DISEÑO DE UN AUTÓMATA PROGRAMABLE PARA AMBIENTE ACADÉMICO

LUIS CARLOS CALVO PULGAR

VÍCTOR HUGO TOVAR POLO

JUAN DAVID COSSIO GUZMAN

UNIVERSIDAD DE LA COSTA CUC

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA ELECTRÓNICA

BARRANQUILLA

2013

DISEÑO DE UN AUTÓMATA PROGRAMABLE PARA AMBIENTE ACADÉMICO

LUIS CARLOS CALVO PULGAR

VÍCTOR HUGO TOVAR POLO

JUAN DAVID COSSIO GUZMAN

Director RUBEN DARÍO SÁNCHEZ DAMS

Trabajo de Grado presentado como requisito parcial para optar al título de:
INGENIERO ELECTRONICO

UNIVERSIDAD DE LA COSTA CUC

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA ELECTRÓNICA

BARRANQUILLA

2013

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

_____

_____

Firma del Jurado

_____

Firma del Tutor

_____

Firma del Revisor

Barranquilla, Junio de 2013

CARTA DE ACEPTACIÓN

AGRADECIMIENTOS

Queremos agradecer principalmente a Dios por el regalo de la vida y permitirnos tener la bendición de estudiar y terminar nuestro pregrado academicamente. A nuestros familiares por ser un apoyo en todo momento tanto en lo económico como en lo personal. A nuestros profesores y amigos de universidad , con los cuales aprendimos y comenzamos nuestro proceso de formacion como profesionales. A nuestro tutor de proyecto de grado Ruben Sanchez Dams quien nos apoyó en este desarrollo, y siempre creyó en que lograríamos terminarlo. A todas las personas que colaboraron durante el desarrollo de cada etapa del proyecto, en especial a Julieth y a Heider.

RESUMEN

En el proyecto explicado aquí, Se implementó un Controlador Lógico Programable (PLC) de tipo académico, también conocido como Autómata programable, basándose en el desarrollo de investigaciones anteriores de PLC con microcontroladores. Además, se desarrolló un sistema de adquisición de datos basado en PC, mediante estándar de comunicación USB 2.0. El sistema está compuesto de un *hardware* de adquisición y un *software* para el PC, comunicados entre sí a través del puerto USB.

El desarrollo del proyecto se llevó a cabo en dos fases:

Para la fase inicial se diseñó e implementó el *hardware* para la adquisición de datos del proceso mediante la utilización de sensores digitales “todo o nada” y la integración de los microcontroladores PIC 16F84 y 18F4550 de Microchip. Se incluyó también el desarrollo de una aplicación basada en LabView llamada “Sistema Supervisorio: Mezclado de sustancias”, para el control y supervisión de los recursos del *hardware*. El desarrollo del sistema de adquisición de datos se basa en la arquitectura propia del PLC, tal como se planteó en los objetivos.

Para finalizar esta fase, se procedió a realizar el diseño de los circuitos impresos mediante el método serigráfico para las tarjetas de cada uno de los módulos que conforman el sistema, con el fin de disponer de buena calidad para el prototipo.

Para la segunda fase se hizo un mapeo de memoria mediante el Sistema Supervisor y se accedió a los temporizadores, entradas y salidas que posee la Unidad Central de Procesamiento (CPU), desde el Computador Personal (PC). Para el desarrollo del *Firmware* del microcontrolador PIC 18F4550 se utilizaron compiladores y herramientas como MPLAB, PIC C Compiler, Proton y Micro C. También se utilizaron las librerías de desarrollo de Microchip, el MCHPUSB Driver para la descripción del Vendor y el descriptor VID para Windows. Para cargar los

archivos .HEX, leer y escribir en la memoria del microcontrolador se utilizó la aplicación PDFSUSB, suministrada por Microchip.

Al finalizar las fases 1 y 2, se procede a crear las plantillas de programación por medio del lenguaje PIC C Compiler para el *Firmware* de la CPU (el microcontrolador), haciendo uso de diferentes métodos con el fin de poder reconfigurar nuevamente los parámetros.

Luego, se realizaron las pruebas de funcionamiento que anteriormente se habían llevado a cabo en el simulador ISIS Proteus, excelente por su simulación para sistemas microcontrolados y su soporte para los modelos existentes, y posteriormente la del sistema supervisor en el PC.

El sistema actual también cuenta con la posibilidad de conectar el equipo a una red de comunicación tipo industrial, gracias a que posee puertos COM, con capa física RS232 y el RS485 externos.

ABSTRACT

In the project explained here, was implemented Programmable Logic Controller (PLC) academic, also known as Programmable automaton, based on previous research development of PLC with microcontrollers. In addition, was developed a data acquisition system based on PC using USB 2.0 communication standard. The system consists of an acquisition hardware and software for the PC, interconnected via the USB port.

The development of the project was carried out in two phases:

For the initial phase was designed and implemented the hardware for process data acquisition using digital sensors "all or nothing" and the integration of the and 18F4550 PIC 16F84 microcontrollers from Microchip. It also included developing a LabView based application called "Supervisory System: Mixed substance" to the control and supervision of hardware resources. The development of the data acquisition system architecture is based on PLC itself, as outlined in the objectives.

To complete this phase, was proceeded to the design of printed circuit boards using the screen printing method for cards of each of the modules that comprise the system, in order to have good quality for the prototype.

For the second phase was mapped memory using the Supervisor System and accessed the timers, inputs and outputs that owns the Central Processing Unit (CPU), from the Personal Computer (PC). The Firmware development for microcontroller PIC 18F4550 were used compilers and tools as MPLAB, PIC C Compiler, Proton and Micro C. Were also used the Microchip development libraries, the MCHPUSB Driver for describing the Vendor and the VID descriptor for Windows. To upload files. HEX, read and write memory of the microcontroller is used PDFSUSB application, provided by Microchip.

At the end of phases 1 and 2, was proceeded to create templates using the programming language for PIC C Compiler CPU Firmware (microcontroller), using different methods in order to be possible reconfigure the parameters.

Then were made performance tests that were previously conducted in the simulator Proteus ISIS, excellent for its microcontrolled systems simulation and its support for existing models, and then the supervisor system on the PC.

The current system also has the ability to connect the equipment to an industrial-type communication network, because it has COM ports with RS232 and RS485 physical layer.

TABLA DE CONTENIDO

LISTA DE GRAFICOS

Pag.

Pag.

INTRODUCCIÓN

Los controladores lógicos programables (PLC) son ampliamente utilizados en el sector industrial, llegando a ser pilares de las líneas de producción de muchas compañías. Este hecho hace que para un estudiante de ingeniería electrónica sea fundamental estudiar el funcionamiento de estos equipos en un ambiente académico, donde se pueden simular situaciones y comportamientos de diversos tipos, preparándose de forma integral para afrontar el ambiente laboral.

En el presente proyecto se muestra el diseño y realización de unautomata programable de ambiente academico, como material de apoyo para que los estudiantes analicen el funcionamiento interno y principales aplicaciones de este tipo de dispositivos. El sistema está compuesto de un *hardware* de adquisición y un *software* en el ordenador, comunicados a través del puerto USB. Una vez finalizado su desarrollo, se mostrarán los resultados obtenidos mediante un artículo científico producto de la investigación realizada y las ventajas del diseño didáctico académico presentado.

El desarrollo del presente sistema, se basó puramente en la arquitectura de un autómata programable y cumple plenamente los objetivos para su posterior integración con el proyecto presentado por el Ing. Rubén Sánchez, en su libro "*Controlador lógico programable, una mirada interna: implementación académica del dispositivo*"[1].

Dicho automata programable se compone de varios módulos o tarjetas, una denominada CPU, tarjetas de entradas y salidas digitales (todo o nada) y tarjetas de salidas digitales. También se incluye un sistema supervisorio en el ordenador que se utiliza para el control y supervisión de los recursos de la CPU y se muestra una aplicación sencilla a nivel industrial

1. OBJETIVOS

1.1. OBJETIVO GENERAL

Desarrollar un Autómata Lógico Programable de tal manera que se pueda usar en ambientes pedagógicos y se logre analizar su comportamiento interno.

1.2. OBJETIVOS ESPECÍFICOS

- Diseñar e implementar un *hardware* didáctico que facilite la comprensión de la estructura interna de un PLC en una ambiente académico.
- Desarrollar un *software* que permita controlar y supervisar las variables de un proceso bajo estudio.
- Implementar un sistema demostrativo de un proceso en donde se aplique el prototipo desarrollado.

2. JUSTIFICACIÓN

Los Autómatas Programables (AP) o Controladores Lógicos Programables (PLC) nacen como una respuesta a la necesidad de contar con cadenas de producción automatizadas que pudieran seguir la evolución de las técnicas de entrada de producción a nivel industrial. Desde ese entonces,Los AP han aportado mucho a la industria ya que son herramientas de fácil manejoque mediante *software* de programación facilitan mucho el trabajo en los laboratorios, independientemente de la ubicación del proceso, además de que mejoran el sistema y permiten la realización de pruebas en tiempo real.

El principal motivo para la realización de este proyecto fue llevar a ejecución un módulo de comunicaciones que se integraría al PLC de ambiente académico formulado teóricamente por el Ing. Rubén Sánchez [1], pero en vista de la necesidad creciente de formar profesionales completamente capacitados en el tema, se decidió implementar el proyecto en su totalidad.

Una de las necesidades que se debió cumplir durante el desarrollo el prototipo del PLC didáctico, fue desarrollar una interfaz entre el operador y el PLC, que la memoria de usuario pudiese leer con las instrucciones que definen las secuencias de control. Normalmente esta interfaz se lleva a cabo a través de un *software* instalado en Computadores personales [2]. Dependiendo del tipo de PLC el equipo de programación produce unos códigos de instrucción directamente ejecutables por el procesador o bien un código intermedio, que es interpretado por un programa residente en el procesador (*firmware*)[3]. En este proyecto se utilizó una forma alternativa para crear el *firmware* y el *software* necesario utilizando el lenguaje de programación de alto nivel C para el diseño de un autómata con sistemas embebidos básicos (microcontroladores), junto con la electrónica básica, la de potencia, los aspectos digitales y análogos.

3. PLANTEAMIENTO DEL PROBLEMA

El ambiente laboral actual para un ingeniero electrónico es un espacio exigente, especialmente a nivel industrial donde los profesionales se enfrentan a diversos retos diariamente. En el ambiente academico es necesario que los estudiantes aprendan sobre el funcionamiento interno de los controladores lógicos programables con el fin de conocer sus principales aplicaciones, generar valor agregado y por tanto ser más competentes. Un profesional que conoce cuál es la estructura interna de un plc, tendrá plena capacidad para implementar cualquier método de programación que le parezca conveniente. Actualmente no se enseña en el programa de ingenieria electronica la estructura interna de un controlador logico programable.

4. CAPÍTULO 1: INTRODUCCIÓN A PLC

4.1. DEFINICIÓN DE PLC.

Según la IEC 61131: “Un autómata programable (AP) o PLC es una máquina electrónica programable diseñada para ser utilizada en un entorno industrial hostil, que posee una memoria programable para el almacenamiento interno de instrucciones orientadas al usuario, para implantar soluciones específicas tales como funciones lógicas, secuenciales, temporizadas, recuentos y funciones aritméticas, con el fin de controlar mediante entradas y salidas ya sean digitales o analógicas diversos tipos de máquinas o procesos industriales” [4].

Un PLC consta de las siguientes partes:

- Unidad Central de Proceso (CPU)
- Memoria
- Fuente de Alimentación
- Reloj en tiempo real
- Puerto de entradas
- Puerto de salidas

4.1.1. CPU – Unidad Central De Proceso.

Como su nombre lo indica, es la unidad central de procesamiento. En ella se llevan a cabo todos los procesos lógico-aritméticos del sistema. Usualmente está conformada por un microcontrolador. Antiguamente se usaban microcontroladores de 8bits, pero hoy en día, para aplicaciones más robustas, se utilizan también microcontroladores de 16 y 32 bits.

Por ser un microcontrolador, cuenta con temporizadores, interrupciones, conversiones ADC y DAC, comunicaciones seriales sincrónicos y asincrónicos, etc. [5]

4.1.2. Memoria

Usualmente se incluye una memoria externa al microcontrolador que puede ser EEPROM y/o FLASH, que hace las veces de banco de datos para la lecto/escritura de datos.

Esta memoria se utiliza para almacenar el programa (funciones, variables, estados, tiempos) desarrollado que se encargará de controlar las entradas y las salidas del PLC. En esta memoria no se almacena la programación del microcontrolador, puesto que éste viene programado de fábrica con un programa que permite administrar las entradas, las salidas y los temporizadores del PLC [5].

4.1.3. Fuente De Alimentación

El PLC tiene una entrada de voltaje de 220VAC o 110VAC o en algunos casos 24VDC, dependiendo de la necesidad, y adicionalmente tiene salidas de 24VAC o DC con el fin de alimentar todos sus módulos y/o sensores [5].

4.1.4. Reloj En Tiempo Real

El generador de frecuencia de referencia para el sistema automatizado, necesita establecer la variable tiempo que es indispensable para poner en marcha temporizadores y contadores [5].

4.1.5. Puerto De Entradas

Las entradas de un PLC debe ser aisladas para proteger al microcontrolador de altos voltajes y en algunos casos, por configuración del sistema, pueden permitir ajustar la intensidad de la entrada, es decir, la corriente de entrada varía gracias a un potenciómetro instalado en el circuito. Adicionalmente, las entradas de un PLC pueden ser analógicas o digitales [5].

4.1.6. Puerto De Salidas

Como en las entradas, las salidas pueden ser analógicas o digitales, y pueden ser de cualquiera de los siguientes tipos:

- 120 VAC.
- 24 VDC.
- 12 – 48 VAC.
- 12 – 48 VDC.
- 5V DC (TTL).
- 230 VAC.
- Otros.

Esto se debe a que sus circuitos internos permiten convertir niveles lógicos a niveles de voltaje externos, y efectivamente, también suelen utilizarse opto-acopladores para proteger a los componentes electrónicos [5].

4.2. COMUNICACIONES

El PLC es un sistema autónomo, sin embargo, no puede autoprogramarse. Para ello es necesaria una interfaz con el usuario y esa la provee el puerto RS232, un cable serial y un computador o un programador portátil.

En un proceso industrial, muchas veces es necesario utilizar más de un PLC o establecer comunicación con diferentes dispositivos inteligentes como termostatos, captadores de radiación solar, sistemas de control de fluidos (agua, gas, aire), motores, detectores de intrusión, cámaras frigoríficas, sistemas de ascensores, calefacción, etc. Con ese fin, fue inventado el bus de campo dedicado a la GTB (gestión técnica del edificio), el cual ofrece la posibilidad de cablear o precablear numerosos equipos inteligentes a bajo costo, teniendo en cuenta que hay tantos protocolos, como fabricantes de dispositivos [5].

4.3. SOFTWARE

Indispensable tanto para programar el PLC, como para monitorearlo. Aquí es donde se unen la informática, las redes y los PLC y Sistemas SCADA [5].

En las siguientes imágenes se muestran los componentes habituales de un sistema automatizado:

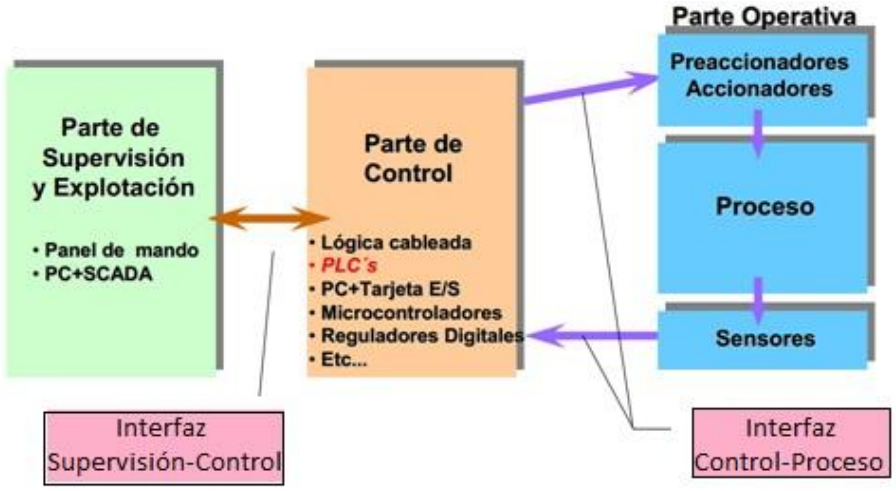


Imagen 1 Diagrama de bloques del sistema automatizado

En la Imagen 1, aparece el Diagrama de bloques del sistema automatizado donde se pueden distinguir las diferentes etapas de las que está compuesto un sistema SCADA.

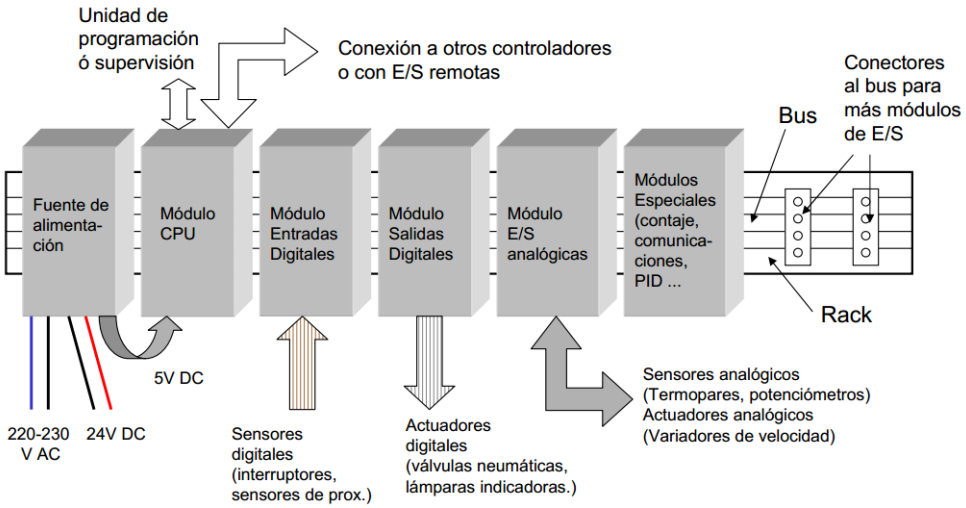


Imagen 2 Arquitectura típica del autómata programable (AP) [6].

En la imagen 2 se distinguen las diferentes etapas en un autómata programable y la interacción entre ellas.

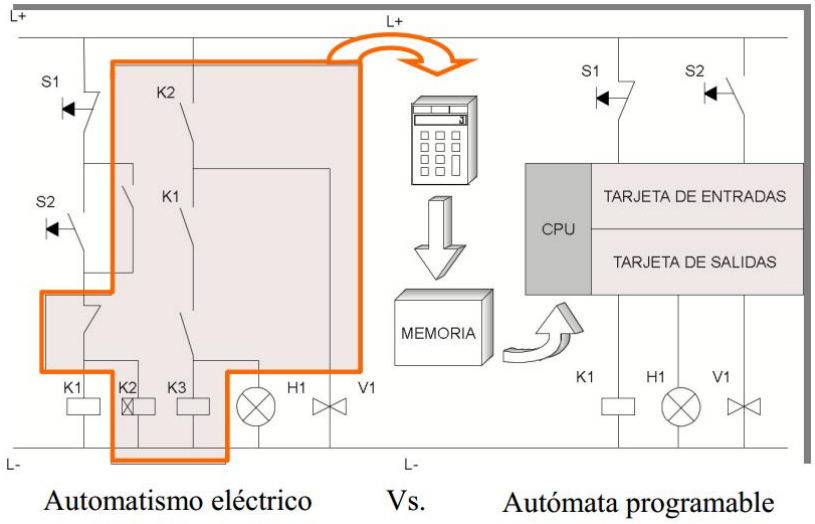


Imagen 3 Concepto gráfico del PLC [6].

La imagen 3 compara el funcionamiento de un PLC frente a su equivalente en lógica cableada.

4.4. LENGUAJES DE PROGRAMACIÓN DEL PLC

Desde el punto de vista del Procesador, un programa es un conjunto de instrucciones o preposiciones bien definidas que le dicen lo que tiene que hacer. Cada instrucción le indica qué operación realizará a continuación, de dónde obtendrá los datos que necesita para realizarla y dónde guardará los resultados de la operación. Desde el punto de vista del usuario, un programa son las especificaciones de un conjunto de operaciones que debe llevar a cabo el computador para lograr resolver una determinada tarea. Un programa se escribe en un lenguaje de programación y a la actividad de expresar un algoritmo en forma

de programa se le denomina programación. A menudo, el lenguaje de programación se denomina *software* de programación cuándo se emplea un término genérico, a fin de distinguirlo del *hardware* [7].

Con el objetivo de estandarizar estas representaciones, se ha establecido una norma internacional IEC 1131-3 que se encarga de los lenguajes de programación [7], cuya estructura se puede apreciar en la Imagen 4.

Esta norma contempla dos tipos de lenguajes de programación:

- **Lenguajes gráficos**: diagrama de escalera (Ladder, LD) y diagrama de bloques funcionales (FBD)
- **Lenguajes literales**: lista de instrucciones (IL) y texto estructurado (ST).

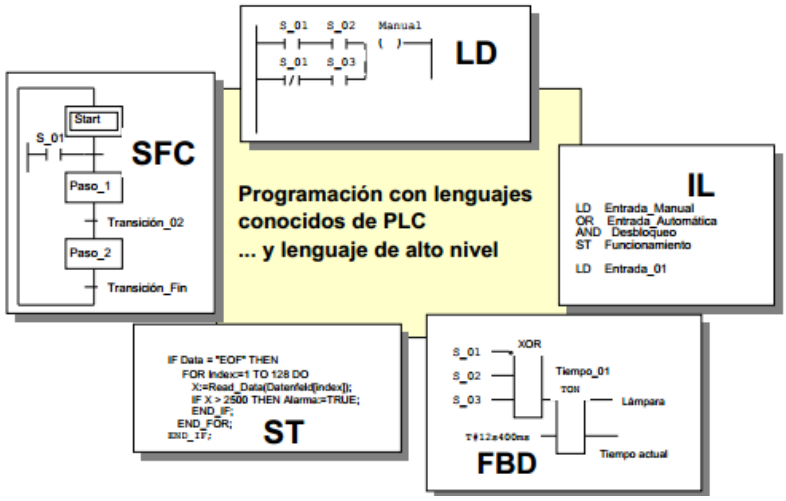


Imagen 4 Estructura de los lenguajes de programación [8].

5. CAPÍTULO 2: BASES TEÓRICAS

5.1. HARDWARE

El término *hardware* se refiere a todas las partes tangibles de un sistema informático; sus componentes son: eléctricos, electrónicos, electromecánicos y mecánicos. Son cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico involucrado; contrariamente, el soporte lógico es intangible y es llamado *software* [9]. El término, aunque sea lo más común, no solamente se aplica a las computadoras; del mismo modo, también un robot, un teléfono móvil, una cámara fotográfica o un reproductor multimedia poseen *hardware* y *software*.

Al hablar de *hardware*, se puede diferenciar entre el básico, estrictamente necesario para el funcionamiento normal del equipo, y el complementario, que realiza funciones específicas de acuerdo a la aplicación. Por ejemplo, un sistema informático básico se compone de una unidad central de procesamiento (UCP/CPU), encargada de procesar los datos, uno o varios periféricos de entrada, que permiten el ingreso de la información y uno o varios periféricos de salida que posibilitan dar salida (normalmente en forma visual o auditiva) a los datos procesados [9].

5.2. MICROCONTROLADORES

Un microcontrolador es un circuito integrado o chip que incluye en su interior las tres unidades funcionales de una computadora: CPU, Memoria y Unidades de E/S. Además de ésto, también posee otros recursos como conversores análogo digital (ADC), módulo de comparación de pulso, comunicación serie asíncrona UART, etc [10].

Los microcontroladores son utilizados en muchísimas aplicaciones, para ello en el mercado hay muchas opciones y marcas. Las marcas más representativas son Motorola, Atmel, Hitachi, Intel, Texas Instruments, entre otras. Microchip ha venido ganando popularidad en el mercado con sus gamas de microcontroladores PIC tipo RISC.

5.2.1. PIC 18F4550

Microchip posee microcontroladores para toda aplicación, entre ellos las series 10 y 12 que corresponden a la gama baja, la serie 16 y 17 que corresponden a la gama media; y la serie 18 que corresponde a la gama alta.

La gama 18 posee muchos más recursos que la anterior gama, puesto que incluyen más capacidad de memoria de datos y de programa, más temporizadores, operación de la CPU a mayor velocidad, posibilidad de utilizar más fuentes de interrupciones, etc [11].

A continuación se mencionarán las principales características de esta gama de microcontroladores:

5.2.1.1. Características generales

Los 18fxx5x vienen en encapsulados de 28 y 40 pines, con igual distribución de pines que las anteriores gama de microcontroladores. Poseen memoria de programa de 32kB distribuida en 16 bancos de memoria que pueden almacenar hasta 16.384 instrucciones de palabra única.

Esta familia tiene más recursos físicos, y de igual manera, muchos más registros de propósitos especiales para el control de los mismos. También tienen muchas fuentes de interrupciones entre las que se destaca la del control de bus USB, utilizando 10 registros para su operación [11].

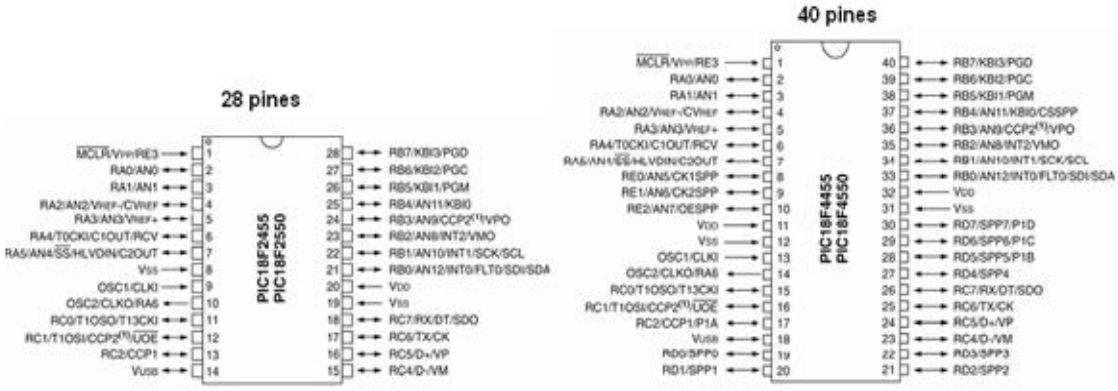


Imagen 5 Tipos de encapsulado para los micro controladores 18Fxx5x [12]

5.2.1.2. Configuración

Una característica importante de la gama de los 18Fxxxx es el oscilador. Éste posee un módulo multiplicador de frecuencia llamado PLL, que hace que cualquier cristal válido compatible (4Mhz, 8Mhz, 12Mhz, 16Mhz, 20Mhz, 24Mhz, 40Mhz ó 48Mhz) trabaje a una frecuencia mayor a su valor, tal como se muestra en la imagen 12. Es decir, se puede tener un módulo trabajando a 48Mhz y la CPU operando a 4Mhz.

La familia de dispositivos PIC18FX455/X550 contiene una interfaz serie compatible con el SIE (serial interface engine-máquina con comunicación serie) USB “full-speed” (2.0) y “de baja velocidad” (1.0) que permite la comunicación rápida entre cualquier dispositivo USB y el microcontrolador PIC.

El SIE puede interconectarse directamente al USB, utilizando el transmisor-receptor interno, o puede conectarse a través un transmisor-receptor externo. El PIC tiene un regulador interno de 3,3V para accionar el transmisor-receptor interno en aplicaciones de 5V.

Se han incluido algunas características especiales en el *hardware* para mejorar el funcionamiento. Se proporciona memoria de puerto dual en la memoria de datos del dispositivo (RAM del USB) para tener acceso directo a la memoria desde el núcleo del microcontrolador y desde el SIE. También se proporcionan unos búffer para que el programador elija libremente el final de la memoria dentro del espacio de la RAM del USB. Existe un puerto paralelo para transmitir datos grandes, por ejemplo al puerto paralelo, y se ha proporcionado la ayuda de transferencia ininterrumpida de volúmenes de datos grandes, por ejemplo datos síncronos, a los buffer de memoria externos [11].

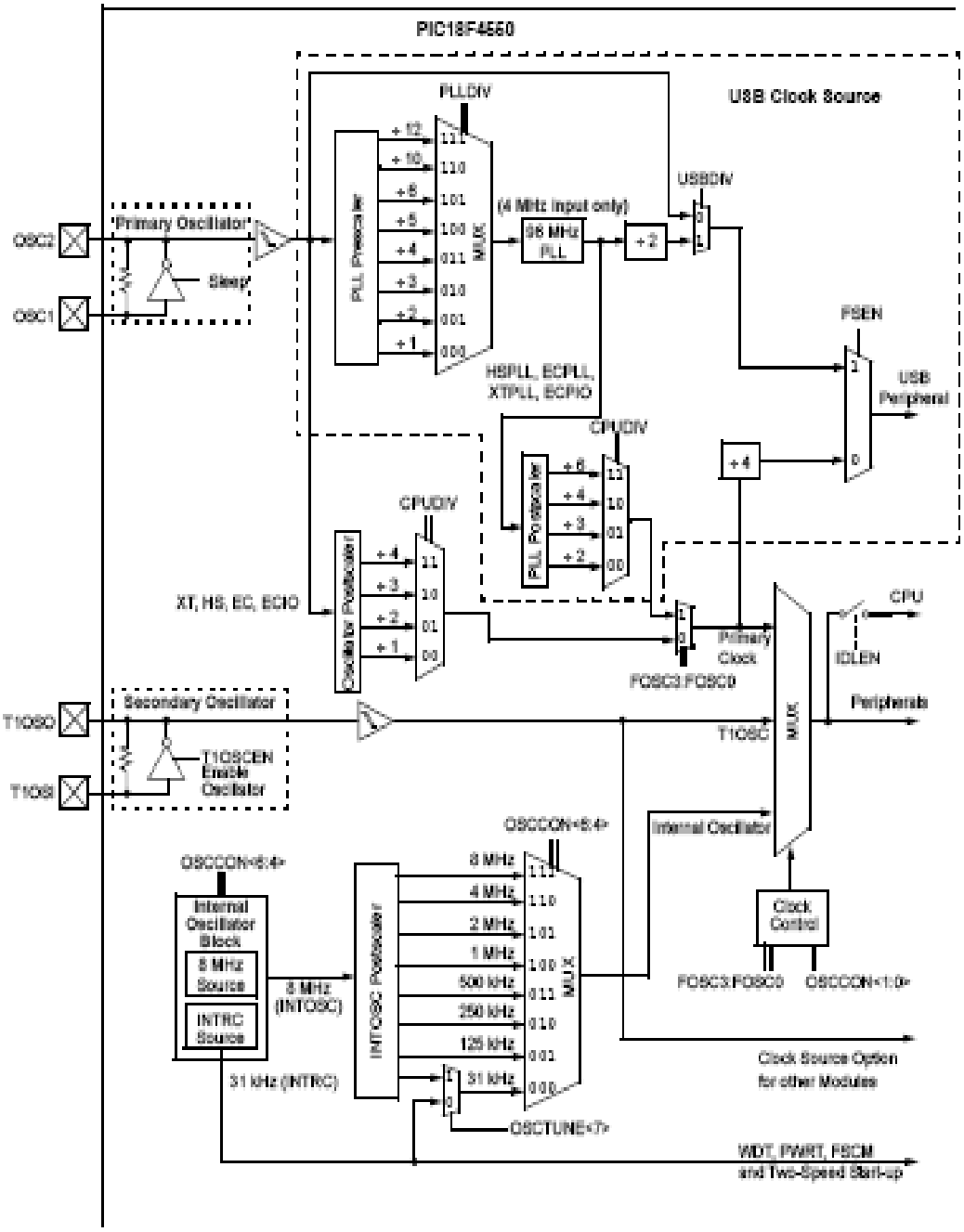


Imagen 6 Diagrama de bloque del oscilador [12]

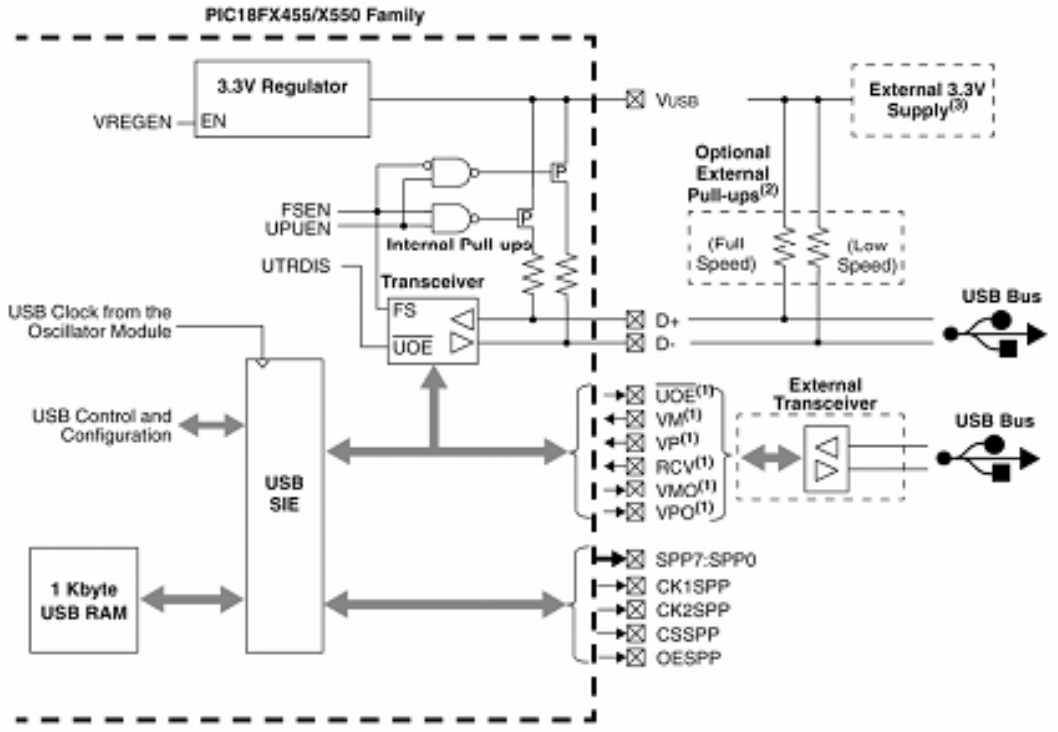


Imagen 7 Manejo de puertos del PCI18F4550 [12]

5.3. ELECTRÓNICA DE CONTROL

La electrónica es la rama de la física y especialización de la ingeniería, que estudia y emplea sistemas cuyo funcionamiento se basa en la conducción y el control del flujo microscópico de los electrones u otras partículas cargadas eléctricamente.

Utiliza una gran variedad de conocimientos, materiales y dispositivos, desde los semiconductores hasta las válvulas termoiónicas. El diseño y la construcción de circuitos electrónicos para resolver problemas prácticos forman parte de la electrónica y de los campos de la ingeniería electrónica, electromecánica y la informática en el diseño de *software* para su control. El estudio de nuevos dispositivos semiconductores y su tecnología se suele considerar una rama de la física, más concretamente, de ingeniería de materiales [13].

La electrónica desarrolla en la actualidad una gran variedad de tareas. Los principales usos de los circuitos electrónicos son el control, el procesado, la distribución de información, la conversión y la distribución de la energía eléctrica. Esta investigación se basa en la rama de la electrónica de control.

Los sistemas de control son aquellos dedicados a obtener la salida deseada de un sistema o proceso. En un sistema general se tienen una serie de entradas que provienen del sistema a controlar, llamado planta, y se diseña un sistema para que, a partir de estas entradas, modifique ciertos parámetros en el sistema planta, con lo que las señales anteriores volverán a su estado normal ante cualquier variación [14].

Hay varias clasificaciones dentro de los sistemas de control. Atendiendo a su naturaleza son analógicos, digitales o mixtos; atendiendo a su estructura (número de entradas y salidas) puede ser control clásico o control moderno; atendiendo a su diseño pueden ser por lógica difusa, redes neuronales. La clasificación principal de un sistema de control es de dos grandes grupos, los cuáles son [14]:

Sistema de lazo abierto: Sistema de control en el que la salida no tiene efecto sobre la acción de control.

- Se caracteriza porque la información o la variable que controla el proceso circulan en una sola dirección desde el sistema de control al proceso.
- El sistema de control no recibe la confirmación de que las acciones se han realizado correctamente.

Sistema de lazo cerrado: Sistema de control en el que la salida ejerce un efecto directo sobre la acción de control.

- Se caracteriza porque existe una relación de realimentación desde el proceso hacia el sistema de control a través de los sensores.
- El sistema de control recibe la confirmación si las acciones ordenadas han sido realizadas correctamente

5.4. INTERFACES DE COMUNICACIÓN

Con la interfaz de comunicación se establece un medio donde se puede comunicar el usuario y el equipo, y las diferentes tarjetas con la computadora en caso que se requiera.

Para lograr ésto se utilizan varias técnicas y herramientas de comunicación, tal como lo son:

- Buses de comunicación serie asíncrona
- RS-232
- Buses de comunicación serie síncrona
- Protocolo I2C
- Protocolo SPI
- Buses de comunicación paralela
- USB (Explicado con mayor profundidad en la sección 6.5)

5.4.1. Buses De Comunicación Serie Asíncrona

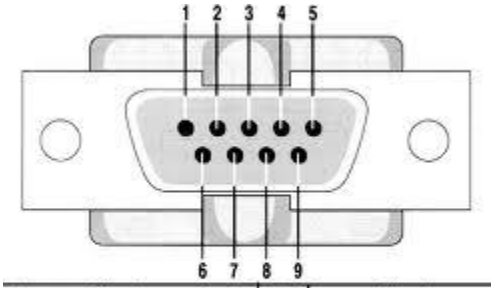
Están constituidos por buses a dos hilos, uno para la transmisión y otro para la recepción. En la CPU está implementado el bus USB y el protocolo RS232, a continuación se explica la implementación de cada bus.

5.4.2. RS-232

La CPU tiene implementado dos puertos de comunicación serie RS232, uno principal y otro auxiliar. Son utilizados para comunicar la CPU con la computadora y principalmente para agregar el PLC a una red de tipo industrial como la MODBUS o PROFIBUS.

Las líneas del COM auxiliar se encuentran conectadas al pin RC0 y RC1 del PIC y al circuito integrado MAX232 que es un conversor de protocolo de TTL a RS232 y viceversa. En la línea TX y RX, transmisión y recepción respectivamente, se encuentran diodos LED (D7, D8) que cambian su estado cuando se está haciendo la comunicación. Para la implementación de este puerto del microcontrolador, se elabora una rutina de código especial.

El COM principal utiliza el *hardware* UART que corresponde al pin RC6 y RC7 del PIC, además de estar conectado al MAX232 también está en la capacidad de cambiar la capa física a niveles RS485, cambiando la posición del jumper para elegir la capa que se quiera utilizar. Éste también posee diodos LED (D5, D6) que indican el paso de datos [15].



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Imagen 8 Descripcion de pines de un conector DB9 para RS232. [15].

5.4.3. Buses De Comunicación Serie Síncrona

Los protocolos I2C y SPI se utilizan para comunicar la tarjeta con otros sistemas (pantallas LCD, memorias, sensores, etc) que soporten estos mismos protocolos.

5.4.4. Protocolo I2C

El I2C es un protocolo serial utilizado para la comunicación entre periféricos. Para llevar a cabo este protocolo se utiliza el *hardware* que trae implementado el micro controlador MSSP (Master Synchronous Serial Port). La línea SCL (reloj) corresponde al pin RB1 y el pin SDA (dato) al pin RB0. Además se a reservados dos línea mas para el control del protocolo.

La CPU ya trae implementadas las resistencias de polarización que exige el protocolo.

5.4.5. Protocolo SPI

Igualmente que el I2C, este utiliza el *hardware* MSSP que corresponde a los pines RA5 de la línea SS, RB0 de la línea SDA, RB1 de la línea SDL y RC7 de la línea SDO.

La CPU solo trabaja en modo maestro cuando la línea SS no se utiliza. Cabe resaltar que este protocolo no se puede utilizar en conjunto con el RS232 ya que comparten la línea SDO.

5.4.6. Buses De Comunicación Paralela

Es un sistema que está compuesto por un bus de datos, un bus de dirección y un bus de control. Se utiliza para comunicar otros tipos de dispositivos de forma paralela como puede ser la tarjeta de entradas y salidas digitales o pantallas LCD.

5.5. USB

En la gran mayoría de los ordenadores actuales se ha eliminado el periférico RS-232, que era el más utilizado para la trasferencia de datos con un dispositivo externo, y han adoptado como único protocolo de comunicación el bus USB.

Imagen 9 Conectores USB Tipo A y B [16]

La importancia del bus USB para el proyecto radica en que es el utilizado por el *hardware* de adquisición para la transmisión de datos con el ordenador. A continuación se explican los aspectos más importantes del estándar.

- La versión actual del USB es el USB 3.0 compatible con sus versiones anteriores. Esta versión permite velocidades hasta de 480Mbps ("High speed"), se define como un modelo único de cableado y conectores, permitiendo que a través del mismo se lleve a cabo la transferencia de datos tanto síncrona como asíncrona y proporcione la alimentación del dispositivo periférico que a través de él se conecte; permite el conectado y desconectado en caliente de aparatos (operación plug&play) sin la necesidad de reiniciar el computador, para que este sea reconocido. [11]
- La familia de dispositivos PIC18FX455/X550 contiene una interfaz serie compatible con el SIE (serial interface engine, máquina con comunicación serie) USB “full-speed” (3.0) y “de baja velocidad” (1.0) que permite la

comunicación rápida entre cualquier dispositivo USB y el micro controlador PIC.

- La CPU es capaz de soportar el protocolo USB gracias al micro controlador PIC18f4550. Este bus tiene principalmente dos objetivos, el primero es cargar el programa compilado (archivo.HEX) desde la computadora a la CPU y el segundo servir de comunicación en un sistema SCADA, entre el ordenador y la CPU, como se puede ver en la imagen 10.
- El *hardware* del bus esta compuesto por un conector USB, un indicador de estado del bus y un circuito de detección de conexión. El conector USB es un tipo A en donde las líneas D+ y D- están conectadas a los pines RC5 y RC4 respectivamente. Estas líneas son las utilizadas para transmitir y recibir datos. El indicador de estado es un diodo LED (D4) conectado al pin RA3 que encienda y apaga cuando la CPU está conectada al ordenador.
- El circuito de detección de conexión es un divisor de voltaje entre dos resistores también incorpora un diodo LED (D1). Para realizar la detección del bus un extremo del divisor de voltaje va conectado al voltaje del bus (VDD) y el otro extremo al pin RA5 del PIC (detección de bus por *hardware*).

También es importante señal la función del capacitor (C3) que se encuentra conectado al pin 18 del PIC (Vusb). Este ayuda a establecer el voltaje en 3.3 voltios que exige el protocolo para poder establecer la comunicación a través del bus USB.

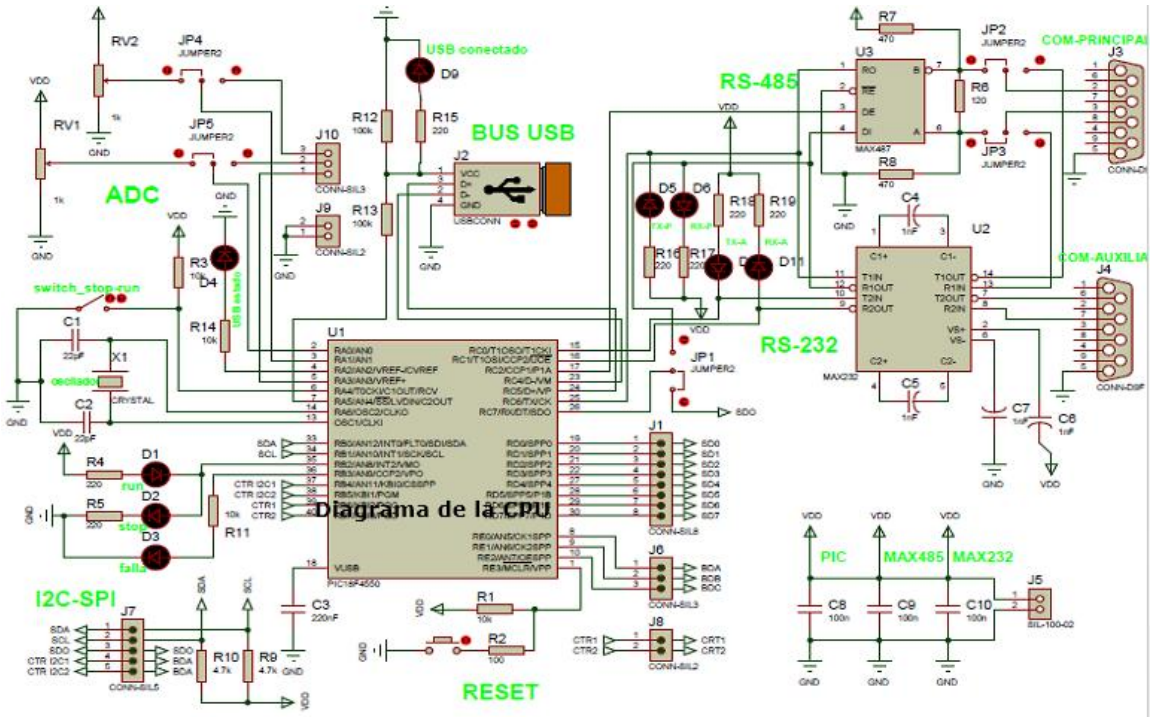


Imagen 10 Diseño inicial de CPU

5.6. LENGUAJES DE PROGRAMACIÓN DEL MICROCONTROLADOR

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que son llevados a cabo por sistemas de cómputo. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de ese sistema, para expresar algoritmos con precisión o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama programación [17].

También la palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos [18]:

- El desarrollo lógico del programa para resolver un problema en particular.
- Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa).
- Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina.
- Prueba y depuración del programa.

Permite especificar de manera precisa sobre qué datos debe operar una computadora, cómo deben ser almacenados o transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural. Una característica relevante de los lenguajes de programación es precisamente que más de un programador pueda usar un conjunto común de instrucciones que sean comprendidas entre ellos para realizar la construcción de un programa de forma colaborativa [17].

A la forma visible de un lenguaje de programación se le conoce como sintaxis. La mayoría de los lenguajes de programación son puramente textuales, es decir, utilizan secuencias de texto que incluyen palabras, números y puntuación, de manera similar a los lenguajes naturales escritos. Por otra parte, hay algunos lenguajes de programación que son más gráficos en su naturaleza, utilizando relaciones visuales entre símbolos para especificar un programa [19].

La sintaxis de un lenguaje de programación describe las combinaciones posibles de los símbolos que forman un programa sintácticamente correcto. El significado que se le da a una combinación de símbolos es manejado por su semántica (ya sea formal o como parte del código duro de la referencia de implementación). Dado que la mayoría de los lenguajes son textuales, este artículo trata de la sintaxis textual.

La sintaxis de los lenguajes de programación es definida generalmente utilizando una combinación de expresiones regulares (para la estructura léxica) y la Notación de Backus-Naur (para la estructura gramática).

5.6.1. Paradigmas

Los programas se pueden clasificar por el paradigma del lenguaje que se use para producirlos. Los principales paradigmas son: imperativos, declarativos y orientación a objetos [17].

Los programas que usan un lenguaje imperativo especifican un algoritmo, usan declaraciones, expresiones y sentencias [20]. Una declaración asocia un nombre de variable con un tipo de dato, por ejemplo: `var x: integer;`. Una expresión contiene un valor, por ejemplo: `2 + 2` contiene el valor 4. Finalmente, una sentencia debe asignar una expresión a una variable o usar el valor de una variable para alterar el flujo de un programa, por ejemplo: `x := 2 + 2; if x == 4 then`

haz_algo();. Una crítica común en los lenguajes imperativos es el efecto de las sentencias de asignación sobre una clase de variables llamadas "no locales".

Los programas que usan un lenguaje declarativo especifican las propiedades que la salida debe conocer y no especifica cualquier detalle de implementación. Dos amplias categorías de lenguajes declarativos son los lenguajes funcionales y los lenguajes lógicos. Los lenguajes funcionales no permiten asignaciones de variables no locales, así, se hacen más fáciles, por ejemplo, programas como funciones matemáticas. El principio detrás de los lenguajes lógicos es definir el problema que se quiere resolver (el objetivo) y dejar los detalles de la solución al sistema. El objetivo es definido dando una lista de sub-objetivos. Cada sub-objetivo también se define dando una lista de sus sub-objetivos, etc. Si al tratar de buscar una solución, una ruta de sub-objetivos falla, entonces tal sub-objetivo se descarta y sistemáticamente se prueba otra ruta.

La forma en la cual se programa puede ser por medio de texto o de forma visual. En la programación visual los elementos son manipulados gráficamente en vez de especificarse por medio de texto [20].

5.6.2. MPLAB

MPLAB es un editor IDE (Entorno de Desarrollo Integrado) gratuito, destinado a productos de la marca Microchip. Este editor es modular, permite seleccionar los distintos microcontroladores soportados, además de permitir la grabación de estos circuitos integrados directamente al programador.Es un programa que corre bajo Windows y como tal, presenta las clásicas barras de programa, de menú, de herramientas de estado, etc. El ambiente MPLAB® posee editor de texto, compilador y simulación (no en tiempo real). Para comenzar un programa desde cero para luego grabarlo al µC en MPLAB® v7.XX los pasos a seguir son:

1. Crear un nuevo archivo con extensión .ASM y nombre cualquiera
2. Crear un Proyecto nuevo eligiendo un nombre y ubicación
3. Agregar el archivo .ASM como un SOURCE FILE
4. Elegir el microcontrolador a utilizar desde SELECT DEVICE del menú CONFIGURE

Una vez realizado esto, se está en condiciones de empezar a escribir el programa respetando las directivas necesarias y la sintaxis para luego compilarlo y grabarlo en el PIC.

Una vez escrito y depurado el programa, se procede a la compilación. Para esto, desde el menú PROJECT se elige la opción BUILD ALL (construir todo) que, si no existen errores, devolverá un mensaje como BUILD SUCCESFULL. Los errores que muestra el compilador son del tipo sintácticos, es decir que si el programa "construido" llegara a tener un error, por ejemplo que esperase a que se ponga un bit en “0” y nunca pasase, se estará en un bucle infinito a pesar de que el compilador compilará perfectamente porque no hay error de sintaxis.

También existen mensajes y advertencias; los mensajes pueden ser, por ejemplo, que se está trabajando en un banco de memoria que no es el bank 0, etc. Las advertencias tienen un poco más de peso, por ejemplo: el PIC seleccionado no es el mismo que esta definido en el programa, etc. En ambos casos, mensajes y advertencias, la compilación termina satisfactoriamente pero hay que tener en cuenta siempre lo que nos dicen estos para prevenir errores.

Terminada la compilación el MPLAB® nos genera un archivo de extensión *.hex* el cual es completamente entendible para el PIC. Es decir, solo resta grabarlo al PIC por medio de una interfaz como por ejemplo el programador Picstart Plus de microchip. Una vez completado esto, se alimenta al mismo y el programa ya se estará ejecutando.

En este proyecto se utiliza MPLAB para crear un entorno compatible con el PIC C Compiler, explicado a continuación.

5.6.3. PIC C Compiler

Es un entorno de desarrollo que contiene un compilador en C para microcontroladores PIC. Además también cuenta con librerías para el manejo de diferentes dispositivos y librerías para el manejo del bus USB, comunicación tipo industrial MODBUS, entre otras.

El compilador se puede integrar con MPLAB IDE, tal como se hizo en este proyecto, y también permite la depuración del *firmware* en Proteus. Para el desarrollo de *firmware* mediante la plantilla es necesario conocer el lenguaje de especialmente el compilador PIC C compiler de CCS.

De acuerdo a la aplicación que se quiera desarrollar se colocan en comentarios o se comentan las directivas escritas en la plantilla.

5.6.4. LABVIEW

LabVIEW es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico. Recomendado para sistemas *hardware* y *software* de pruebas, control y diseño, simulado o real y embebido, pues acelera la productividad. El lenguaje que usa se llama G, donde la letra simboliza que es lenguaje Gráfico [21].

Labview fue utilizado en el proyecto porque, a diferencia de otros entornos de simulación, permite agregar una cantidad ilimitada de tags o variables, permite explicar mejor el funcionamiento interno del PLC, puesto que la configuración requiere de más detalles de bajo nivel. Además, su entorno de desarrollo basado en un ambiente gráfico, permite entender más fácilmente la lógica planteada.

Imagen 11 Iniciando labview 2011 y 2012

Este programa fue creado por National Instruments (1976) para funcionar sobre máquinas MAC, salió al mercado por primera vez en 1986. Ahora está disponible para las plataformas Windows, UNIX, MAC y GNU/Linux. La última versión es la 2012, con la increíble demostración de poderse usar simultáneamente para el diseño del *firmware* de un instrumento RF de última generación, a la programación de alto nivel del mismo instrumento, todo ello con código abierto.

Los programas desarrollados con LabVIEW se llaman Instrumentos Virtuales, o VIs, y su origen provenía del control de instrumentos, aunque hoy en día se ha expandido ampliamente no sólo al control de todo tipo de electrónica (Instrumentación electrónica) sino también a su programación embebida, comunicaciones, matemáticas, etc. Un lema tradicional de LabVIEW es: *"La potencia está en el Software"*, que con la aparición de los sistemas multinúcleo se ha hecho aún más potente. Entre sus objetivos están el reducir el tiempo de desarrollo de aplicaciones de todo tipo (no sólo en ámbitos de Pruebas, Control y Diseño) y el permitir la entrada a la informática a profesionales de cualquier otro campo. LabVIEW consigue combinarse con todo tipo de *software* y *hardware*, tanto del propio fabricante -tarjetas de adquisición de datos, PAC, Visión, instrumentos y otro *Hardware*- como de otros fabricantes.

Su principal característica es la facilidad de uso, válido para programadores profesionales como para personas con pocos conocimientos en programación

pueden hacer programas relativamente complejos, imposibles para ellos de hacer con lenguajes tradicionales. También es muy rápido hacer programas con LabVIEW y cualquier programador, por experimentado que sea, puede beneficiarse de él. Los programas en LabView son llamados instrumentos virtuales (VIs) Para los amantes de lo complejo, con LabVIEW pueden crearse programas de miles de VIs (equivalente a millones de páginas de código texto) para aplicaciones complejas, programas de automatizaciones de decenas de miles de puntos de entradas/salidas, proyectos para combinar nuevos VIs con VIs ya creados, etc. Incluso existen buenas prácticas de programación para optimizar el rendimiento y la calidad de la programación. El labView 7.0 introduce un nuevo tipo de subVI llamado VIs Expreso (Express VIS). Estos son VIs interactivos que tienen una configuración de caja de diálogo que permite al usuario personalizar la funcionalidad del VI Expreso. El VIs estándard son VIs modulares y personalizables mediante cableado y funciones que son elementos fundamentales de operación de LabView [21].

Presenta facilidades para el manejo de:

- Interfaces de comunicaciones:
  - Puerto serie
  - Puerto paralelo
  - GPIB
  - PXI
  - VXI
  - TCP/IP, UDP, DataSocket
  - Irda
  - Bluetooth
  - USB
  - OPC...
- Capacidad de interactuar con otros lenguajes y aplicaciones:

    - DLL: librerías de funciones
    - .NET
    - ActiveX
    - Multisim
    - Matlab/Simulink
    - AutoCAD, SolidWorks, etc
- Herramientas gráficas y textuales para el procesado digital de señales.
- Visualización y manejo de gráficas con datos dinámicos.
- Adquisición y tratamiento de imágenes.
- Control de movimiento (combinado incluso con todo lo anterior).
- Tiempo Real estrictamente hablando.
- Programación de FPGAs para control o validación.
- Sincronización entre dispositivos.

Como se ha dicho es una herramienta gráfica de programación, esto significa que los programas no se escriben, sino que se dibujan, facilitando su comprensión. Al tener ya pre-diseñados una gran cantidad de bloques, se le facilita al usuario la creación del proyecto, con lo cual en vez de estar una gran cantidad de tiempo en programar un dispositivo/bloque, se le permite invertir mucho menos tiempo y dedicarse un poco más en la interfaz gráfica y la interacción con el usuario final. Cada VI consta de dos partes diferenciadas [21]:

- *Panel Frontal*: El *Panel Frontal* es la interfaz con el usuario, que se utiliza para interactuar con el usuario cuando el programa se está ejecutando. Los usuarios podrán observar los datos del programa actualizados en tiempo real(como van fluyendo los datos, un ejemplo sería una calculadora, donde tú le pones las entradas, y te pone el resultado en la salida). En esta interfaz se definen los *controles* (usados como entradas, pueden ser botones, marcadores etc...) e *indicadores* (usados como salidas, pueden ser gráficas....).

- *Diagrama de Bloques*: es el programa propiamente dicho, donde se define su funcionalidad, aquí se colocan íconos que realizan una determinada función y se interconectan (el código que controla el programa --. Suele haber una tercera parte *icono/conector* que son los medios utilizados para conectar un VI con otros VIs.

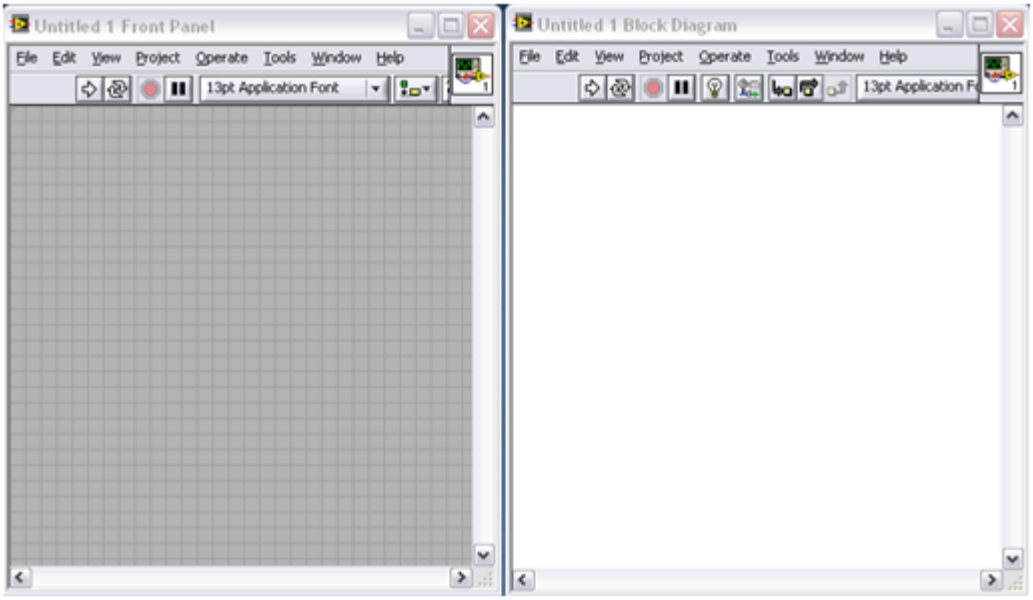


Imagen 12 Panel frontal (izquierda) y diagrama de Bloques (derecha) LabView [22].

En el panel frontal, se encuentran todo tipos de controles o indicadores, donde cada uno de estos elementos tiene asignado en el diagrama de bloques una terminal, es decir el usuario podrá diseñar un proyecto en el panel frontal con controles e indicadores, donde estos elementos serán las entradas y salidas que interactuaran con la terminal del VI. Se puede observar en el diagrama de bloques, todos los valores de los controles e indicadores, como van fluyendo entre ellos cuando se está ejecutando un programa VI [21].

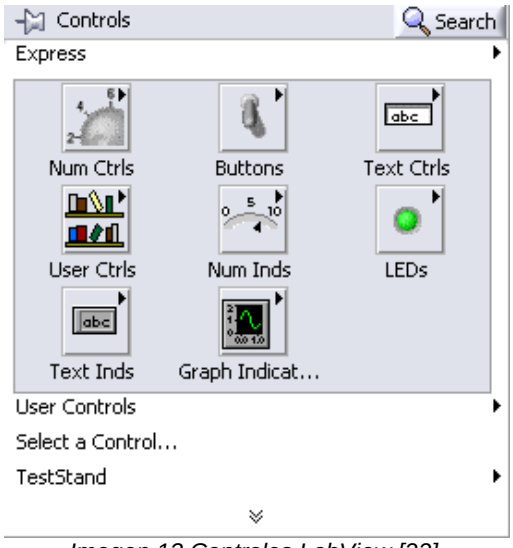


Imagen 13 Controles LabView [22].

Es importante hacer notar que un control es una *entrada* de datos, y un indicador es una *salida*. Aquí se pueden ver unos ejemplos:

Imagen 14 Ejemplo de Controles vistos en panel frontal y Diagrama de Bloques LabView [22].

En la parte superior se observa un control numérico y uno booleano (botón), mientras que en la inferior se observa un indicador numérico y uno booleano (un LED). De igual forma cada control e indicador tiene una etiqueta única, dando doble clic sobre la etiqueta puede modificarse.

En la Paleta de Controles de LabVIEW, se encuentran una amplia variedad de controles e indicadores, con ellos se puede armar interfaces de usuarios muy amigables y personalizados.

En resumen, cada control e indicador colocado en el Panel Frontal tiene un bloque correspondiente en el Diagrama de Bloques. Y precisamente estos bloques son los que se utilizan para realizar la programación.

Al hacer clic derecho en el Diagrama de Bloques, abre la Paleta de Funciones:

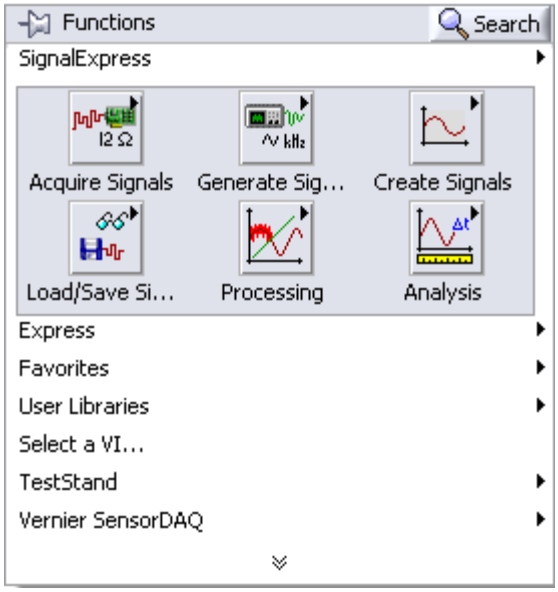


Imagen 15 Funciones de LabView [22].

En la Paleta de Funciones se encuentran todas las librerías necesarias para realizar la programación de la aplicación. Por ejemplo, en la categoría Programming se tienen funciones numéricas, de comparación, arreglos, etc.

Imagen 16 Ejemplo de Funciones en Diagrama de Bloques LabView [22].

Como es de esperarse, las funciones tienen uno o más entradas y salidas.

Además de las funciones básicas de programación, LabVIEW cuenta con librerías de funciones para usos más avanzados, algunos ejemplos son [22]:

- **Matemáticas**: Ajuste de curvas, geometría, interpolación, álgebra lineal-
- **Procesamiento de Señales**: Filtros, FFTs, análisis espectral.
- **Adquisición de Datos**: Librerías para adquirir datos de instrumentos (vía GPIB), y dispositivos de National Instruments (PCI, USB, PXI).
- **Comunicación de Datos**: Serial, TCP/IP, UDP, USB.

6. CAPITULO 3: ANTECEDENTES, PLANIFICACION Y DISEÑO

6.1. ANTECEDENTES

Como antecedente principal del proyecto se puede mencionar la investigación realizada por el Ingeniero Rubén Sánchez [1], que consistió en el desarrollo de un PLC para ambiente académico con microcontrolador. Esta investigación sirvió de base para la realización del proyecto aquí ejecutado.

A continuación se describe lo que se logró en la anterior investigación.

El principal producto obtenido a partir del desarrollo del proyecto, fue una CPU sobre el micro controlador 16F84A, compuesta por un sistema de comunicación serie para el intercambio de información con ordenadores y con capacidad de direccionar ocho entradas y ocho salidas digitales. En la Imagen 17 se comparan la tarjeta incial y la totalmente construida.

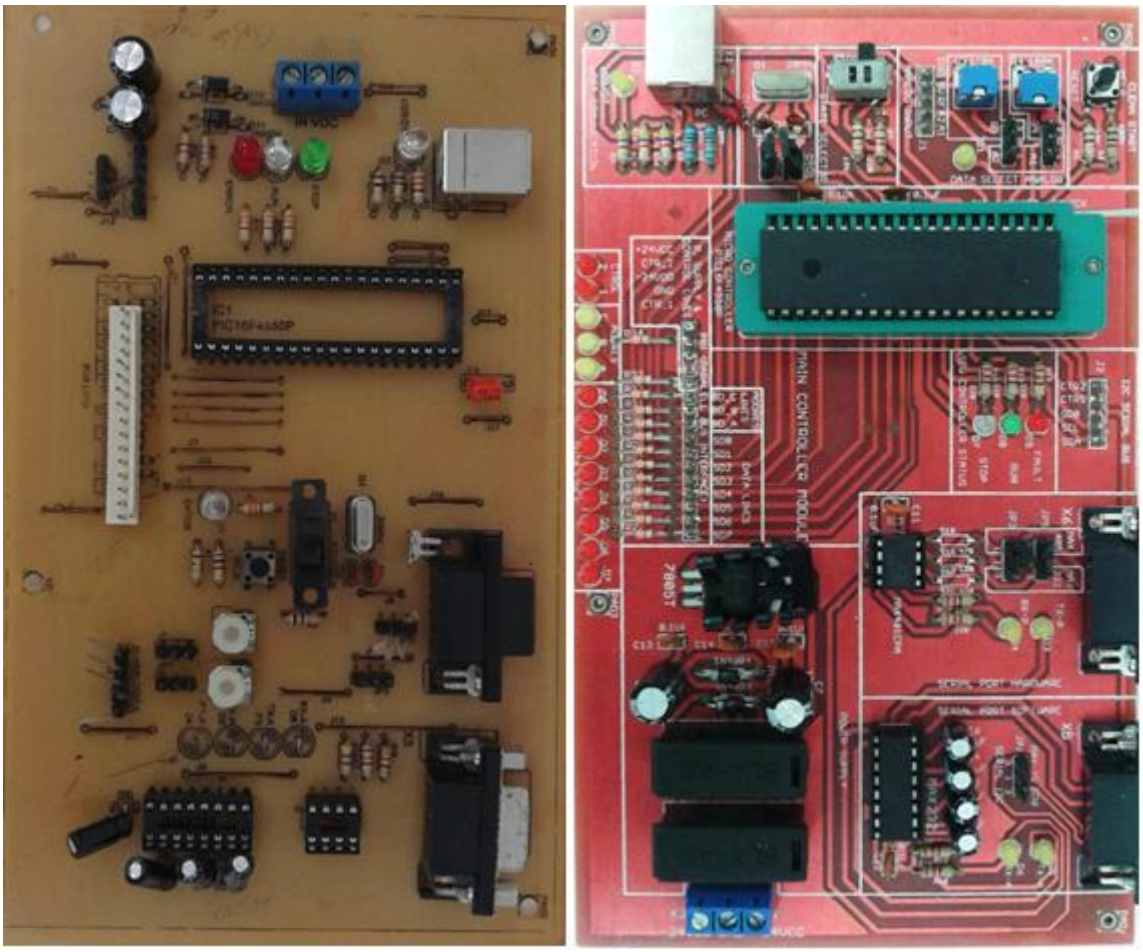


Imagen 17.Comparativa de tarjetas de la CPU. Izquierda CPU Inicial. Derecha, CPU final

Por otra parte se desarrolló una tarjeta para la gestión de salidas y entradas digitales, basadas en salidas a relays y entradas optoacopladas. Se ilustra en la Imagen 18.

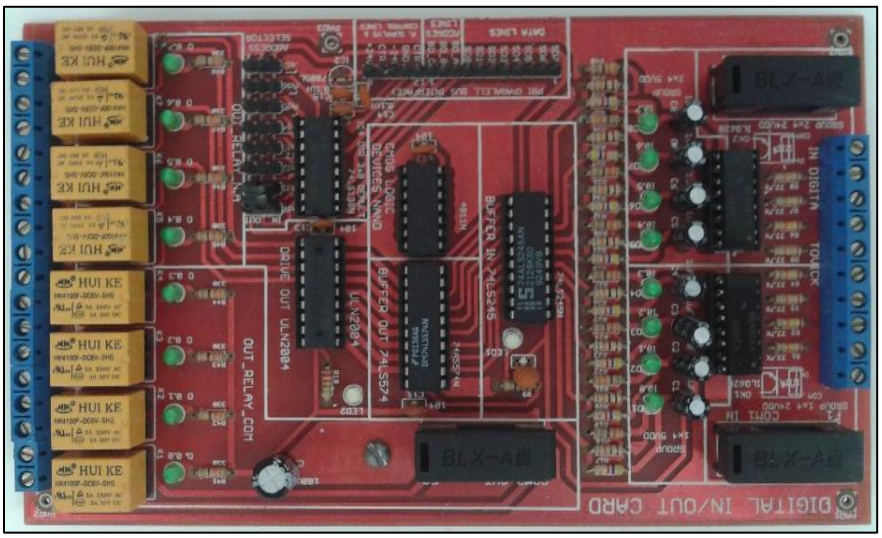


Imagen 18. Tarjeta de Entradas y salidas Digitales. [1].

6.2. PLANIFICACIÓN

Durante la ejecución del proyecto se presentaron ciertos obstáculos que significaron un retraso de tres años para su finalización. Dichos retrasos se debieron principalmente a que, con el fin de cumplir con los requisitos planteados, se debieron investigar más a fondo muchos conceptos, para luego plasmarlos en físico y conectarlos entre sí. De igual manera, se intentó desarrollar una tarjeta de entradas y salidas analógicas para mostrar el comportamiento del autómata bajo este tipo de señales, pero se desistió de esa idea porque ya el proyecto había sufrido un retraso considerable.

Las primeras cuatro semanas en el cronograma establecido inicialmente para su desarrollo, fueron dedicadas exclusivamente a la planificación. En esas semanas se fijaron las pautas generales a seguir, se definieron las etapas a cubrir en el proyecto, objetivos, plazos y expectativas. Estas etapas consisten en:

6.2.1. Requisitos Del Sistema

Los requisitos del sistema se dividen en dos partes, la primera comprende los requisitos para el *hardware* que incluye la CPU, entradas y salidas (tarjeta de adquisición y control) y la segunda para el *software* que estará instalado en el ordenador. El medio de comunicación entre estos dos es el bus USB, definido desde el principio.

6.2.1.1. Requisitos Para Las Tarjetas CPU y Entradas/Salidas

La tarjeta de adquisición o CPU debe tener como requisitos mínimos: el soporte de entradas y salidas todo o nada (digitales), entradas y salidas analógicas,

comunicación con el ordenador mediante el bus USB, implementación de protocolos de comunicación serie síncrono y asíncrono para intercambiar información con otros dispositivos, o acceso a redes tipo industriales. Debe estar construida cumpliendo las normas que definen un circuito de estado sólido, teniendo el menor espacio posible en un diseño doble capa, con su respectiva demarcación de componentes. De igual manera debe tener un diseño que incluya perforaciones para soportarla a una estructura o base al momento del ensamble final. Las tarjetas adicionales digitales deben de igual manera cumplir con estos requisitos.

6.2.2. Requisitos Para El Software

El *software* de aplicación debe hacer la función de un sistema SCADA, permitiendo el control y la supervisión de los recursos de la CPU (entradas, salidas, etc.). En un sistema embebido el *hardware* y el *software* se complementan de tal manera que el uno no funciona sin el otro. Cabe resaltar en este caso que los requisitos del *software* también incluyen unos requisitos de *hardware*; estos requisitos parten desde el microcontrolador escogido, en este caso PIC18F4550. Además hay una necesidad de *software* embebido en el microcontrolador que es el código que este ejecutará en él. El algoritmo fue ejecutado en entornos de programación tales como: C Compiler (desglosado en el apartado “Lenguaje de programación), MPLAB, PROTEUS, y LabVIEW.

6.3. ALTERNATIVAS DEL DESARROLLO

6.3.1. Alternativas Para El Desarrollo Del Software

Teniendo en cuenta que el proyecto comprende la utilización de dos tipos de *software*, uno para el funcionamiento del microcontrolador (*firmware*) y otro para el sistema de supervisión (ordenador), es necesario conocer las alternativas existentes para el desarrollo de ambos.

Inicialmente se mencionan las alternativas para el desarrollo del *firmware* y posteriormente se comentarán las alternativas para la realización del sistema de supervisión.

6.3.1.1. Alternativas Para El Desarrollo Del Firmware

Además de los fabricantes de microcontroladores existen empresas que desarrollan compiladores. A continuación se mencionan los entornos de desarrollo más utilizados en la programación de microcontroladores relacionados con Microchip.

- **PIC C Compiler**

Es un entorno de desarrollo que contiene un compilador para lenguaje C para microcontroladores PIC. Además también cuenta con librerías para el manejo de diferentes dispositivos y librerías para el manejo del bus USB, comunicación tipo industrial MODBUS, entre otras. El compilador se puede integrar con MPLAB IDE y también permite la depuración del *firmware* en Proteus [23].

- **MikroC**

Es un entorno de desarrollo que ofrece muchas herramientas para la depuración de *firmware*, soporta una gran cantidad de marcas de microcontroladores entre las cuales se encuentra Microchip y Atmel. Este entorno posee un compilador para lenguaje C y herramientas para la realización de la comunicación serie, USB, entre otras [24].

- **Proton IDE**

Es un entorno de desarrollo creado especialmente para el trabajo en el lenguaje de programación Basic, posee muchas herramientas para el trabajo con dispositivos como LCD gráficos y un asistente para el establecimiento de comunicación USB entre el dispositivo y el ordenador [25].

- **MPLab IDE**

Es el entorno de desarrollo nativo para los microcontroladores PIC. Teniendo la capacidad de soportar muchos compiladores entre los cuales se encuentra el ASM, C y Basic. Además cuenta con muchas herramientas para la depuración del *firmware* [26].

6.3.2. Alternativas Para El Desarrollo Del Supervisorio

Existen muchas herramientas para el desarrollo de *software*, a continuación se consideran las más destacadas y que cumplen con los requisitos para el desarrollo de la aplicación.

6.3.2.1. VISUAL STUDIO

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE) para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros [27].

6.3.2.2. DELPHI

Delphi es un entorno de desarrollo de *software* diseñado para la programación de propósito general con énfasis en la programación visual. En Delphi se utiliza como lenguaje de programación una versión moderna de Pascal llamada Object Pascal. Es producido comercialmente por la empresa estadounidense CodeGear. En sus diferentes variantes, permite producir archivos ejecutables para Windows, GNU/Linux y la plataforma .NET [28].

6.3.2.3. NETBEANS

NetBeans es una plataforma para el desarrollo de aplicaciones de escritorio a partir del lenguaje de programación Java y un entorno de desarrollo integrado (IDE) usando la Plataforma NetBeans [29].

6.3.2.4. LABVIEW

LabVIEW es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico. Recomendado para sistemas *hardware* y *software* de pruebas, control y diseño, simulado o real y embebido, pues acelera la productividad. El lenguaje que usa se llama G, donde la letra simboliza que es lenguaje Gráfico [21].

Este *software* es descrito con mayor profundidad en la sección 6.6.4.

6.3.3. Alternativas Para El Desarrollo Del *Hardware*

Un paso importante es definir la arquitectura del *hardware* sobre la cual se va a desarrollar el proyecto y las herramientas de desarrollo.

Se definió una arquitectura con base en un microcontrolador teniendo en cuenta el costo y la complejidad del sistema, así que solo se mencionan las alternativas tenidas en cuenta para la elección de éste.

Las herramientas de desarrollo de *hardware* comprenden los *software* para la realización de PCB (circuito impreso).

6.3.3.1. Alternativas de Selección Del Micro-Controlador

Antes de seleccionar un microcontrolador es imprescindible analizar los requisitos de la aplicación que se quiere desarrollar. A continuación se indican algunos de los aspectos que normalmente hay que tener en cuenta a la hora de realizar la elección:

Procesamiento de datos: puede ser necesario que el microcontrolador realice cálculos críticos en un tiempo limitado. En ese caso se debe asegurar la selección de un dispositivo suficientemente rápido para ello. Por otro lado, se debe tener en cuenta la precisión de los datos a manejar: si no es suficiente con un microcontrolador de 8 bits, puede ser necesario acudir a microcontroladores de 16 o 32 bits, o incluso a *hardware* de coma flotante.

Entrada/Salida: para determinar las necesidades de Entrada/Salida del sistema es conveniente conocer el diagrama de bloques del mismo, de tal forma, que sea sencillo identificar la cantidad y tipo de señales a controlar. Una vez realizado este análisis puede ser necesario añadir periféricos externos o cambiar a otro microcontrolador más adecuado para ese sistema.

Consumo: algunos productos que incorporan microcontroladores están alimentados con baterías. Lo más conveniente en un caso como éste puede ser que el microcontrolador esté en estado de bajo consumo pero que despierte ante la activación de una señal (una interrupción) y ejecute el programa adecuado para procesarla.

Memoria: en cuanto a la cantidad de memoria necesaria se debe hacer una estimación de cuánta memoria volátil y no volátil es necesaria y si es conveniente disponer de memoria no volátil modificable.

Además de todo lo mencionado también es importante tener en cuenta la documentación y herramientas de desarrollo disponibles para cada microcontrolador.

A continuación se mencionan algunas de las marcas que se adaptan a los aspectos mencionados y a los requisitos del sistema.

6.3.3.2. ATMEL

Atmel es una compañía de semiconductores, que tiene la familia de microcontroladores AVR. Esta familia está basada en una nueva arquitectura RISC que incorpora memoria Flash para el programa y memoria EEPROM para los datos. Además esta arquitectura fue diseñada para ser totalmente compatible con lenguaje C, permitiendo trabajar en alto nivel.

Los microcontroladores AVR vienen equipados con muchos recursos, como timer, módulo UART, módulo PWM, conversor análogo digital, entre otros.

Entre los microcontroladores AVR más destacados se encuentra la gama AT9, que tienen la característica de soportar bus USB y puede ser programado en lenguaje ensamblador o C [30].

6.3.3.3. MOTOROLA

Motorola es una compañía que desarrolla semiconductores, posee una familia de microcontroladores Motorola 68HC11. Los 68HC11 son derivados del microprocesador Motorola 6800, siguen la Arquitectura de von Neumann, en la que la memoria de programa, de datos y de entrada/salida se direcciona en un único mapa de memoria [31].

6.3.3.4. PARALLAX

Es una importante empresa desarrolladora de semiconductores, posee diferentes familias de microcontroladores, entre ellas están las siguientes [32]:

BASIC Stamp: Esta línea de microcontroladores se caracteriza, como su nombre lo indica, por ser programable en el idioma PBASIC, con un editor que se distribuye en forma gratuita, en versiones para Windows, MAC y LINUX; cuenta con una gran cantidad de información en varios idiomas y explicaciones detalladas de la conexión electrónica, para diversos circuitos y accesorios; y es indicado para el estudio y el desarrollo de aplicaciones (Tabla comparativa de los distintos modelos de BASIC Stamps ; Manuales en formato PDF en inglés y castellano).

SX: Estos microcontroladores, que pueden trabajar a más de 75 MHz en modo "turbo", se programan con un editor de distribución gratuita (SX/B), que permite la programación en BASIC y/o Assembly, la línea dispone de varios modelos en SMD y DIP, siendo el más popular el SX28AC/DP (20 I/O), pese a un costo moderado, por sus características de alta velocidad, permite hacer aplicaciones que otros microcontroladores no pueden.

Propeller: Es un nuevo concepto en microcontroladores, en realidad son ocho micro controladores en un mismo chip; trabajan al unísono con un mismo CLK, que cubre frecuencias, desde 0 más de 80 MHz., esto permite hacer programas que se ejecuten en verdadera "multifunción", no como es costumbre hacer "multitarea" por medio de "interrupciones". El editor que se distribuye en forma gratuita, permite la combinación de programación en idioma SPIN y Assembly (Manual en formato PDF en inglés y castellano.

6.3.3.5. MICROCHIP

Es una importante empresa desarrolladora de semiconductores, posee diferentes familias de microcontroladores, entre ellas están las siguientes [33]:

BASIC Stamp: Esta línea de microcontroladores se caracteriza, como su nombre lo indica, por ser programable en el idioma PBASIC, con un editor que se distribuye en forma gratuita, en versiones para Windows, MAC y LINUX; cuenta con una gran cantidad de información en varios idiomas y explicaciones detalladas de la conexión electrónica, para diversos circuitos y accesorios; y es indicado para el estudio y el desarrollo de aplicaciones (Tabla comparativa de los distintos modelos de BASIC Stamps ; Manuales en formato PDF en inglés y castellano).

SX: Estos microcontroladores, que pueden trabajar a más de 75 MHz en modo "turbo", se programan con un editor de distribución gratuita (SX/B), que permite la programación en BASIC y/o Assembly, la línea dispone de varios modelos en SMD y DIP, siendo el más popular el SX28AC/DP (20 I/O), pese a un costo moderado, por sus características de alta velocidad, permite hacer aplicaciones que otros microcontroladores no pueden.

Propeller: Es un nuevo concepto en microcontroladores, en realidad son ocho micro controladores en un mismo chip; trabajan al unísono con un mismo CLK, que cubre frecuencias, desde DC hasta más de 80 MHz., esto permite hacer programas que se ejecuten en verdadera "multifunción", no como es costumbre hacer "multitarea" por medio de "interrupciones". El editor que se distribuye en forma gratuita, permite la combinación de programación en idioma SPIN y Assembly (Manual en formato PDF en inglés y castellano.

6.3.4. Herramientas Para Diseño Del Hardware

En el mercado se encuentran muchos *software* para el desarrollo de circuitos impresos. A continuación se mencionan los más importantes y los más utilizados.

6.3.4.1. Proteus

Es un entorno de desarrollo integrado que comprende dos módulos, uno denominado ISIS y el otro ARES. ISIS se utiliza para desarrollar el esquema, simulación y depuración del circuito, mientras que ARES se utiliza para hacer el circuito impreso (PCB). El potencial de Proteus radica en la simulación y en la depuración de circuitos, mas no la realización de PCB [34].

6.3.4.2. Eagle Cadsoft

ECadSoft EAGLE es un sencillo programa que te permite el desarrollo de circuitos impresos. El programa consta de tres módulos, un Diagramador, un Editor de esquemas y un Autorouter que están integrados, por lo que no resulta necesario convertir los *netlist* (conjunto de paquetes sueltos vendidos como una suite y conectados mediante archivos externos) entre esquemas y diseños. Es una potente aplicación con la que diseñar circuitos impresos y realizar esquemas electrónicos. Eagle son las siglas de Easily Aplicable Graphical Layout Editor [35].

6.3.4.3. Orcad

La herramienta CAD (Diseño Asistido por Computadora) además de ser empleado en el diseño de planos y dibujos, es también ampliamente utilizada en el diseño de circuitos electrónicos. Las herramientas CAD permiten al usuario simular el funcionamiento que tendría el producto sin necesidad de ser armado, con esto abaratando la producción de *hardware* [36].

6.3.4.4. Altium Designer

Altium Designer es un conjunto de programas para el diseño electrónico en todas sus fases y para todas la disciplinas, ya sean esquemas, simulación , diseño de circuitos impresos , implementación de FPGA, o desarrollo de código para microprocesadores.

No se trata de un conjunto de paquetes sueltos vendidos como una suite y conectados mediante archivos externos (*netlist*), sino de un programa único (dxp.exe) que crea un (entorno){front-end} y comunica al usuario con los distintos servidores (por ejemplo, editor de texto, editor de esquemas, editor de PCB...) [37].

6.4. DISEÑO

6.4.1. Diseño General Del Proyecto

El desarrollo del autómata programable de ambiente académico se basó en la arquitectura planteada en el proyecto del Ingeniero Rubén Sánchez [1], tutor de

este proyecto.De acuerdo a la estructura de un PLC, el *hardware* de adquisición se compone de varios módulos o tarjetas. Una denominada CPU (unidad central de procesamiento) y tarjetas de entradas y salidas digitales (todo o nada). El sistema también se compone de un sistema supervisorio en el ordenador que se utiliza para el control y supervisión de los recursos de la CPU.

A continuación se muestra el diagrama de bloques del sistema:

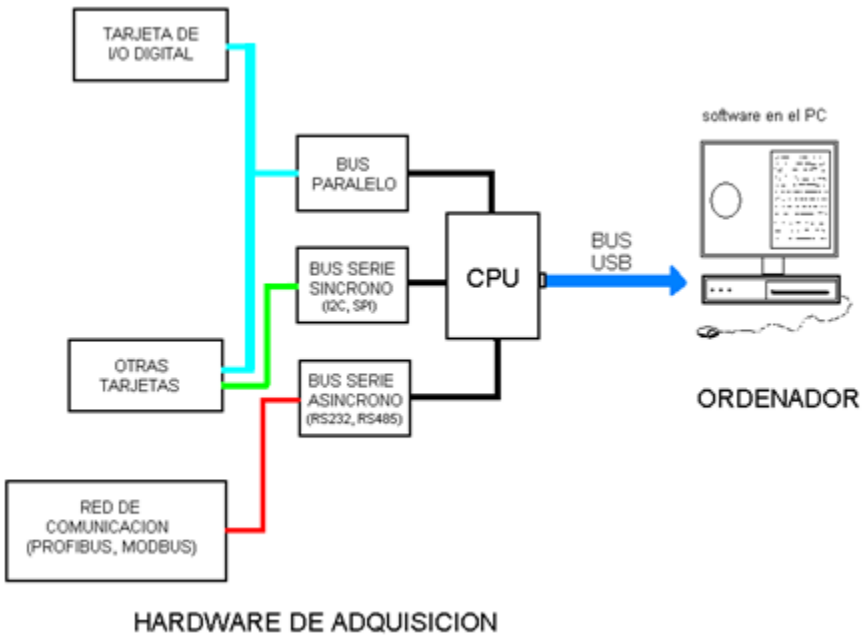


Imagen 19 Diagrama de bloques del Sistema.

La CPU está conectada al PC mediante el bus USB como se muestra en la Imagen 19. En dicho PC se encuentra el sistema de supervisión. Esto permite que mediante una aplicación (sistema supervisorio) en el ordenador se pueda acceder a los recursos que posea la CPU. Así mismo, el sistema se puede conectar a una red de comunicación tipo industrial, gracias a los puertos COM, que poseen capa física RS232 y RS485.

Para cumplir con estos requisitos y luego de estudiar las alternativas disponibles, se decidió utilizar los siguientes métodos y entornos de desarrollo:

- Selección del microcontrolador: Se decidió utilizar un microcontrolador MICROCHIP [33], debido a su amplio uso tanto en ambientes académicos como en desarrollos independientes. Dentro de la familia MICROCHIP, se decidió utilizar el PIC18F4550 ya que ofrecía todas las características que se requerían para el soporte de las comunicaciones utilizadas en el proyecto y el procesamiento de la tareas en un tiempo óptimo, además de ser económico y ampliamente disponible en el mercado.
- Selección del entorno de desarrollo del *hardware*: Se decidió utilizar EAGLE [35] debido a que cuenta con herramientas de diseño que facilitan y optimizan la creación de los PCB a nivel profesional, ya que para optimizar el espacio en las tarjetas debían tener un diseño a doble capa. Otra de las razones que se tuvieron en cuenta en la selección de EAGLE, fue que su versión freeware (acceso gratuito) cuenta con todos los módulos necesarios para el desarrollo del sistema.
- Selección del entorno de desarrollo del *software*: Se utilizó MPLAB porque está ampliamente difundido en el ambiente acdémico siendo el entorno que comunmente se enseña en universidades y entidades educativas del pais. Permite programar tanto en lenguaje asembler como en otros lenguajes mas difundidos, como el C++.
- Selección del entorno de desarrollo del sistema supervisorio: Se utilizó LABVIEW ya que está diseñado para desarrollar aplicaciones a nivel académico e industrial, es fácil de programar y permite la integración de cualquier dispositivo a través de protocolos de comunicación estándares como el RS232 y el USB, entre otros. Su interface grafica permite un rapido aprendizaje al mismo tiempo que se desarrolla la aplicación, y cuenta con amplias librerias.

6.4.2. Diseño Del Hardware

El *hardware* del sistema está compuesto básicamente por una CPU (unidad de procesamiento central) y una tarjeta de entrada salida todo o nada (digitales). A continuación se describirá cada parte.

6.4.2.1. Hardware De La CPU

La CPU está constituida por el microcontrolador PIC18F4550 gama alta de MICROCHIP, buses de comunicación serie asíncrona (USB y RS232) y síncrona (SPI y I2C), y un bus de comunicación en paralelo. En la Imagen 20 se muestra el esquema eléctrico detallando cada parte. A continuación se explica cada etapa.

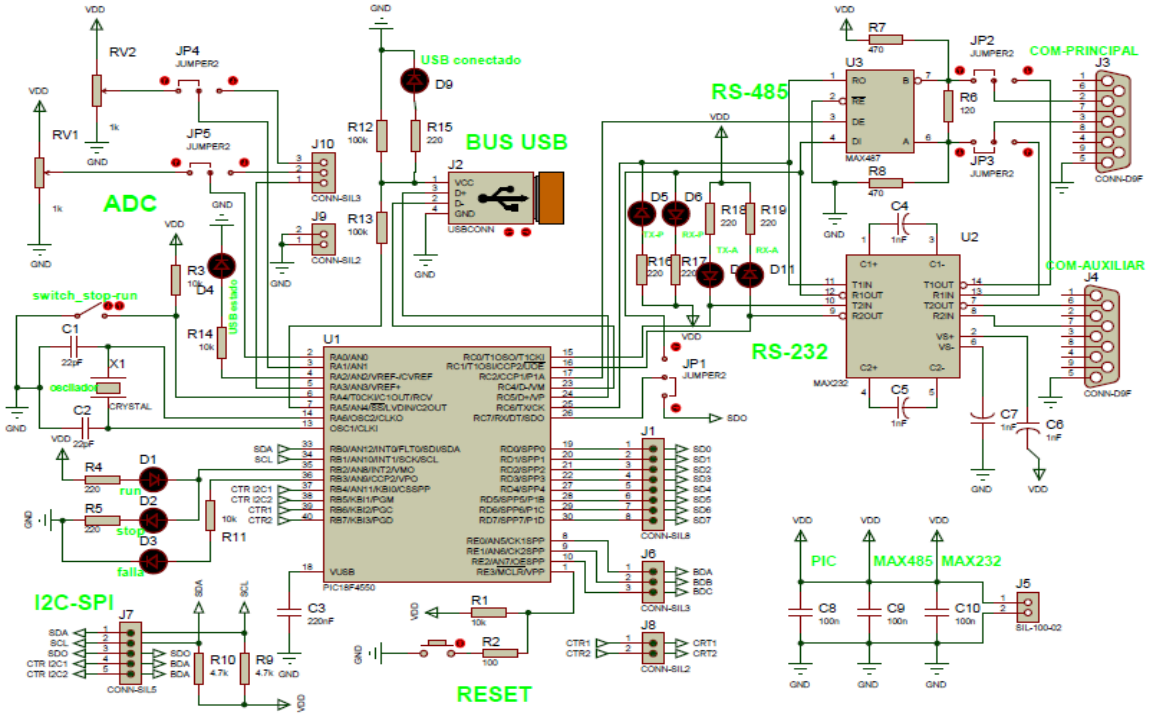


Imagen 20 Diagrama Esquemático de la CPU

6.4.2.2. Microcontrolador PIC18F4550

La configuración del PIC en la CPU está dada por un circuito de reset, una etapa de oscilación y un circuito *stop-run*, tal como se ilustra en la Imagen 21. Su función es controlar los periféricos conectados a los buses y procesar datos provenientes de los mismos.

El circuito de reset, conectado al pin MCLR del PIC consta del paralelo de un resistor de 10K ohmios en serie con un pulsador y un resistor de 100 ohmios, tiene como objetivo de eliminar la sobrecorriente y evitar cortocircuitos.

El circuito de oscilación, conectado a la fuente de oscilación primaria (OSC1 Y OSC2), consta de un cristal de cuarzo a 20MHz y 2 capacitores a 22pF, con el fin de lograr una señal de reloj más pura.

El circuito *stop-run* es un switch bi-estado conectado al pin RA4 del microcontrolador que se utiliza para cambiar la CPU de modo *Stop* a modo *Run*. A este circuito están asociados dos diodos LED que indican el modo en que se encuentra la CPU.

Para el caso de este proyecto, tanto el programa de control del autómata programable como el programa de usuario, se encuentran dentro de la memoria del microcontrolador.

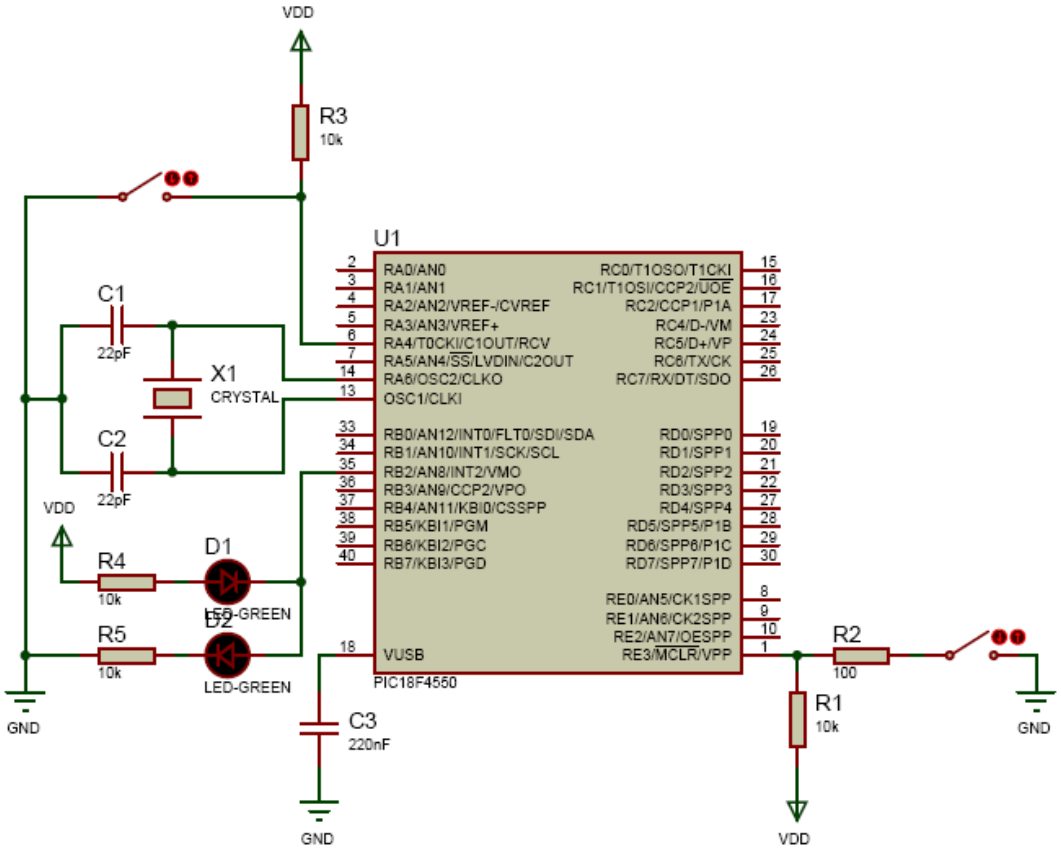


Imagen 21 Circuito de Oscilación de la CPU

6.4.2.3. Buses de comunicación serie asíncrona

Están constituidos por buses a dos hilos uno para la transmisión y otro para la recepción. En la CPU, está implementado el bus USB y el protocolo RS232, a continuación se explica la implementación de cada bus.

6.4.2.3.1. El bus USB

La CPU es capaz de soportar el protocolo USB gracias al microcontrolador PIC18f4550. Este bus tiene principalmente dos objetivos, el primero es cargar el programa compilado (archivo .hex) desde la computadora a la CPU y el segundo servir de comunicación en un sistema SCADA, entre el ordenador y la CPU.

El *hardware* del bus está compuesto por un conector USB, un indicador de estado del bus y un circuito de detección de conexión. En la parte superior izquierda de la Imagen 22 se ilustra el esquema eléctrico.

El conector USB es un tipo A en donde las líneas D+ y D- están conectadas a los pines RC5 y RC4 respectivamente. Estas líneas son las utilizadas para transmitir y recibir datos.

El indicador de estado es un diodo LED (D4) conectado al pin RA3 que enciende y apaga cuando la CPU está conectada al ordenador.

El circuito de detección de conexión es un divisor de voltaje entre dos resistores, que también incorpora un diodo LED (D1). Para realizar la detección del bus un extremo del divisor de voltaje va conectado al voltaje del bus (VDD) y el otro extremo al pin RA5 del PIC (detección de bus por *hardware*).

También es importante señalar la función del capacitor (C3), que se encuentra conectado al pin 18 del PIC (Vusb). Este ayuda a establecer el voltaje (3.3 voltios) que exige el protocolo para poder establecer la comunicación a través del bus USB.

6.4.2.3.2. El protocolo RS232

La CPU tiene implementado dos puertos de comunicación serie RS232, uno principal y otro auxiliar. Son utilizados para comunicar la CPU con la computadora

y principalmente para agregar el PLC a una red de tipo industrial como la MODBUS o PROFIBUS. En la Imagen 22 se ilustra el esquema de conexión del COM principal y auxiliar.

Las líneas del COM auxiliar se encuentran conectadas al pin RC0 y RC1 del PIC y al circuito integrado MAX232 que es un conversor de protocolo de TTL a RS232 y viceversa. En la línea TX y RX, transmisión y recepción respectivamente, se encuentran los diodos LED (D7, D8) que cambian su estado cuando se está haciendo la comunicación. Para la implementación de este puerto en el microcontrolador se realiza una rutina de código (comunicación por *software*).

El COM principal utiliza el *hardware* UART que corresponde al pin RC6 y RC7 del PIC, además de estar conectado al MAX232 también estas en la capacidad de cambiar la capa física a niveles RS485, mediante jumper que pueden ser cambiados de posición para elegir la capa que se quiera utilizar. Este también posee diodos LED (D5, D6) que indican el paso de datos.

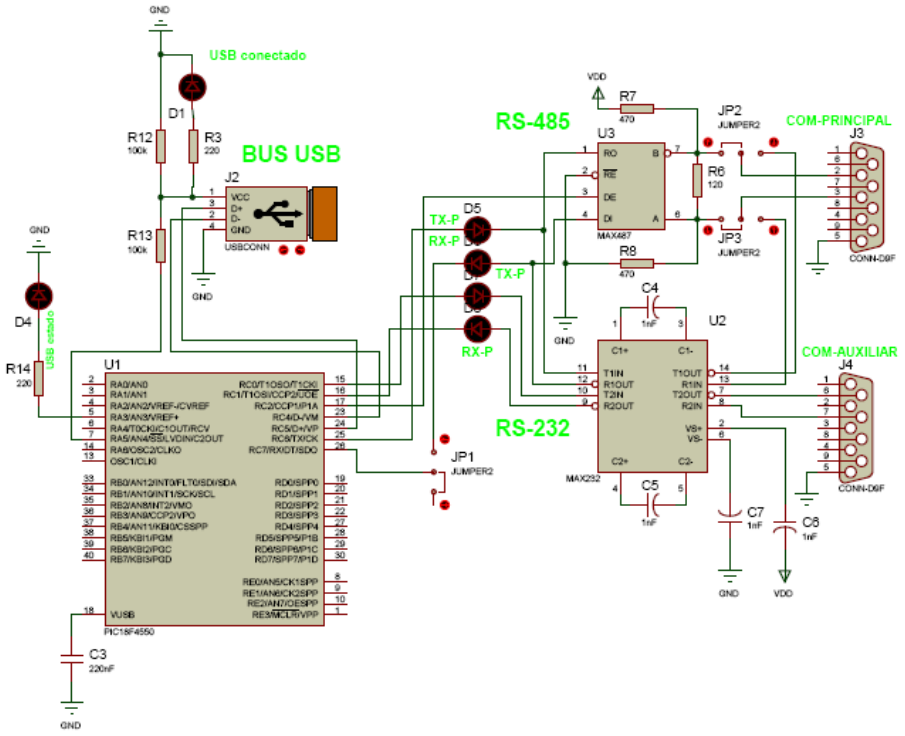


Imagen 22 Diagrama de buses de comunicación serie asíncrona (USB, RS232, 485)

6.4.2.4. Buses de comunicación serie síncrona.

La CPU (Unidad central de proceso) del PLC puede soportar los protocolos I2C y SPI. Éstos se utilizan para comunicar la tarjeta con otros sistemas (pantallas LCD, memorias, sensores, etc) que soporten estos mismos protocolos. En la Imagen 23 se muestra su esquema eléctrico y como está constituido.

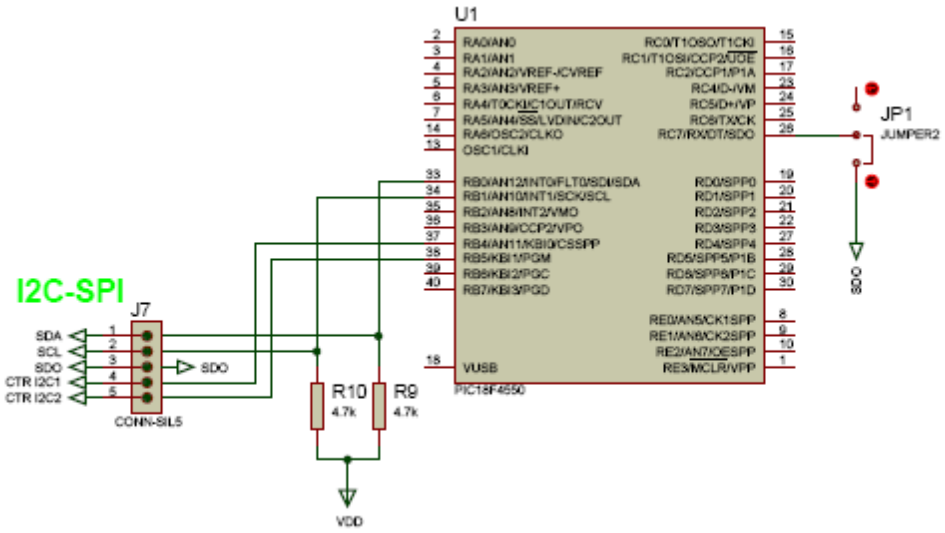


Imagen 23 Diagrama buses de comunicación serie síncrona (I2C, SPI)

6.4.2.4.1. Protocolo I2C

Para implementar este protocolo se utiliza el *hardware* que trae implementado el microcontrolador MSSP (Master Synchronous Serial Port). La línea SCL (reloj) corresponde al pin RB1 y el pin SDA (dato) al pin RB0. Además se han reservados dos líneas más para el control del protocolo.

La CPU ya trae implementadas las resistencias de polarización que exige el protocolo.

6.4.2.4.2. Protocolo SPI

Igual que para el I2C, éste utiliza el *hardware* MSSP, que corresponde a los pines RA5 la línea SS, RB0 la línea SDA, RB1 la línea SDL y RC7 la línea SDO.

La CPU sólo trabajará en modo maestro ya que la línea SS no se está utilizando y cabe resaltar que éste protocolo no se podrá utilizar en conjunto con el RS232 ya que comparten la línea SDO.

6.4.2.4.3. Buses de comunicación paralela

Es un sistema que está compuesto por un bus de datos, un bus de dirección y un bus de control. Se utiliza para comunicar otros tipos de dispositivos de forma paralela como puede ser la tarjeta de entradas y salidas digitales o pantallas LCD.

El bus de datos es a ocho bit ubicado en el puerto D del micro controlador, es bidireccional se puede leer y escribir.

El bus de direcciones es a tres bit ubicado en el puerto E del micro controlador, sólo se puede escribir y se utiliza para direccionar el número de tarjetas que se conecten al bus paralelo.

El bus de control es a dos bit ubicado en los pines RB6 y RB7 respectivamente, se utiliza para el control de las tarjetas que se conecten al bus paralelo.

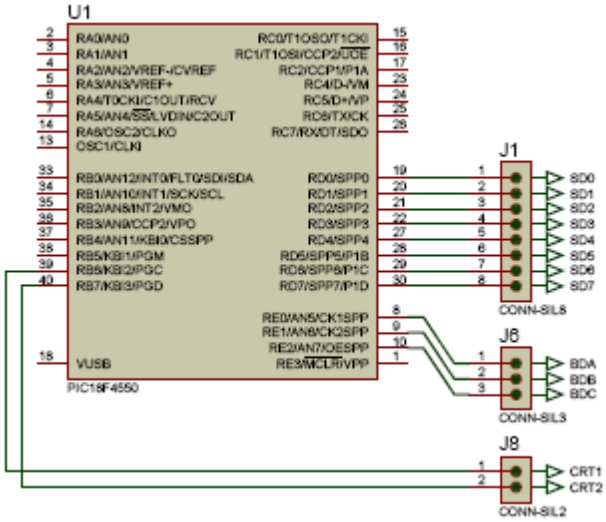


Imagen 24 Diagrama Buses de comunicación paralela.

6.4.2.5. Hardware De Tarjetas De Entradas Y Salidas Digitales

La tarjeta de entradas y salidas digitales se conecta directamente en el bus de comunicación paralela de la CPU y se divide principalmente en dos etapas, una parte de entradas y una de salidas. En la Imagen 25 se muestra el esquema eléctrico de la tarjeta.

Como la etapa de entrada y salida comparten el mismo bus de datos, es necesario realizar un proceso de multiplexar, activar entrada y desactivar salida y viceversa. La multiplexación se encarga de realizar gestionar el bus de dirección y de control.

6.4.2.5.1. Etapa de entradas

Está compuesta por ocho líneas de entrada a 24V, aislada eléctricamente por opto acopladores. Las líneas están conectadas al buffer 74LS245 (flip-flop) que es

controlado por el decodificador 74LS138, encargado de habilitar o deshabilitar el búffer.

6.4.2.5.2. Etapa de salidas

Está compuesta por ocho líneas de salida a relay, cuatro a 24V y las otras configurables a un voltaje menor a 120V.

Cada línea está conectada a un driver que suministra corriente para activar y desactivar los relay. En las entradas del driver se encuentra el buffer 74LS575 que de acuerdo a la configuración de las líneas de dirección y de control refleja el dato del mismo bus en las salidas.

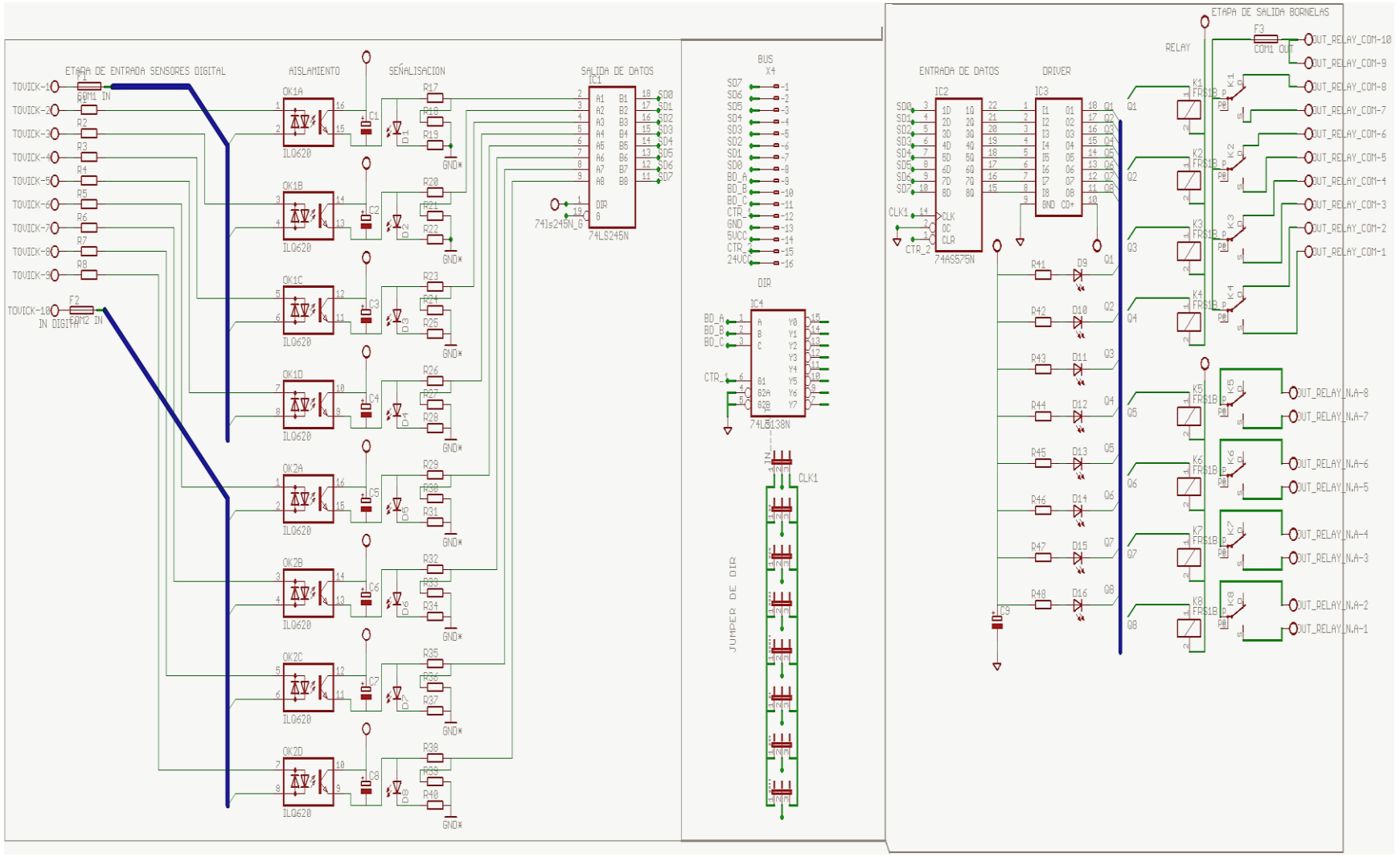


Imagen 25 Diagrama de Tarjetas de entradas y salidas digitales.

6.4.2.6. Diseño del circuito impreso (pcb)

El circuito impreso es una parte importante del sistema, conformado por los diferentes elementos y sus interconexiones. Para su desarrollo, se requiere *software* especializado como lo son Proteus, Orcad, Altium Designer, CadSoft Eagle, entre otros. La elección de la herramienta debe cumplir con los requisitos mínimos de diseño y por ello se eligió CadSoft Eagle, que cuenta con todas las herramientas necesarias y es de fácil manejo, tal como se mostrará a continuación.

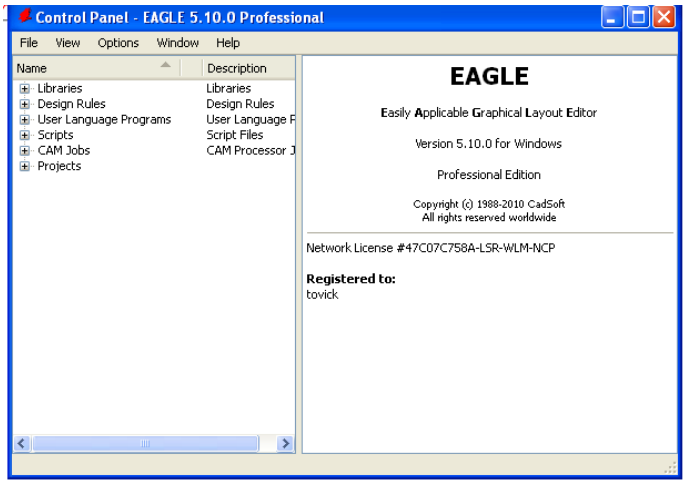


Imagen 26 Software EAGLE 5.10.0 profesional

Eagle es un potente *software* desarrollado por CadSoft para el diseño de circuitos impresos y realizar esquemas electrónicos. Se compone de dos módulos, uno llamado Schematic y el otro Board.

El módulo Schematic se utiliza para el desarrollo del esquema eléctrico. Aquí se puede seleccionar el componente adecuado y mediantes comando, realizar sus conexiones.

El módulo Board se utiliza para realizar el circuito impreso. Aquí se puede elegir la posición de los componentes, su ubicación en la placa, el tamaño de las pistas, etc.

6.4.2.6.1. Diseño del esquemático

Para iniciar la realización del esquema se inicia un nuevo esquemático en *file – new – Schematic*, como se muestra en la imagen 27.

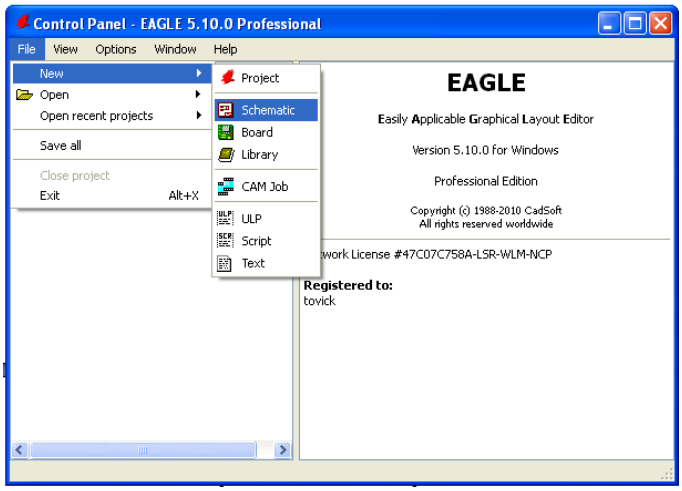


Imagen 27 Abrilr la ventana para nuevo esquemático.

De esta manera se inicia el modulo esquemático, donde puede observar en la barra lateral izquierda las herramientas.

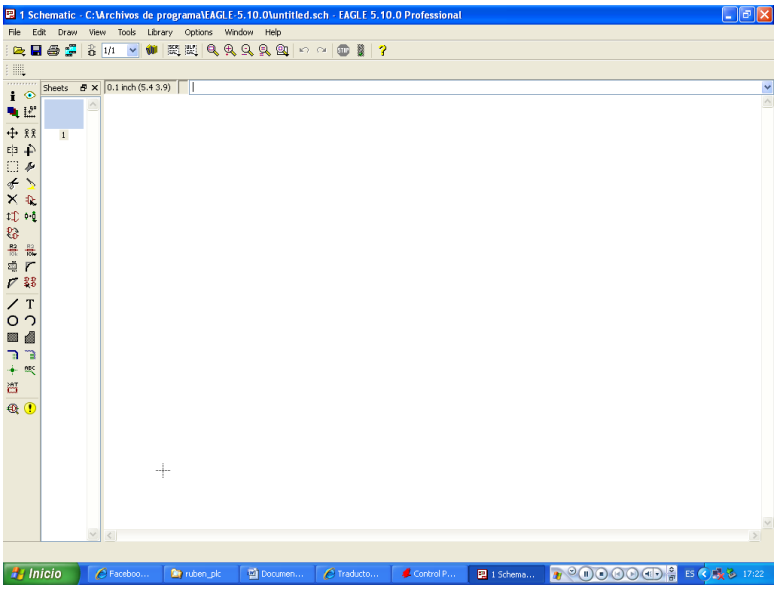


Imagen 28 Ventana para el desarrollo del esquemático en Eagle

Habiendo identificado las herramientas se pasa a la selección de los componentes. Para esto se hace click en *Library – Open library*.

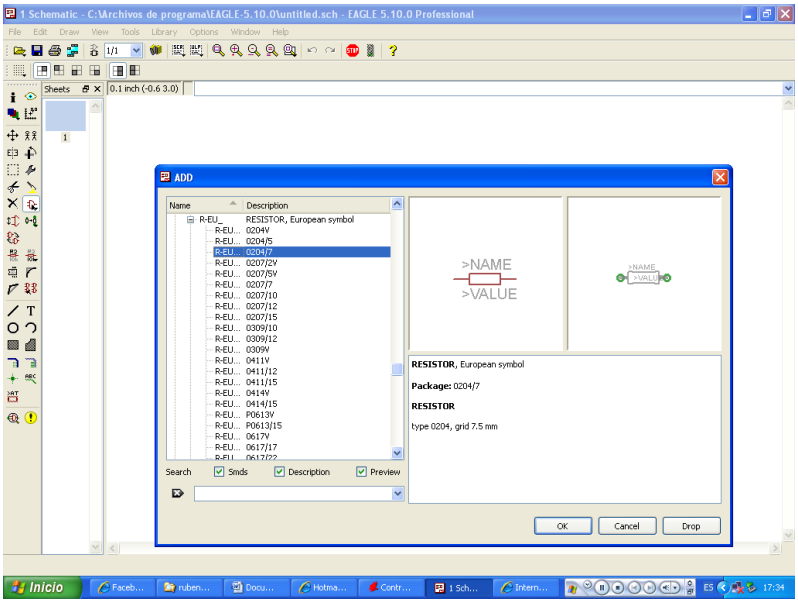


Imagen 29 Agregando elementas con ADD de EAGLE.

Se seleccionan los símbolos en la biblioteca y se dibuja un esquema con éstos, haciendo uso de las herramientas disponibles. De esta manera, se elabora el circuito de entradas y salidas digitales del proyecto.

Ya terminado el esquemático se pasa a la elaboración del PCB del circuito digital, haciendo clic en el ícono *Switch to Board*. Los paquetes de componentes aparecerán junto a una ventana con las líneas de conexión.

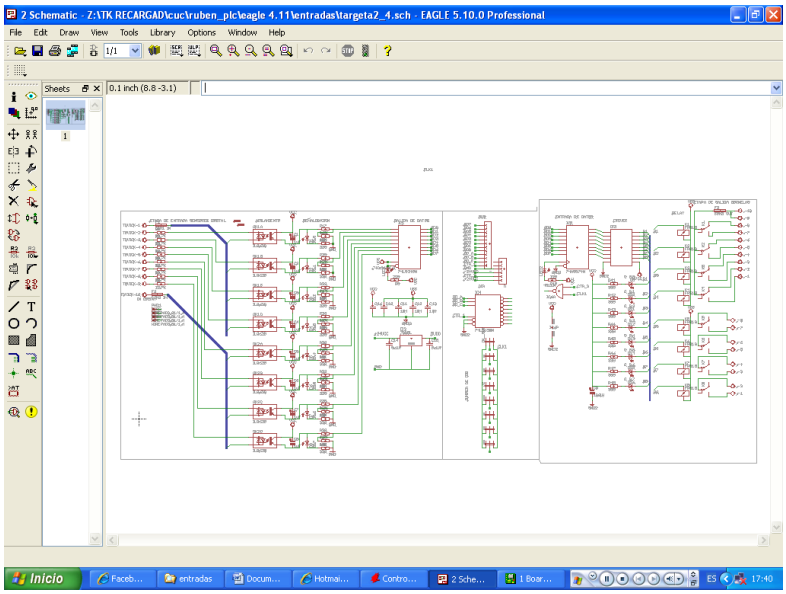


Imagen 30 Esquema desarrollado el EAGLE

Para la elaboración de la board se debe conocer la definición y desarrollo de un PCB en el programa de EAGLE. Se pueden organizar los componentes de la board con las funciones y herramientas ofrecidas por el programa.

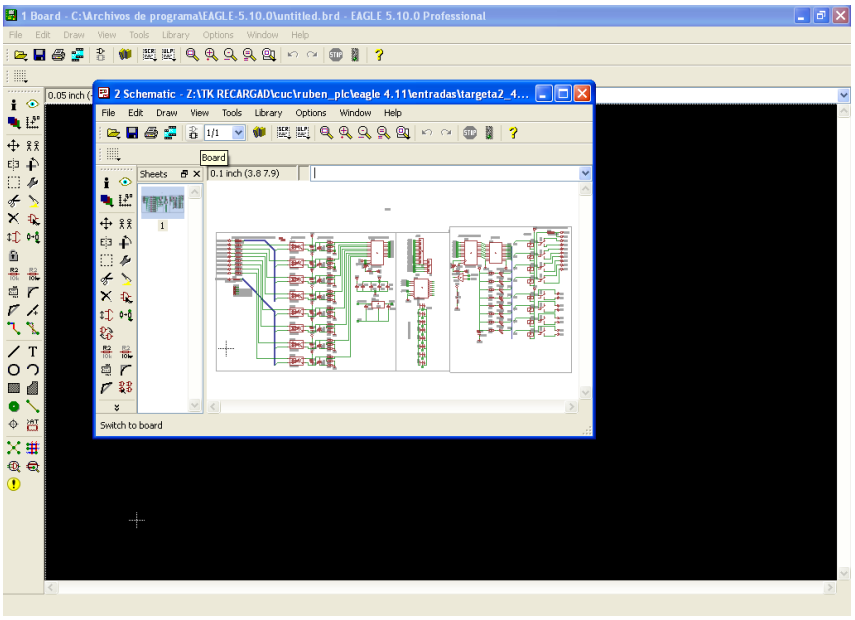


Imagen 31 Módulo Schematic vs. Módulo Board en EAGLE

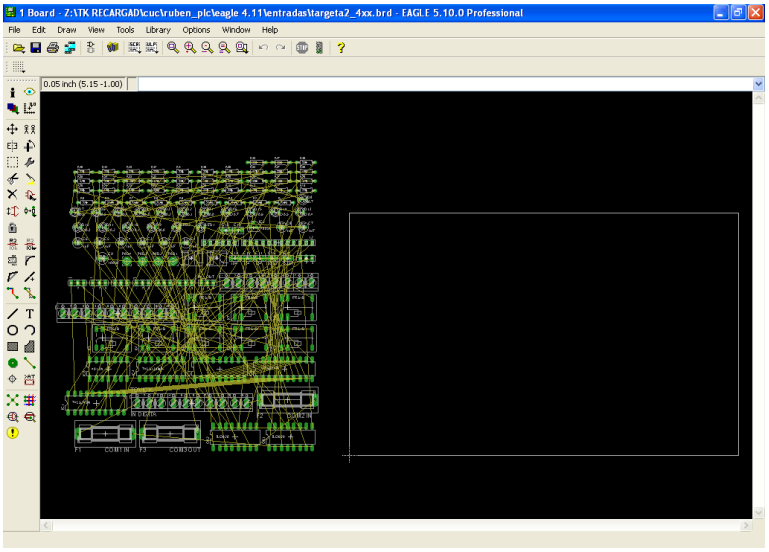


Imagen 32 Diseño del PCB

Para el diseño del impreso, se deben acomodar todos los componentes del circuito con el fin de realizar el trazado de las pistas.

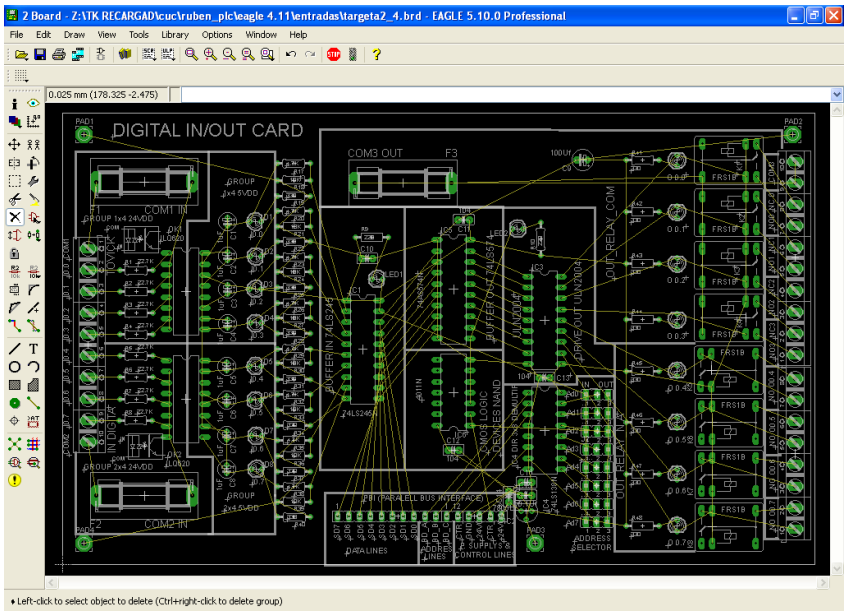


Imagen 33 Ordenar los componentes del circuito para el trazado de pistas.

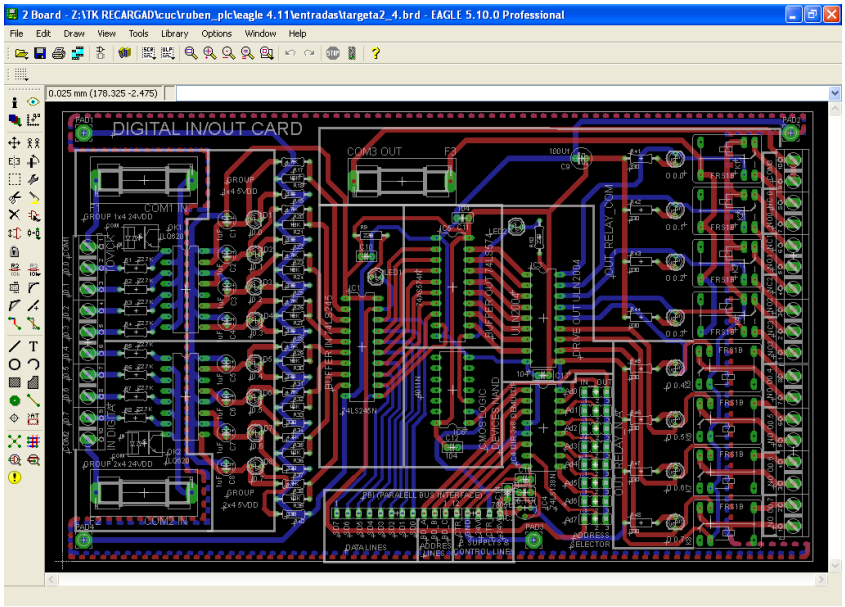


Imagen 34 Trazado de pistas del circuito terminado.

Las conexiones establecidas en el esquemático se convierten en pistas. Este procedimiento se puede hacer a mano, o usando la herramienta AutoRoute, que lo hace de forma automática. Es recomendable hacer a mano las conexiones críticas para mayor seguridad.

Para descartar la posibilidad de errores causados por el trazado manual de pistas, EAGLE lleva a cabo una verificación de Normas de Diseño (RDC). Esta herramienta comprueba si hay cortocircuitos en el tablero o si la distancia entre las pistas pistas no es la adecuada.

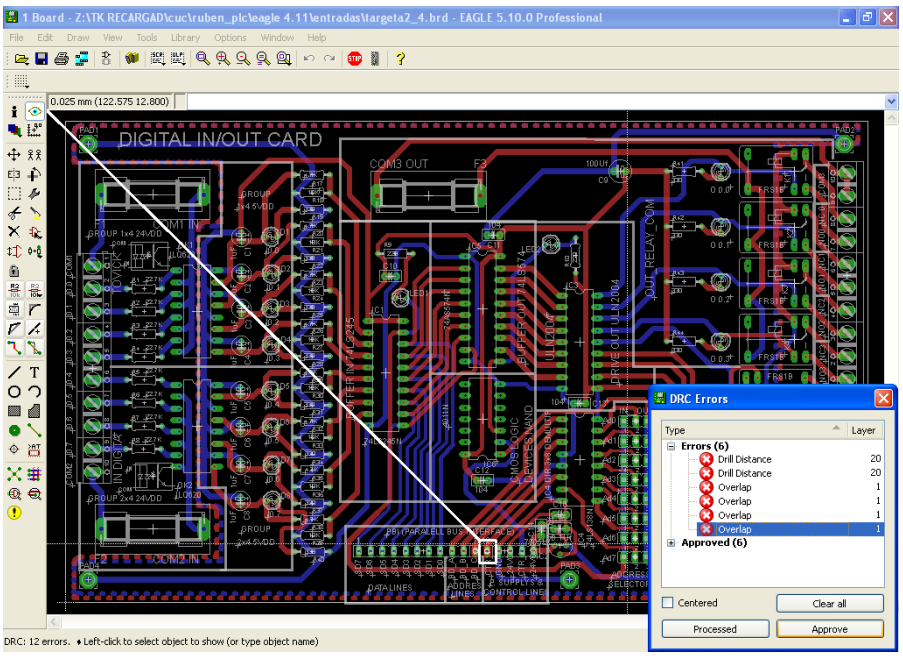


Imagen 35 Verificación de Normas de Diseño (RDC)

.

Se puede modificar o ampliar las bibliotecas existentes para cumplir las necesidades particulares del proyecto. De esta forma, se puede decidir la apariencia de los símbolos, la etiqueta, qué tamaño debe tener, forma y tamaño de los pines, orientación, entre otros.

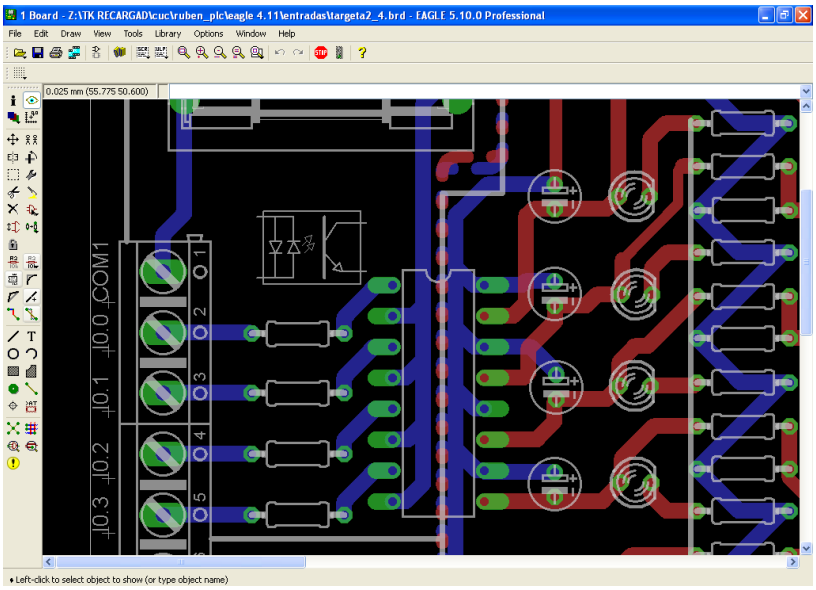


Imagen 36 Diseño de un símbolo para un optoacoplador ILQ620.

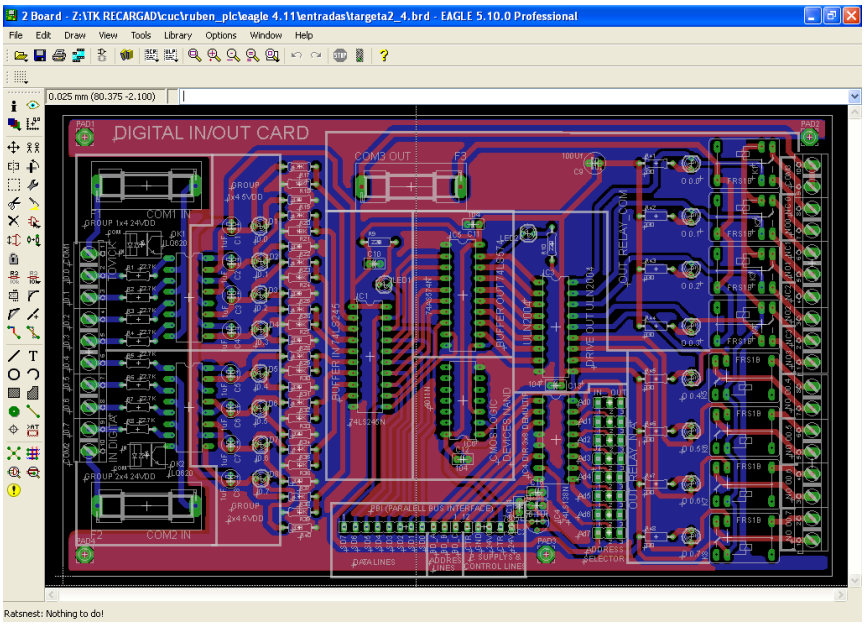


Imagen 37 Diseño del PCB entrada y salida digital.

6.4.3. Diseño Del Sistema Supervisorio

El sistema supervisorio debe cumplir con dos requisitos básicos:

- Probar de manera completa la capacidad de operación de la CPU junto con sus tarjetas de entradas y salidas.
- Estar basado en una aplicación que se encontraría en el ambiente industrial, para que sirva a los fines pedagógicos del proyecto.

Siguiendo estos requisitos se realiza el siguiente diseño de un sistema de mezclado de dos sustancias:

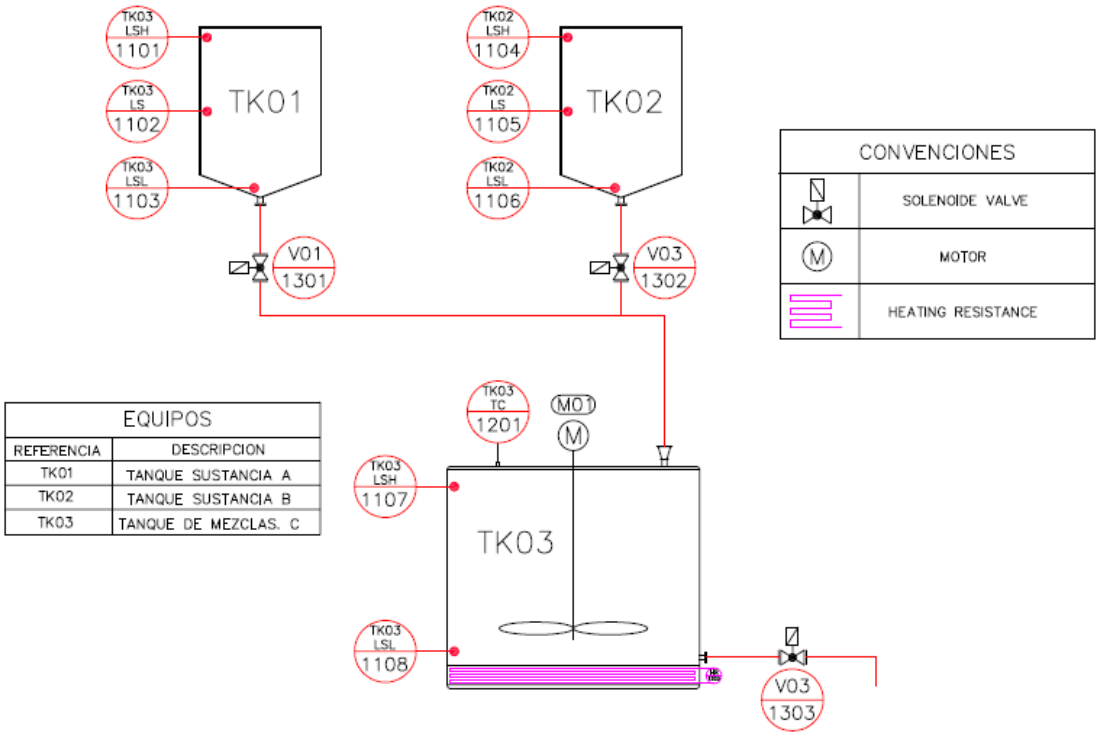


Imagen 38 Sistema de mezclado de dos sustancias.

Como lo muestra la Imagen 38, este sistema realiza la mezcla de dos sustancias contenidas en los tanques TK01 y TK02, en el tanque TK03. La dosificación de estas sustancias se hace a través de electroválvulas, la cantidad a dosificar se controla con sensores de nivel discretos, 3 en los tanques TK01 y TK02, 2 en el tanque TK03. En el tanque TK03 se encuentra un agitador y una resistencia de calor, para lograr una mezcla homogénea de las dos substancias. Finalmente el producto se deposita a través de una tercera electroválvula.

Para cumplir con este procedimiento se comienza por diseñar una tarjeta capaz de controlar todas estas señales e interactuar con la tarjeta entradas y salidas digitales de este PLC. En primera instancia, se realiza el diseño de una fuente de voltaje que permita manejar los niveles de voltaje adecuados para la electrónica. Luego se desarrolla una etapa que permite realizar la medición de nivel, esto con el integrado ULN2803. Para el control de cargas como las electroválvulas, el agitador y la resistencia de calor se utilizan triacs, relevos y optoacopladores, y todo el sistema se integra al PLC.

El esquemático de este sistema se puede observar en la siguiente imagen dividido por etapas.

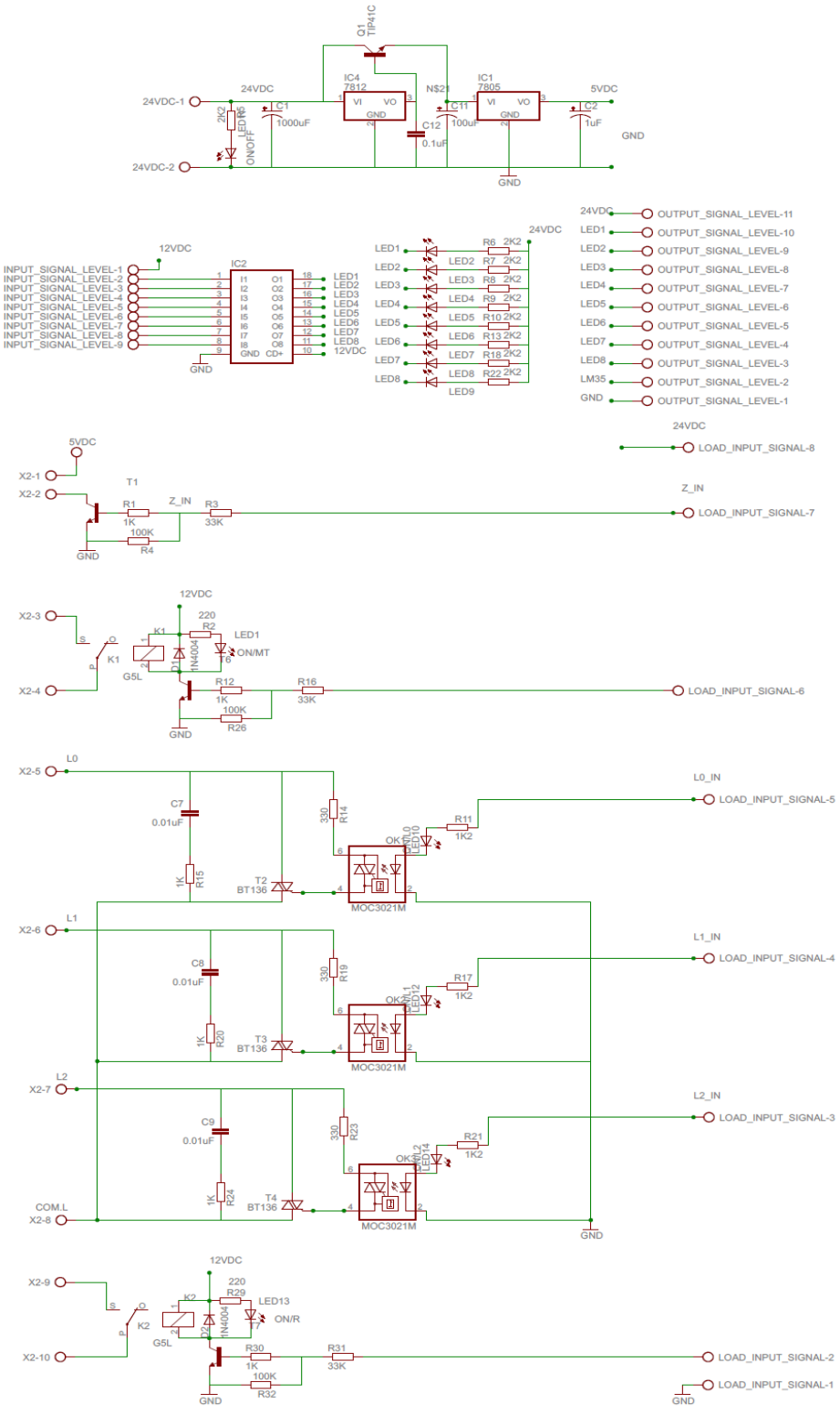


Imagen 39 Sistema de mezclado de dos sustancias, esquematico.

7. CAPITULO 4: IMPLEMENTACIÓN Y PUESTA EN FUNCIONAMIENTO

7.1. CONSTRUCCIÓN DEL HARDWARE

Existen muchas técnicas de fabricación de circuitos impresos, desde las que involucran toda una serie de maquinarias a nivel industrial, hasta las técnicas rudimentarias que utilizan los aficionados. Antes de elegir una técnica, es necesario tener en cuenta el costo de la misma y el grado de complejidad del circuito a desarrollar.

Los circuitos que se pretenden desarrollar poseen muchos elementos y también se requiere que el tamaño del diseño no sea muy grande, esto hace que el diseño sea muy complejo debido al gran número de interconexiones (pistas) entre elementos. Un método que brinda la posibilidad de trabajar con un grueso de pista milimétrico y con costo muy reducido es el de la serigrafía.

La serigrafía consiste en la reproducción de documentos e imágenes sobre cualquier material, y consiste en transferir una tinta a través de una gasa tensada en un marco, el paso de la tinta se bloquea en las áreas donde no habrá imagen mediante una emulsión o barniz, quedando libre la zona donde pasará la tinta [38].

A continuación se explicará el proceso que se llevó a cabo con la serigrafía para la creación de los circuitos impresos.

7.1.1. Construcción de marco o bastidor

El marco es en donde se va a grabar la imagen del circuito que se va a estampar. Está construido en madera y en una de sus caras está recubierto de una

membrana o seda sintética especial para serigrafía. La seda utilizada en el presente proyecto posee una resolución de 120 orificios por centímetro lineal suficiente para que salgan las pistas correctamente.

Como el marco es reutilizable, se puede grabar cientos de veces; se hace con dimensiones que puedan servir para un PCB mucho más grande que los desarrollado en este proyecto. En este caso las dimensiones son de veinte por treinta centímetro. La Imagen 40 muestra el marco terminado.

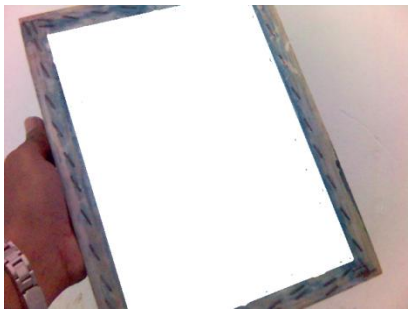


Imagen 40 Marco serigráfico

7.1.2. Etapa de revelado

El revelado es el proceso en donde la imagen del circuito queda grabada en la seda del marco. Para hacer este proceso es necesario contar con una emulsión y una máquina de revelado.

Esta emulsión es aplicada al marco en el lado de la seda. Es recomendable que la capa de emulsión quede lo más fina posible. Posteriormente es secada con aire caliente, cabe señalar que este proceso se realiza en ausencia de luz. Luego de estar un poco seca la emulsión en la seda, el marco se coloca en la máquina de revelado colocándolo por encima del circuito impreso que se encuentra grabado en una hoja de block.

Se enciende la luz de la máquina de revelado y se deja hay durante unos minutos (tres aproximadamente). Luego el marco es retirado e inmediatamente se coloca en la llave del agua hasta que se vea el circuito grabado en la seda. El resultado se muestra en la Imagen 41.

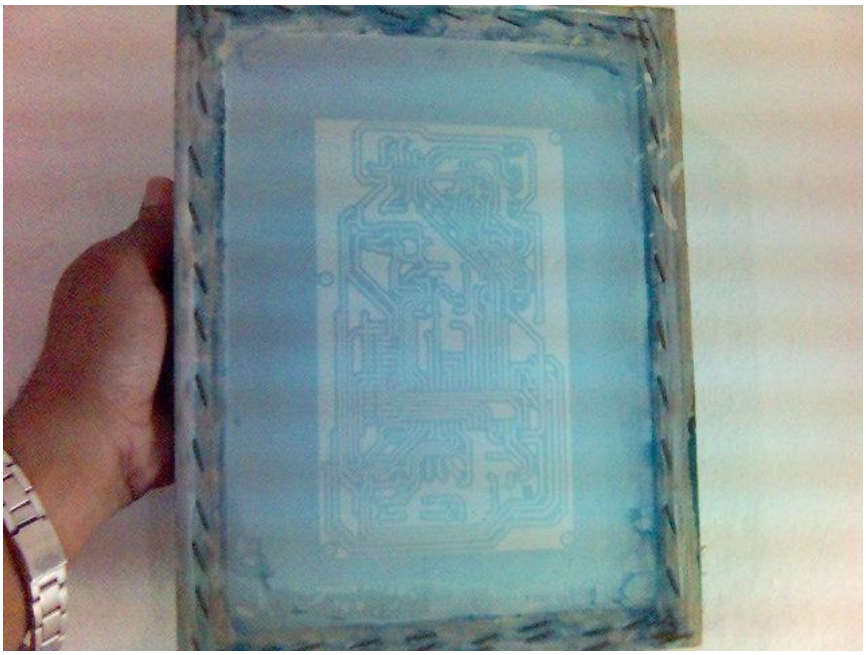


Imagen 41 Revelado De un circuito impreso

7.1.3. Emulsión

Es una sustancia preparada a base de dos componentes que al mezclarse se vuelve sensible a la luz.

7.1.4. Máquina de revelado

Esta es una caja normalmente de madera con un cristal como tapa y con luces de color blanca en su interior, se observa en la Imagen 42.

Imagen 42 Máquina de revelado

7.1.5. Estampado del circuito en la baquelita

Ya teniendo el revelado del circuito en la seda que tiene el marco, se procede a estampar el circuito en la baquelita. Esto se realiza colocando el marco por encima de la baquelita y aplicando una pequeña capa de pintura especial para estampar; luego de regar la pintura con un objeto plano, se levanta el marco y se observa que ya quedo el circuito dibujado en la baquelita (Imagen 43).

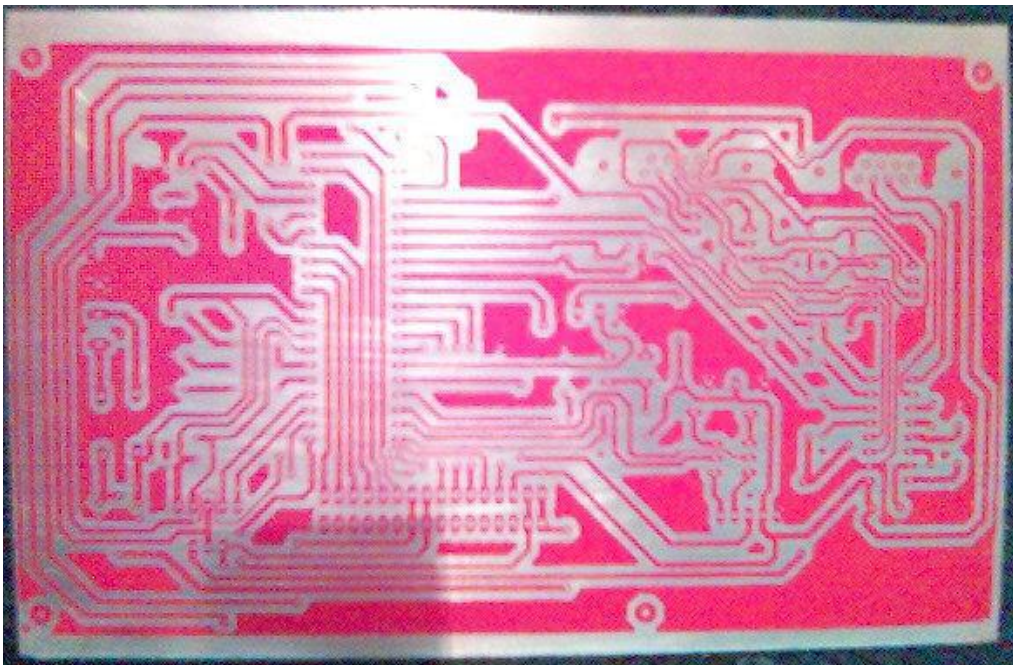


Imagen 43 Circuito de la CPU revelado en la baquelita.

Luego de llevar a cabo estos pasos en el desarrollo del *hardware*, se tiene como resultado la tarjeta de la CPU de este proyecto. Este mismo proceso se aplicó para el desarrollo de las tarjetas de entradas y salidas digitales. Sin embargo, a medida que se desarrolló el proyecto se realizaron una serie de modificaciones sobre la CPU, como se puede observar en la siguiente imagen comparativa:

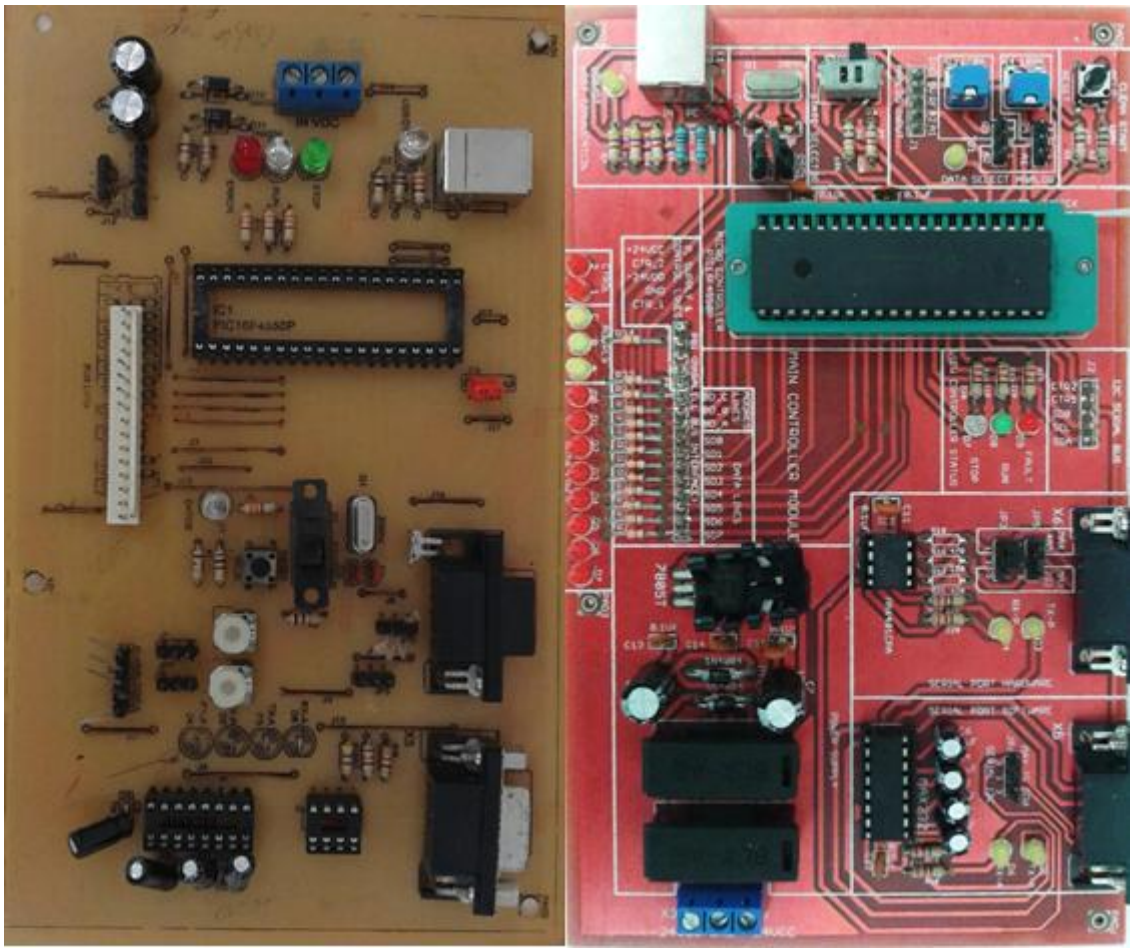


Imagen 44 Comparativa de tarjetas de la CPU. Izquierda CPU Inicial. Derecha, CPU final

A continuación, se muestran tambien imágenes de las tarjetas de entradas y salidas terminadas junto al diseño de la distribucción de pistas que arroja el *Software*.

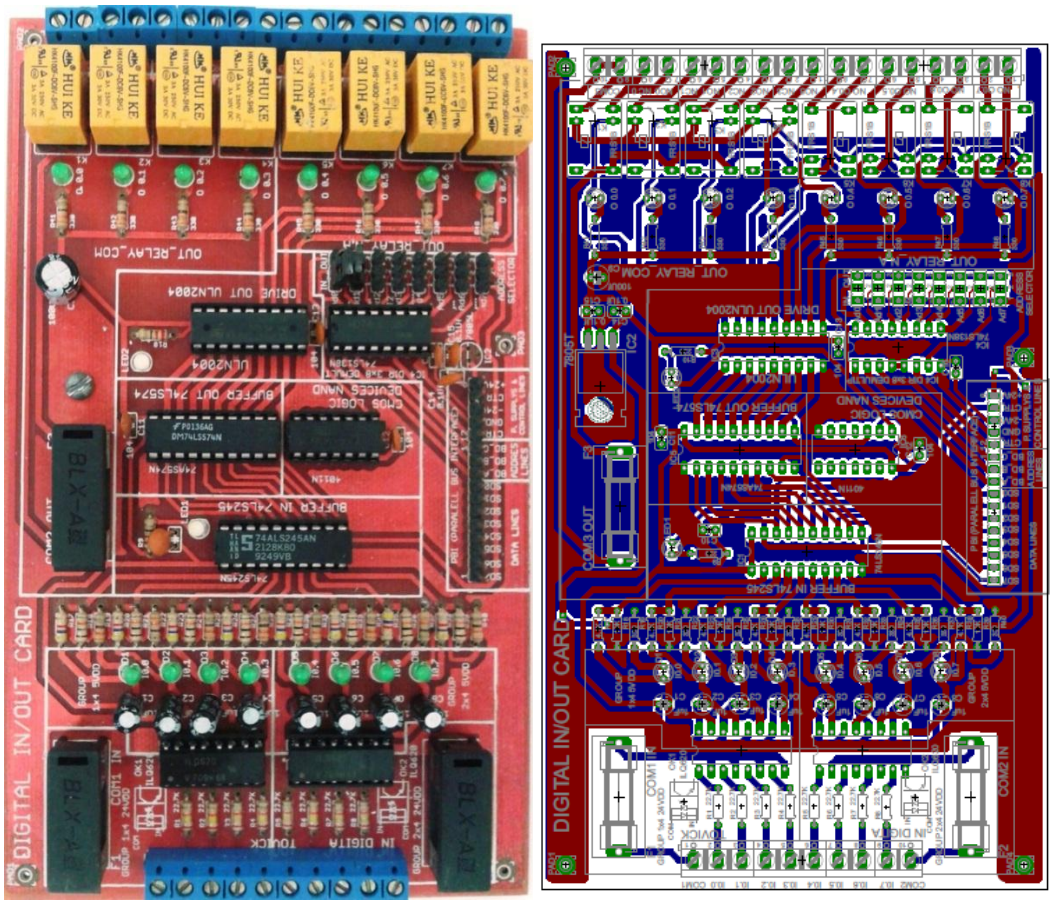


Imagen 45 Tarjeta de entradas y salidas digitales terminada vs diseño.

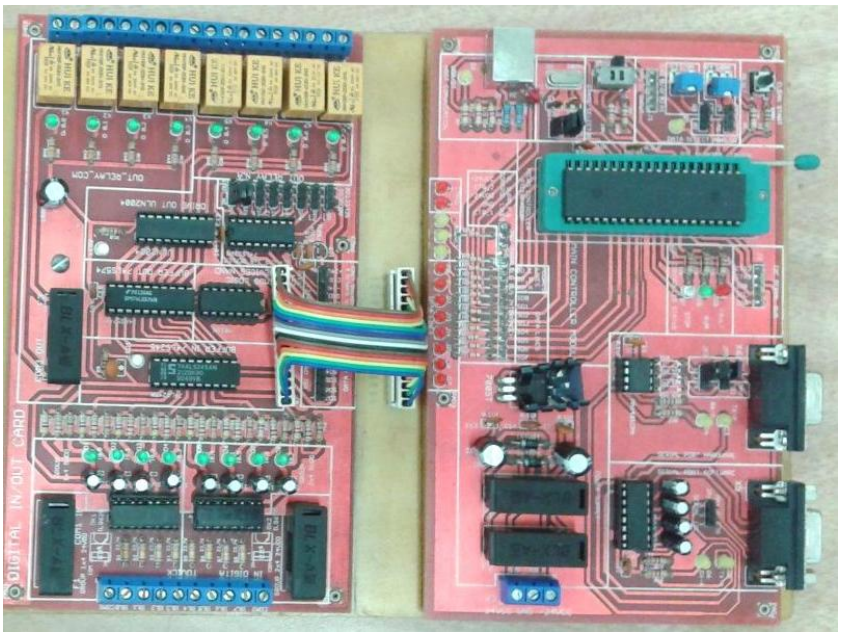


Imagen 46 Hardware completo. Tarjeta de entradas y salidas digitales, y CPU. De izquierda a derecha.

7.2. DESARROLLO DE SOFTWARE

7.2.1. Desarrollo Del Firmware

Evaluando las alternativas para el desarrollo de *firmware* para microcontroladores de microchip, se eligió el lenguaje C por medio del entorno de desarrollo CCS para la programación de PIC. Por ser un compilador muy popular, contar con mucha documentación en internet y por ser fácil de usar.

A continuación se describe la composición de los códigos desarrollados para el manejo de la CPU y sus recursos.

7.2.1.1. Archivo Principal

El archivo principal fue desarrollado para facilitar la programación de aplicaciones del PLC con extensión ".C". Se compone de las cabeceras pertenecientes a la referencia del microcontrolador utilizado, llamado en las librerías a utilizar (RS232, periféricos, etc), la función principal "Void Main()", directivas del preprocesador y demás funciones o declaraciones propias de la aplicación que se vaya a desarrollar.

```
//-------------------------------------------------------------------------//
//                    Configuracion del dispositivo                        //
#include <18F4550.h>                                  // micro controlador utilizado
#device adc=8                                         // conversor analogo digital a 16 bit
#fuses HSPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL5,CPUDIV1,VREGEN
#use delay(clock=20000000)                            // velocidad de la CPU 48 mega herz
//-------------------------------------------------------------------------//

//-------------------------------------------------------------------------//
//                  Configuracion del bus USB                              //
#define USB_HID_DEVICE     FALSE                      //habilitar o desabilitar el modo HID
#define USB_EP1_TX_ENABLE  USB_ENABLE_BULK            //habilitar el modo bulk
#define USB_EP1_RX_ENABLE  USB_ENABLE_BULK            //habilitar el modo bulk
#define USB_EP1_TX_SIZE    15                         //endepoint TX a utilizar y su tamaño
#define USB_EP1_RX_SIZE    15                         //endepoint RX a utilizar y su tamaño
//#define USB_CON_SENSE_PIN  PIN_A5                     //habilitar la deteccion del bus por hardware
//-------------------------------------------------------------------------//

//-------------------------------------------------------------------------//
//               librerias para el manejo del bus USB                      //
#include <pic18_usb.h>                                //Microchip PIC18Fxx5x Hardware layer for CCS's PIC USB driver.
#include "usb_desc_scope.h"                           //
#include <usb.c>                                      //handles usb setup tokens and get descriptor reports.

//-------------------------------------------------------------------------//
//                   Librerias para las tarjetas utilizadas                //
#include "mult_entradasSalidas.c"
//-------------------------------------------------------------------------//
```

Imagen 47 Encabezados y directivas del preprocesador.

```
//-------------------------------------------------------------------------//
//                    Librerias para el manejo de puerto serie             //
#use RS232(UART1, baud=9600)                          //COM principal
//#use RS232(baud=9600, xmit=PIN_C0, rcv=PIN_C1)        //COM auxiliar
//-------------------------------------------------------------------------//

//-------------------------------------------------------------------------//
//       Definicion del area de memoria reservada para el bootloader       //
#define LOADER_END 0x7FF
#define LOADER_SIZE 0x6FF
#build(reset=LOADER_END+1, interrupt=LOADER_END+9)
#org 0, LOADER_END {}
//-------------------------------------------------------------------------//
```

Imagen 48 Encabezados y directivas del preprocesador.

LIBRERÍA “mult_entradasSalidas.c”

Esta librería fue desarrollada para el manejo de las tarjetas de entradas y salidas digitales. En ella se realiza una multiplexacion y a través de un bus se direcciona los datos a leer y a escribir.

Las funciones que contiene la librería son:

- mult_init(): Inicializa registros, variables y la multiplicación entre las tarjetas involucradas. Imagen 50.
- mult_leer(int dirección): Se utiliza para lee los valores de una tarjeta de entradas digitales. Retorna como parámetro el dato leído y acepta como parámetro la dirección (un numero de 0 a 255) de la tarjeta asociada. Imagen 51.
- mult_escribir(int dato, int dirección): Se utiliza para escribir los valores de una tarjeta de salida digital. Acepta como parámetro una dirección y un dato que se va a escribir.

```
//-------------------------------------------------------------------------------//
//             Libreria para el manejo de las entradas y salidas                 //
//                                                                               //
//-------------------------------------------------------------------------------//

//-------------------------------------------------------------------------------//
//                          declaracion de funciones                             //
void mult_init();                              //inicializa la multipleacion
int  mult_leer(int direccion);                  //leer un dato del puerto
void mult_escribir(int dato, int direccion); //escribir un dato en el puerto
int  mult_escr_leer(int dato, int direccion); //lectura y escritura en el puerto
//-------------------------------------------------------------------------------//

//-------------------------------------------------------------------------------//
int  entradas = 0;
int  Salida   = 0;
//-------------------------------------------------------------------------------//
```

```
//-------------------------------------------------------------------------------//
// QOX - SALIDAS
#define Q00 Salida,0
#define Q01 Salida,1
#define Q02 Salida,2
#define Q03 Salida,3
#define Q04 Salida,4
#define Q05 Salida,5
#define Q06 Salida,6
#define Q07 Salida,7

// QOX - ENTRADAS
#define I00 bit_test(entradas,0)
#define I01 bit_test(entradas,1)
#define I02 bit_test(entradas,2)
#define I03 bit_test(entradas,3)
#define I04 bit_test(entradas,4)
#define I05 bit_test(entradas,5)
#define I06 bit_test(entradas,6)
#define I07 bit_test(entradas,7)
//-------------------------------------------------------------------------------//
```

Imagen 49 Contenido Librería "mult_entradasSalidas.c".

```
//-------------------------------------------------------------------------------//
#byte portd     = 0xf83
#bit  conf_CTR1 = 0xf93.6
#bit  conf_CTR2 = 0xf93.7

#bit  CTR1      = 0xf81.6
#bit  CTR2      = 0xf81.7
//-------------------------------------------------------------------------------//

//-------------------------------------------------------------------------------//
//                        INICIALIZACION DE LA MULTIPLEZACION                    //
void mult_init()
{
   conf_CTR1=0;
   conf_CTR2=0;

   CTR1=1;

   set_tris_e(255);
   output_e(0x00);
}
//-------------------------------------------------------------------------------//
```

Imagen 50 Contenido Librería "mult_entradasSalidas.c".

```
//-------------------------------------------------------------------------//
//                          LECTURA DE ENTRADAS                            //
int mult_leer(int direccion)
{
   int dato;

   CTR1=1;
   delay_us(10);
   CTR2=1;

   switch (direccion)
   {
      case 0x00: output_e(0);
                 break;
      case 0x01: output_e(0b00000001);
                 break;
      case 0x02: output_e(0b00000010);
                 break;
      case 0x03: output_e(0b00000011);
                 break;
      default:   //printf("\n\rDireccion no Implementada");
                 break;
   }
   set_tris_d(255);
   portd=0;
   dato=portd;
   CTR1=0;
   return(dato);
}
//-------------------------------------------------------------------------//
```

Imagen 51 Contenido Librería "mult_entradasSalidas.c".

```
//-------------------------------------------------------------------------//
//                        ESCRITURA DE LAS SALIDAS                         //
void mult_escribir(int dato, int direccion)
{
   CTR1=1;
   CTR2=1;

   switch (direccion)
   {
      case 0x00: output_e(0);
                 break;
      case 0x01: output_e(0b00000001);
                 break;
      case 0x02: output_e(0b00000010);
                 break;
      case 0x03: output_e(0b00000011);
                 break;
      default:   //printf("\n\rDireccion no Implementada");
                 break;
   }

   set_tris_d(0);
   output_d(dato);
   CTR1=0;
   CTR2=0;
   delay_us(10);
   CTR2=1;
}
//-------------------------------------------------------------------------//
```

Imagen 52 Contenido Librería "mult_entradasSalidas.c".

Para el desarrollo de aplicaciones, se desarrolló una plantilla que contiene las configuraciones y librerías para comunicarse con las tarjetas de entradas y salidas digitales.

7.3. DESARROLLO DEL SUPERVISORIO.

Para el desarrollo del Sistema supervisorio se utilizó el *software* labview. Luego de comparar este sistema con las otras opciones, se determinó que es el óptimo puesto que cuenta con todas las herramientas necesarias, su implementación es sencilla, ofrece un entorno completo y es fácil de manejar.

Para cumplir con los requisitos planteados para el programa supervisorio, se debieron realizar eficientemente las siguientes funciones.

- Establecimiento de comunicación con CPU
- Control de salidas
- Visualización de entradas

De igual manera debe tener una interfaz que represente la aplicación desarrollada físicamente.

Al evaluar las alternativas y realizar la comparación para el desarrollo del *firmware* para el microcontrolador, se opto por utilizar el compilador de Pic C Compliler CCS, debido a que se contaba con experiencia en el manejo del *software*, mayor documentación en la internet y por lo potente de las librerías para el desarrollo de aplicaciones.

7.3.1. Establecimiento de comunicación con CPU.

Para una mayor facilidad de desarrollo de aplicaciones basadas en el bus USB, Microchip ha creado un archivo .dll en el que proporciona las funciones de acceso al puerto USB con un microcontrolador de la familia PIC18Fxx5x. Para un funcionamiento correcto, se necesita el driver mchpusb.sys. Este archivo funciona perfectamente al accederlo desde LabView [39].

7.3.1.1. Funciones

MPUSBGETDLLVERSION (VOID)
Lee el nivel de revisión del MPUSAPI.dll. Es un nivel de revisión de 32bits. Esta función no devuelve la versión del código, no realiza nada con el USB. Devuelve la versión de la dll en formato Hexadecimal de 32bits.

MPUSBGETDEVICECOUNT (PVID_PID)
Devuelve el número de dispositivo con VID_PID asignado.

- *PVID_PID:* Input: cadena de caracteres del número de identificación asignado.

MPUSBOPEN (INSTANCE, PVID_PID, PEP, DW DIR, DW RESERVED)
Devuelve el acceso al pipe del Endpoint con el VID_PID asignado.
Todas las pipes se abren con el atributo FILE_FLAG_OVERLAPPED. Esto permite que MPUSBRead, MPUSBWrite y MPUSBReadInt tengan un valor de time-out.
Nota: el valor del time-out no tiene sentido en una pipe síncrona.

- *INSTANCE:* Input: Un número de dispositivo para abrir. Normalmente, se utiliza primero la llamada de MPUSBGetDeviceCount para saber cuantos dispositivos hay. Es importante entender que el driver lo comparten distintos dispositivos. El número devuelto por el MPUSBGetDeviceCount tiene que ser igual o menor que el número de todos los dispositivos actualmente conectados y usando el driver genérico.
  Ejemplo: Si hay tres dispositivos con los siguientes PID_VID conectados:

  - Dispositivo tipo 0, VID 0x04d8, PID 0x0001
  - Dispositivo tipo 1, VID 0x04d8, PID 0x0002
  - Dispositivo tipo 2, VID 0x04d8, PID 0x0003

Si el dispositivo que nos interesa tiene VID=0x04d8 y PID=0x0002 el MPUSBGetDeviceCount devolverá un '1'. Al llamar la función tiene que haber un mecanismo que intente llamar MPUSOpen() desde 0 hasta MAX_NUM_MPUSB_DEV. Se tiene que contar el número de llamadas exitosas. Cuando este número sea igual al número devuelto por MPUSBGetDeviceCount, hay que dejar de hacer las llamadas porque no puede haber más dispositivos con el mismo VID_PID.

- *PVID_PID:* Input: String que contiene el PID&VID del dispositivo objetivo. El formato es "vid_xxxx&pid_yyyy". Donde xxxx es el valor del VID y el yyyy el del PID, los dos en hexadecimal.
  Ejemplo: Si un dispositivo tiene un VID=0x04d8 y un PID=0x000b, el string de entrada es: "vid_0x04d8&pid_0x000b".

- *PEP:* Input: String con el número del Endpoint que se va a abrir. El formato es"\\MCHP_EPz" o "\MCHP_EPz" dependiendo del lenguaje de programación. Donde z es el número del Endpoint en decimal.
  Ejemplo: "\\MCHP_EP1" o "\MCHP_EP1"Este argumento puede ser NULL (nulo) para crear lazos con Endpoints de funciones no específicas.
  Las funciones específicas son: MPUSBRead, MPUSBWrite,MPUSBReadInt.
  Nota: Para utilizar MPUSBReadInt(), el formato de pEP tiene que ser"\\MCHP_EPz_ASYNC". Esta opción sólo está disponible para un Endpoint interrupción IN. La pipe de datos abierta con "_ASYNC" debe almacenar datos con el intervalo especificado en el Endpoint descriptor con un máximo de 100 recepciones. Cualquier otro dato recibido después de llenar el buffer del driver se ignora.
  La aplicación del usuario tiene que llamar MPUSBReadInt() a menudo sin superar el máximo de 100.

- *DWDIR:* Especifica la dirección del Endpoint:
    - MP_READ: para MPUSBRead y MPUSBReadInt
    - MP_Write: para MPUSBWrite

- *DWRESERVED:* por ahora nada.

MPUSBREAD (HANDLE, PDATA, DW LEN, PLENGTH, DW MILLISECONDS)

- *HANDLE:* Input: Identifica la pipe del Endpoint que se va a leer. La pipe unida tiene que crearse con el atributo de acceso MP_READ.
- *PDATA:* Output: Puntero al buffer que recibe el dato leído de la pipe.
- *DWLEN:* Input: Especifica el número de bytes que hay que leer de la pipe.
- *PLENGHT:* Output: Puntero al número de bytes leídos. MPUSBRead pone este valor a cero antes de cualquier lectura o de chequear un error.
- *DWMILLISECONDS:* Input: Especifica el intervalo de time-out en milisegundos. La función vuelve si transcurre el intervalo aunque no se complete la operación. Si dwMilliseconds=0, la función comprueba los datos de la pipe y vuelve inmediatamente. Si dwMilliseconds es infinito, el intervalo de time-out nunca termina.

MPUSBWRITE (HANDLE, PDATA, DW LEN, PLENGTH, DW MILLISECONDS)

- *HANDLE:* Input: Identifica la pipe del Endpoint que se va a escribir. La pipe unida tiene que crearse con el atributo de acceso MP_WRITE.

- *PDATA:* Output: Puntero al buffer que contiene los datos que se van a escribir en la pipe.

- *DWLEN:* Input: Especifica el número de bytes que se van a escribir en la pipe.

- *PLENGHT:* Output: Puntero al número de bytes que se escriben al llamar esta función. MPUSBWrite pone este valor a cero antes de cualquier lectura o de chequear un error.

- *DWMILLISECONDS:* Input: Especifica el intervalo de time-out en milisegundos. La función vuelve si transcurre el intervalo aunque no se complete la operación. Si dwMilliseconds=0, la función comprueba los datos de la pipe y vuelve inmediatamente. Si dwMilliseconds es infinito, el intervalo de time-out nunca termina.

MPUSBREADINT (HANDLE, PDATA, DW LEN, P LENGTH, DW MILLISECONDS)

- *HANDLE:* Input: Identifica la pipe del Endpoint que se va a leer. La pipe unida tiene que crearse con el atributo de acceso MP_READ.

- *PDATA:* Output: Puntero al buffer que recibe el dato leído de la pipe.

- *DWLEN:* Input: Especifica el número de bytes que hay que leer de la pipe.

- *PLENGHT:* Output: Puntero al número de bytes leídos. MPUSBRead pone estevalor a cero antes de cualquier lectura o de chequear un error.

- *DWMILLISECONDS:* Input: Especifica el intervalo de time-out en milisegundos. La función vuelve si transcurre el intervalo aunque no se complete la operación. Si dwMilliseconds=0, la función comprueba los datos de la pipe y vuelve inmediatamente. Si dwMilliseconds es infinito, el intervalo de time-out nunca termina.

MPUSBCLOSE (HANDLE)

Cierra una determinada unión.

- *HANDLE:* Input: Identifica la pipe del Endpoint que se va a cerrar.

7.3.2. Entorno Visual Del Sistema De Control Y Lógica De Funcionamiento.

La interfaz visual está basada en la aplicación elegida para la demostración del funcionamiento del sistema: Mezclado de dos sustancias. Utilizando las librerías de Labview se recreó con simbología industrial, con el fin de darle un aspecto similar al utilizado en sistemas supervisorios en plantas industriales. En la siguiente imagen se muestra la pantalla principal del proceso.

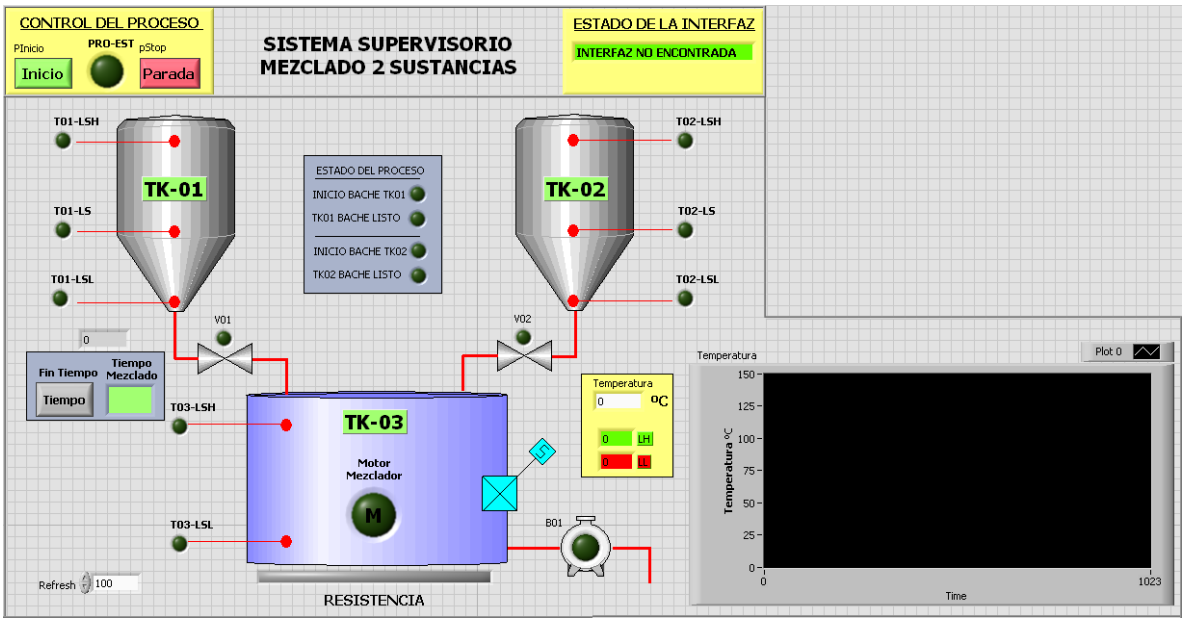


Imagen 53. Entorno visual del supervisorio en LabView.

El entorno visual consiste en dos tanques (TK-01 y TK-02), cada uno con sustancias diferentes. En cada tanque se encuentran instalados tres sensores de

nivel discretos, para medición de nivel: bajo, medio y alto. Ambos tanques dosifican dichas sustancias a un tercer tanque (TK-03), de mayor capacidad. Esta dosificación se realiza actuando dos válvulas solenoides que al momento de abrirlas permiten la caída de las sustancias al tercer tanque por gravedad. El tanque TK-03 cuenta con dos sensores de nivel, uno para el nivel alto y otro para el bajo. Una vez alcanzado el nivel alto, se detiene la dosificación, se activa el mezclador y la resistencia de calefacción durante un tiempo preestablecido por el usuario. Una vez transcurrido este tiempo, se activa la bomba que permite enviar el producto terminado a otra sección de la planta e iniciar el proceso nuevamente. Adicionalmente, esta pantalla tiene:

- Una gráfica que se actualiza cada dos segundos y que muestra la tendencia de la temperatura durante el proceso.
- Un campo donde se visualiza en tiempo real el valor de la temperatura en grados centígrados.
- Indicadores que informan al usuario cuando el sistema está listo para iniciar el proceso.

En la siguiente imagen, se observa el código en el que se establece como condición que cuando no hay conexión de la tarjeta se deshabilitan los botones de control y las luces de indicación.

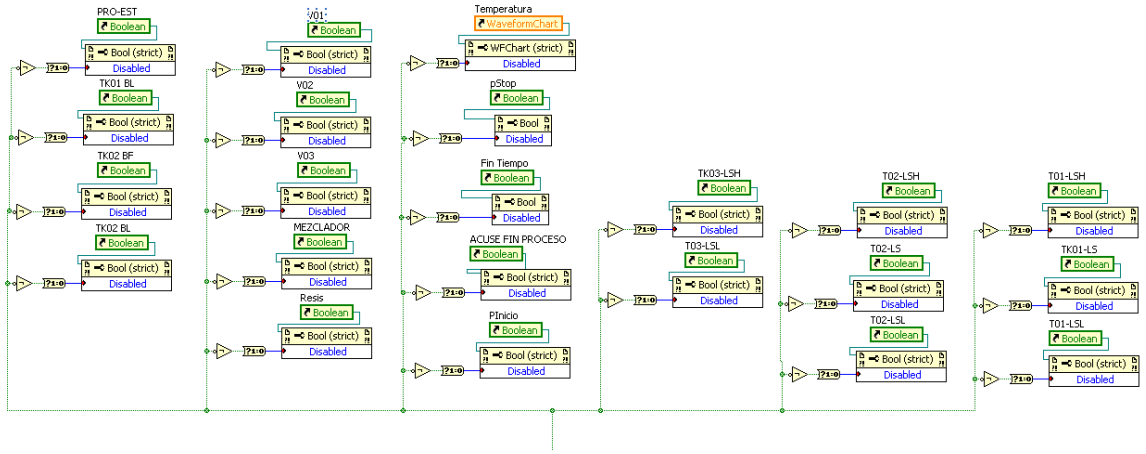


Imagen 54 Lógica de control en LabView – Control de desactivación.

Luego se encuentra el VI que realiza el intercambio de datos con la tarjeta. Se encarga de realizar la actualización de datos cada vez que es modificado desde la pantalla.

El VI acepta como parámetro el identificador PID_&_VID para establecer la comunicación con la tarjeta a través del bus USB.

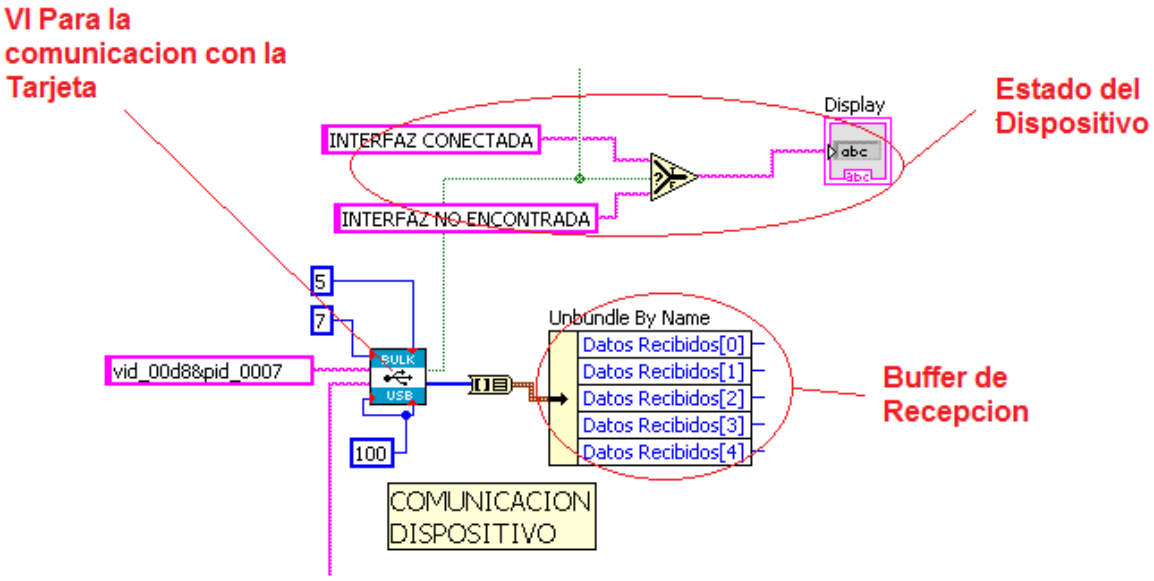


Imagen 55 Lógica de control en LabView – Envío y recepción de datos con la tarjeta.

En la imagen 55, se ilustra la forma en cómo se reciben los datos y se organizan. en la imagen 56, se muestra la forma en cómo se organizan los tipos datos digitales.

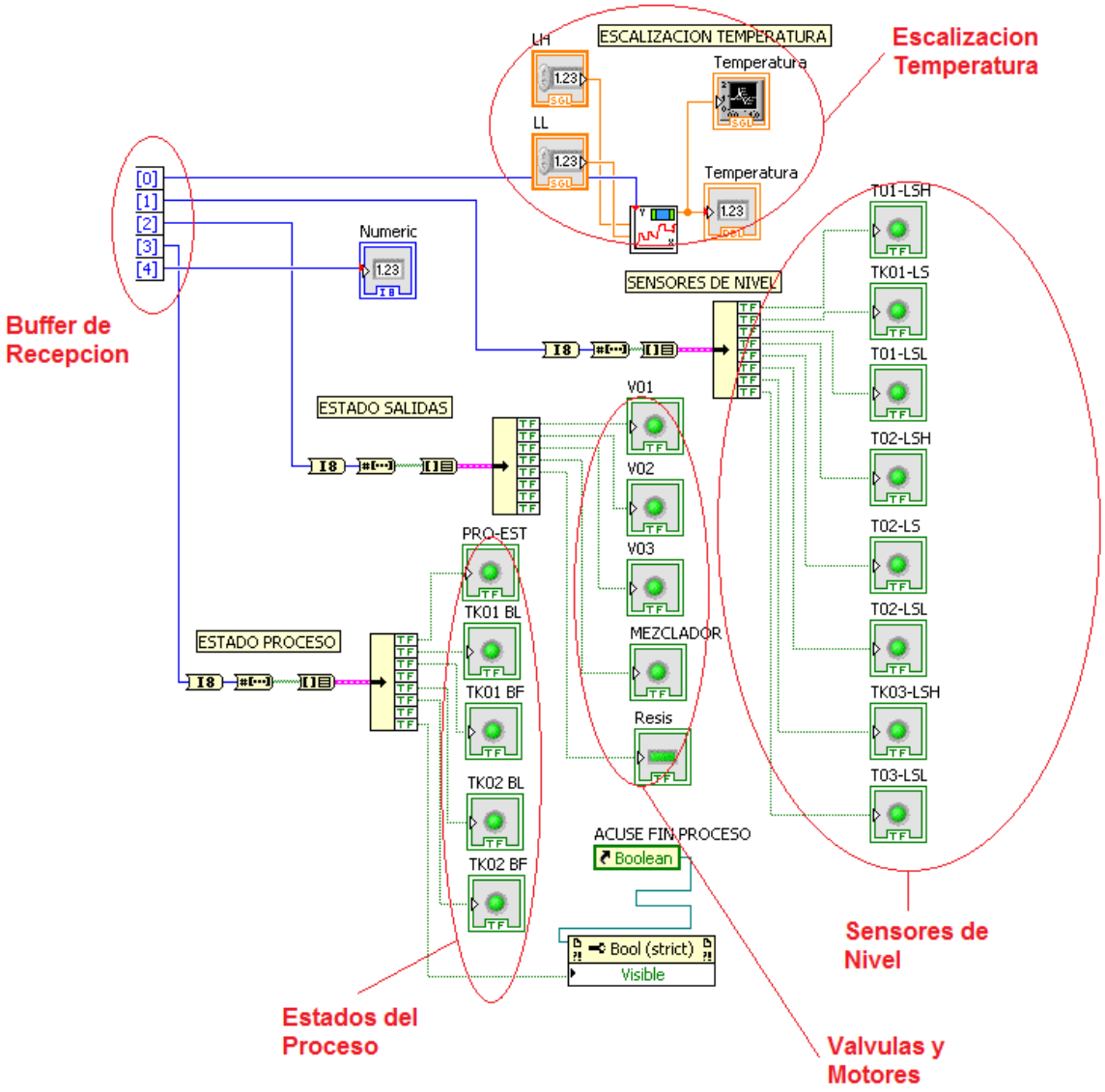


Imagen 56 Lógica de control en LabView – Organización de los datos recibidos.

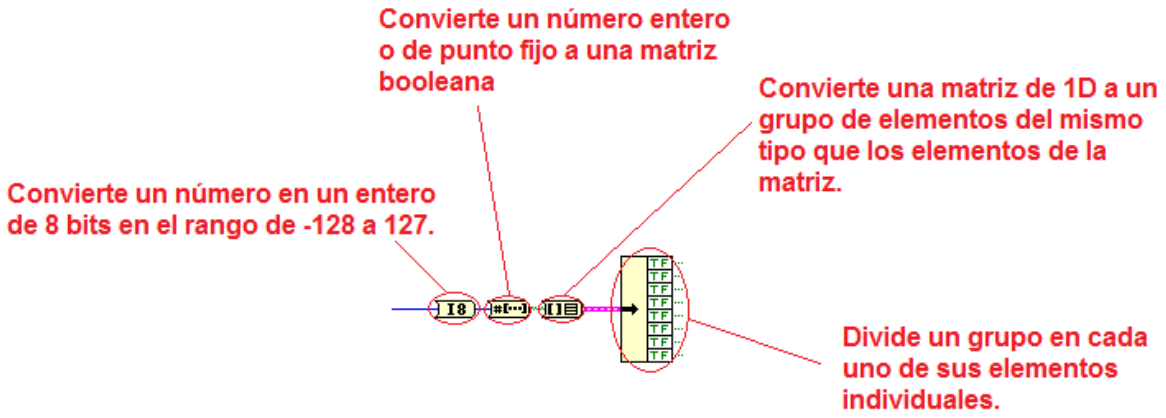


Imagen 57 Lógica de control en LabView – Interpretación y organización de los datos recibidos.

En la imagen 58 se muestra la forma en que se envían los comandos de control a la tarjeta, los comandos que se envían son datos tipo carácter (chart) y se concatenan y se envían como un string al buffer de envio.

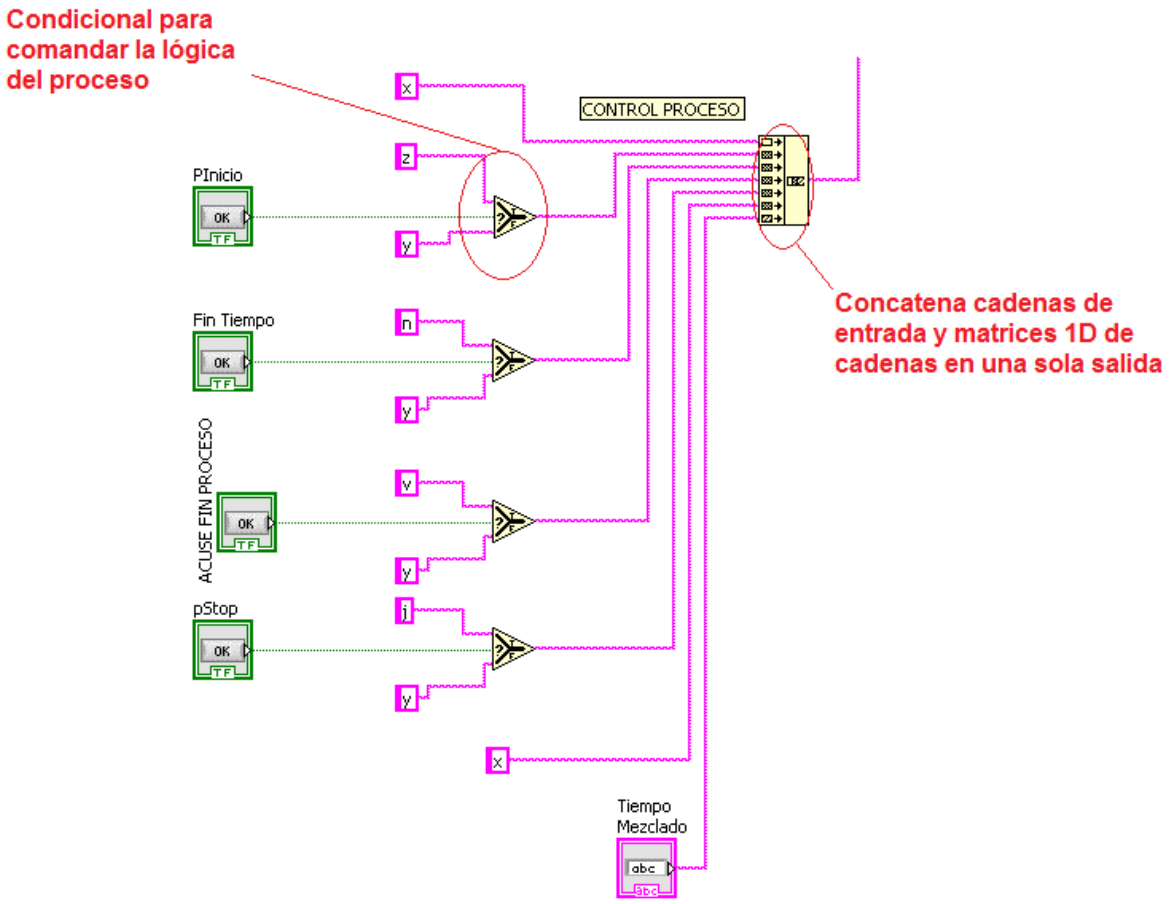


Imagen 58 Lógica de control en LabView – Envío de comandos a la tarjeta.

En la imagen 59 se ilustra la forma en como se controla el tiempo de actualización de los datos, entre la aplicación en labview y la tarjeta.

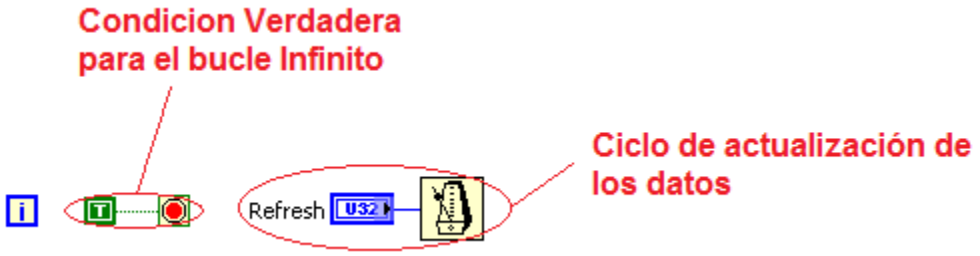


Imagen 59 Lógica de control en LabView – Tiempo para la actualización de dato.

7.4. DESARROLLO DE APLICACIÓN

Siguiendo lo planteado durante la etapa de diseño, se construyó la maqueta que simula el proceso que controlara al PLC. Para esto se utilizaron materiales comúnmente encontrados en ferreterías y misceláneas. A continuacion se muestran imágenes de de la maqueta terminada.

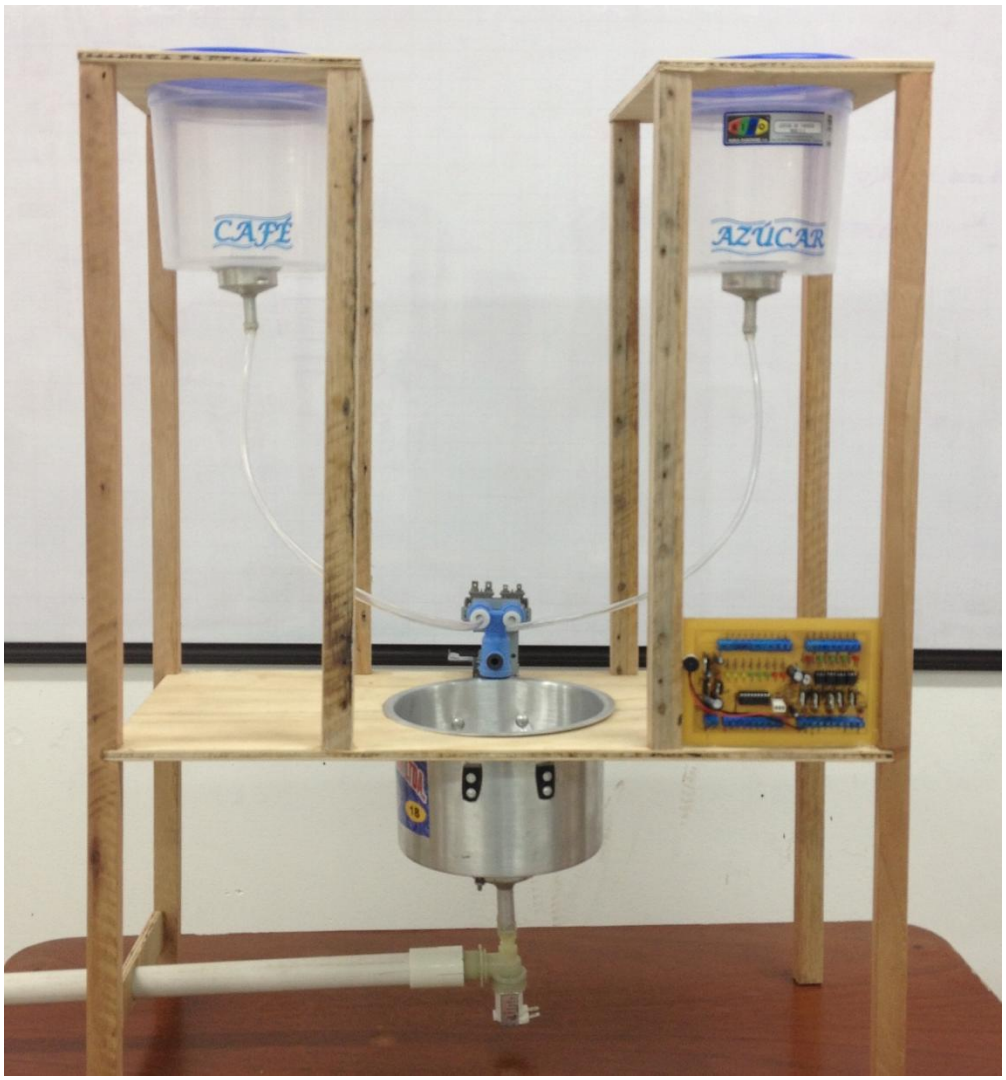


Imagen 60 Maqueta para simulación terminada.

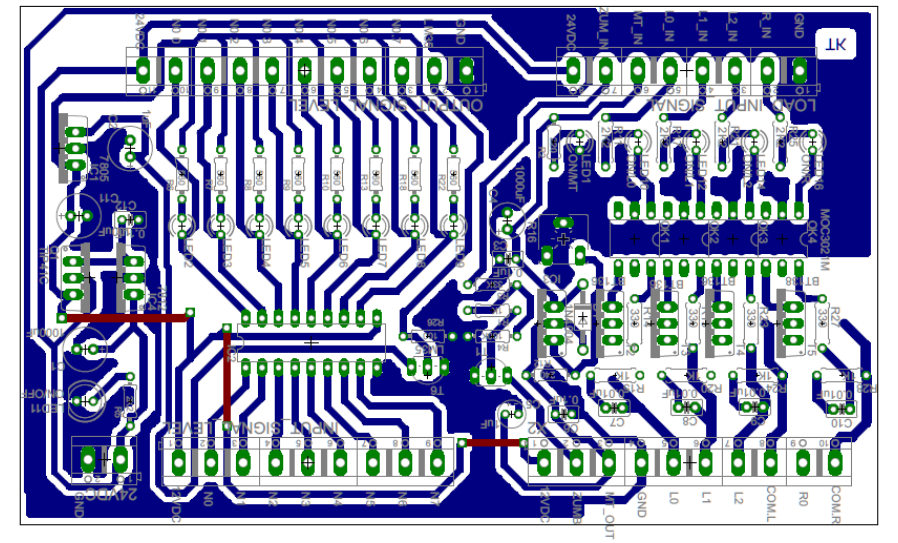


Imagen 61 Tarjeta de control de maqueta. Diseño vs estado final.

8. CAPITULO 5: PRUEBAS DE FUNCIONAMIENTO Y ENTREGA.

Una vez terminada la etapa de desarrollo, se procedió a realizar pruebas sobre el autómata programable. Primero, se hicieron pruebas de comunicación entre la CPU y la tarjeta de entradas y salidas digitales, cargando un programa que activaba secuencialmente las salidas, con el que se obtuvo un buen funcionamiento. Luego se cargó otro programa que activaba las salidas dependiendo del estado de las entradas digitales, también obteniendo resultados positivos. Estas pruebas permitieron no sólo verificar el correcto funcionamiento de la tarjeta de entradas y salidas, sino también la capacidad de procesamiento de la CPU, su velocidad y el óptimo funcionamiento del bus paralelo.

Posteriormente, se configuró el driver de comunicación USB, usando la librería USB MPUSBApi.dll y se creó una pantalla sencilla en Labview para observar el estado de las entradas y forzar el estado de las salidas, obteniendo resultados óptimos.

Con estas pruebas se verificó la comunicación entre el autómata y Labview.

Terminada esa fase de pruebas, se procedió a realizar los ensayos finales con el Sistema supervisorio actualizado, la lógica de control para dicho proceso en la CPU y la maqueta de aplicación conectaba a la tarjeta de entradas y salidas. En este punto se corrigieron errores de direccionamiento, mejoró la visualización, se cambiaron algunos materiales de la maqueta hasta lograr simular varios batches del procesos de forma continua.

A continuación, se encuentra un enlace a un video de la aplicación en funcionamiento: http://youtu.be/IxuKaQQfDj8

9. CONCLUSIONES

- Se implemento un sistema demostrativo de un proceso industrial, en donde se pudo probar el prototipo desarrollado y verificar su funcionamiento validando su aplicabilidad en múltiples sistemas de control.
- Las alternativas utilizadas para el desarrollo del *software*, *hardware* y sistema supervisorio cumplieron a cabalidad con las exigencias de este proyecto. El MPLAB y su integración con el PIC C Compiler, ayudaron a desarrollar una aplicación más estable y versátil, las librerías preestablecidas y el *firmware* desarrollado para el control del *hardware* permiten aplicar este autómata en una gran diversidad de aplicaciones que se encuentran comúnmente en la industria y con las cuales se puede enseñar el funcionamiento de los mismos. La implementación de EAGLE permitió desarrollar tarjetas de estado sólido, de tamaño reducido, que garantizan un funcionamiento óptimo. El sistema de supervision emula los equipos o máquinas que se pueden encontrar en la industria, esto a su amplia librería de elementos.
- En comparación a autómatas programables de tipo genérico, el creado en este proyecto presenta como desventaja el no tener un entorno dedicado de programación propiamente organizado, ni soporte de los lenguaje de programación mayormente difundidos en este tipo de equipos, y establecidos como estandar por la IEEE, sin embargo el programar en C permite un buen empalme entre los lenguajes de programación convencionales y los ya mencionados.
- Es posible agregar tarjetas de entradas y salidas analógicas, con el fin de analizar el comportamiento del autómata bajo este tipo de señales y de esta manera, que los estudiantes afiancen aún más sus conocimientos en automatización.

GLOSARIO

A continuación se presentan los términos, abreviaciones y siglas utilizadas en el proyecto.

- **PLC:** Controlador Lógico Programable (en inglés Programmable Logical Controller).
- **AP:** Autómata programable.
- **API:** Application Program Interface. Conjunto de funciones provistas por una librería para extender la funcionalidad de un lenguaje de programación para una funcionalidad particular.
- **USB**: Bus De Serie Universal.
- **Endpoint:** entidad lógica que usan los dispositivos para comunicarse por el protocolo USB.
- **PCB:** Printer Circuit Board; Circuito Impreso.
- **Pipe**: En USB, es el canal de comunicación de conecta al host controlador con el endpoint.
- **PIC:** IDE y compilador para programación de PIC Microchip en el lenguaje C.
- **E/S:** Entradas y Salidas.
- **SCADA:** es un *software* para ordenadores que permite controlar y supervisar procesos industriales a distancia. Facilita retroalimentación en tiempo real con los dispositivos de campo (sensores y actuadores) y controlando el proceso automáticamente. Provee de toda la información que se genera en el proceso productivo (supervisión, control calidad, control de producción, almacenamiento de datos, etc.) y permite su gestión e intervención.
- ***Firmware*:** bloque de instrucciones de máquina para propósitos específicos, grabado en una memoria de tipo de solo lectura (ROM, EEPROM, flash,

etc), que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo.

10. BIBLIOGRAFÍA

[1] R. Sánchez, Controlador lógico programable, una mirada interna: implementación académica del dispositivo. Educosta, 2009.

[2] “Que es un PLC (Avanzado).” [Online]. Available: http://www.rocatek.com/forum_plc2.php. [Accessed: 19-Mar-2013].

[3] “Autómata programable - Wikipedia, la enciclopedia libre.” [Online]. Available: http://es.wikipedia.org/wiki/Aut%C3%B3mata_programable. [Accessed: 19-Mar-2013].

[4] “Welcome to IEC - International Electrotechnical Commission.” [Online]. Available: http://www.iec.ch/. [Accessed: 22-Mar-2013].

[5] “definicion-de-Los Controladores Lógicos Programables | ingeniaste.com.” [Online]. Available: http://www.ingeniaste.com/ingenias/telecom/tutorial-plc.html. [Accessed: 19-Mar-2013].

[6] “Automatizacion industrial mediante PLCs.” [Online]. Available: http://davidrojasticsplc.wordpress.com/. [Accessed: 19-Mar-2013].

[7] “Manual para el participante: Control Lógico Programable,” Instituto Nacional Tecnológico, Dirección General de formación profesional, Jun. 2008.

[8] F. Mateos, “Estandarización con autómatas programables,” Universidad de Oviedo, Nov-2001.

[9] “Hardware,” Hardware, Feb-2010. [Online]. Available: http://40138hardware.blogspot.com/2010_02_01_archive.html. [Accessed: 19-Mar-2013].

[10] E. Erazo, “Diseño e implementación de una computadora a bordo de control de mantenimiento en un vehículo Suzuki Forsa.” LATACUNGA / ESPE / 2009, Ene-2009.

[11] “USB y PIC 18F2550: Desarrollo de proyectos en ASM.” [Online]. Available: http://www.ucontrol.com.ar/forosmf/programacion-en-asm/usb-y-pic-18f2550-desarrollo-de-proyectos-en-asm/. [Accessed: 19-Mar-2013].

[12] Microchip, “PIC18F2455/2550/4455/4550 Data Sheet.” Microchip Technology Inc., 2006.

[13] “Dispositivos de medición y captadores de datos Dispositivo para el monitoreo de temperatura, humedad relativa, luz.” [Online]. Available: http://dspace.uniminuto.edu:8080/xmlui/handle/10656/1309. [Accessed: 19-Mar-2013].

[14] “Electrónica de control - Wikipedia, la enciclopedia libre.” [Online]. Available: http://es.wikipedia.org/wiki/Electr%C3%B3nica_de_control. [Accessed: 19-Mar-2013].

[15] “RS232 Tutorial on Data Interface and cables.” [Online]. Available: http://www.arcelect.com/rs232.htm. [Accessed: 19-Mar-2013].

[16] “Robolution | Upcoming.” [Online]. Available: http://robolution.in/upcoming. [Accessed: 19-Mar-2013].

[17] “Lenguajes de Programacion | Carlos2512’s Blog.” [Online]. Available: http://carlos2512.wordpress.com/lenguajes-de-programacion/. [Accessed: 19-Mar-2013].

[18] “paoye.” [Online]. Available: http://paoye.blogia.com/. [Accessed: 19-Mar-2013].

[19] “CONOCIMIENTOS Y LAS NUEVAS TECNOLOGIA: Que es un Lenguaje de programación.” [Online]. Available: http://losprogramadores65.blogspot.com/2011/02/que-es-un-lenguaje-de-programacion.html. [Accessed: 19-Mar-2013].

[20] B. Wilson, Comparative Programming Languages, Second. Addison-Wesley, 1993.

[21] “LabVIEW - Wikipedia, la enciclopedia libre.” [Online]. Available: http://es.wikipedia.org/wiki/LabVIEW. [Accessed: 19-Mar-2013].

[22] “Simulación de Circuitos con Multisim: Tutorial Introductorio de LabVIEW – Parte 1.” [Online]. Available: http://blogmultisim.blogspot.com/2011/06/tutorial-introductorio-de-labview-parte.html. [Accessed: 19-Mar-2013].

[23] “CCS, Inc. - CCS C Compiler Details.” [Online]. Available: http://www.ccsinfo.com/content.php?page=compiler-details. [Accessed: 19-Mar-2013].

[24] “mikroC PRO for PIC - C compiler for Microchip PIC microcontrollers.” [Online]. Available: http://www.mikroe.com/mikroc/pic/. [Accessed: 19-Mar-2013].

[25] “Proton BASIC Compiler - Proton IDE.” [Online]. Available: http://www.protonbasic.co.uk/content.php/935-Proton-IDE. [Accessed: 19-Mar-2013].

[26] “MPLAB Integrated Development Environment.” [Online]. Available: http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002. [Accessed: 19-Mar-2013].

[27] “Office Developer Tools for Visual Studio 2012 | Microsoft Visual Studio.” [Online]. Available: http://www.microsoft.com/visualstudio/eng/products/visual-studio-overview. [Accessed: 19-Mar-2013].

[28] “Lenguaje Delphi - Wikilibros.” [Online]. Available: http://es.wikibooks.org/wiki/Lenguaje_Delphi. [Accessed: 19-Mar-2013].

[29] “Welcome to NetBeans.” [Online]. Available: http://netbeans.org/. [Accessed: 19-Mar-2013].

[30] “Atmel Corporation - Microcontrollers, 32-bit, and touch solutions.” [Online]. Available: http://www.atmel.com/. [Accessed: 19-Mar-2013].

[31] “Motorola Solutions - Motorola Solutions America Latina.” [Online]. Available: http://www.motorolasolutions.com/XL-ES/Home. [Accessed: 20-Mar-2013].

[32] “Parallax Home.” [Online]. Available: http://www.parallax.com/. [Accessed: 20-Mar-2013].

[33] “Microchip Technology Inc.” [Online]. Available: http://www.microchip.com/. [Accessed: 20-Mar-2013].

[34] “Labcenter Electronics - Professional PCB Design and Simulation Software.” [Online]. Available: http://www.labcenter.com/index.cfm. [Accessed: 20-Mar-2013].

[35] “CadSoft EAGLE PCB Design Software - EAGLE Support, Tutorials, Shop.” [Online]. Available: http://www.cadsoftusa.com/. [Accessed: 20-Mar-2013].

[36] “Cadence OrCAD Solutions.” [Online]. Available: http://www.cadence.com/products/orcad/pages/default.aspx. [Accessed: 20-Mar-2013].

[37] “Altium - Next generation electronics design.” [Online]. Available: http://www.altium.com/. [Accessed: 20-Mar-2013].

[38] “Serigrafía - Wikipedia, la enciclopedia libre.” [Online]. Available: http://es.wikipedia.org/wiki/Serigraf%C3%ADa. [Accessed: 22-Mar-2013].

[39] “libreria MPUSBApi.” [Online]. Available: http://www.scribd.com/doc/97751964/libreria-MPUSBApi. [Accessed: 21-Mar-2013].

Anexos

UNIVERSIDAD DE LA COSTA	NORMAS PARA LA ENTREGA DE TESIS Y TRABAJOS DE GRADO A LA UNIDAD DE INFORMACION	VERSION: 02
		FECHA: Junio 2012
		CODIGO:DOC-VACRE-NETGUDI

ANEXO 1
CARTA DE ENTREGA Y AUTORIZACIÓN DE LOS AUTORES PARA LA CONSULTA, LA REPRODUCCIÓN PARCIAL O TOTAL, Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO DE TESIS Y TRABAJOS DE GRADO

Barranquilla, Fecha

Marque con una X
Tesis [X] **Trabajo de Grado** [ ]

Yo JUAN DAVID COSSIO GOZMAN, identificado con C.C. No. 1140.818.356, actuando en nombre propio y como autor de la tesis y/o trabajo de grado titulado DISEÑO DE UN AUTOMATA PROGRAMABLE PARA AMBIENTE ACADEMICO presentado y aprobado en el año 2013 como requisito para optar al título de INGENIERO ELECTRONICO;
hago entrega del ejemplar respectivo y de sus anexos de ser el caso, en formato digital o electrónico (DVD) y autorizo a la UNIVERSIDAD DE LA COSTA, CUC, para que en los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia, utilice y use en todas sus formas, los derechos patrimoniales de reproducción, comunicación pública, transformación y distribución (alquiler, préstamo público e importación) que me corresponden como creador de la obra objeto del presente documento.

Y autorizo a la Unidad de información, para que con fines académicos, muestre al mundo la producción intelectual de la Universidad de la Costa, CUC, a través de la visibilidad de su contenido de la siguiente manera:

Los usuarios puedan consultar el contenido de este trabajo de grado en la página Web de la Facultad, de la Unidad de información, en el repositorio institucional y en las redes de información del país y del exterior, con las cuales tenga convenio la institución y Permita la consulta, la reproducción, a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato DVD o digital desde Internet, Intranet, etc., y en general para cualquier formato conocido o por conocer.

El AUTOR - ESTUDIANTES, manifiesta que la obra objeto de la presente autorización es original y la realizó sin violar o usurpar derechos de autor de terceros, por lo tanto la obra es de su exclusiva autoría y detenta la titularidad ante la misma. PARÁGRAFO: En caso de presentarse cualquier reclamación o acción por parte de un tercero en cuanto a los derechos de autor sobre la obra en cuestión, EL ESTUDIANTE - AUTOR, asumirá toda la responsabilidad, y saldrá en defensa de los derechos aquí autorizados; para todos los efectos, la Universidad actúa como un tercero de buena fe.

Para constancia se firma el presente documento en dos (02) ejemplares del mismo valor y tenor, en Barranquilla D.E.I.P., a los 16 días del mes de MAYO de Dos Mil Trece 2013

EL AUTOR - ESTUDIANTE._____
FIRMA

UNIVERSIDAD DE LA COSTA	NORMAS PARA LA ENTREGA DE TESIS Y TRABAJOS DE GRADO A LA UNIDAD DE INFORMACION	VERSION: 02 FECHA: Junio 2012 CODIGO:DOC-VACRE-NETGUDI

ANEXO 1
CARTA DE ENTREGA Y AUTORIZACIÓN DE LOS AUTORES PARA LA CONSULTA, LA REPRODUCCIÓN PARCIAL O TOTAL, Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO DE TESIS Y TRABAJOS DE GRADO

Barranquilla, Fecha

Marque con una X
Tesis ☐ Trabajo de Grado ☐

Yo Luis carlos Calvo Pulgar, identificado con C.C. No. 72 056626, actuando en nombre propio y como autor de la tesis y/o trabajo de grado titulado Diseño de un Automata programable de Ambiente Academico presentado y aprobado en el año 2013 como requisito para optar al título de Ingeniero electronico;
hago entrega del ejemplar respectivo y de sus anexos de ser el caso, en formato digital o electrónico (DVD) y autorizo a la UNIVERSIDAD DE LA COSTA, CUC, para que en los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia, utilice y use en todas sus formas, los derechos patrimoniales de reproducción, comunicación pública, transformación y distribución (alquiler, préstamo público e importación) que me corresponden como creador de la obra objeto del presente documento.

Y autorizo a la Unidad de información, para que con fines académicos, muestre al mundo la producción intelectual de la Universidad de la Costa, CUC, a través de la visibilidad de su contenido de la siguiente manera:

Los usuarios puedan consultar el contenido de este trabajo de grado en la página Web de la Facultad, de la Unidad de información, en el repositorio institucional y en las redes de información del país y del exterior, con las cuales tenga convenio la institución y Permita la consulta, la reproducción, a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato DVD o digital desde Internet, Intranet, etc., y en general para cualquier formato conocido o por conocer.

El AUTOR - ESTUDIANTES, manifiesta que la obra objeto de la presente autorización es original y la realizó sin violar o usurpar derechos de autor de terceros, por lo tanto la obra es de su exclusiva autoría y detenta la titularidad ante la misma. PARÁGRAFO: En caso de presentarse cualquier reclamación o acción por parte de un tercero en cuanto a los derechos de autor sobre la obra en cuestión, EL ESTUDIANTE - AUTOR, asumirá toda la responsabilidad, y saldrá en defensa de los derechos aquí autorizados; para todos los efectos, la Universidad actúa como un tercero de buena fe.

Para constancia se firma el presente documento en dos (02) ejemplares del mismo valor y tenor, en Barranquilla D.E.I.P., a los 21 días del mes de Mayo de Dos Mil Trece 2013

EL AUTOR - ESTUDIANTE. _____
FIRMA

ANEXO 1
CARTA DE ENTREGA Y AUTORIZACIÓN DE LOS AUTORES PARA LA CONSULTA, LA REPRODUCCIÓN PARCIAL O TOTAL, Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO DE TESIS Y TRABAJOS DE GRADO

Barranquilla, Fecha

Marque con una X
Tesis ☐ **Trabajo de Grado** ☐

Yo VICTOR HUGO TOVAR POLO, identificado con C.C. No. 8.646.977., actuando en nombre propio y como autor de la tesis y/o trabajo de grado titulado DISEÑO DE UN AUTOMATA PROGRAMABLE PARA AMBIENTE ACADEMICO presentado y aprobado en el año 2013 como requisito para optar al título de INGENIERO ELECTRONICO;
hago entrega del ejemplar respectivo y de sus anexos de ser el caso, en formato digital o electrónico (DVD) y autorizo a la UNIVERSIDAD DE LA COSTA, CUC, para que en los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia, utilice y use en todas sus formas, los derechos patrimoniales de reproducción, comunicación pública, transformación y distribución (alquiler, préstamo público e importación) que me corresponden como creador de la obra objeto del presente documento.

Y autorizo a la Unidad de información, para que con fines académicos, muestre al mundo la producción intelectual de la Universidad de la Costa, CUC, a través de la visibilidad de su contenido de la siguiente manera:

Los usuarios puedan consultar el contenido de este trabajo de grado en la página Web de la Facultad, de la Unidad de información, en el repositorio institucional y en las redes de información del país y del exterior, con las cuales tenga convenio la institución y Permita la consulta, la reproducción, a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato DVD o digital desde Internet, Intranet, etc., y en general para cualquier formato conocido o por conocer.

El AUTOR - ESTUDIANTES, manifiesta que la obra objeto de la presente autorización es original y la realizó sin violar o usurpar derechos de autor de terceros, por lo tanto la obra es de su exclusiva autoría y detenta la titularidad ante la misma. PARÁGRAFO: En caso de presentarse cualquier reclamación o acción por parte de un tercero en cuanto a los derechos de autor sobre la obra en cuestión, EL ESTUDIANTE - AUTOR, asumirá toda la responsabilidad, y saldrá en defensa de los derechos aquí autorizados; para todos los efectos, la Universidad actúa como un tercero de buena fe.

Para constancia se firma el presente documento en dos (02) ejemplares del mismo valor y tenor, en Barranquilla D.E.I.P., a los 16 días del mes de MAYO de Dos Mil Trece 2013.

EL AUTOR - ESTUDIANTE.

FIRMA

ANEXO 2
FORMULARIO DE LA DESCRIPCIÓN DE LA TESIS O DEL TRABAJO DE GRADO

TÍTULO COMPLETO DE LA TESIS O TRABAJO DE GRADO:

DISEÑO DE UN AUTOMATA PROGRAMABLE PARA AMBIENTE ACADEMICO :

SUBTÍTULO, SI LO TIENE:

AUTOR AUTORES

Apellidos Completos	Nombres Completos
COSSIO GOZMAN	JUAN DAVID

DIRECTOR (ES)

Apellidos Completos	Nombres Completos
CALVO POLGAR	LUIS CARLOS

JURADO (S)

Apellidos Completos	Nombres Completos
TOVAR POLO	VICTOR HUGO

ASESOR (ES) O CODIRECTOR

Apellidos Completos	Nombres Completos
SANCHEZ DAMS	RUBEN

TRABAJO PARA OPTAR AL TÍTULO DE: INGENIERO ELECTRONICO

FACULTAD: INGENIERIA

PROGRAMA: Pregrado ____ Especialización ____

NOMBRE DEL PROGRAMA INGENIERIA ELECTRONICA.