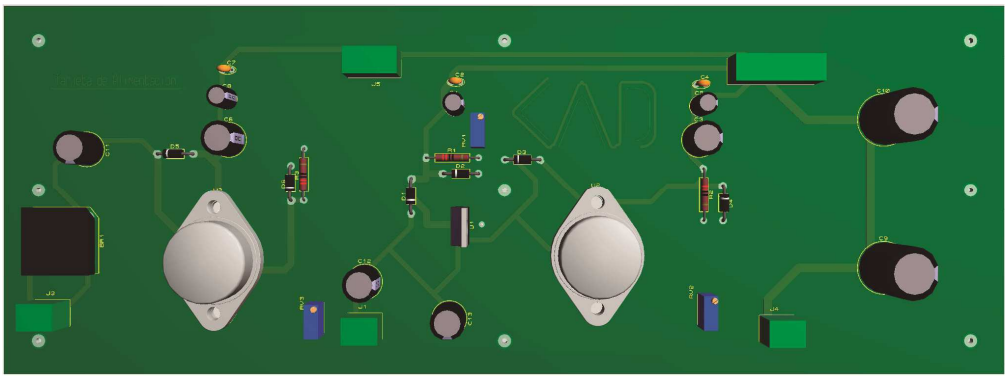

Figura 4.7 Tarjeta de Alimentación del controlador nuevo

4.1.5. Estructura Externa Del Controlador (Caja)

En la caja del controlador se tiene en el panel frontal, todos los elementos que permiten el manejo del sistema como tal, es decir, el selector de modo de operación, pulsador de *"ARM POWER"*, botón de paro de emergencia, LCD indicador, y el conector necesario para la comunicación con el *Teach Pendant.* Su ubicación se presenta a continuación en la Figura 4.8

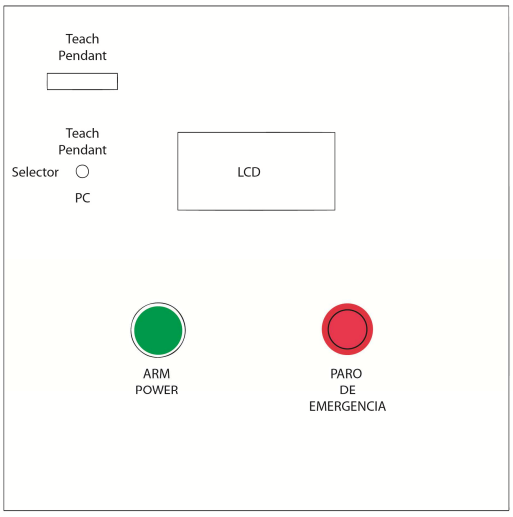

Figura 4.8 Vista Frontal del controlador nuevo

En el panel posterior se tiene el botón para encender el controlador, los conectores para los cables de potencia y de realimetación, así como dos conectores DB9: uno para la conexión RS232 con el computador y el otro para los frenos, esto se puede apreciar en la Figura 4.9.

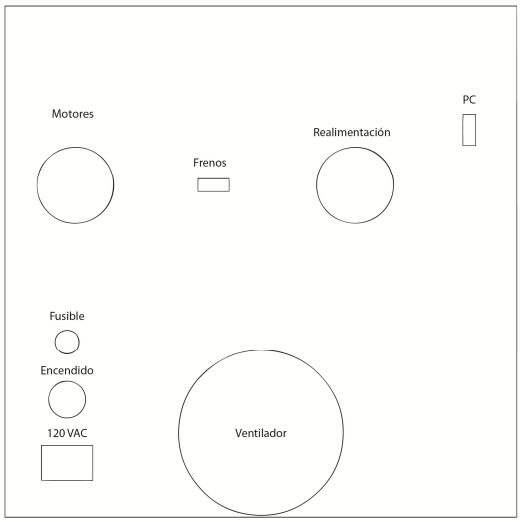

Figura 4.9 Vista Posterior del controlador nuevo

A continuación se presenta la parte frontal del controlador implementado:

Figura 4.10 Controlador Implementado

4.1.6. Teach Pendant

En el *teach pendant* se tiene un conector DB-25 en la parte inferior, el cual sirve para su conexión hacia el controlador. En su parte frontal se tiene un *display* indicador y el teclado que cuenta con 16 pulsadores para la manipulación de las 5 articulaciones, incremento/decremento de velocidad, *HOME*, y un pulsador extra para nuevas funciones que se deseen implementar.

Figura 4.11 *Teach Pendant*

A continuación en la Figura 4.12 se observa la conexión del controlador con el *teach pendant.*

Figura 4.12 Controlador y *Teach Pendant*

CAPITULO V

PRUEBAS Y RESULTADOS

5.1. PRUEBAS

Inicialmente se consideró la utilización de un Puente H en un circuito integrado (L298), el cual, acorde con las necesidades de los motores DC que se desean manejar soporta hasta 46 VDC, y 2 A máximo, sin embargo la potencia máxima que puede disipar es de 25 W, motivo por el cual no se pudo utilizarlo.

Las pruebas que se consideraron para el presente proyecto y sus correspondientes resultados se presentan a continuación:

- **Movimiento total del manipulador robótico**

Como primera prueba se realizó la comprobación de movimiento total de las articulaciones 1, 2 y 3, su correspondiente error en relación a la cantidad de pulsos que se deberían obtener para el movimiento, de extremo a extremo, de la misma. Esto obedece a la resolución que se tiene de los *encoders* incrementales en cada motor y su reducción con la siguiente fórmula:

$$Pulsos_{contados} = \frac{movimiento_{total}[°]}{resolucion\ del\ encoder[°]}$$

Para las articulación 1, 2 y 3 se presentan los resultados en las Tablas a continuación:

Tabla 5.1 Resultados de movimiento de Articulación 1

Velocidad	Movimiento		Pulsos Contados		Error	
	[°]	pulsos	CW	CCW	CW	CCW
1	350	70000	71340	72168	2%	3%
2	350	70000	70914	70980	1%	1%
3	350	70000	73019	71257	4%	2%

Tabla 5.2 Resultados de movimiento de Articulación 2

Velocidad	Movimiento		Pulsos Contados		Error	
	[°]	Pulsos	CW	CCW	CW	CCW
1	110	22000	21980	23098	4%	5%
2	110	22000	22031	23707	2%	8%
3	110	22000	20953	24160	-5%	10%

Tabla 5.3 Resultados de movimiento de Articulación 3

Velocidad	Movimiento		Pulsos Contados		Error	
	[°]	pulsos	CW	CCW	CW	CCW
1	130	26000	27890	28094	7%	8%
2	130	26000	26939	29030	4%	12%
3	130	26000	27052	28507	4%	10%

Para la sintonización de la constante de proporcionalidad, se consideró el cambio de ciclo de trabajo que se deseaba tener en función del error que se tiene. A continuación en las Tablas se presentan los resultados obtenidos con cada uno de los valores de constante de proporcionalidad sugeridos, tomando en cuenta que los valores de velocidad presentados, tanto de *SP* (valor deseado) como de error, están dados en pulsos contados cada 52.4 [ms]:

Tabla 5.4 Error con KP = 1/15

SP	Articulación 1		Articulación 2		Articulación 3	
	CW	CCW	CW	CCW	CW	CCW
100	20	15	30	15	25	30
250	35	25	35	20	40	35
400	60	40	50	20	45	35

Tabla 5.5 Error con KP = 1/20

SP	Articulación 1		Articulación 2		Articulación 3	
	CW	CCW	CW	CCW	CW	CCW
100	20	20	15	30	20	15
250	25	20	15	40	30	20
400	25	20	20	45	35	25

Tabla 5.6 Error con KP = 1/30

SP	Articulación 1		Articulación 2		Articulación 3	
	CW	CCW	CW	CCW	CW	CCW
100	10	15	30	30	10	20
250	10	10	45	50	15	15
400	15	20	45	60	15	25

- **Velocidad real**

En la Tabla 5.7 se presenta la velocidad real obtenida en cada una de las articulaciones con el control implementado, los valores están dados en [°/s], los mismos que obedecen a la fórmula:

$$w_{[°/seg]} = \frac{\text{pulsos\_contados}}{52.4 * 10^{-3}} * \frac{360\,°}{1000\ \text{pulsos}} * relación\ de\ reduccion$$

Tabla 5.7 Velocidad aproximada de las Articulaciones 1, 2 y 3 [°/s]

Velocidad	Articulación 1		Articulación 2		Articulación 3	
	CW	CCW	CW	CCW	CW	CCW
1	8,68	8,87	8,30	10,02	8,11	8,59
2	22,33	22,90	23,19	21,95	22,81	24,43
3	36,26	37,79	35,97	39,89	36,07	40,36

CAPITULO VI

CONCLUSIONES Y RECOMENDACIONES

6.1. CONCLUSIONES

- Se logró implementar un controlador, capaz de controlar tres grados de libertad del manipulador robótico CRS-A255, cumpliendo con los requerimientos propuestos para el presente proyecto.

- El manual con el que se desarrolló el proyecto es de un manipulador robótico similar al CRS-A255, por lo que no cuenta con un detalle acorde en su totalidad al robot, y es aquí donde se encontraron varios problemas, uno de estos ejemplos es el de los frenos, motivo por el cual se modificó la conexión de los mismos. En cuanto a los motores se constató que estos son de Corriente Contínua (DC) y no Servo Motores como se indica en la guía de usuario.

- Las señales provenientes de los enconders incrementales con los que cuentan los motores tiene demasiado ruido para poder leerlas directamente por el microcontrolador. Un circuito RC ayuda a atenuar el ruido, sin embargo, es necesaria la utilización de una compuerta inversora con histéresis adicional para obtener la forma de onda deseada.

- Para la selección de los elementos de potencia es necesario tomar en cuenta, además de las características de voltaje, corriente y potencia, la frecuencia de conmutación a la que pueden trabajar, en el caso particular de la modulación por ancho de pulso (PWM) se refiere al mínimo ciclo de trabajo que se va a utilizar

- Se logró implementar un programa para el manejo del robot desde el computador, el mismo que fue desarrollado en NETBEANS 7.2, que trabaja sobre la plataforma Java. Este entorno de trabajo permite el manejo de puertos del computador, y por ser de conocimiento general entre los estudiantes de la carrera permitirá a los estudiantes el desarrollo la adición de nuevos componentes, acciones, etc cumpliendo así con la necesidad de ser un proyecto escalable.

6.2. RECOMENDACIONES

- Este proyecto se presenta como una solución escalable, por lo que se recomienda que exista continuidad en el mismo, es decir, que se continúe con las siguientes etapas como pueden ser generación de trayectorias, almacenamiento de puntos en el espacio, etc.

- Dar un mejor mantenimiento a las partes mecánicas del manipulador robótico

- Dedicar una clase a la explicación detallada, desmontaje y montaje, del manipulador robótico a fin de que los estudiantes de la carrera conozcan físicamente los actuadores, sensores, reducciones y transmisiones presentes en el mismo.

REFERENCIAS BIBLIOGRÁFICAS

- Thermo CRS. (2002). *CataLyst-5 Robot System User Guide.*

- International Rectifier. (s.f.). *International Rectifier (IRF).* Obtenido de AN-978 HV Floating MOS-Gate Driver ICs: http://www.irf.com/technical-info/appnotes/an-978.pdf

- García Saquicela, M. R. (2010). *DISEÑO DEL SISTEMA DE CONTROL DEL BRAZO ROBÓTICO CRS A255 UTILIZANDO LA PLATAFORMA KINETIX DE ALLEN BRADLEY.* Escuela Politécnica del Ejército, Quito.

- García Breijo, E. (2008). *Compilador C CCS y Simulador PROTEUS Para Microcontroladores PIC* (Primera Edición ed.). México: Alfaomega Grupo Editor.

- *Foro: Todo PIC*. (s.f.). Obtenido de www.todopic.com.ar/foros/index.php?topic=23978.0

- Balcells, J., Daura, F., Esparza, R., & Pallás, R. (1992). *INTERFERENCIAS ELECTROMAGNÉTICAS EN SISTEMAS ELECTRÓNICOS.* Alfaomega.

- Ott, H. W. (1988). *Noise Reduction Techniques in Electronic Systems.* John Wiley & Sons.

ÍNDICE DE FIGURAS

CAPITULO II

Figura 2.1 Descripción de Articulaciones del Manipulador CRS A255 18
Figura 2.2 Vista lateral del volumen de trabajo del manipulador robótico CRS A255 20
Figura 2.3 Vista superior del volumen de trabajo del manipulador robótico CRS A255 21
Figura 2.4 Ubicación de los motores de las articulaciones 2, 3, 4 y 5 23
Figura 2.5 Forma de onda de la salida del *encoder* en rotación horaria 24
Figura 2.6 Disposición de conectores en la parte posterior del brazo robótico 277
Figura 2.7 Conector para Alimentación de potencia de los motores [5] 288
Figura 2.8 Cable umbilical de Alimentación de Potencia de los motores 28
Figura 2.9 Conector para Realimentación de los *encoders* 3030
Figura 2.10 Cable umbilical de Realimentación 30
Figura 2.11 Conector para Servogripper 33
Figura 2.12 Conector para *Gripper Neumático* [9] 34
Figura 2.13 Organización Funcional de un robot 35
Figura 2.14 Controlador C500 37
Figura 2.15 Esquema básico de una fuente de alimentación 37
Figura 2.16 Etapa de Alimentación del controlador C500 38
Figura 2.17 Modulo de entrada de alimentación del controlador C500 [10] 388
Figura 2.18 Tarjeta de Aislamiento del controlador C500 40
Figura 2.19 Tarjeta de acoplamiento de los motores y el controlador C500 40
Figura 2.20 Tarjeta de control 41
Figura 2.21 *Teach Pendant* 422
Figura 2.22 Captura de pantalla inicial del software *ROBCOMM3* 433

CAPITULO III

Figura 3.1 Configuración Puente H 50
Figura 3.2 Conexión Driver IR2110 [12] 52
Figura 3.3 Opto-acoplador 4n33 54
Figura 3.4 (a) Estructura de los motores del robot CRS A255 55
Figura 3.5 Circuito de potencia para un freno 56
Figura 3.6 Circuito de potencia para un motor 58
Figura 3.7 Configuración Maestro-Esclavo 61
Figura 3.8 Condiciones de Inicio y Finalización de Comunicación I_2C 61

Figura 3.9 Conectores en lo que está dividido el conector de realimentación 67
Figura 3.10 Circuito de acoplamiento 70
Figura 3.11 Circuito de enclavamiento..... 70
Figura 3.12 Circuito para el *Teach Pendant* 72
Figura 3.13 Esquema de la comunicación I2C 75
Figura 3.14 Esquema de conexión al *Teach Pendant* 77
Figura 3.15 Articulaciones que controla el Esclavo A..... 80
Figura 3.16 Conexiones del Esclavo A 80
Figura 3.17 Lazo de Control..... 83
Figura 3.18 Comparación de velocidad del motor [rpm] vs. Pulsos contados cada 52.4 [ms]... 84
Figura 3.19 Articulaciones que controla el Esclavo B..... 866
Figura 3.20 Conexiones del Esclavo B 87
Figura 3.21 Articulaciones que controla el Esclavo B..... 92
Figura 3.22 Conexiones del Esclavo C 92
Figura 3.23 Esquema de fuente no regulada..... 98
Figura 3.24 Esquema de fuente regulada..... 99
Figura 3.25 Fuentes de Alimentación de 5V, 12V y 35V 100
Figura 3.26 Comunicación Controlador - Computador 102
Figura 3.27 Conexión MAX232 103
Figura 3.28 Pantalla de HMI..... 1030

CAPITULO IV

Figura 4.1 Ubicación de los elementos en la tarjeta de control 114
Figura 4.2 Tarjeta de Control 116
Figura 4.3 Ubicación de los elementos en la tarjeta de potencia 117
Figura 4.4 Ubicación de los elementos en el circuito de potencia..... 118
Figura 4.5 Ubicación de los elementos que controlan cada uno de los frenos y solenoide..... 119
Figura 4.6 Tarjeta de Potencia..... 120
Figura 4.7 Tarjeta de Alimentación del controlador nuevo 120
Figura 4.8 Vista frontal del controlador nuevo 121
Figura 4.9 Vista posterior del controlador nuevo 122
Figura 4.10 Controlador Implementado 122
Figura 4.11 Teach Pendant..... 123
Figura 4.12 Controlador y Teach Pendant..... 123

ÍNDICE DE TABLAS

CAPITULO II

Tabla 2. 1 Especificaciones físicas y ambientales del manipulador robótico CRS A255 [1] 19
Tabla 2.2 Descripción de movimiento y velocidad de las articulaciones del robot [2] 21
Tabla 2.3 Descripción del Motor DC [3] 22
Tabla 2.4 Motores de las articulaciones 2, 3, 4 y 5 23
Tabla 2.5 *Encoders* de las articulaciones 2, 3, 4 y 5 24
Tabla 2.6 Relación de reducción de engranajes [4] 26
Tabla 2.7 Detalle de conectores del cable umbilical de alimentación de potencia de los motores[6] 29
Tabla 2.8 Detalle de conectores del cable umbilical de realimentación de los *encoders* y solenoide[7] 311
Tabla 2.9 Detalle del conector para *Servo Gripper* [8] 33
Tabla 2.10 Características del transformador 39
Tabla 2.11 Características del controlador C500 [11] 41
Tabla 2.12 Requerimientos de energía para el controlador 44

CAPITULO III

Tabla 3.1 Tabla comparativa BJT vs MOSFET 49
Tabla 3.2 Tabla comparativa de MOSFET´S 50
Tabla 3.3 Características del driver IR2110 51
Tabla 3.4 Características del driver IR2110 52
Tabla 3.5 Características del transistor C1162 56
Tabla 3.6 Señales requeridas para activación del motor 57
Tabla 3.7 Opciones para la Directiva de la Comunicación I_2C 63
Tabla 3.8 Funciones usadas para la Comunicación I_2C 64
Tabla 3.9 Distribución de señales del Conector de Realimentación 67
Tabla 3.10 *Teach Pendant* 73
Tabla 3.11 Dirección de los esclavos 75
Tabla 3.12 Tabla de lectura de datos desde el *Teach Pendant* 78
Tabla 3.13 Descripción de los pines del microcontrolador 81
Tabla 3.14 Descripción de los pines del Esclavo B 877
Tabla 3.15 Descripción de los pines del Esclavo C 93

Tabla 3.16 Detalle de Alimentación 97
Tabla 3.17 Comandos enviados desde el controlador hacia el computador 106

CAPITULO IV

Tabla 4.1 Descripción de elementos en la tarjeta de control 115
Tabla 4.2 Descripción de elementos en la tarjeta de potencia 117
Tabla 4.3 Descripción de elementos en la tarjeta de potencia 119

CAPITULO V

Tabla 5.1 Resultados de movimiento de Articulación 1 124
Tabla 5.2 Resultados de movimiento de Articulación 2 124
Tabla 5.3 Resultados de movimiento de Articulación 3 124
Tabla 5.4 Error con KP = 1/15..... 125
Tabla 5.5 Error con KP = 1/20..... 125
Tabla 5.6 Error con KP = 1/30..... 126
Tabla 5.7 Velocidad aproximada de las Articulaciones 1, 2 y 3 [°/s] 126