UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERIA

TESIS

INTERFAZ ANALÓGICA Y DIGITAL PARA PC BASADA EN MICROCONTROLADORES DE LA FAMILIA HC(S)08

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO ELÉCTRICO ELECTRÓNICO

P R E S E N T A:
EDUARDO OSWALDO ESCORCIA CARRANZA

Director de Tesis: M.I. Antonio Salvá Calleja

Ciudad universitaria junio 2014

A hortensia mi madre, a Pablo mi padre y a mi hermano Aarón, por siempre creer en mí.

Eterna gratitud a mis profesores que supieron guiarme y enseñarme, de manera especial al Maestro Antonio Salvá Calleja, por su tiempo y paciencia.

ÍNDICE PÁGINA

INTRODUCCIÓN

El objetivo de este trabajo de tesis es diseñar y construir una interfaz analógica y digital cuya operación y control se realice desde una PC (*Personal Computer*[1]).

La razón de dotar a un PC con una Interfaz como lo es la IADPC (Interfaz Analógica y Digital para PC), es integrarla a sistemas de control y/o instrumentación, mediante la interacción con sensores y actuadores asociados a una determinada aplicación. Por otra parte se realiza la creación de los algoritmos de control empleando la alta capacidad de cómputo con que cuenta una PC. De esta forma se logra hacer de una PC un dispositivo de control apto para abordar procesos tan complejos como lo permitan la programación y capacidad de la PC.

Es importante considerar que PC es un dispositivo ampliamente versátil, con una gran capacidad para realizar algoritmos complejos, tanto de bajo como de alto nivel; además, cuenta con muchas otras prestaciones como conectividad y autonomía. Debido a las tendencias y necesidades del mundo actual, el desarrollo tecnológico en este campo no se detiene.

La mayoría de las aplicaciones basadas en computadoras son de tipo administrativo que no requieren interfaz con el mundo exterior. Dada la gran capacidad de cómputo de las computadoras PC eventualmente estas se emplean para desarrollar aplicaciones de control e instrumentación; sin embargo, para estas circunstancias se requerirá contar con hardware adicional para fines de interfaz con dispositivos relacionados con la aplicación, como pueden ser sensores y actuadores tanto digitales como analógicos. Para este fin en la industria existen dos tendencias para fines del conexionado del hardware de interfaz con la PC, estas son:

- Tarjeta con hardware de interfaz conectada a la computadora a través de un slot de expansión ISA (*Industrial Standard Architecture*[2]) o PCIA (*Periferial component interconnect Architecture*[3]).

- Módulo de hardware de interfaz ligado a la PC a través de un puerto USB de la misma.

A través de la IADPC se brinda al programador la capacidad de adaptar a la PC a las necesidades del sistema que se requiera automatizar o controlar. En función de la creatividad del programador de la PC se pueden aprovechar al máximo las prestaciones de la combinación PC - IADPC. La capacidad de la PC para abordar aplicaciones genéricas, habilitada con una interfaz de esta naturaleza, constituyen un instrumento de control muy versátil y capaz.

Durante el desarrollo de este trabajo se abordará el proceso de diseño y prueba de cada uno de los módulos de hardware y software que hacen posible la IADPC. A continuación se describe en forma breve la forma en que se pretende diseñar el dispositivo aquí descrito.

El primer capítulo es una descripción de la tarjeta de desarrollo MINICON_08A**,** la cual está basada en el microcontrolador MC68HC908GP32 de Freescale. Esta tarjeta se utiliza en como enlace entre la PC y el hardware analógico y digital de interfaz con que cuenta la IADPC.

[1] Computadora personal.
[2] Arquitectura Industrial Normalizada
[3] Arquitectura de interconexión de componentes periféricos

En el segundo capítulo se aborda el diseño del hardware de los puertos de entradas y salidas digitales. Para el diseño del sistema se determinó que las salidas digitales estarían asociadas a relevadores, pues son la opción adecuada al área de aplicación para la cual se desarrollo la interfaz. Las entradas digitales son aisladas por medio de opto acopladores, los cuales cumplen la función de proteger tanto a la IADPC como a los dispositivos conectados a los puertos digitales de ésta.

En el capítulo tres se describe el diseño de los módulos de entrada y salida analógicos con los que cuenta la IADPC. En éste capítulo también se describe el diseño del hardware de adecuación de entrada que requiere el convertidor analógico digital propio del microcontrolador empleado como núcleo de la IADPC

En el cuarto capítulo se describe el software de base ejecutable en el MCU de la IADPC, de modo que éste reciba comandos desde la PC para los siguientes fines:

- Colocar salidas binarias en los diversos puertos que para este fin tiene la IADPC.
- Colocar niveles de salida analógicos en los puertos que para este fin tiene la IADPC.
- Leer niveles binarios de entrada presentes en los puertos de entrada que este fin tiene la IADPC.
- Leer niveles analógicos de entrada presentes en las terminales que para este fin tiene la IADPC.

En el capítulo cinco se describe el software ejecutable en la PC que se utiliza para poder controlar a la interfaz, a modo de una plataforma de prueba. Desde este software es posible controlar todas las salidas y leer todas las entradas, ya sean analógicas o digitales. Se emplea el lenguaje de programación basado en objetos de Visual Basic por su capacidad de crear interfaces de usuario y aplicaciones.

En el capítulo seis es un ejemplo de aplicación validado con la IADPC, esta aplicación tiene como objetivo demostrar las capacidades de monitoreo y control de la interfaz.

CAPÍTULO 1

Tarjeta de desarrollo MINICON_08A

1.1 El microprocesador 68HC908GP32CP

Un microcontrolador se compone de tres bloques fundamentales: la unidad de procesamiento central, la memoria y el bloque de entradas y salidas. Estos bloques se interconectan entre si mediante líneas eléctricas llamadas buses, los buses que transportan información sobre las direcciones de memoria o de una entrada o salida se llaman buses de direcciones. Los buses que transportan datos o instrucciones se llaman buses de datos. Finalmente, los buses que transportan señales de control se llaman buses de control.

El CPU (*Central Processing Unit*[1]) es el cerebro de un microcontrolador y esta bajo el control del software programado en la memoria. Las tareas del CPU consisten en adquirir las instrucciones guardadas en la memoria, interpretarlas y luego ejecutarlas. El CPU también incluye la circuitería necesaria para realizar operaciones aritmericas y lógicas con datos binarios. Ésta circuitería especial se llama la unidad lógica y aritmética.

Un microcontrolador puede ser considerado una microcomputadora construida en una solo circuito integrado o bien, en un chip. Un importante parámetro para los microcontroladores es el tamaño de sus registros internos (8, 16, 32 o 64bits), pues estos determinan el número de bits que pueden ser procesados simultáneamente. El MCU (*Microcontroler Unit*[2]) 68HC908GP32CP, del cual se hablara más adelante, tiene registros de 8 bits.

Por otra parte, los microcontroladores son usados en una gran variedad de aplicaciones. Éstos pueden ser encontrados en la industria automovilística, sistemas de comunicación, instrumentación electrónica, equipo de hospitales, equipo y aplicaciones industriales, etc.

La tarjeta de desarrollo MINICON_08A tiene cómo núcleo funcional al MCU 68HC908GP32CP. Éste microcontrolador es fabricado por FREESCALE, todos los MCU, en su familia utilizan la unidad central de procesos CPU08. Dicha unidad de procesos, junto con el hardware estructural del MCU, son esenciales para soportar funciones como temporizadores, manejo de la memoria RAM y FLASH, además de operaciones lógicas y aritméticas.

[1] Unidad Central de Procesos
[2] Microcontrolador

A continuación, en la figura 1.1 se muestra el diagrama de bloques funcional del MCU.

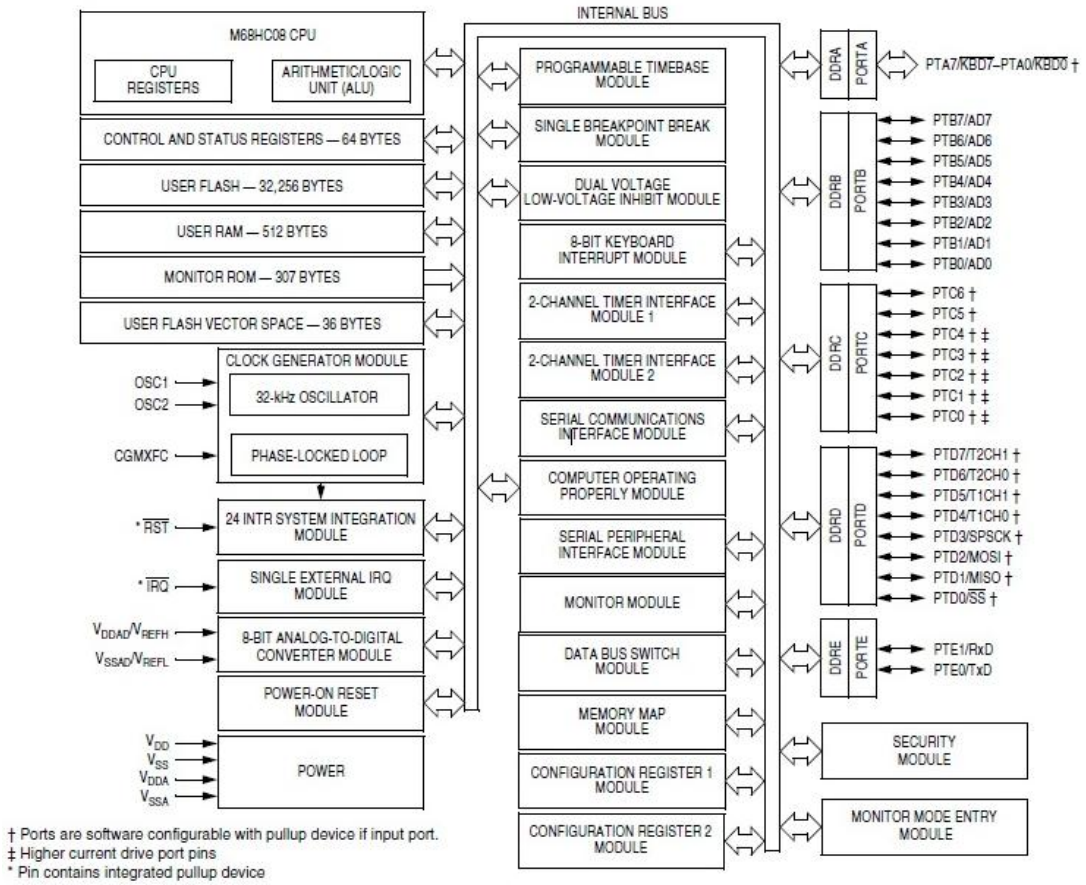


FIGURA 1.1: *Diagrama de bloques funcional del MCU 68HC908GP32CP, que describe la conexión entre los módulos del MCU y a sus puertos periféricos.*

1.1.1 Características del MCU 68HC908GP32CP

Las características del microprocesador se listan a continuación:

- Alto rendimiento de la arquitectura del M68HC08, optimizada para compiladores de C.
- 8 MHz de frecuencia en el bus interno.
- Sistema de protección que incluye:
  - Un restablecimiento opcional COP (*Computer Operating Properly*[1]).
  - Detección de bajo voltaje con un restablecimiento opcional y la capacidad de poder seleccionar la operación con 3 y 5 Volts.

[1] Computadora Operando Apropiadamente

- Detección de código de operación ilegal con restablecimiento.
- Detección de dirección ilegal con restablecimiento.
- Diseño para bajo consumo de energía eléctrica; con paro y modos de espera.
- Pin de restablecimiento maestro y restablecimiento al encendido.
- 32 Kbytes de memoria embebida en el chip.
- Módulo de interface con periféricos seriales.
- Módulo de comunicación serial.
- Dos módulos de interface con temporizadores de 16 bit y dos canales (TIM1 y TIM2) con entrada de captura seleccionable, comparación de salidas y capacidad para PWM (*Pulse Width Modulation*)[1] en cada salida.
- Convertidor analógico digital de ocho canales. Conversión por aproximaciones sucesivas de 8 bits.
- Módulo generador de reloj a bordo del chip con un cristal de 32 KHz, compatible con PLL (*Phase Locked Loop*[2]).
- Más de 33 pines entrada/salida de propósito general incluyendo 26 I/O pines con funciones compartidas.
- Pull-up[3] seleccionables en las entradas de los puertos A,C y D. Durante el modo de salida los Pull-up son desacoplados.
- Capacidad de alta corriente 10mA sink/10mA source en todos los pines de los puertos.
- Capacidad de corriente más alta 15mA sink/source en PTC0-PTC4.
- Módulo de tiempo base con circuito preescalador del reloj para ocho interrupciones periódicas de tiempo real, seleccionables por el usuario con la opción de mantener la fuente del reloj activa mientras se encuentra en modo de paro.
- El Puerto C es de 5 bits: PTC0-PTC4.
- El Puerto D e es de ocho bits.

1.1.2 Características del CPU08

- Modelo de programación HC05 mejorado.
- Amplia funcionalidad en el control de bucle.
- 16 modos de direccionamiento.
- 16 bits de index register[4] y stack pointer[5].
- Transferencias memoria a memoria.
- Rápida instrucción de multiplicación 8x8.
- Rápida instrucción de división 16/8.
- Instrucciones en BCD (*Binary Coded Decimal*[6]).
- Optimización para aplicaciones del controlador.
- Eficiente manejo del lenguaje C, Ensamblador y MiniBas.

[1] Modulación por ancho de pulso
[2] Lazo dependiente de la fase
[3] Entrada verificada en nivel alto de voltaje
[4] Registro de indexado
[5] Registro de pila
[6] Decimal Codificado en Binario

1.2 La tarjeta MINICON_08A

La tarjeta MINICON_08A es el bloque funcional de la IADPC que arbitra la funcionalidad de los elementos de entrada y salida con los que cuenta la interfaz; esto se hace mediante software de base ejecutable en el MCU 68HC908GP32 presente en la tarjeta. Este software es arbitrado desde la PC mediante un enlace serie entre la tarjeta y la PC. Para el desarrollo del software de base aquí mencionado se usó un sistema para desarrollo, diseñado en la Facultad de Ingeniería de la UNAM, denominado AIDA08 (Ambiente Integrado para Desarrollo y Aprendizaje con Microcontroladores de la Familia 68HC(9)08). La tarjeta MINICON_08A es parte de AIDA08.

Los componentes funcionales del sistema AIDA08 se muestran en la figura 1. A continuación se describe brevemente los componentes y funcionalidad del sistema AIDA08.

1.2.1 El sistema AIDA08

El sistema AIDA08 (figura 1.2) está integrado por dos componentes funcionales de hardware y software, los cuales son:

A) Tarjeta para desarrollo MINICON_08A enlazada mediante el puerto serie con una PC donde corre un software manejador.

B) Software manejador denominado PUMMA_08+, el cual cuenta con un ensamblador cruzado denominado ENS08 y un compilador cruzado de BASIC denominado MINIBAS8A.

FIGURA 1.2: *Esquema del sistema AIDA08. De izquierda a derecha se encuentran la PC el enlace serie y finalmente la tarjeta MINICON_08A.*

Las principales características de la tarjeta MINICON_08A son:

- Está basada en el MCU 68HC908GP32CP fabricado por FREESCALE.
- Contiene interfaz de seis hilos compatible con la tarjeta de programación IP_ASC_08B, lo cual permite probar aplicaciones partiendo de microcontroladores sin firmware en memoria FLASH, empleando para ello al manejador AMIGO_08.

- Capacidad para operar con microcontroladores de la familia HC908 que contengan el firmware NBCP8.
- Capacidad para operar con microcontroladores de tipo CHIPBAS8 GP32, que contienen el firmware NBCP8_BIBAS8 compatible con el compilador cruzado MINIBAS8A.
- Contiene postes (conocidos como *headers*) con acceso diversos puntos de interés, como pueden ser líneas de entrada y/o salidas de puertos y diversos puntos de prueba.
- Incluye auxiliares didácticos tales como LEDs[1] testigo para el puerto A.
- Contiene interfaz para desplegados alfanuméricos populares en la industria.
- Capacidad para ejecución autónoma de programas previamente cargados en memoria FLASH con facilidades propias del manejador PUMMA_08+.
  NOTA: NBCP8 son siglas que denotan lo siguiente: Núcleo Básico de Comunicaciones con PUMMA_08+. NBCP8_BIBAS8 denota firmware que contiene las bibliotecas de MINIBAS8A y al NBCP8.

Las características principales del software manejador PUMMA_08+ son:

- Ejecutable bajo WINDOWS (98/M/XP/V/7).
- Capacidad para cargar en la TD archivos S19, que pudieran contener datos o programas en lenguaje de máquina.
- Capacidad para ejecutar un programa previamente cargado en la TD.
- Capacidad para escribir datos a memoria o puerto en la TD.
- Capacidad para examinar memoria o puertos en la TD.
- Capacidad para programar la memoria FLASH contenida en el MCU presente en la TD.
- Contiene un editor básico para que el usuario pueda escribir y almacenar los programas asociados con una determinada aplicación.
- Contiene un ensamblador cruzado, denominado **ENS08**.
- Contiene un compilador cruzado de lenguaje **BASIC**, denominado **MINIBAS8**.
- Capacidad para obtener, a partir del código fuente en ensamblador, presente en la ventana del editor, el archivo S19 que contiene el código de máquina ejecutable en el MCU de la TD.
- Capacidad para obtener, a partir del código fuente en BASIC, presente en la ventana del editor, el archivo S19 que contiene el código de máquina ejecutable en el MCU de la TD.
- Capacidad para ejecutar de inmediato el código generado a partir de un determinado programa fuente, escrito en lenguaje ensamblador y presente en la ventana del editor.
- Capacidad para ejecutar de inmediato el código generado a partir de un determinado programa fuente escrito en lenguaje BASIC y presente en la ventana del editor.
- Contiene un emulador de terminal básico mediante el cual se realiza la consola de interfaz con el usuario para fines de la ejecución de programas en BASIC.

En la tabla 1.1 se muestra la disposición de los conectores presentes en la tarjeta MINICON_08A y su función.

[1] Diodo Emisor de Luz

TABLA 1.1: *Tabla que muestra la disposición de los conectores más representativos de la tarjeta MINICON_08A*

Conector	Funcionalidad	Ubicación
CON1	Alimentación desde eliminador de batería	II
CON2	Terminal DB9 hembra para cable RS232	SI
CON3	Interfaz para desplegado LCD (*liquid Cristal Display*[1])	SD
CON4	Fuente externa de 5 V	SI
CON5	Interfaz de teclado	II e ID
CON6PTA	Acceso a pines del puerto A	ID
CON6PTB	Acceso a pines del puerto B	II e ID
CON6PTC	Acceso a pines del puerto C	SD
CON6PTD	Acceso a pines del puerto D	SD
CON7	Interfaz para programar chips nuevos empleando la tarjeta de programación IP_ASC_08B	ID y SD

NOTA 1:

II denota: cuadrante inferior izquierdo.
ID denota: cuadrante inferior derecho.
SI denota: cuadrante superior izquierdo.
SD denota: cuadrante superior derecho.

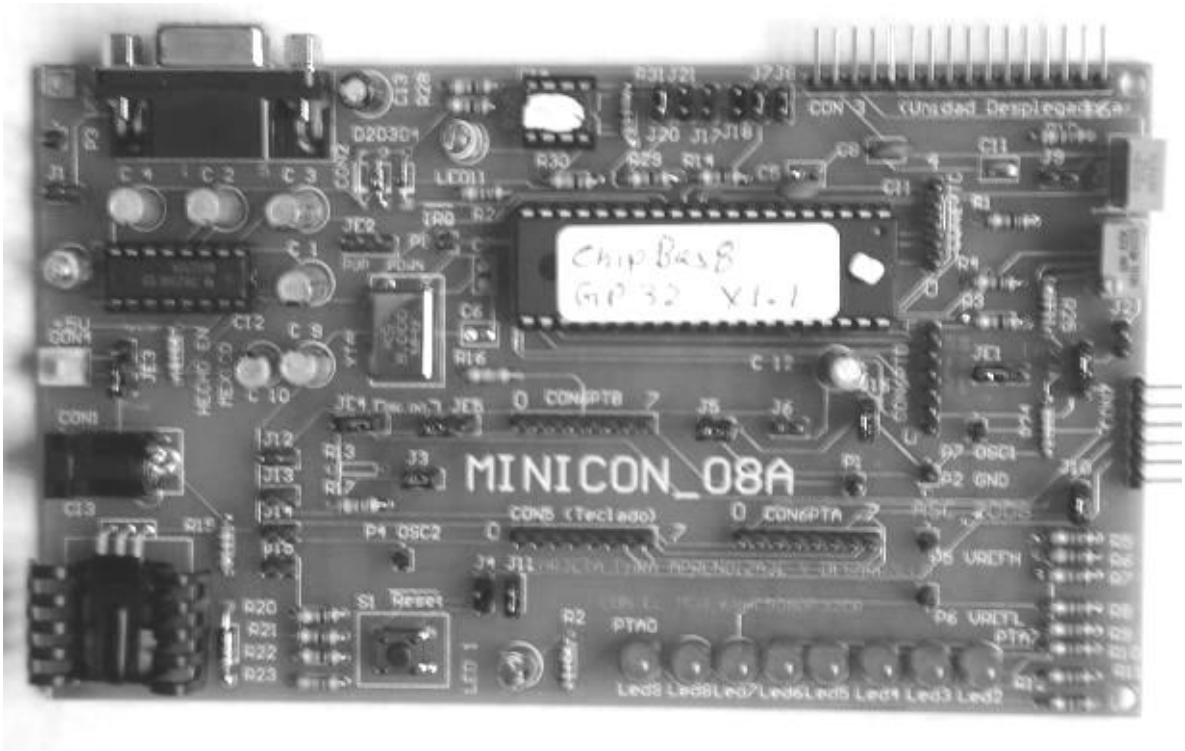


FIGURA 1.3: *Fotografía de la tarjeta MINICON_08A*

[1] Indicador de Cristal Líquido

TABLA 1.2: *Tabla que muestra la disposición de los postes presentes en la tarjeta MINICON_08A*

Poste	Funcionalidad	Ubicación
1	Testigo de pin PTB7/AN7	ID
2	Tierra	ID
3	+VCC (+5 V)	SI
4	Testigo de pin OSC2	II
5	Testigo de pin VREFH	ID
6	Testigo de pin VREFL	ID
7	Testigo de pin OSC1	ID
8	Testigo de pin IRQ	ID

NOTA 1:

II denota: cuadrante inferior izquierdo.
ID denota: cuadrante inferior derecho.
SI denota: cuadrante superior izquierdo.
SD denota: cuadrante superior derecho.

1.3 La Interfaz Analógica y Digital para PC (IADPC)

Cómo se menciona con anterioridad, el objetivo de este trabajo de tesis es el desarrollo de la IADPC, desde su diseño esquemático hasta la construcción de un prototipo funcional. La IADPC es un módulo de interfaz de aplicación general y externo a la computadora personal (por sus siglas en inglés PC) y ligado a esta última por medio del puerto de serie.

En la industria la función de una interfaz de aplicación general, es la de transmitir a la PC (o a un controlador) la información del proceso a través de sensores, o aplicarle a éste las variables de salida mediante los correspondientes actuadores. Tanto las señales eléctricas de entrada como las de salida pueden ser digitales o analógicas. A través de la IADPC es posible controlar y coordinar desde la PC, las entradas y salidas adecuadas al proceso. Para ello es necesario que la IADPC tenga entre sus características, la capacidad de manejar tanto señales analógicas como digitales. En la figura 1.4 se muestran los diferentes tipos de puertos que posee la IADPC, además se hace referencia a la tarjeta MINICON_08A como bloque funcional de la IADPC.

La IADPC consta de:

- Dos puertos de entrada digital de ocho bits cada uno.
- Dos puertos de salida digital ocho bits cada uno.
- Dos salidas analógicas de 0 V a 10 V.
- Cuatro entradas analógicas de rango seleccionable.
- Un puerto de comunicación serie.

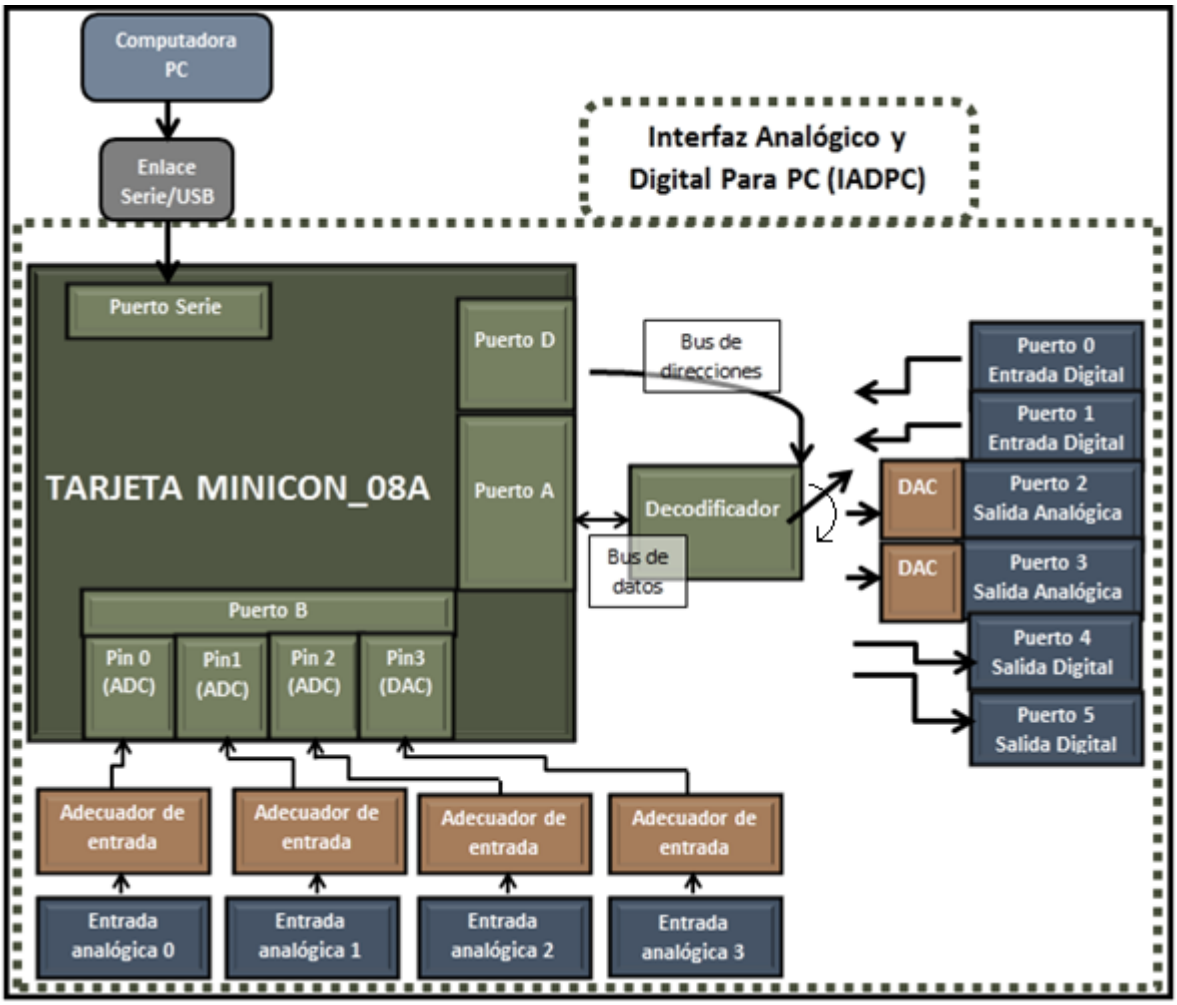


FIGURA 1.4*: Diagrama de bloques funcional de la IADPC, muestra la conexión entre los puertos de la interfaz y la tarjeta MINICON_08A*.

1.3.1 La tarjeta de desarrollo MINICON_08A como núcleo lógico de la IADPC

Para los fines funcionales de la IADPC, la tarjeta MINICON_08A tiene la función de enlazar la PC con los puertos analógicos y digitales de la IADPC. Una vez programada, la tarjeta es capaz de interpretar los comandos provenientes de la PC y en consecuencia, coordinar las acciones solicitadas, es decir, leer o modificar el estado de los puertos analógicos y digitales de la IADPC.
Debido a que la tendencia tecnológica en el desarrollo de las PC llevo a sustituir el uso de los puertos de comunicación serie por los USB; la mayoría de las computadoras de última generación ya no cuentan con puerto serie, en especial las PC denominadas Laptop. En caso de no tener disponible un puerto serie, es necesario utilizar un convertidor USB-Serie, para enlazar a IADPC desde el puerto USB de la computadora. Dicho convertidor es popular en la industria.

A continuación se explica la función que se asignó a módulos propios del MCU de la tarjeta MINICON_08A para fines de la IADPC.

a) **Puerto serie:** El puerto serie es utilizado para comunicarse con la PC. A través de este medio se envían a la IADPC los comandos de control asociados a la lectura y modificación del estado de los puertos. Por este mismo medio, se envían hacia la PC los resultados de las solicitudes de lectura de los puertos de la IADPC.

b) **Puerto A:** Es en éste donde los puertos de entrada digital de la IADPC depositan los niveles binarios resultado de su interacción con el mundo físico. Es también en este puerto donde la IADPC coloca los niveles binarios que corresponden a los puertos de salida digital. La función de terminal entrada o salida del puerto A se determina mediante software en la programación de la tarjeta MINICON_08A. Como se verá más adelante, las salidas analógicas son, en principio, salidas digitales de ocho bits transformadas a niveles analógicos por medio de un convertidor Digital-Analógico (CDA).

c) **Puerto B:** Las líneas del puerto B están habilitadas para ser utilizadas como entradas del convertidor Analógico-Digital propio del MCU 68HC908GP32CP instalado en la tarjeta de desarrollo MINICON_08A. Con el objetivo de mantener al rango de voltaje de entrada en niveles adecuados para el MCU (entre 0 V y 5 V) en los puertos del convertidor analógico digital de la tarjeta de desarrollo, se utiliza un adecuador de voltaje (descrito más adelante).

d) **Puerto D:** Este puerto se utiliza para direccionar el puerto digital (ya sea entrada o salida) que se desea usar. Para lograr este objetivo se enlaza el nivel binario del puerto D con un decodificador, el cual habilita solo un puerto de la IADPC para su lectura o escritura. De esta forma solo se puede leer o modificar un puerto a la vez.

Por ahora se han revisado las propiedades que definen a la tarjeta de desarrollo MINICON_08A, sus características del software y del hardware y al sistema AIDA08. La razón para elegir un sistema basado en un MCU, para el desarrollo de la IADPC, es que este dispositivo cuenta con la capacidad de manejar y administrar los datos necesarios para caracterizar un proceso. Por otra parte, cuenta con comunicación serie para enviar y recibir datos comandos desde la PC.

En la industria existe una gran cantidad de sensores y actuadores de distinto tipo destinados a la automatización de procesos, por esta razón la IADPC fue diseñada con una variedad de entradas y salidas tanto analógicas como digitales. En los capítulos consecuentes se dará una revisión del diseño de cada uno de los módulos de hardware, se mostrarán sus características, funciones y la descripción de los componentes principales, necesarios para realizar la construcción del hardware base de la IADPC.

CAPÍTULO 2

Diseño y prueba del hardware de interfaz digital requerido.

2.1 Señales digitales

Los sensores digitales generan señales eléctricas que sólo toman un número finito de niveles o estados entre un máximo un mínimo. Las más utilizadas son las binarias, que solo pueden tener dos niveles de voltaje, que se asignan a los números 0 y 1. Una variable binaria recibe el nombre de bit. El criterio de asignación de los estados es totalmente arbitrario. Si se asigna el valor 1 al valor más alto y 0 al valor más bajo, se dice que es una lógica positiva. Si por el contrario, se asigna el 1 al valor más bajo y el 0 al valor más alto se dice que es una lógica negativa. Para representar una información se necesita un número *n* de variables binarias cuyo valor depende de la precisión que se desee. Las *n* variables binarias se pueden representar de dos formas diferentes:

- Mediante señales binarias independientes.
- Mediante una secuencia de niveles cero y uno.

La IADPC es capaz ambos tipos de representaciones binarias, en función de los comandos recibidos por la PC.

Como se mencionó anteriormente la IADPC tiene dos puertos digitales de entrada y dos de salida. Adicionalmente, dos canales analógicos de salida y cuatro de entrada. En este capítulo se abordará el análisis de la circuitería digital. Se revisará la importancia que las variables digitales tienen en la industria, justificando el porque es necesario que la IADPC tenga la capacidad para manejarlas. Es necesario recordar que los canales de salida analógicos de la IADPC son, en principio, puertos de salida digital acoplados a un conversor digital-analógico.

El hardware digital de la IADPC esta compuesto por tres módulos que interactúan entres si para ejecutar los comandos provenientes de la tarjeta MINICON_08A. Estos módulos son:

- Módulo manejador de direcciones.
- Puertos de entradas digitales.
- Puertos de salidas digitales.

Los puertos de salidas digitales a su vez, se dividen en puertos de salida digital acoplados a relevadores, y en puertos de salida acoplados a conversores digital analógico.

Para poder controlar los múltiples puertos digitales, la PC envía un comando al MCU de la IADPC, quien se encarga de interpretarlo. Consecuente el MCU, por medio del módulo manejador de direcciones, selecciona el puerto digital a utilizar.

2.2 Módulo manejador de direcciones.

El módulo manejador de direcciones se diseñó para poder seleccionar el puerto digital a utilizar. Mediante el puerto D del MCU se ingresa el número binario que el puerto a utilizar tiene asignado, al módulo manejador de direcciones y en consecuencia éste habilita el puerto seleccionado.

Esta clase de comportamiento es característico de un decodificador.

El módulo manejador de direcciones se constituye de un decodificador de 3 a 8 popular en la industria (74HC138). Las tres líneas de entrada del decodificador se conectan con el puerto D del MCU, donde se ingresa el número asignado al puerto digital; las ocho salidas se conectan de la manera siguiente: cuatro líneas de salida habilitan puertos de entrada, mientras que otras dos líneas de salida del decodificador habilitan los puertos de salida.

Cuando un puerto de entrada o salida es habilitado, se permite su conexión con el bus de datos del sistema que en este caso es el puerto A del MCU de la tarjeta MINICON_08A. Como se mencionó anteriormente el MCU tiene la capacidad de configurar al puerto A como entrada o salida digital.

Dependiendo del comando recibido por la PC, el MCU habilita como entrada o salida al puerto A y al mismo tiempo selecciona el puerto correspondiente por medio del módulo manejador de direcciones. Por ejemplo, al seleccionar el puerto con la dirección 000, también tiene que configurar el puerto A como entrada digital (véase la tabla 2.3)

2.2.1 El decodificador 74HC138N

Como se puede apreciar en el diagrama de bloques de la figura 1.4 del capítulo 1, la IADPC tiene un módulo conocido cómo decodificador, cuya función es la de determinar que puerto digital (ya sea entrada o salida) se conecta al bus de datos del puerto A. El decodificador es un circuito integrado cuyo comportamiento depende de la combinación en sus terminales de entrada de la siguiente manera:

Para cada combinación de variables en la entrada (A, B y C), habrá exactamente una de las líneas de salida cuyo valor será uno. La figura 2.1 ilustra un decodificador de tres a ocho líneas.

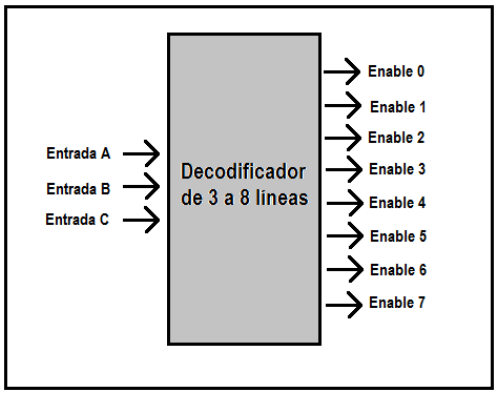


FIGURA 2.1: *Diagrama básico de un decodificador digital*

En general, un decodificador de *n* a 2^n líneas genera 2^n productos. Por ejemplo, para fines del desarrollo de la IADPC se utiliza un decodificador de 3 líneas de entrada a 8 líneas de salida. De esta manera el decodificador genera 2^3 salidas, de las cuales solo una tiene el valor 1 (ver tabla 2.1).

Específicamente la IADPC utiliza el decodificador 74HC138N el cual es un decodificador de 3 a 8 lineas. Este circuito integrado es una compuerta CMOS y sus pines son compatibles con los TTL Schottky de bajo poder.

TABLA 2.1: *Tabla de verdad de un decodificador de 3 a 8 líneas.*

Entradas			Salidas							
Entrada A	**Entrada B**	**Entrada C**	**Salida 0**	**Salida 1**	**Salida 2**	**Salida 3**	**Salida 4**	**Salida 5**	**Salida 6**	**Salida 7**
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

El 74HC138N se caracteriza por tener tres entradas de habilitación (enable), dos que se activan en nivel bajo ($\overline{E1}$ y $\overline{E2}$) y una que se activa en nivel alto (E3). Todas las salidas del decodificador estarán en un nivel alto amenos de que $\overline{E1}$ y $\overline{E2}$ esten en un nivel bajo y E3 en un nivel alto (ver tabla 2.2).

Esta función de multi-hablilitación podría permitir expansión en paralelo del 74HC138N a un decodificador de 5 a 32 líneas, con solo cuatro de estos circuitos integrados.

El diagrama lógico de este circuito integrado se muestra en la figura 2.2

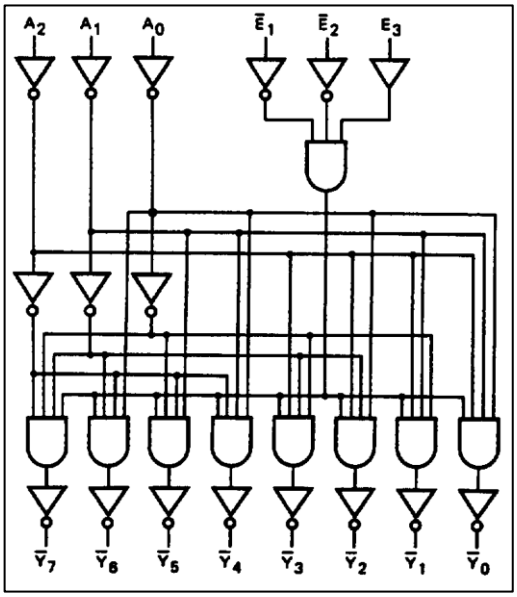


FIGURA 2.2: *Diagrama lógico del decodificador 784HC138N.*

2.2.2 Principales características del 74HC138N

- Capacidad de demultiplexor.
- Múltiples entradas de habilitación para una fácil expansión.
- Salidas invertidas y mutuamente exclusivas.
- Capacidad de salida: estándar.

Las terminales $\overline{E1}\ y\ \overline{E2}$ se mantienen en un nivel bajo permanentemente, mientras que E3 se mantiene en un nivel alto permanentemente; de esta manera se permite que el circuito funcione como un decodificador, la tabla 2.2 describe este comportamiento. La dirección del puerto que se desea activar es colocada en el puerto D de la tarjeta MINICON_08A; específicamente las entradas A0 A1 y A2 del 74HC138N se conectan las tres primeras líneas del puerto D PTD0, PTD1 y PTD2 (Véase la figura 2.2 y la figura 1.1.

TABLA 2.2: *Tabla de estados del decodificador 74HC138N*

Entradas						Salidas							
$\overline{E1}$	$\overline{E2}$	E3	A0	A1	A2	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$	$\overline{Y4}$	$\overline{Y5}$	$\overline{Y6}$	$\overline{Y7}$
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

Cada una de las salidas $\overline{Yn}$ se encuentra asociada a cada uno de los puertos digitales y dependiendo de su estado (alto o bajo) es posible habilitar un puerto digital, pero solo uno a la vez. Más adelante se explicara la interacción entre el puerto digital y el módulo manejador de direcciones (Véase la tabla 2.3).

TABLA 2.3: *Relación entre las salidas del decodificador y cada uno de los puertos digitales de la IADPC.*

Salidas del decodificador	Puerto digital asociado	Número binario asignado (dirección)
$\overline{Y0}$	Puerto 0 (entrada digital)	000
$\overline{Y1}$	Puerto 1 (entrada digital)	001
$\overline{Y2}$	Puerto 2 (salida digital)	010
$\overline{Y3}$	Puerto 3 (salida digital)	011
$\overline{Y4}$	Puerto 4 (salida digital)	100
$\overline{Y5}$	Puerto 5 (salida digital)	101
$\overline{Y6}$	N/A	110
$\overline{Y7}$	Puerto fantasma	111

La salida $\overline{Y7}$ no se encuentra conectada a ningún dispositivo, se le nombró puerto fantasma por esta razón. Cuando se le envía la dirección del puerto fantasma al decodificado, el bus de direcciones del puerto A queda desconectado de todos los latch. La desactivación del bus del puerto A, se justifica en la necesidad de no crear conflictos en los procesos de lectura o escritura (descritos más adelante).

La desactivación del puerto A, bien se pudo realizar al asociar una línea de uno de los puertos periféricos de la tarjeta MINICON_08A, con una de las terminales de habilitación del decodificador ($\overline{E1}$, $\overline{E2}$ o E3). Además de comprometer uno de los puertos de la tarjeta MINICON_08A, las rutinas de escritura o lectura se complicarían al tener que manejar más de un puerto, además de que el tiempo de respuesta de la IADPC se incrementaría. Debido a estos argumentos se determino que la mejor opción era la de utilizar un puerto fantasma.

2.3 Latch (memoria y controlador del puerto digital)

Cada uno de los puertos digitales tiene asignado un latch, que es el circuito integrado encargado de habilitar la conexión del puerto digital con el bus de datos del sistema (puerto A del MCU) y su funcionamiento depende directamente del módulo manejador de direcciones.

2.3.1 El Latch de entradas 74HC573

La función de éste circuito integrado es habilitar la conexión de un puerto digital de entrada con el bus de datos del sistema, es decir, permitir que el puerto A del MCU pueda adquirir el estado de cada uno de los bits del puerto de entrada habilitado.

A continuación se hace una descripción del circuito integrado 74HC573, más adelante se abordara la interacción de éste con el resto del sistema.

El 74HC573 es una compuerta CMOS de alta velocidad y sus pines son compatibles con los TTL Schottky de bajo poder. Este circuito integrado se puede definir como un encapsulado de ocho compuertas de bus tipo D, independientes una de otro y diseñadas específicamente para aplicaciones de manejo de bus.

Posee una entrada que habilita el Latch (*LE*) y una entrada ($\overline{OE}$) común para todos las compuertas del latch que habilita las salidas Q_n.

Características principales del 74HC573:

- Entradas y salidas en lados opuestos del encapsulado, permitiendo una fácil interface con microprocesadores
- Útil para puertos de entrada o salida de microprocesadores o microcontroladores
- Salidas de tres estados no invertidos para aplicaciones orientadas a buses
- Capacidad de salida: manejador de bus
- Habilitación común para las salidas de tres estados

La figura 2.3 ilustra el diagrama lógico del 74HC573.

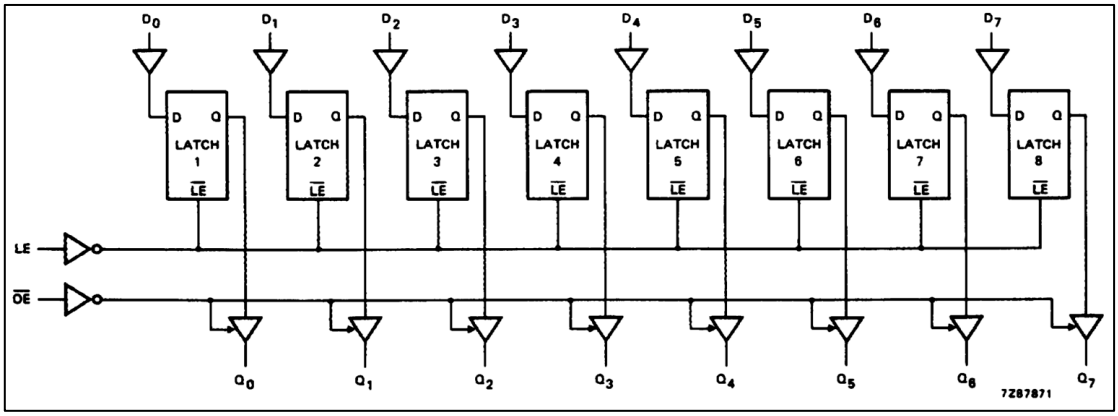


FIGURA 2.3: *Diagrama lógico del Latch de entradas 74HC573*

La combinación de estados de los pines de control $\overline{LE}$ y $\overline{OE}$ tiene como consecuencia distintos comportamientos (modos de operación) en el latch, la tabla 2.4 describe tales patrones.

TABLA 2.4: *Modos de operación del latch de entradas 74HC573*

MODOS DE OPERACIÓN	ENTRADAS			LATCH INTERNOS	SALIDAS DE Q0 A Q7
	$\overline{OE}$	$\overline{LE}$	D_n		
Habilitar y leer registro (modo transparente)	L	H	L	L	L
	L	H	H	H	H
Mantener y leer registro	L	L	l	L	L
	L	L	h	H	H
Mantener registro y deashabilitar salidas	H	L	l	L	Z
	H	L	h	H	Z

NOTAS:

H = Nivel de voltaje alto.
h = Nivel de voltaje alto un tiempo antes de la transición de alto a bajo de $\overline{LE}$.
L = Nivel de voltaje bajo.
l = Nivel de voltaje bajo un tiempo antes de la transición de bajo a alto de $\overline{LE}$.
Z = Alta impedancia, estado apagado.

2.3.1.1 Control del latch de un puerto de entrada digital

Funcionamiento del pin $\overline{LE}$.

1. Cuando $\overline{LE}$ esta en alto los datos en las entradas D_n entran al latch. En esta condición los ocho latch son transparentes y la salida cambiará cada vez que la entrada D_n cambie.

2. Cuando $\overline{LE}$ esta en bajo los latch mantienen la información que se había presentado en la entrada del latch D_n antes de la transición de alto a bajo de la entrada $\overline{LE}$.

Funcionamiento del pin $\overline{OE}$.

1. Cuando $\overline{OE}$ esta en bajo los contenidos de los ocho latch están disponibles en las salidas.
2. Cuando $\overline{OE}$ esta en alto las salidas hacen la transición a alta impedancia. La operación de $\overline{OE}$ no afecta en los estados de los latch.

Estos comportamientos se pueden apreciar en la tabla 2.4. Esta tabla corresponde a cada uno de los modos de operación del dispositivo.

Con el objetivo de controlar la conexión del puerto de entradas digitales con el bus de datos del sistema (puerto A) las salidas del decodificador del módulo manejador de direcciones asignadas a los puertos de entrada digital ($\overline{Y0}$ y $\overline{Y1}$) se conectan a la terminal $\overline{OE}$ del latch correspondiente a cada puerto de entrada digital.

Como la IADPC tiene dos puertos de entrada digital, se utilizaron dos 74HC573 para poder controlarlos. Para funcionar como puerto, cada 74HC573 tiene conectada del lado de las entradas D_n una línea del puerto de entrada digital y cada salida Q_n esta conectada a una línea del bus del sistema (PTA0 conecta con Q0 y así sucesivamente). La terminal $\overline{LE}$ se conecta permanentemente a un nivel alto, permitiendo que la lectura del puerto sólo dependa del estado de la terminal $\overline{OE}$.

2.3.2 El Latch de salidas 74HC574.

La función de éste circuito integrado es habilitar la conexión de un puerto digital de salida con el bus de datos del sistema, es decir, permitir que el puerto A del MCU pueda modificar el estado de cada uno de los bits del puerto de salida habilitado. Por otra parte, también cumple la función de memoria, es decir que mantiene el estado del puerto hasta que sea modificado nuevamente.

A continuación se hace una descripción del circuito integrado 74HC574, más adelante se abordara la interacción de éste con el resto del sistema.

El 74HC574 es un circuito integrado que encapsula ocho flip-flops, con salidas no invertidas y de tres estados (nivel alto, nivel bajo y alta impedancia), diseñadas para aplicaciones para manejo de bus.

Para habilitar la carga del latch tiene una entrada de reloj (CP). Para activar la carga de las variables digitales en la entrada del Latch, se debe aplicar un flanco de subida de voltaje en la terminal CP. La entrada($\overline{OE}$) es común para todos los flip-flops en el circuito integrado y su función es habilitar las salidas. Su diagrama lógico se muestra a continuación en la figura 2.4.

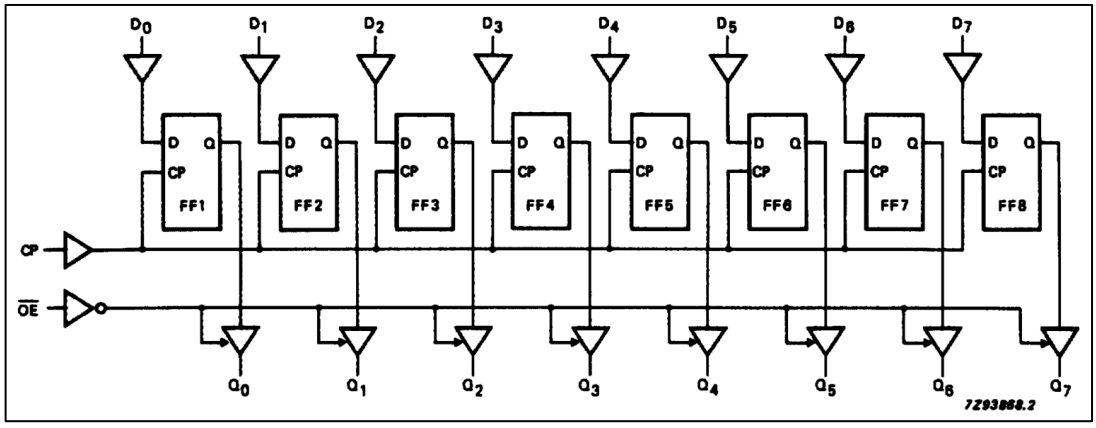


FIGURA 2.4: *Diagrama lógico del latch de salida 74HC574.*

Dependiendo del estado de las terminales de control el comportamiento del latch

La combinación de estados de los pines de control $\overline{OE}$ y CP tiene como consecuencia distintos comportamientos (modos de operación) en el latch, la tabla 2.5 describe tales patrones.

TABLA 2.5: *Modos de operación del latch de salidas 74HC574*

MODOS DE OPERACIÓN	ENTRADAS			FLIP-FLOPS INTERNOS	SALIDAS DE Q0 A Q7
	$\overline{OE}$	CP	Dn		
Cargar y leer registro	L	↑	l	L	L
	L	↑	h	H	H
Cargar registro y deshabilitar salidas	H	↑	l	L	Z
	H	↑	h	H	Z

NOTAS:
H= Nivel alto de voltaje
h=Nivel alto de voltaje un tiempo antes de la transición de nivel bajo a alto en CP
L= Nivel bajo de voltaje
l= Nivel bajo de voltaje un tiempo antes de la transición de nivel bajo a alto en CP
Z= Alta impedancia, OFF-state
↑=Transición de nivel bajo a alto en el reloj

Características del 74HC574

- Salidas no invertidas de tres estados para aplicaciones orientadas al manejo de bus.
- Registros de ocho bits disparados por flanco positivo.
- Entrada común para habilitar salidas.
- Registro y buffer de tres estados independiente.
- Capacidad de salida: manejo de bus.

2.3.2.1 Control del latch de los puertos de salida digital

Funcionamiento de la terminal CP

Los ocho Flip-flops mantienen el estado anterior de entradas D_n y cambian la salida (cargan el registro) hasta que se encuentran con un flanco de subida de voltaje en la terminal CP, el estado de las salidas no cambiara una vez que el voltaje se encuentra en un nivel alto.

Funcionamiento de la terminal $\overline{OE}$

Cuando $\overline{OE}$ esta en bajo, se habilita el contenido de los ocho flip-flops en las salidas. La operación de $\overline{OE}$ no afecta los estados de los flip-flops.

Estos comportamientos se pueden apreciar en la tabla 2.5, esta tabla corresponde a cada uno de los modos de operación del dispositivo.

Al tener cuatro puertos con salidas digitales (dos acoplados a relevadores y dos acoplados a convertidores CDA), la IADPC utiliza cuatro circuitos integrados 74HC574.

Las salidas del decodificador de $\overline{Y2}$ a $\overline{Y5}$ del módulo manejador de direcciones, se conectan con las terminales CP de cada uno de los latch respectivo con el objetivo de poder seleccionar el puerto cuyo estado se va a modificar.

Cuando el MCU modifica el estado de un puerto de salida seleccionado, lo hace mediante un flanco de subida en la terminal CP, mientras deposita en el bus de datos del sistema el valor binario que desea que el puerto despliegue, habiendo antes habilitado al puerto A como puerto de salida.

2.4 Integración del módulo manejador de direcciones con los latch asociados a los puertos digitales

En la tabla 2.6 se muestra la forma en que del bus del sistema se conecta directamente con las terminales de los latch (tanto los asociados con las entradas digitales como los asociados con salidas digitales).

TABLA 2.6: *Conexiones entre el bus de datos del sistema y los latch asociados a los puertos*

Bus de datos del sistema (puerto A)	Terminales del Latch *n* asociado con las salidas digitales	Terminales del Latch *n* asociado con las entradas digitales
PTA0 conecta con →	D0	Q0
PTA1 conecta con →	D1	Q1
PTA2 conecta con →	D2	Q2
PTA3 conecta con →	D3	Q3
PTA4 conecta con →	D4	Q4
PTA5 conecta con →	D5	Q5
PTA6 conecta con →	D6	Q6
PTA7 conecta con →	D7	Q7

Como se menciona anteriormente, es en el puerto D del MCU donde se coloca la dirección binaria asociada a cada puerto digital (véase tabla 2.3). Las terminales del puerto D se conectan con el módulo manejador de direcciones, específicamente a las terminales de entrada del decodificador, estas conexiones se muestran a detalle en la tabla 2.7.

TABLA 2.7: *Conexiones entre el puerto D y las terminales del decodificador del módulo manejador de direcciones*

Terminales del puerto D	Terminales del decodificador
PTD0 conecta con →	A0
PTD1 conecta con →	A1
PTD2 conecta con →	A2

A continuación en la figura 2.5 se muestra el diagrama del módulo manejador de direcciones integrado a todos los latch que se asocian con los puertos de entrada y salida, para propósitos del presente trabajo, a este hardware se le denomina MD-L.

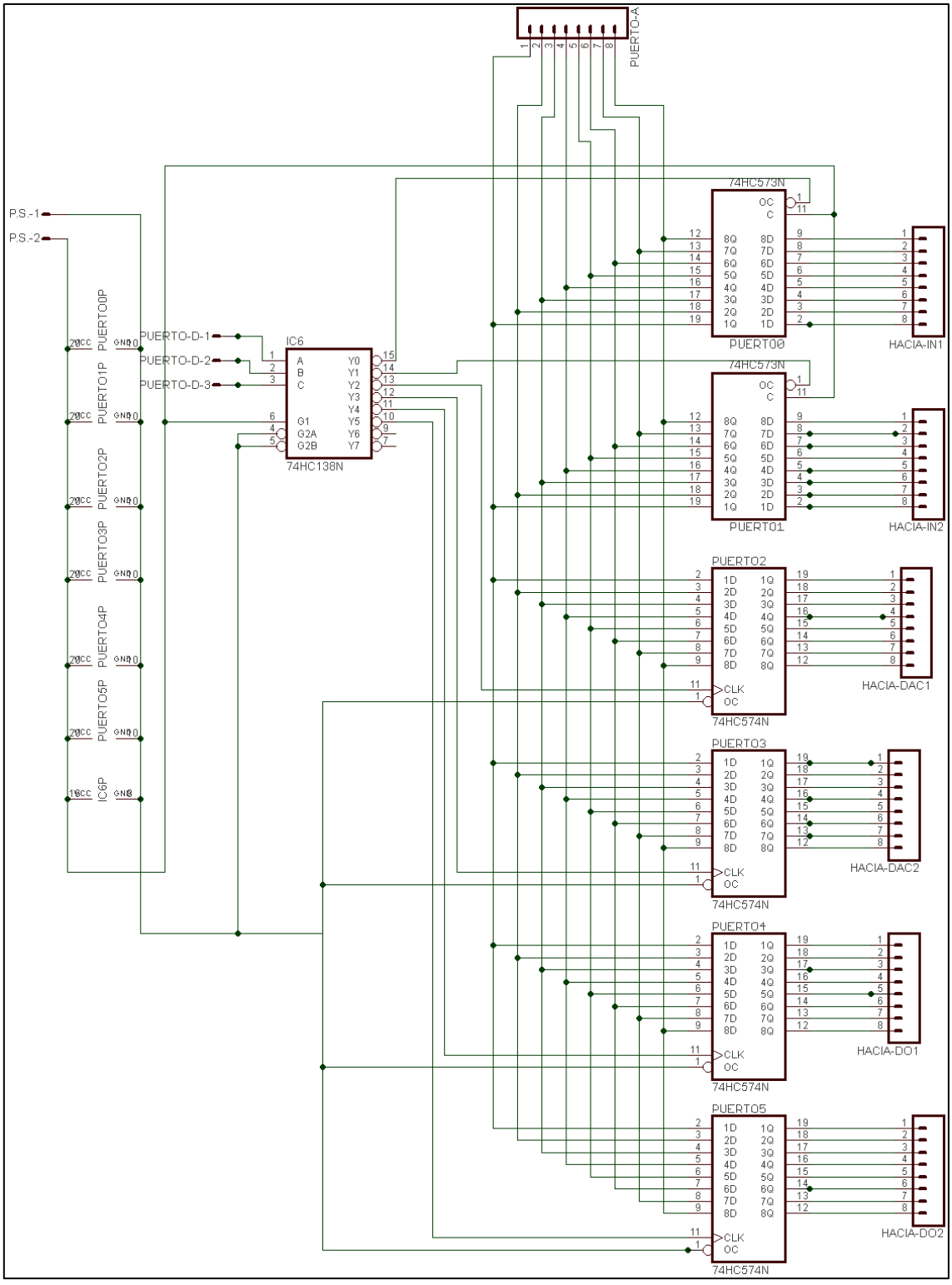


FIGURA 2.5: *Módulo manejador de direcciones integrado a los latch de entrada y salida (MD-L).*

2.5 Proceso de lectura y escritura de los puertos digitales

En este apartado se explicaran los procesos de lectura y escritura, basados en la interacción entre el decodificador y los latch de entrada y salida. A continuación se muestran los algoritmos propios de cada uno de los procesos antes mencionados.

2.5.1 Proceso de lectura de un puerto de entrada digital

Mediante el puerto D de la tarjeta MINICON_08A, se envía por un bus de direcciones de tres hilos un número binario a las terminales de entrada A0, A1 Y A2 del decodificador. El decodificador tiene sus salidas $\overline{Y0}$ y $\overline{Y1}$ conectadas a la terminal $\overline{OE}$, de cada uno de los latch de entrada 74HC573 (ver tabla 2.3). Los pasos para leer un puerto son los siguientes:

1) El puerto A esta habilitado como puerto de entrada todo el tiempo.
2) Enviar una dirección desde al puerto D al decodificador, por ejemplo la dirección “000”.
3) En este momento el puerto digital esta conectado al puerto A, ahora es posible leer y almacenar los datos del puerto digital.
4) Ahora se envía la dirección del puerto fantasma al decodificador y ningún latch queda conectado al bus del puerto A.

Para que la IADPC pueda cumplir con su objetivo de interfaz es necesario que los datos leídos en el puerto de entrada digital regresen a la PC. La dirección del puerto y la orden de leerlo vienen desde la PC, y la IADPC realiza el algoritmo expuesto en éste apartado para, finalmente, regresar a la PC la información contenida en el puerto digital.

2.5.2 Proceso de activación de un puerto de salida

Mediante el puerto D de la tarjeta MINICON_08A se envía por un bus de direcciones de tres hilos un número binario a las terminales de entrada A0, A1 Y A2 del decodificador. El decodificador tiene sus salidas $\overline{Y2}$, $\overline{Y3}$, $\overline{Y4}$ y $\overline{Y5}$ conectadas a las terminales CP de cada uno de los latch de salida (ver tabla 2.3). Los pasos para escribir en los puertos digitales son los siguientes:

1) El puerto A se habilita como salida.
2) Se escribe el dato a enviar en el bus del puerto A.
3) Se envía una dirección desde el puerto D al decodificador, por ejemplo la dirección “010” generando el flanco de subida, necesario para colocar en las salidas del latch los datos en el bus de datos del puerto A.
4) Se envía la dirección del puerto fantasma al decodificador y ningún puerto queda conectado al bus del puerto A.
5) El puerto A se habilita como entrada (vuelve a su estado inicial).

De la PC vienen la orden de escritura, el número del puerto y el dato binario que se desea escribir. Puede haber más de una modalidad de escritura, la IADPC esta capacitada para escribir un byte de 8 bits o modificar un solo bit de un puerto digital de salida determinado.

2.6 Puertos de entrada digital

El hardware de la IADPC cuenta con dos puertos de entrada digital de ocho bits, que se encargan de detectar las variables de tipo digital. En general, el diseño de un puerto digital de entrada, depende de tres conceptos:

- El tipo de variable, que puede ser de entrada o de salida.
- El tipo de tensión de alimentación utilizada, que puede ser de corriente continua (c.c.) o alterna (c.a).
- La forma de realizar el acoplamiento, que puede ser con aislamiento galvánico o directo (sin aislamiento galvánico).

Las variables digitales las generan los sensores todo-nada (como por ejemplo termostatos, presostatos, interruptores de caudal, finales de carrera, pulsadores, interruptores, etc). Dichos sensores cierran o abren un contacto libre de potencial o hacen que un transistor PNP o NPN, un trisistor o un triac se encuentre en estado de saturación o de corte. Por ello según el tipo de sensor y el tipo de alimentación que se utilice el circuito puede estar alimentado en corriente continua o corriente alterna.

Además, la conexión de un sensor a la interfaz se puede realizar de dos formas diferentes:

a) **Sin aislamiento galvánico**: Se dice que la conexión entre un sensor que proporciona una variable digital y el puerto se realiza sin aislamiento galvánico cuando ambos tienen al menos un punto unidos eléctricamente, es decir, están al mismo potencial eléctrico.

b) **Con aislamiento galvánico:** Se dice que el acoplamiento entre un sensor que proporciona una variable digital y el puerto se realiza con aislamiento galvánico cuando el valor de la tensión a la salida influye en la tensión de la terminal de entrada de la interfaz sin que exista una conexión eléctrica entre ambos. Se logra así que, una sobretensión o una sobrecorriente en el circuito de entrada no afecte a la interfaz.

La IADPC esta diseñada para utilizarse en aplicaciones de instrumentación, adquisición de datos y procesos de control y automatización. Por esta razón se optó por el aislamiento galvánico mediante optoacopladores. Un optoacoplador es un dispositivo de emisión y recepción que funciona como un interruptor excitado mediante la luz emitida por un LED que satura un componente optoelectrónico normalmente en forma de fototransistor o fototriac.

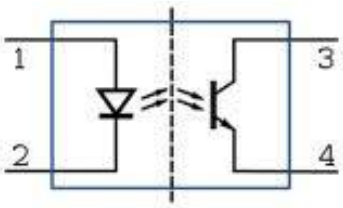
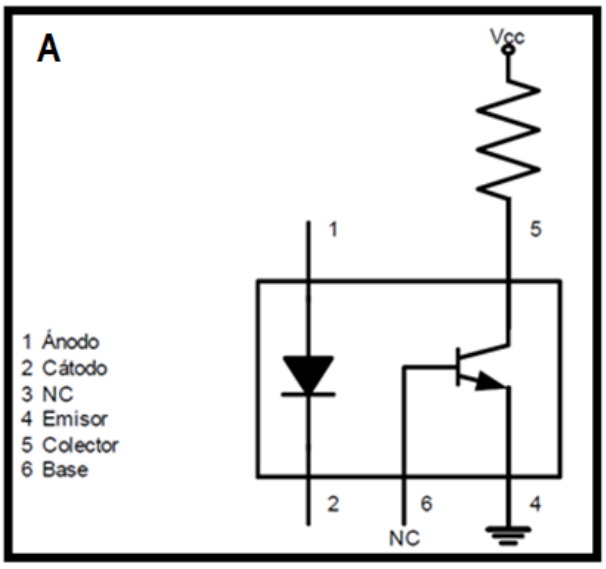


Figura 2.6: *Esta imagen muestra un LED activando un fototransistor con el objetivo de transmitir la variable digital mientras ambos lados se encuentran aislados.*

2.7 Diseño del optoacoplamiento

El circuito de optoacoplamiento se diseño para proteger a cada una de las terminales de los puertos de entrada digital. Los puertos de entrada digital de la IADPC aceptan como entrada 5 volts positivos de corriente continua, en cada una de sus terminales.

Como se puede apreciar en la figura 2.7B, el circuito tiene a la entrada un diodo externo al optoacoplador. La razón de poner el diodo es evitar dañar al dispositivo si se comete un error al conectar al sensor digital, es decir, si se invierte la polaridad aceptada por el puerto digital.



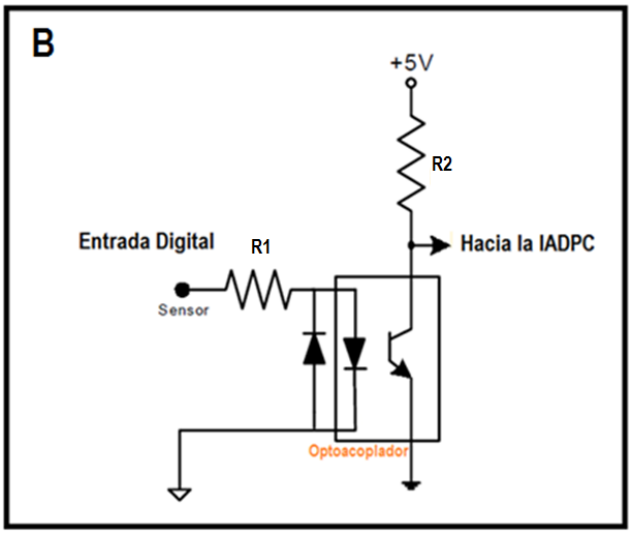


FIGURA 2.7: En A se muestra el esquema y terminales *del optoacoplador 4N26, mientras que en B se muestra el esquema de conexiones del optoacoplamiento para cada uno de los bits de los puertos de entrada digital de la IADPC*

Debido a la configuración de sus conexiones, el optoacoplador invierte la lógica de la entrada digital, es decir, que si a la entrada del puerto se tiene un nivel alto de voltaje, la IADPC detectará un nivel bajo.

Cuando el dispositivo se conecta un nivel bajo o no se suministra voltaje en la entrada del optoacoplador, es decir en la resistencia conectada al ánodo, las terminales 5 y 4 no tienen conductividad (hay alta impedancia entre ellas), por tanto el voltaje leído por la IADPC es el mismo que tiene a la entrada de R2, 5 V.

Cuando se suministra un voltaje positivo a la entrada del optoacoplador las terminales 5 y 4 tienen conductividad, de tal forma que ahora la terminal que va a la IADPC se encuentra conectada a un nivel bajo, es decir 0 volts. Ver figura 2.7.

Específicamente la IADPC utiliza en sus puertos de entrada digital al optoacoplador 4N26 (ver figura 2.8) cuyas características son:

- Prueba de aislamiento con 5300 VRMS.
- Interfaz con familias lógicas comunes.
- Capacitancia de conexión entrada-salida menor a 0.5 pF.
- Encapsulado industrial estándar, de seis terminales.

El 4N26 es un fototransistor de acoplamiento y se constituye de un LED y un fototransistor NPN de silicón. Para fines de la IADPC, el lado del puerto se conecta al LED y del lado del fototransistor se conecta el latch de entrada correspondiente.

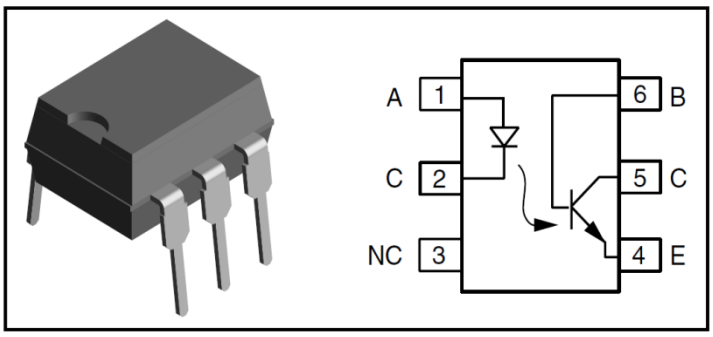


FIGURA 2.8: *Optoacoplador 4N26*

Como se mencionó anteriormente hay un circuito de optoacoplamiento para cada bit de cada puerto de entrada digital, de tal manera que se tiene el diagrama de la figura 2.9 que es el optoacoplamiento para cada uno de los puertos de entrada digital de la IADPC.

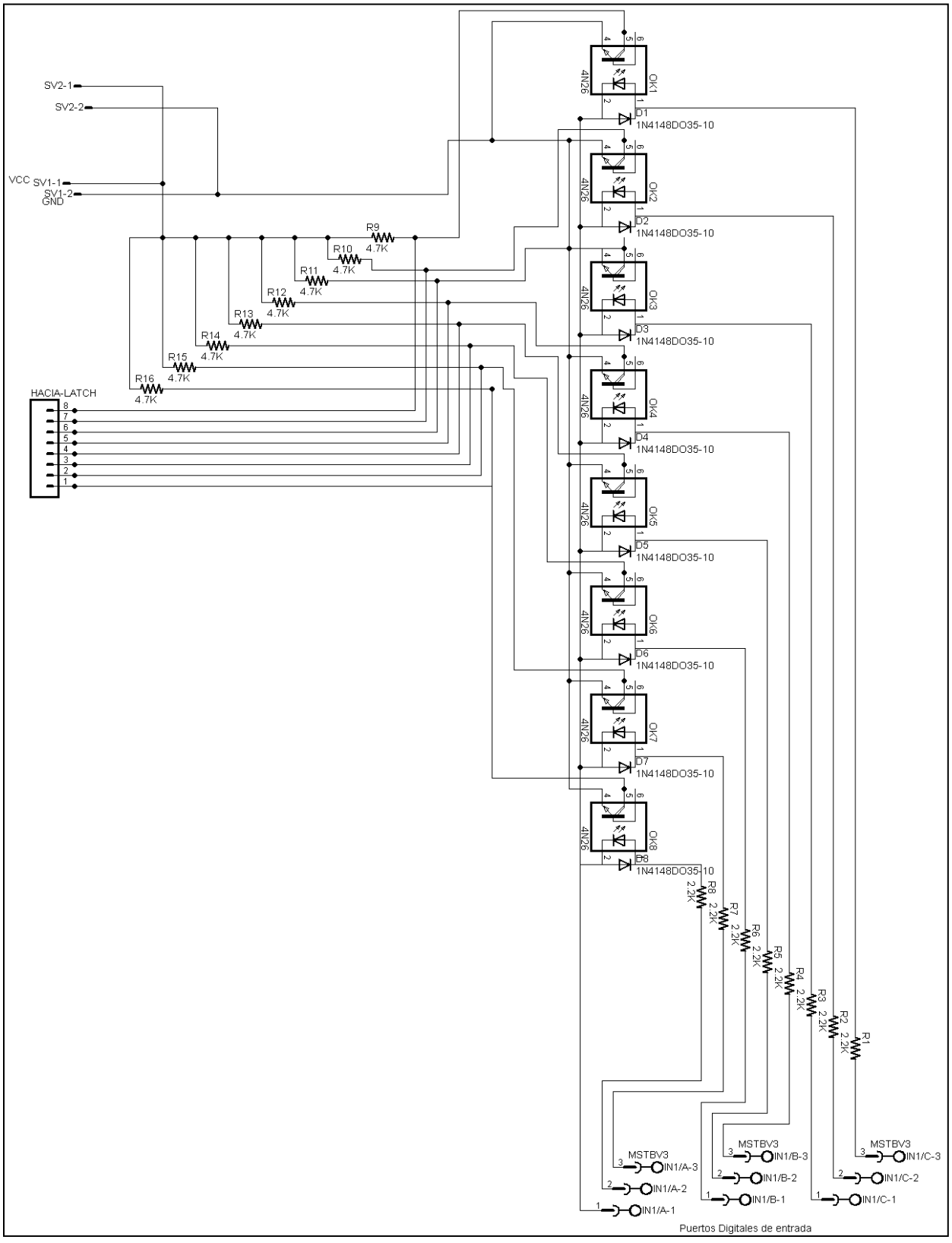


FIGURA 2.9: *Diagrama de conexiones del optoacoplamiento para cada uno de los puertos de entrada digital de la IADPC*

NOTAS:
SV1 = VCC (5 V).
SV2 = GND.
LATCH = Conexión con el latch correspondiente a este puerto.
OK*n* = Optoacopladores.
MSTBV3 = Conectores del puerto de entrada digital, aquí se conectan los sensores digitales.

2.8 Puertos de salida Digital acoplados a relevador

Las interfaces de salida digital, en general, están implementadas con aislamiento galvánico para evitar que las sobrecorrientes que se produzcan en los circuitos de salida puedan afectar a la interfaz. Según el tipo de dispositivo de conmutación utilizado, la interfaz de salida puede ser de tipo relé, de tipo transistor (PNP o NPN) de tipo triac o tiristor. Si la alimentación es alterna la salida tiene que ser de tipo relé, triac o tiristor. En cualquier caso el objetivo de un circuito de salida es hacer que el dispositivo de potencia se active o desactive de acuerdo con las acciones que ejecuta el programa de control de la PC.

Se determinó que para los fines con que la IADPC fue diseñada, la mejor opción son los relevadores, debido a su versatilidad y durabilidad. Las aplicaciones para este tipo de salidas solo están limitadas por los requerimientos de consumo de la carga y la frecuencia de conmutación que se necesite (un relevador no puede conmutar en altas frecuencias). La IADPC cuenta con dos puertos de salidas digitales de ocho bits, acoplados a relevadores.

Con este tipo de interfaz se pueden utilizar contactos normalmente abiertos “NA” y normalmente cerrados “NC”. Al ser salidas libres de potencial, permiten que las cargas se puedan alimentar en continua o en alterna y que, además, cada salida pueda activar actuadores con distinto tipo de alimentación (alterna o continua) y con valores de tensión diferentes.

Para poder proteger a la carga y a la IADPC de una posible sobrecorriente en las terminales de un puerto digital acoplado con relevadores se incluye en el diseño un fusible, el cual es fácil de cambiar y de bajo costo.

Para cada bit de cada puerto de salida digital de la IADPC se utilizó un relevador como el de la figura 2.11. Debido a que los latch fueron diseñados específicamente para manejo de bus digital, los niveles de corriente que son capaces de entregar, no son adecuados para poder activar un relevador, por lo que es necesario operar las salidas de relevadores mediante un driver (amplificador de corriente).

Este driver tiene la función de buffer invertido, es decir que lo que hace es devolver el valor negado del nivel lógico que tiene a la entrada. Físicamente provee la capacidad de manejar altas corrientes para poder suministrar la corriente de control al relevador. Ver figura 2.10.

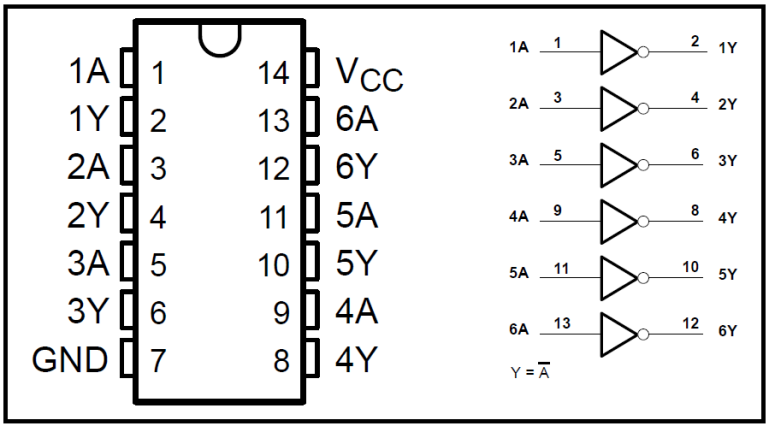


FIGURA 2.10: *Diagrama lógico del Buffer SN7416*

La IADPC utiliza el driver SN7416 cuyas características son:

- Convertir los niveles de voltaje TTL en niveles MOS.
- Gran capacidad de corriente de drenaje (máximo 40 mA).
- Manejador de colector abierto para lámparas indicadoras y relevadores.
- Entradas totalmente compatibles con la mayoría de circuitos TTL.

Este circuito TTL de seis buffers inversores se caracteriza por sus salidas de alto voltaje y colector abierto, lo que lo hace útil en aplicación de interfaz con circuitos de alto nivel de voltaje (como los MOS) o de manejar cargas de alto consumo de corriente (como lámparas y relevadores). Tiene capacidad para entregar hasta 40 mA de corriente.

Finalmente cada salida del buffer (1Y, 2Y,… y 6Y) se conecta a la bobina del relevador como lo indica la figura 2.11. Para proteger al relevador se coloca un diodo conectado con la polaridad invertida entre la alimentación del relevador y la salida Yn.

Para proteger al relevador mismo de una sobre corriente proveniente de la carga, es necesario colocar un fusible en el común da cada uno de los relés, esta conexión se realiza internamente y el usuario final de la IADPC solo tiene que preocuparse por conectar correctamente la carga al relevador y que el consumo de corriente de la carga sea menor a la capacidad del fusible.

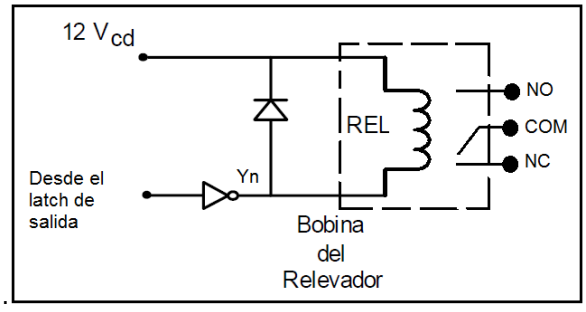


FIGURA 2.11: *Conexión entre la salida del buffer "Yn" y un relevador.*

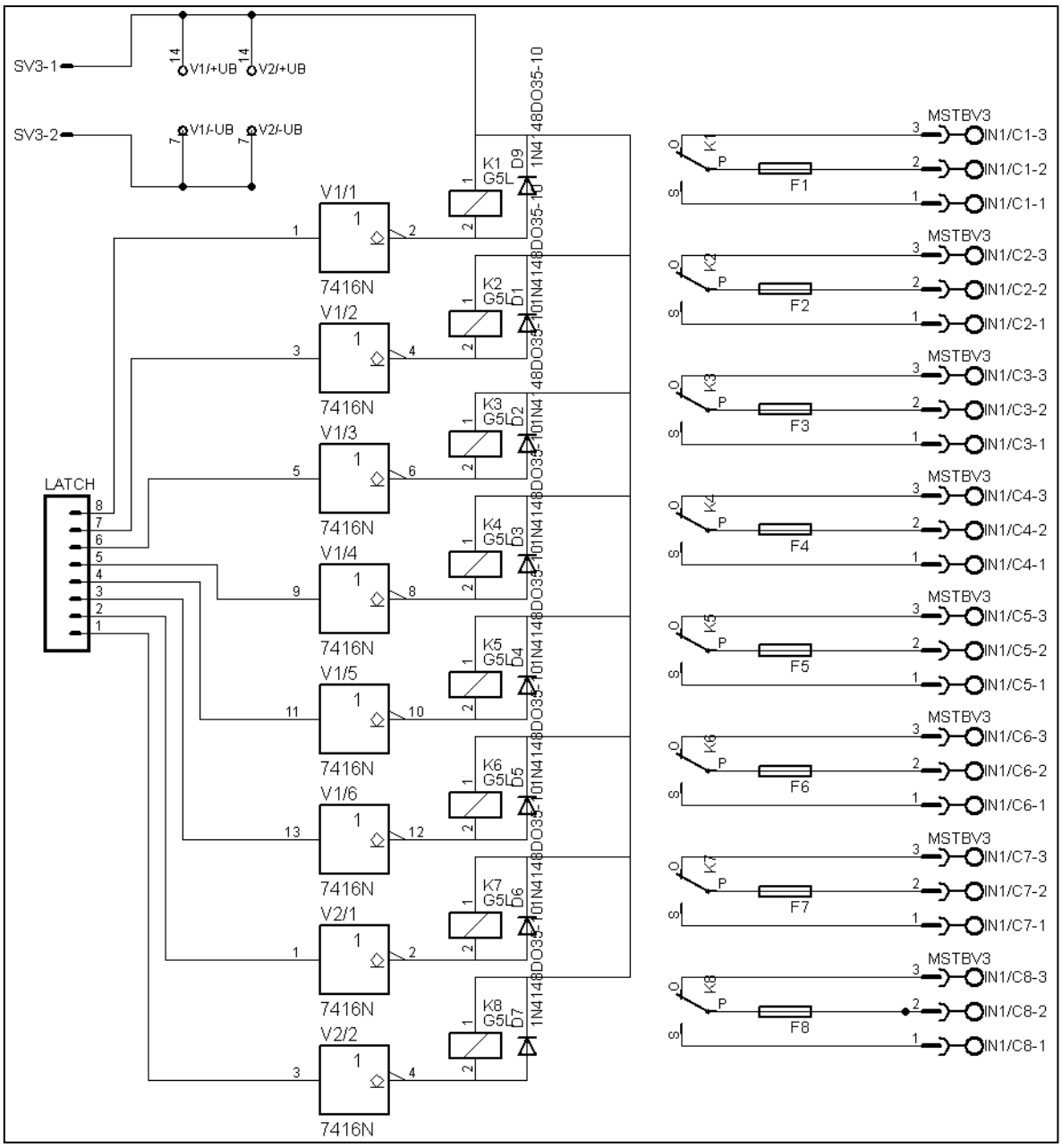


Figura 2.13:*Diagrama de conexiones para cada uno de los puertos de salida digital de la IADPC*

NOTAS:

SV3-1 = VCC (5 V).
SV3-2 = GND.
LATCH = Conexión con el latch de salida correspondiente a este puerto.
K*n* =Relevadores.
MSTBV3 = Conectores del puerto de salida digital, aquí se conectan los actuadores digitales.

2.9 Puertos de salida Digital empleados por los Convertidores Digital – Analógico (CDA) que contiene la IADPC

Los CDA de la IADPC requieren, como entrada, un número binario de ocho bits para transformarlo en un nivel analógico. Tal byte es colocado por la IADPC en sendos puertos de salida digitales que ésta contiene para conectarse con la entrada del CDA. El proceso de escritura a cada puerto es similar al empleado para una salida digital, la diferencia es que ésta va acoplada a un CDA. La circuitería del convertidor digital-analógico se abordará más adelante.

2.10 Pruebas a las que se sometieron los circuitos digitales de la IADPC

Las pruebas se realizaron para observar los resultados de la interacción del software de control y el hardware digital. Gracias a las pruebas de funcionamiento se pudieron identificar posibles oportunidades de mejora y optimización.

2.10.1 Armado del hardware

Se realizó el armado de la circuitería, siguiendo los diagramas antes mostrados (ver figuras 2.5, 2.9 y 2.13). Se conectaron en las salidas digitales LEDs testigos y DIP switch en las entradas digitales, con el único fin de realizar pruebas de funcionamiento con el MD-L. De esta manera se puede prescindir de los relevadores y los optoacopladores durante las pruebas del módulo. Una vez armados se revisan nuevamente los diagramas de conexiones y se procede a alimentar la circuitería.

Para motivos de prueba, cada circuito integrado obtuvo la alimentación de los postes P2 (GND) y P3 (VDD) de la tarjeta MINICON_08A (ver tabla 1.2). Más adelante, en el prototipo final, se utilizará otra fuente de alimentación independiente de la tarjeta, para no comprometer a ésta última en caso de una falla eléctrica.

Una vez alimentados los circuitos integrados, se procede a observar como se comportan durante un corto periodo de tiempo. El sobrecalentamiento de uno de los circuitos integrados, es señal de que existe un error en su conexión o que existe una falla en la fabricación de éste.

Al no encontrar posibles errores y observar un comportamiento normal de la circuitería, se procede a conectar el MD-L con la tarjeta MINICON_08A, por medio de buses. Es necesario que tanto la circuitería como la tarjeta estén desenergizadas durante la conexión. La forma de conectar al módulo con la tarjeta se muestra en las tablas 2.6 y 2.7.

Una vez conectadas, se pueden energizar la tarjeta y el módulo manejador de direcciones. Para realizar las pruebas requeridas es necesario programar rutinas que asemejen el funcionamiento básico de la IADPC. Como se explicó en el capítulo anterior, el desarrollo de este software se realiza mediante el sistema AIDA08.

2.10.2 Programas de prueba

Los programas de prueba se diseñaron para observar el comportamiento resultado de la interacción del software de la IADPC con el hardware digital. Se desarrollaron programas en lenguaje ensamblador, cuya labor era la de testificar la existencia de comunicación entre módulos y el correcto funcionamiento de éstos. Algunas de estas pruebas sirvieron como precedente para el diseño del software de control final.

Se recomienda apoyarse del Datasheet del MCU para la mejor interpretación de los segmentos de código que se presenta a continuación.

2.10.2.1 Programa de prueba de rutinas de entrada y salida digital

El siguiente programa se desarrollo para fines de validación de las rutinas de escritura y lectura digital, denominadas "escport" y "leeport", respectivamente. Éste programa se diseño para ejecución en la memoria RAM del MCU de la tarjeta MINICON_08A, utilizando lenguaje ensamblador. Su objetivo es leer el dato del puerto 01 (Latch asociado con el puerto de entradas digitales con la dirección 001) y devolverlo en el puerto 04 (Latch de salida con la dirección 04). De esta forma comprobamos que la interacción entre el MCU y el MD-L se lleve de manera adecuada.

Constantes usadas:

```
pta equ $00 ;          Dirección del puerto A
ddra equ $04 ;         Registro de dirección del puerto A
ptd equ $03 ;          Dirección del puerto D
ddrd equ $07 ;         Registro de dirección del puerto D
```

Variables usadas:

```
numport equ $0a ;      Dirección que almacena el numero de puerto a usar
datsal  equ $a1 ;      Dirección que  almacena el dato de salida
datent  equ $a2 ;      Dirección que almacena el dato de entrada
numfant equ $07 ;      Dirección del puerto fantasma
```

```
        org $0100 ;          Programando en RAM

        mov #$07,ddrd ;      El puerto D es salida en bits 0,1 y 2
        mov #numfant,ptd ;   Selecciona puerto fantasma
```

Ciclo principal

```
ciclo:  mov #$01,numport ;   Selecciona el puerto 1
        bsr leeport  ;       Salto a rutina para leer
        mov datent,datsal ;  Dato de salida ← Dato de entrada
        mov #$04,numport ;   Selecciona el puerto 4
        bsr escport  ;       Salta a rutina de escritura
        bra ciclo  ;         Salta siempre a "ciclo"
```

Subrutina “escport” Escribe un byte dato en un puerto de salida de la IADPC. Antes de invocar se deben cumplir las condiciones siguientes:

- (datsal) ←btye a escribir
- (numport) ←Número de puerto digital de salida en la IADPC

Al retornar ya se habrá escrito el dato en el puerto implicado.

```
escport:  mov #$ff,ddra ;        El Puerto A es declarado como salida
          mov datsal,pta ;       Escribe al bus de datos (Líneas del puerto A)
          mov numport,ptd ;      Selecciona puerto de salida (Puerto D←numport)
          nop
          mov #numfant,ptd ;     Se genera flanco de subida en línea de selección del puerto y se
queda apuntando al puerto fantasma.

          mov #$00,ddra ;        El puerto A regresa a su estado de entrada
          rts
```

Subrutina “leeport”: Lee un puerto de entrada de la IADPC, antes de invocar:

- (numport)←número de puertodigital de entrada a leer
- Al retornar: (datent) ←byte leído

```
leeport:  mov numport,ptd ;      Selecciona el puerto digital de entrada a leer
          mov pta,datent  ;      Lee el dato en el puerto A
          mov #numfant,ptd ;     Regresa al puerto fantasma
          rts
```

2.10.2.2 Programa de prueba de las rutinas básicas de comunicación serie entre la tarjeta MINICON_08A y la PC

El presente programa se desarrollo para comprobar el desempeño de la comunicación del puerto serie. Para fines de prueba, el presente programa se ejecuta en la memoria RAM del MCU. A las rutinas de envío y recepción de datos de les denominó respectivamente TXPS y RXPS. Para fines de la prueba, al recibir un número proveniente de la PC, éste se coloca en el puerto A del MCU, después se incrementa y se transmite así a la PC. Para testificar la funcionalidad de este experimento, se usó el emulador de terminal presente en el software manejador PUMMA_08+. El programa en cuestión es el siguiente:

```
scc2 equ $14
scc3 equ $15
scs1 equ $16
scs2 equ $17
scdr equ $18
scbr equ $19
```

Las siguientes direcciones corresponden a la dirección del puerto A y su registro de dirección:

```
pta equ $00
ddra equ $04
```

Esta es la inicialización del Puerto Serie:

```
        bset 6,scc1 ;       ensci<--1, habilita sci
        mov #$0c,scc2 ;     TE<--1,RE<--1,habilita TX y RX
        mov #$30,scbr ;     Baudaje: bps=9600 @ fc=8.000 MHz
```

Rutina Principal

```
        org $0100  ;        Programando en RAM
        mov #$ff,ddra ;     Habilita Puerto A como salida

otro:   bsr rxsci ;         Alto a rutina de recepción de dato
        sta pta ;           Salto a rutina de recepción de dato
        inca ;              a=a+1
        bsr txsci ;         Salto a rutina de transmición de dato
        bra otro ;          Regresa siempre a "otro"
```

Subrutina "txsci" para la transmisión de datos

```
txsci:    brclr 6,scs1,txsci ;   Espera a que este listo el puerto para la transmición
          sta scdr  ;            Envía el dato del registro  a por el puerto serie
          rts ;                  Regresa a la rutina principal
```

Subrutina "rxsci" para la recepción de datos

```
rxsci:    brclr 5,scs1,rxsci  ;  Espera a que la recepción termine
          lda scdr ;             Aloja el dato recibido en el registro a
          rts ;                  Regresa a la rutina principal
```

Hasta este punto, el hardware digital se comportó de manera esperada, los puertos devuelven valores congruentes y la comunicación con la PC se hace correctamente. El software definitivo, desarrollado para el control de los puertos digitales se analizará en el capítulo 4, mientras que el software para el intercambio de datos residente en la PC se encuentra en el capítulo 5.

El diseño del hardware digital representa el primer paso para lograr un sistema capaz de controlar un proceso industrial. Existen algunos sistemas que únicamente emplean variables digitales para fines de su control; sin embargo, en muchos casos será necesario el uso de variables analógicas. En el siguiente capítulo se aborda el diseño del hardware analógico, que complementa la funcionalidad de la IADPC.

CAPÍTULO 3

Diseño y prueba del hardware analógico requerido.

3.1 Las señales analógicas

Como se menciono anteriormente, la IADPC cuenta con cuatro entradas analógicas de rango seleccionable y dos puertos de salida analógica. en general las señales analógicas también pueden ser unipolares y bipolares. Las unipolares solo pueden ser positivas o negativas con respecto a un terminal de referencia, mientras que las bipolares pueden ser tanto positivas como negativas. En la industria se prefiere el uso de voltajes positivos debido a su compatibilidad con los circuitos digitales.

Las entradas analógicas fueron integradas a la IADPC debido a la necesidad de adquirir datos desde sensores analógicos. Éstos tienen una mejor capacidad para monitorear un proceso, pues su salida es un nivel de voltaje cuyo valor es función de la variable física que se desea medir. El rango de voltaje que entregan a la salida suele variar, dependiendo del tipo de sensor e incluso de la marca.

Los actuadores analógicos tienen la función de amplificar la señal a su entrada y transformarla en una variable física que tiene influencia en el control de un proceso; por ejemplo, la temperatura de la resistencia de un horno eléctrico. El hardware analógico de la IADPC esta constituido por los siguientes módulos:

- El convertidor analógico-digital (CAD) presente en el MCU de la tarjeta MINICON_08A.
- Cuatro circuitos adecuadores de entrada, para sendos canales del convertidor analógico digital aquí mencionado.
- Dos convertidores digital-analógico.

3.2 El convertidor digital-analógico DAC0800

Las señales analógicas de salida, en la IADPC, se crean mediante un convertidor digital-analógico. Como se mencionó en el capítulo anterior, este dispositivo se acopla a una salida digital de ocho bits, para transformar el número binario en un nivel de voltaje. A continuación se revisarán las características del convertidor DAC0800, usado para las salidas digitales de la IADPC. La figura 3.1 muestra su diagrama de bloques funcionales.

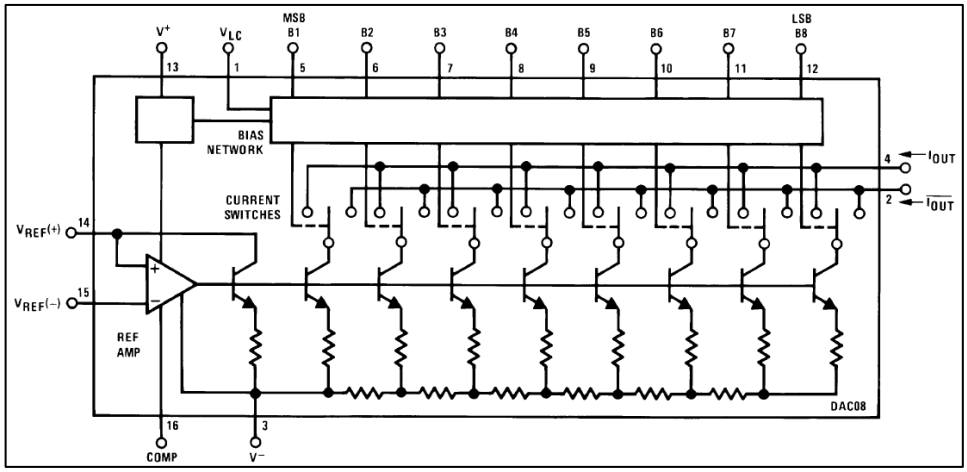


FIGURA 3.1: *Diagrama de bloques del convertidor digital analógico DAC0800*

El DAC0800 es un convertidor digital-analógico de ocho bits, de alta velocidad con tiempo de estabilización de 100ns. También dispone de la corriente suficiente para permitir a su salida voltajes pico a pico de 20V, en el caso de que la carga se conforme de resistores.

La inmunidad al ruido en sus entradas le da la capacidad de aceptar varios niveles lógicos. El comportamiento y características del dispositivo prácticamente no cambian, mientras el rango del voltaje de alimentación se encuentre entre +- 4.5V al +-18V. Si al dispositivo se le alimenta +-5 Volts su consumo es de tan solo 33mW, midiendo independientemente a los voltajes lógicos. La figura 3.2 muestra la configuración mínima necesaria de sus terminales, para que pueda operar, mediante el cambio de esta configuración, es posible hacer que el DAC0800 entregue a su salida diferentes rangos de voltaje.

3.2.1 Características principales:

- Rápida estabilización de la corriente de salida: 100ns
- Error en escala completa: +- el bit menos significativo
- No linealidad por temperatura: +-0.1%
- Salida de alto rendimiento: de -10V a +18
- Interface directa con TTL, CMOS,PMOS y otros
- Amplio rango de voltaje de alimentación: de +-4.5V a +-18V
- Bajo consumo de energía: 33 mW a +-5V.

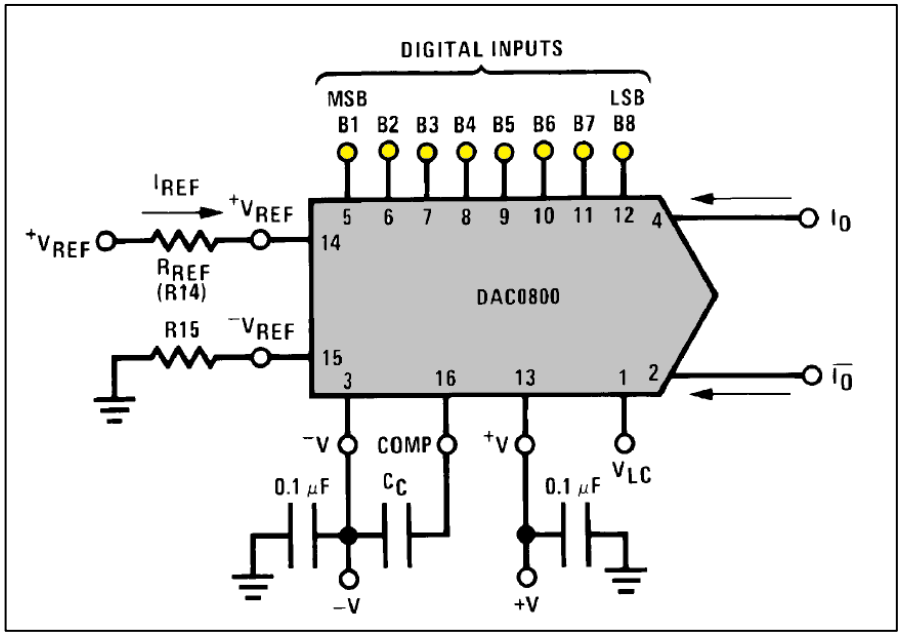


FIGURA 3.2: *Configuración básica de las terminales del DAC0800*

La IADPC tiene dos salidas analógicas de voltaje, conectadas a los puertos digitales 2 y 3 del MD-L. Como se mencionó anteriormente, cada uno de los latch de salida, tienen la capacidad de manejar ocho bits, en el caso de las salidas analógicas estos ocho bits son utilizados para enviar la información requerida por el CDA, para generar un nivel de voltaje analógico. Éste nivel de voltaje varia dependiendo el rango configurado en el hardware del CDA.

Para obtener un nivel de voltaje del convertidor DAC0800, éste recibe en su entrada un número en un rango que va desde el “00000000” al “11111111” (ocho bits provenientes del Latch correspondiente), para después generar a su salida, un nivel de voltaje diferente para cada uno de los números que se presenten en su entrada, dando como resultado 255 niveles.

La IADPC tiene la entrega en sus salidas analógicas, voltajes bipolares, de -10 a 10 V. Para lograr este fin se utilizo al DAC0800 en su modalidad simétrica, esta configuración se muestra en la figura 3.3. Para esta configuración el DAC0800 se integra a un amplificador operacional de tal forma que la salida de voltaje Eo se calcula de acuerdo a la ecuación 3.1.

$$E_0 = V_{REF}\left(\frac{-255}{256} + \frac{2X}{256}\right) \qquad (3.1)$$

Donde X es el número binario en la entrada del CDA, transformado a una base decimal. Por otra parte para que el circuito trabaje correctamente, es necesario que tanto las resistencias de las terminales 4 y 2 y la resistencia asociada al voltaje de referencia, sean iguales. Otra de las consideraciones que se descubrieron durante las pruebas del circuito, fue que el voltaje de alimentación debía estar balanceado, se detallara esta particularidad, más adelante.

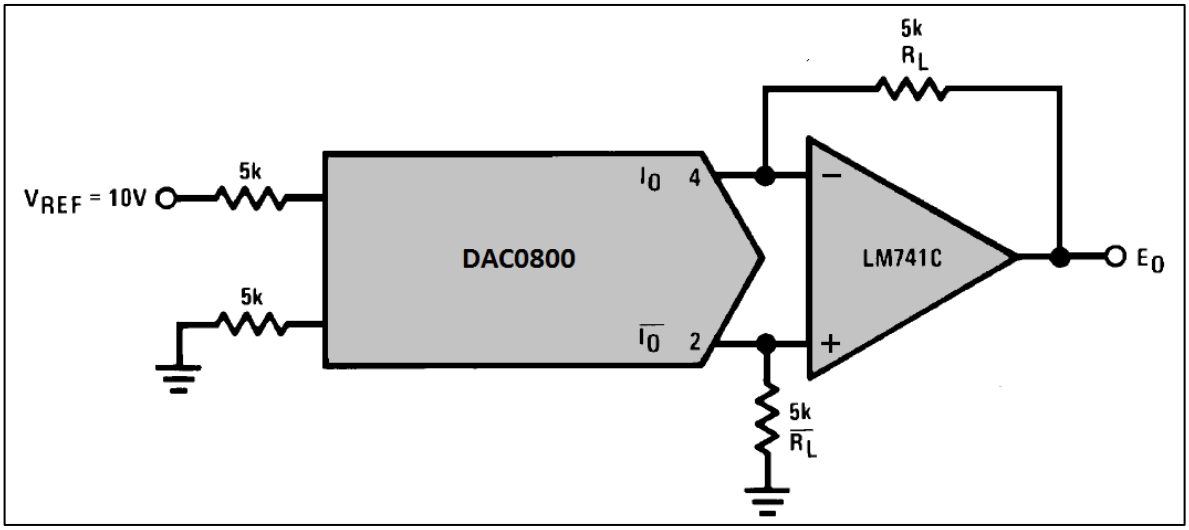


FIGURA 3.3: *Configuración necesaria para la operación simétrica del DAC0800*

Gracias a esta configuración, el DAC0800 tiene la capacidad de entregar a su salida una escala de voltajes simétrica. En la tabla 3.1 se muestra el valor de voltaje esperado en la terminal E0, para los números binarios que se encuentran a los extremos positivo y negativo de la escala.

Tabla 3.1: *Muestra el valor de voltaje correspondiente a cada uno de los números, localizados en los extremos de la escala del 0 al 255.*

	B0	B1	B2	B3	B4	B5	B6	B7	Eo
Escala completa positiva	1	1	1	1	1	1	1	1	9.96
Escala completa positiva - bit menos significativo	1	1	1	1	1	1	1	0	9.88
(+) cero	1	0	0	0	0	0	0	0	0.4
(-) cero	0	1	1	1	1	1	1	1	-0.4
Escala completa negativa+ bit menos significativo	0	0	0	0	0	0	0	1	-9.88
Escala completa negativa	0	0	0	0	0	0	0	0	-9.96

La resolución del CDA, es el resultado de dividir al rango de niveles de voltaje posibles a la salida del CDA, entre el número de combinaciones posibles en la entrada del CDA (ocho bites). Como la salida del CDA esta configurada para poder entregar de -10 a 10 (20 volts). El resultado es la diferencia entre dos niveles correspondientes a dos números binarios consecutivos, tal como se muestra en las ecuaciones 3.2 y 3.3.

$$\Delta = \frac{20}{256} \qquad (3.2)$$

$$\Delta = .078125 \qquad (3.3)$$

El circuito del módulo de salidas analógicas de la IADPC, se muestra en la figura 3.4; se incluye el hardware de cada uno de los dos CDA y los conectores para el conexionado de las entradas de los CDA, con los puertos binarios de la IADPC asociados con la entrada de cada uno de los dos CDA.

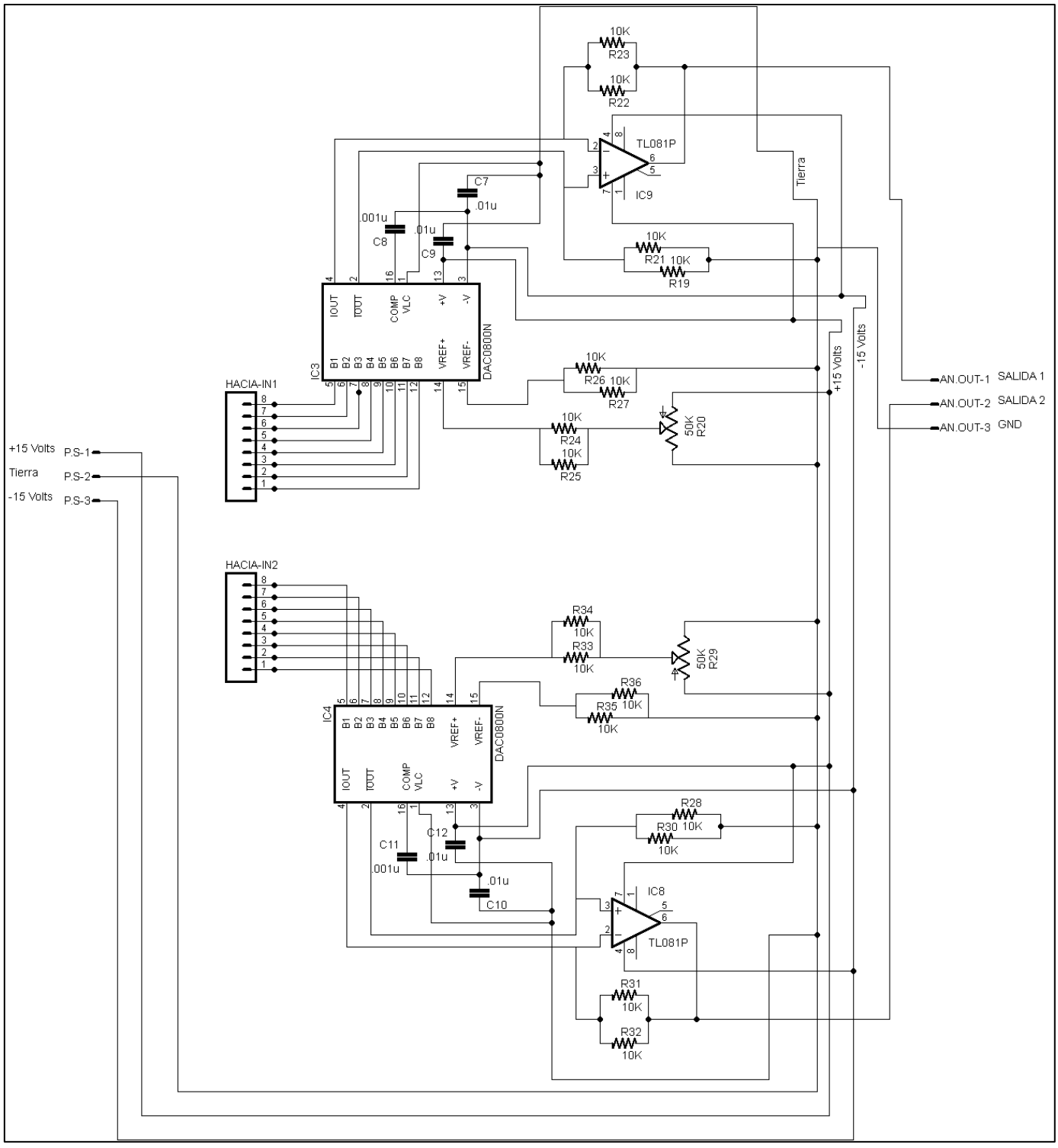


Figura 3.4: *Diagrama de conexiones del módulo de salidas analógicas, incluye los dos CDA*

3.3 El convertidor Analógico-Digital

Para adquirir los datos de un sensor de salida analógica, se utiliza un convertidor analógico digital. Éste último dispositivo tiene la función de generar un número binario a partir de un nivel de voltaje analógico. La IADPC utiliza los canales del convertidor analógico-digital (CAD) embebido en el MCU 68HC908GP32CP de la tarjeta MINICON_08A.

El MCU solo tiene un CAD, el cual es multiplexado a ocho canales, contenidos en las ocho líneas del puerto B. Como puede apreciarse en la figura 3.5 el puerto B puede ser configurado, mediante sus registros de control, como un puerto digital de ocho bits, o bien, como ocho canales de entrada analógica, conectados con el CAD del MCU.

Para poder utilizar al CAD, es necesario seleccionar al bit del puerto B que se utilizara en ese momento como canal analógico, hecho esto se debe esperar a que el CAD termine la conversión, para finalmente poder extraer el dato del registro de resultados del convertidor. Todo el proceso descrito anteriormente se realiza mediante software, el cual se analizará detalladamente durante el próximo capítulo.

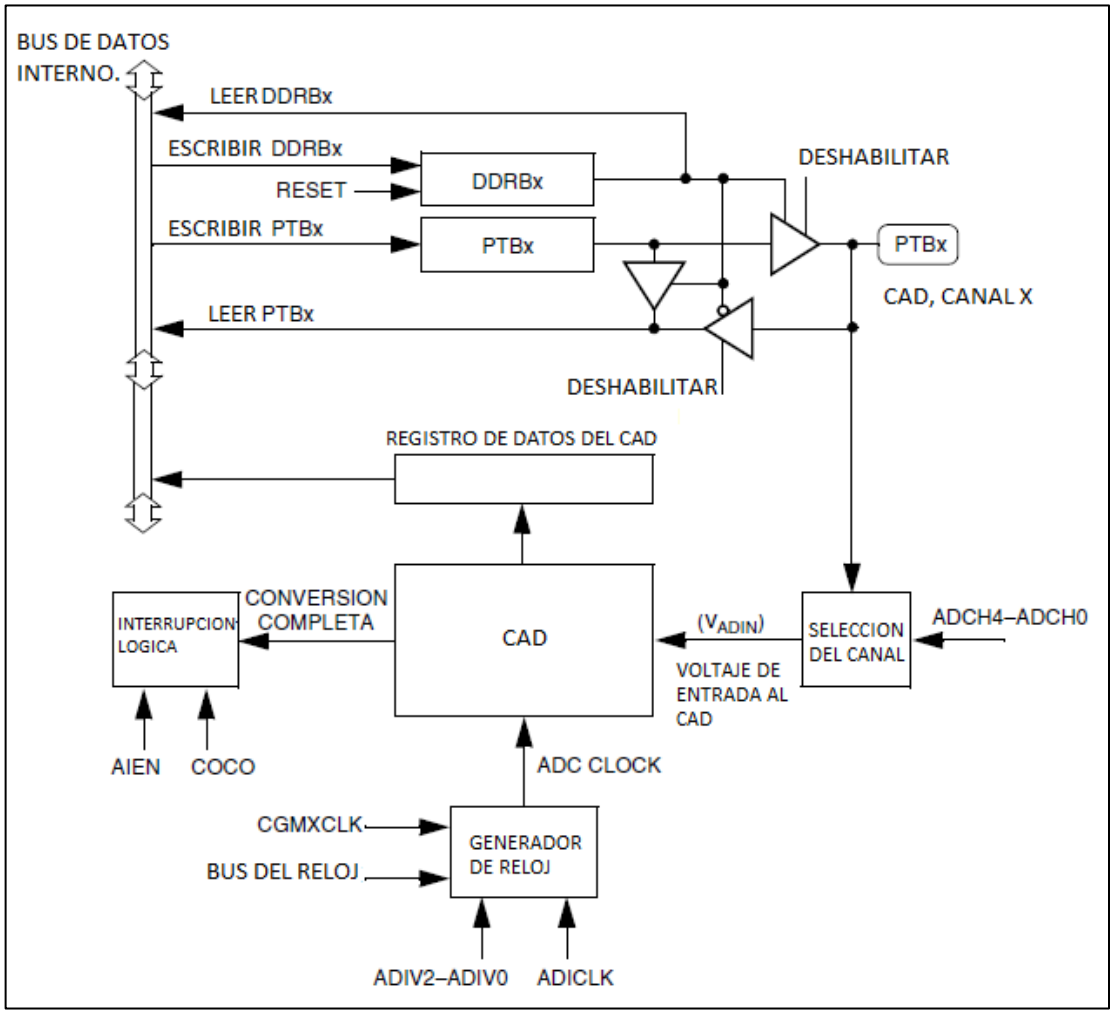


FIGURA 3.5: *Diagrama de bloques del convertidor analógico-digital del MCU 68HC908GP32CP.*

Debido a que el voltaje que soportan los canales del CAD varía en un rango de 0 a 5 volts, y a que el tamaño de los registros del MCU es de ocho bits, la resolución de los convertidores se calcula dividiendo el rango de voltaje configurado entre el número de combinaciones posibles con ocho bits, de manera que se obtienen las ecuaciones 3.4 y 3.5:

$$\Delta = \frac{5}{256}[V] \qquad 3.4$$

$$\Delta = .01953\ [V] \qquad 3.5$$

Lo que significa que el CAD es capaz de detectar a su entrada, cambios de voltaje no menores a .01953 [V], cualquier cambio menor a este valor será imperceptible para el CAD.

El CAD del MCU, por si solo quedaría muy limitado, debido a que el rango que el CAD del MCU comprende de los 0V a los 5V. Para brindarle más versatilidad a las aplicaciones que la IADPC es capaz de realizar, se diseño un circuito cuya finalidad es la de adaptar distintos rangos de voltaje, tanto positivos como negativos, en un rango aceptable para el convertidor analógico-digital del MCU. Es importante mencionar que si un voltaje de entrada fuera de los rangos antes mencionados podría dañar a la IADPC.

Para este fin se diseño un circuito que aquí denominamos como adecuador de entrada, sin embargo, antes de empezar con la descripción de las características de éste circuito, es necesario hablar de su principal elemento constitutivo, el amplificador operacional LM324.

3.4 El amplificador operacional LM324.

EL LM 324 es un circuito integrado (CI), conformado por cuatro amplificadores operacionales de alta ganancia y de frecuencia compensada internamente. Los amplificadores operacionales de éste están diseñados para funcionar con una sola fuente de poder y de varios rangos de voltaje. A continuación sus principales características:

- Frecuencia internamente compensada.
- Amplio ancho de banda (ganancia unitaria): 1MHZ (Compensado por temperatura).
- Amplio rango de fuente de alimentación: [3VDC , 30VDC] o fuentes bipolares: [+-1.5, +-15] VDC.
- Baja corriente de drenaje en la fuente (1mW/ amp op a +5VDC).
- Baja corriente de polarización: 45 nA DC (compensado por temperatura).
- Bajo voltaje de offset a la entrada: 2mDC y corriente de offset de 5nA DC.
- Rango del voltaje diferencial de entrada, igual al de la fuente de alimentación.
- Voltaje de salida: [0 VDC, VCC – 1.5]

El siguiente esquema, de la figura 3 muestra como se distribuyen los cuatro amplificadores operacionales en el LM324.

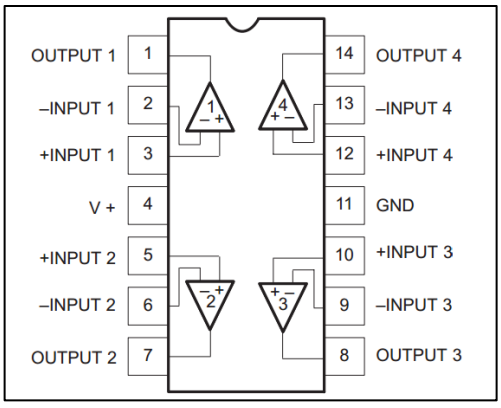


FIGURA 3.6: *Diagrama funcional del LM324*

3.5 El circuito adecuador de entrada

Para lograr hacer que los rangos de voltaje a la entrada de la IADPC fueran compatibles con el rango de voltaje aceptado por el CAD del MCU, se opto por realizar un circuito cuya salida de voltaje fuese una función lineal, de su voltaje de entrada. El circuito adecuador de entrada se diseño con seis rangos de voltaje seleccionables por el usuario:

- Rango bipolar de -10V a +10V
- Rango bipolar de -5V a +5V
- Rango bipolar de -1V a +1V
- Rango unipolar de 0V a +10V
- Rango unipolar de 0V a +5V
- Rango unipolar de 0V a +1V

Dependiendo del rango que se seleccione en el adecuador de entrada, este devolverá a su salida un voltaje en función a su entrada. Todas las operaciones a las que se somete al voltaje de entrada, consisten en divisiones, multiplicaciones y sumas, las cuales pueden ser fácilmente implementadas mediante un arreglo de amplificadores operacionales.

A continuación se describen las de los amplificadores operacionales en el circuito adecuador de entrada, más adelante se describe el diseño de éste.

3.5.1 Adecuador del rango [-10, 10] al rango [0, 5].

Para cualquier nivel entre -10V y +10V a la entrada del adecuador, el voltaje a su salida se mantiene entre 0V y 5V, como se muestra a continuación en la ecuación 3.6:

$$V_{salida} = \frac{V_{entrada}}{4} + 2.5 \qquad (3.6)$$

La figura 3.7 muestra la forma en que el amplificador debe ser configurado, de manera que realice las operaciones necesarias para adecuar un voltaje de entrada correspondiente a este rango, en un rango de 0 a 5 volts.

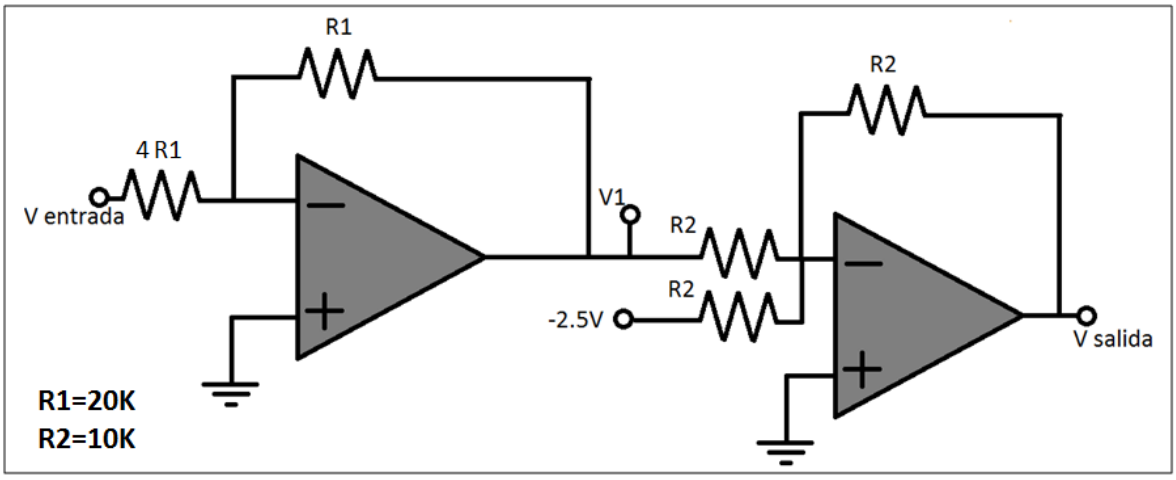


FIGURA 3.7: *Configuración correspondiente al circuito adecuador del nivel de voltaje de +10V a -10V*

En el primer amplificador se realiza un proceso de atenuación. Para realizar una atenuación no se puede utilizar un amplificador con una configuración no inversora, pues la mínima ganancia que puede entregar esta última configuración es de "1". Debido a este comportamiento, se utiliza un amplificador en configuración inversora, y de esta forma se realiza una atenuación y una inversión del voltaje de entrada: V1= (-1)*(1/4)*(Voltaje de entrada).

Finalmente, en el segundo amplificador operacional, se utiliza una configuración de sumador inversor, de tal manera que a V1 se le suma el offset necesario y se vuelve a invertir, corrigiendo la polaridad del voltaje y adecuándolo en un rango de 0V a 5V.

3.5.2 Adecuador del rango [-5, 5] al rango [0, 5]

Para cualquier nivel entre -5V y +5V a la entrada del adecuador, el voltaje a la salida se mantiene entre 0V y 5V mediante la ecuación 3.7:

$$V_{salida} = \frac{V_{entrada}}{2} + 2.5 \qquad (3.7)$$

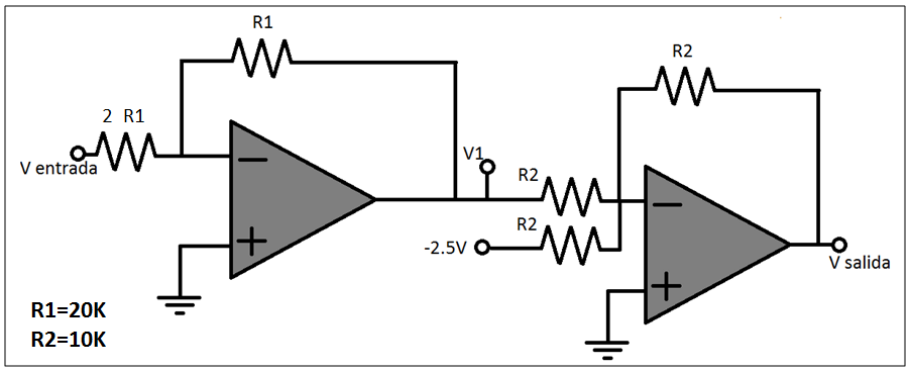


FIGURA 3.8: *Configuración correspondiente al circuito adecuador del nivel de voltaje de +5V a -5V*

3.5.3 Adecuador del rango [-1, 1] al rango [0, 5]

Para cualquier nivel entre -1 V y +1 V a la entrada del adecuador, el voltaje a la salida se mantiene entre 0V y 5V, de acuerdo a la ecuación 3.8:

$$V_{salida} = 2.5 * V_{entrada} + 2.5 \qquad 3.8$$

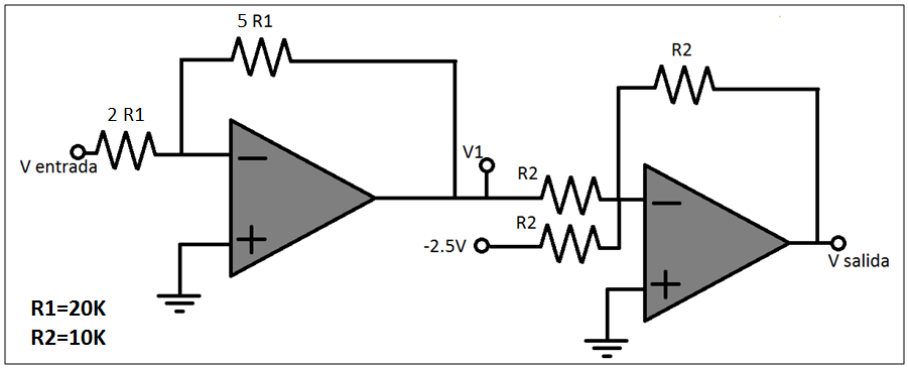


FIGURA 3.9: *Configuración correspondiente al circuito adecuador del nivel de voltaje de +1V a -1V*

3.5.4 Adecuador del rango [0, 10] al rango [0, 5]

Para cualquier nivel entre 0V y +10V a la entrada del adecuador, el voltaje a la salida se mantiene entre 0V y 5V, mediante la ecuación 3.9:

$$V_{salida} = \frac{V_{entrada}}{2} \qquad (3.9)$$

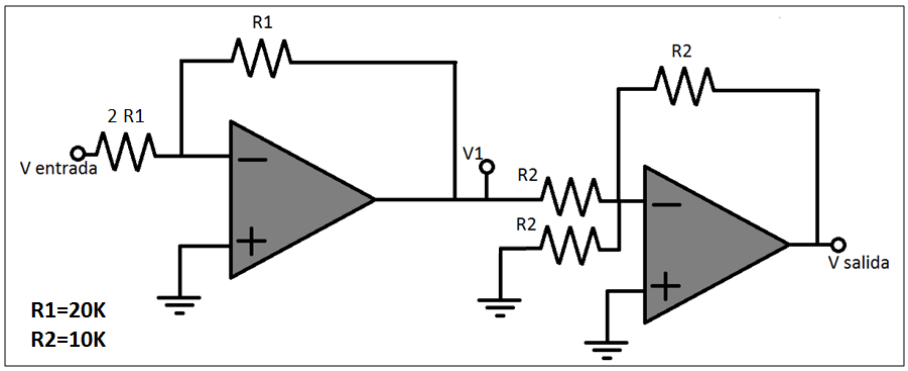


FIGURA 3.10: *Configuración correspondiente al circuito adecuador del nivel de voltaje de 0 V a 10 V*

3.5.5 Adecuador del rango [0, 5] al rango [0, 5]

Para cualquier nivel entre 0V y +5V a la entrada del adecuador, el voltaje a la salida se mantiene entre 0V y 5V mediante la ecuación 3.10:

$$V_{salida}=V_{entrada} \qquad (3.10)$$

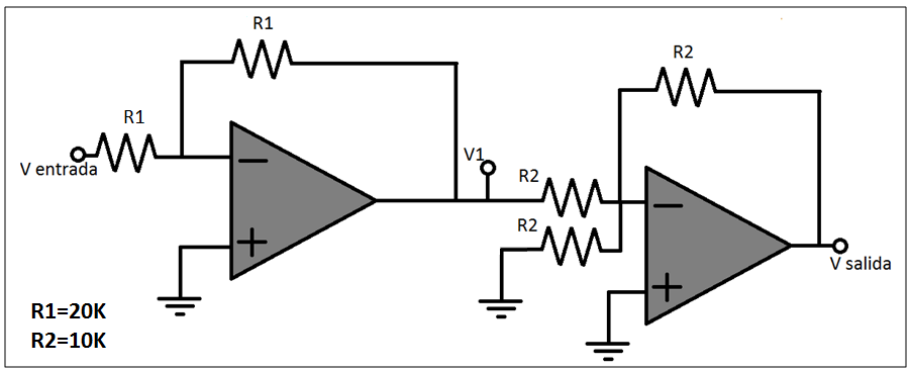


FIGURA 3.11: *Configuración correspondiente al circuito adecuador del nivel de voltaje de 0V a 5V*

3.5.6 Adecuador del rango [0, 1] al rango [0, 5]

Para cualquier nivel entre 0V y +1V a la entrada del adecuador, el voltaje a la salida se mantiene entre 0V y 5V mediante la ecuación 3.11:

$$V_{salida} = 5 * V_{entrada} \qquad (3.11)$$

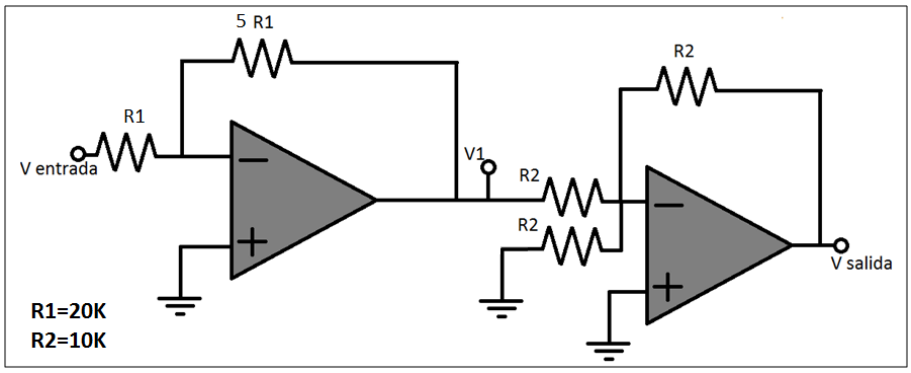


FIGURA 3.12: *Configuración correspondiente al circuito adecuador del nivel de voltaje de 0V a 1V*

3.5.7 Circuito de adecuación de rango seleccionable

El circuito de adecuación de rango seleccionable es un dispositivo conformado por cuatro amplificadores operacionales, dos de ellos están conectados entre si, en la forma mostrada en las figuras 3.7 a 3.12, contando con interruptores selectores de componentes resistivos, que se fijan de acuerdo con cada uno de los rangos de entrada analógica configurables en la IADPC. Los otros dos amplificadores operacionales son utilizados para aislar la impedancia que se tiene, tanto a la entrada del circuito de adecuación, como a su salida. En la figura 3.13 se muestra al circuito de adecuación de rango seleccionable.

La IADPC cuenta con cuatro de estos circuitos para dos entradas analógicas. Por motivos de experimentación este circuito se hizo compatible con otros MCU de la familia HC(S)08 que admiten a su entrada 3 V. Según la posición del jumper “J1” (véase la figura 3.13) es posible obtener a la salida un rango de [0,5] Volts o uno de [0,3] Volts.

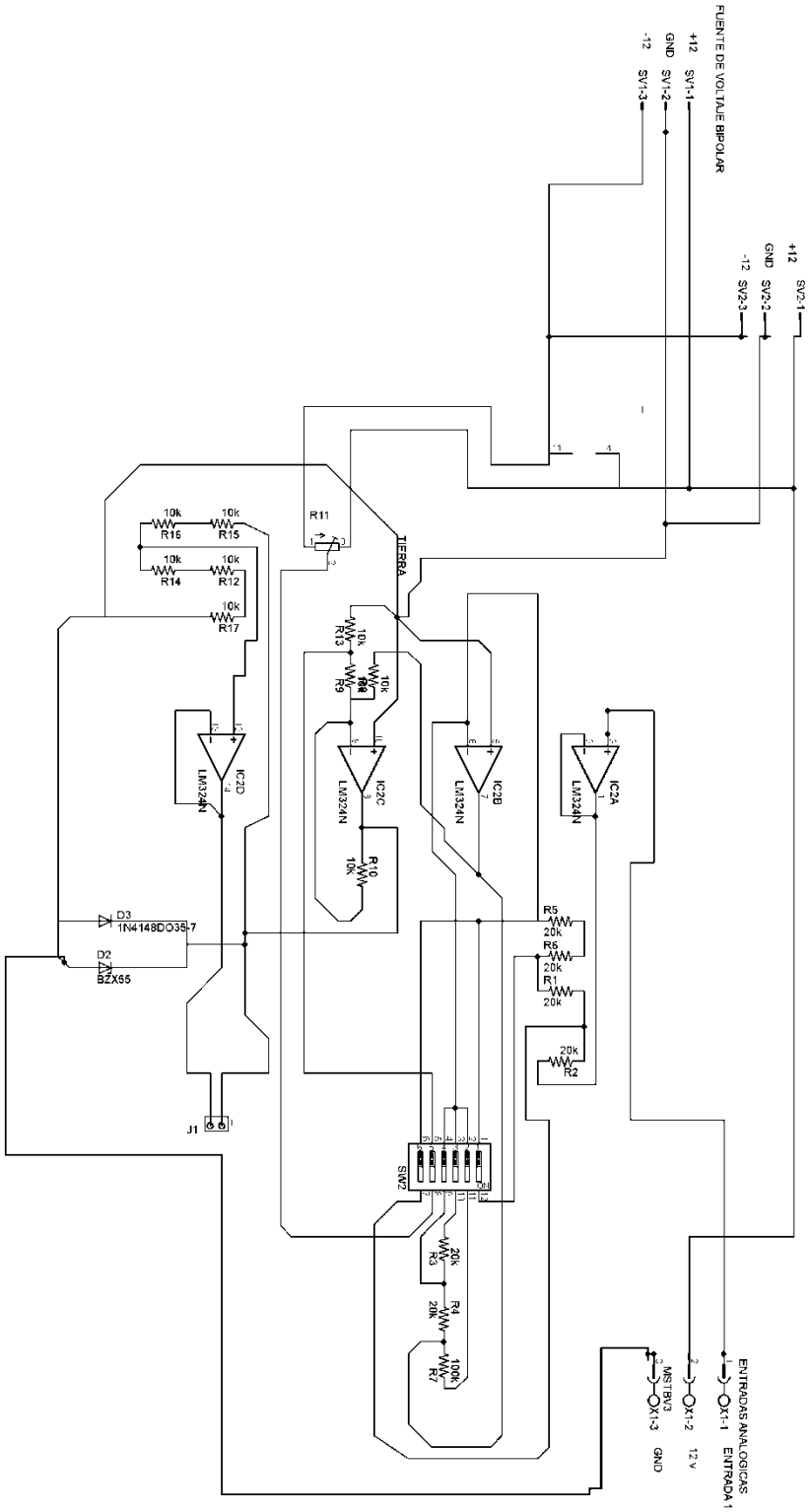


FIGURA 3.13: *Diagrama de conexiones del circuito de adecuación de rango seleccionable, incluye una terminal de 12 Vcd y una terminal a tierra.*

El control del circuito adecuador de entrada se realiza por medio de un dip-switch de seis contactos, y por medio de ellos se reconfiguran las resistencias del amplificador encargado de la amplificación (o atenuación, según el caso). Además, por medio de uno de estos contactos se permite la conexión al voltaje de referencia de -2.5V, para los rangos de voltaje bipolares. La tabla 3.2 muestra la manera de posicionar al dip-switch, dependiendo del rango de voltaje de entrada analógica que se desee emplear en un momento dado.

TABLA 3.2: *Relación entre la configuración del dip-switch, correspondiente a cada entrada analógica, y el rango de voltaje que se desea conectar a la entrada del adecuador.*

CONTACTOS	RANGO ENTRDA ANALÓGICA					
	[-10,+10]	[-5,+5]	[-1,+1]	[0,10]	[0,5]	[0,1]
S1	OFF	ON	ON	ON	ON	OFF
S2	OFF	OFF	ON	OFF	OFF	ON
S3	OFF	OFF	OFF	OFF	ON	OFF
S4	ON	ON	OFF	ON	OFF	OFF
S5	ON	ON	ON	OFF	OFF	OFF
S6	OFF	OFF	OFF	OFF	OFF	ON

La conexión del adecuador con el convertidor analógico-digital del MCU se encuentra protegida por un diodo rectificador y un diodo zener, que en conjunto tienen la función de evitar que un voltaje mayor que 5 V, o de polarización negativa, dañen al convertidor del MCU. Ver la figura 3.14.

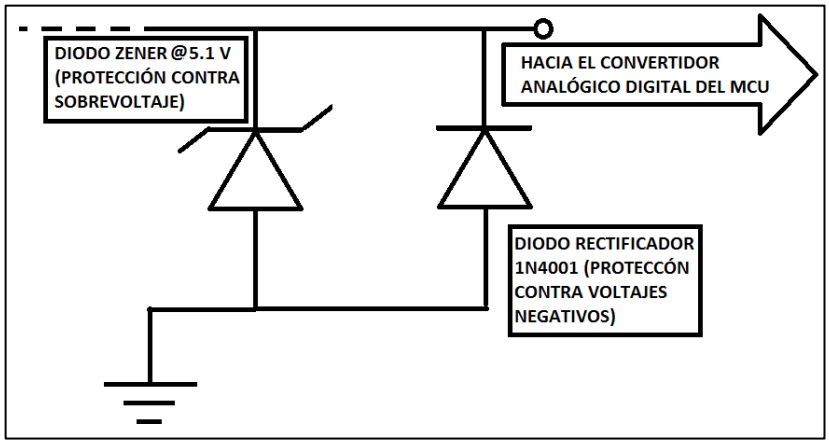


FIGURA 3.14: *Configuración del circuito de protección de la salida que va, del adecuador al convertidor analógico digital del MCU.*

Gracias a esta protección se puede garantizar que el adecuador siempre entregará un voltaje adecuado para el MCU (funcionando en las condiciones para las que éste fue diseñado), incluso si el usuario comete un error al configurar un rango inadecuado para el voltaje de entrada.

Finalmente, el adecuador de entrada conecta sus salidas a las terminales del MCU asociadas con las entradas analógicas AD0, AD1, AD2 y AD3. Véase la tabla 3.3.

TABLA 3.3: *Conexionado entre las salidas* de los circuitos de adecuación de rango seleccionable *y las terminales del puerto B.*

Circuito de adecuación de rango seleccionable	Terminales del Puerto B
1	PTB0/AD0
2	PTB1/AD1
3	PTB2/AD2
4	PTB3/AD3

3.6 Pruebas a las que se sometieron los circuitos analógicos de la IADPC

3.6.1 Pruebas del hardware

Para comprobar el diseño de la IADPC, los circuitos del CDA y el módulo de entradas analógicas fueron armados en Proto-Board. El proceso para comprobar al armado del hardware y el estado de los circuitos integrados es similar al descrito en el capítulo anterior, para el armado de los circuitos digitales.

Estos circuitos no se alimentaron con la fuente propia de la tarjeta de desarrollo MINICON_08A, debido a que necesitan voltajes bipolares para su correcto funcionamiento. Para este fin, fue necesario armar una fuente bipolar (± 12 Volts) que alimentara los circuitos tanto del CDA y el módulo entradas analógicas.

3.6.2 Armado y prueba del hardware del CDA

Antes de iniciar con el proceso, se comprueba que el hardware del CDA está conectado correctamente y que los circuitos integrados no están dañados. Hecho esto último, las entradas binarias de los dos CDA se conectan respectivamente con los latch 2 y 3 presentes en el MD-L. Finalmente se alimenta con una fuente bipolar de ±12 V y se acopla la tierra del CDA, con la de la tarjeta de desarrollo MINICON_08A.

Como se puede aprecia en la figura 3.4, cada convertidor digital-analógico tiene asociado un potenciómetro. Este último esta conectado, en cada caso, a la terminal VREF+ (pin 14) del circuito integrado DAC0800. Cada potenciómetro debe estar ajustado de tal manera que en la terminal VREF+, de cada DAC0800, se tenga un nivel de voltaje de +10 V. Ahora el CDA esta listo para funcionar.

3.6.3 Armado y prueba del hardware del circuito de adecuación de rango seleccionable

El primer diseño del circuito de adecuación de rango seleccionable se conformaba por potenciómetros en lugar de resistencias fijas, esto derivó en que el circuito fuera poco estable y en un alto costo. Por ese motivo se optó por sustituir los potenciómetros por resistencias de precisión, las cuales tienen un mejor precio y proveen una mejor estabilidad al circuito. Antes de iniciar con el proceso se comprueba que el hardware del Circuito adecuador esta conectado correctamente y que los circuitos integrados no están dañados.

De la misma forma que en el caso anterior, éste circuito se alimenta con una fuente bipolar de ±12 Volts y se acopla la tierra del adecuador, con la de la tarjeta de desarrollo MINICON_08A.

Como se puede apreciar en la figura 3.13, cada circuito adecuador, del módulo de entradas analógicas tiene asociado un potenciómetro, el cual establece el voltaje de referencia. Para cada caso, cada potenciómetro debe ser ajustado de tal manera que, su terminal 2, tenga un nivel de voltaje de 2.5 V.

Para comprobar que el circuito adecuador devolvía a su salida un voltaje de 0 V a 5 V, se estableció en la entrada de voltaje, un potencial de prueba mediante un potenciómetro. Esta prueba se realizo con cada una de las configuraciones posibles para este circuito. Ver tabla 3.2.

Ahora que el módulo de entradas analógicas esta listo para funcionar, cada circuito adecuador conecta sus salidas a las terminales del puerto B, como lo muestra la tabla 3.3.

3.6.4 Programas de prueba

Los programas de prueba se desarrollaron para comprobar la interacción entre la tarjeta MINICON_08A y el hardware analógico, algunas partes del código de estos programas, sirvieron como precedentes en el desarrollo del software de base requerido en el MCU para fines de la IADPC.

3.6.4.1 Programa de prueba para el CDA

El siguiente programa se desarrollo para fines de validación de las rutinas de escritura analógica. Éste programa se diseño para ejecución en la memoria RAM del MCU de la tarjeta MINICON_08A. Su objetivo es devolver en el canal de salida analógica, asociado al latch 02, una rampa de voltaje.

Para lograr este objetivo se crea un contador de la siguiente manera: el número que se almacena en el registro a, se incrementa cíclicamente en uno, y va de 0 a 255. En cada ciclo, el número contenido en a, es depositado en el latch 02 del MD-L. Por su parte, el CDA genera un nivel de voltaje, para cada uno de los números generados por el contador, de la misma manera como se mostró en la tabla 3.1.

El resultado es un cambio de voltaje respecto al tiempo, en el canal de salida analógica correspondiente al latch 02. Para poder comprobar la salida del CDA, es necesario utilizar un osciloscopio, de manera de que se pueda observar mediante éste una señal periódica con una forma de onda conocida como diente de sierra. El programa de prueba aquí mencionado se muestra a continuación:

Constantes usadas:

pta equ $00 ;	Dirección del puerto A
ddra equ $04 ;	Registro de dirección del puerto A
ptd equ $03 ;	Dirección del puerto D
ddrd equ $07 ;	Registro de dirección del puerto D

Variables usadas:

numport equ $0a ;	Dirección que almacena el numero de puerto a usar
datsal equ $a1 ;	Dirección que almacena el dato de salida
datent equ $a2 ;	Dirección que almacena el dato de entrada
numfant equ $07 ;	Dirección del puerto fantasma

```
        org $0100 ;          Programando en RAM

       mov #$07,ddrd ;       El puerto D es salida en bits 0,1 y 2
       mov #numfant,ptd ;    Selecciona puerto fantasma
       mov #$02,numport ;    Selecciona el latch 02 (Canal de salida analógica).
```

Ciclo principal:

```
ciclo: sta datsal ;          Dato de salida ← a
       bsr escport  ;        Salta a rutina de escritura
       inca;                 a se incrementa en 1
       bra ciclo  ;          Salta siempre a la etiqueta  “ciclo”
```

Subrutina “escport”: Escribe un byte dato en puerto de salida de la Interfaz analógica y digital para PC (IADPC). Antes de invocar:

- (datsal) ←btye a escribir.
- (numport) ←Número de puerto digital de salida en la IADPC.
- Al retornar ya se habrá escrito el dato en el puerto implicado.

```
escport:  mov #$ff,ddra ;     El Puerto A es declarado como salida
          mov datsal,pta ;    Escribe dato al bus externo (Líneas del puerto A)
          mov numport,ptd ;   Selecciona puerto de salida  (Puerto D←numport)
          nop
          mov #numfant,ptd ;  Se genera flanco de subida en línea de selección del puerto y se
queda apuntando al puerto fantasma.

       mov #$00,ddra ;        El puerto A regresa a su estado de entrada
       rts
```

Como resultado de esta prueba se obtiene una señal cíclica en forma de diente de sierra.

3.6.4.2 Programa de prueba para el CAD

El siguiente programa se desarrollo para comprobar el funcionamiento del CAD embebido en el MCU. El objetivo de este programa es, enviar periódicamente al puerto A de la tarjeta MINICON_08A, el resultado de la conversión del canal 07 del CAD embebido en el MCU. A continuación se muestra el código del programa antes mencionado:

Las siguientes direcciones corresponden al convertidor analógico-digital:

```
 adscr equ $3c
adr   equ $3d
adclk equ $3e

pta equ $00
ddra equ $04
          org $0100  ;              Programando en RAM
            mov #$ff, ddra ;        Habilitar al puerto a como salida digital
```

Esta es la inicialización del convertidor:

```
          lda #$60  ;               reloj del CAD de 1MH
          sta adclk ;               reloj del CAD de 1MH
```

Rutina Principal:

```
conversión:   mov #$07,adscr ;          Selección del canal 07 del CAD
 checoco:     brclr 7,adscr,checoco ;   Espera a que termine la conversión
              lda adr ;                 a← resultado de la conversión

               sta pta                  Resultado enviado al puerto A

               bra  conversión  ;       Salta a la siguiente conversión
```

De las pruebas que a las que se sometió el hardware desarrollado para la IADPC, se puede concluir que los resultados obtenidos son los esperados; la interacción entre el software de prueba y el hardware analógico, cumple con los objetivos para los que fue diseñado. El desarrollo de las pruebas dio como resultado el mejoramiento del funcionamiento del hardware y la modificación del diseño, a favor de obtener un mejor costo.

Por otra parte, el software desarrollado en esta etapa sirve como referencia para la creación del software base programado en el MCU, para el manejo de las entradas y salidas analógicas. Dicho software se revisara de manera detallada en el siguiente capítulo.

Con las entradas y salidas analógicas en funcionamiento, en conjunto con los demás módulos descritos en capítulos anteriores, se da por terminado el diseño del hardware de la IADPC. Durante este proceso se afianzaron los conocimientos necesarios, acerca del MCU y la tarjeta de desarrollo MINICON_08A, para desarrollar el software de base necesario para el funcionamiento adecuado de la IADPC. Gracias a esto fue más fácil el desarrollo de los bloques de código que componen las distintas funciones de control que se le programaron al MCU.

CAPÍTULO 4

Desarrollo del software de base ejecutable en el MCU 68HC908GP32

Durante el presente capítulo se abordará el análisis del software programado en el MCU de la tarjeta MINICON_08A. Este software fue desarrollado con el sistema AIDA08, mencionado en el capítulo 1. El software de base requerido ejecutable en el MCU, se constituye por bloques de código, dedicados a una tarea específica, y relacionados entre ellos mediante variables y saltos a subrutinas. Estos segmentos o bloques de código son:

1) Bloque de inicialización.
2) Lazo principal (Administración del Buffer y rutina de interrupción).
3) Rutina para la interpretación de comandos.
4) Bloque de rutinas y subrutinas de control.
5) Rutina de interrupción.

El bloque de inicialización, se desarrollo debido a la necesidad de adecuar los parámetros que definen la funcionalidad del MCU, para los fines propios de la IADPC. Entre estos parámetros se encuentran el reloj del convertidor analógico digital, la velocidad del puerto serie o bien, la configuración de los registros de dirección de los puertos del MCU, utilizados en la IADPC.

Como se revisó en capítulos anteriores, la IADPC tiene la función de actuar como un seguidor de una computadora PC, utilizando el puerto serie como medio de comunicación, a través del cual se envían y reciben datos y comandos. Cada comando se constituye por un número binario dividido en dos partes, las cuales corresponden al tipo de acción y al número de puerto que se desea modificar o leer.

El microcontrolador es el encargado de recibir y administrar los datos. Cuenta con un espacio en la memoria habilitado como “Buffer” de recepción, al cual llegan los comandos desde la PC y se almacenan ahí, hasta que son invocados por el bloque de procesamiento de comandos. Dicho buffer es administrado por dos apuntadores de dirección, los cuales se coordinan para efectuar el ingreso y análisis de los comandos recibidos desde la PC. La interacción entre el Buffer y los apuntadores se explicara a detalle más adelante.

La rutina para la interpretación de comandos se encarga de interpretar los comandos contenidos en el Buffer, y una vez que se tienen los datos necesarios para llevar a cabo la instrucción, se invoca la subrutina que efectúa la acción apropiada para el comando recibido. Véase la figura 4.1 y figura 4.2.

En caso de que la instrucción sea de lectura, de alguno de los puertos de entrada, dicha acción incluirá una subrutina de transmisión a la PC, donde el dato que se envía es el número binario que se obtuvo en la lectura.

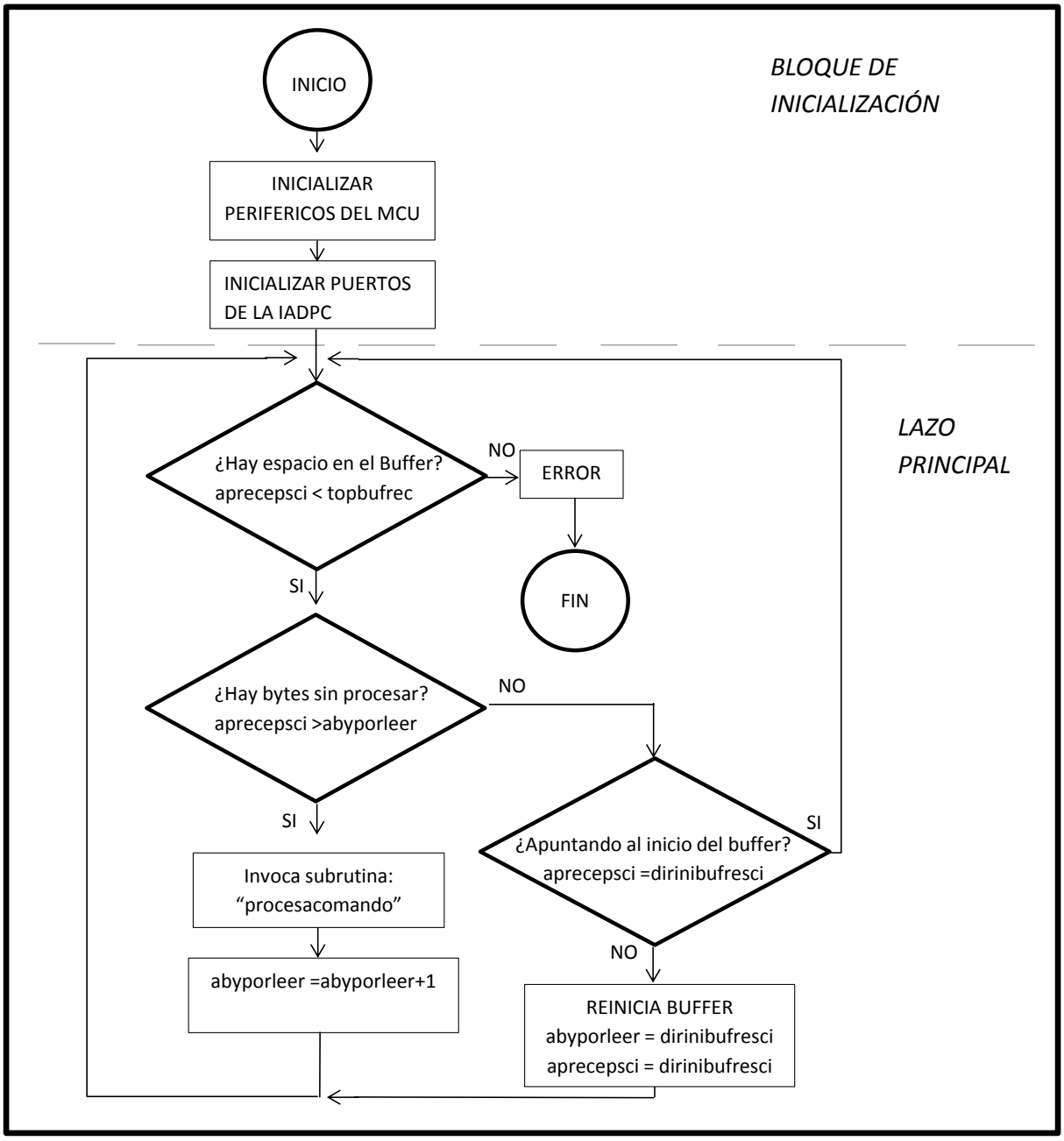


FIGURA 4.1: *Diagrama de flujo básico del software embebido en la IADPC. La cadena "dirinibufresci" esta asociada a la dirección inicial del buffer, mientras que "topbufrec" a la dirección final de mismo. Los apuntadores "aprecepsci" y "abyporleer" se definen en 4.2.1*

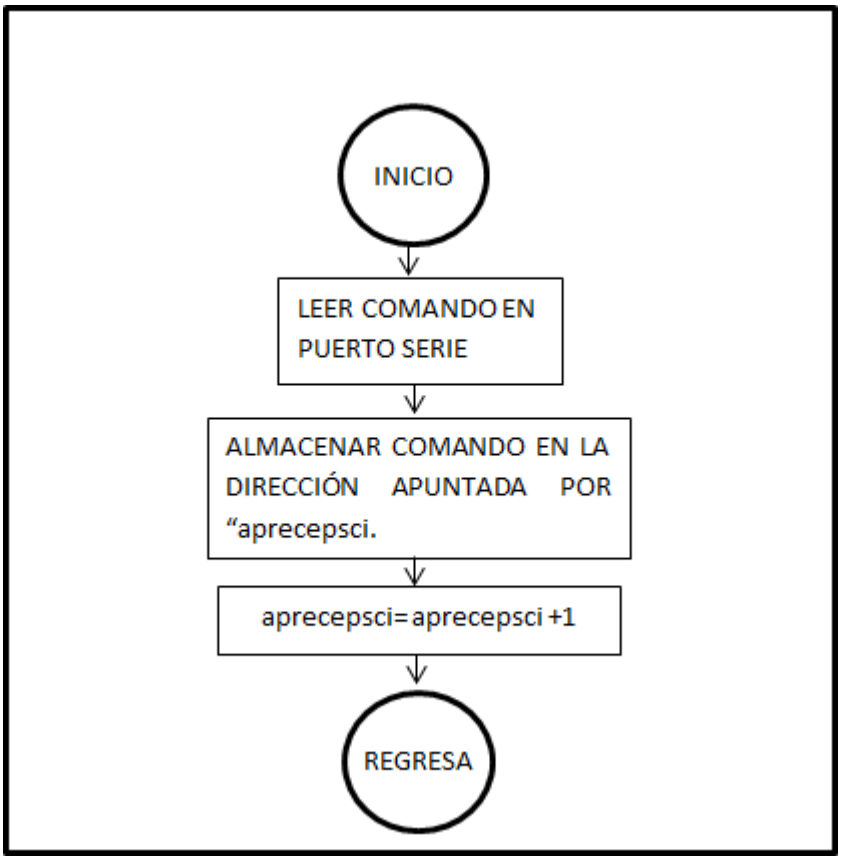


FIGURA 4.2: *Diagrama de flujo que muestra la rutina de servicio a la interrupción. Ésta se puede ejecutar aleatoriamente en cualquier parte del lazo principal. Se activa cada vez que se recibe un comando por el puerto serie. El apuntador "aprecepsci"" se define en 4.2.1*

A continuación se expondrán y revisarán cada uno de estos bloques de código; para una mejor comprensión de los mnemónicos utilizados por el ensamblador, se recomienda consultar el manual del MCU 68HC908GP32, así como el manual de usuario del sistema AIDA08.

4.1 Bloque de inicialización

Para el desarrollo del software se utilizaron múltiples variables, utilizadas para almacenar datos que posteriormente tendrán un tratamiento y una interpretación. También son usadas como auxiliares en el procesamiento de datos y en la inicialización de parámetros.

Una variable es una cadena de caracteres, a la cual se le asigna una locación de memoria. Dicha locación de memoria puede estar relacionada con uno de los sistemas internos del MCU (como el control de dirección de datos de un puerto o el convertidor analógico-digital), o bien, puede ser una locación disponible en el mapa de memoria, utilizada como auxiliar en los procesos programados en el MCU.

Gracias al uso de cadenas o “variables” se simplificó el desarrollo del software, pues en lugar de programar utilizando las direcciones de las locaciones de memoria, se usó una cadena referida a la función que tiene la variable o a que sistema esta asociada. Para asignar una variable se utiliza el mnemónico “equ” cuya función es relacionar a la cadena con la dirección de memoria que se le asignó.

El bloque de inicialización también comprende la configuración de los parámetros de aquellos subsistemas, embebidos en el MCU, antes mencionados (puerto serie, CAD y registro de dirección de datos de los puertos). Este bloque sólo se ejecuta una vez, cada que la IADPC es energizada y puesta en operación, con el fin de que cada subsistema funcione como se requiere. Este bloque no forma parte de los procesos cíclicos de la IADPC.

A continuación se expondrá al bloque aquí mencionado, se incluye al principio las definiciones de las cadenas propias de funcionalidades del MCU, empleados por el software de base completo.

Variables dedicadas al manejo del buffer:

```
dirinibufram      equ $00b0 ;
dirinibufresci    equ $0100 ;
topbufrec         equ $0200 ;
apcomposlcd       equ $a4
aprecepsci        equ $a6 ;
apbyporleer       equ $a8 ;
byrecibido        equ $aa
```

Variables auxiliares del puerto serie:

```
scc1 equ $13
scc2 equ $14
scc3 equ $15
scs1 equ $16
scs2 equ $17
scdr  equ $18
scbr equ $19
txsci  equ $fc87
```

Registro de configuración del MCU:

```
config equ $1f
```

Variables correspondientes a la configuración de los puertos, “A”, “C” y “D”:

```
pta equ $00
ddra equ $04
ptc equ $02
ddrc equ $06
ptd equ $03
ddrd equ $07
```

Variables para las rutinas de escritura a puertos de la IADPC:

```
numport equ $b0
datsal  equ $b1

tessal2 equ $b2
tessal3 equ $b3
tessal4 equ $b4
tessal5 equ $b5
```

Variables para las rutinas de escritura a puertos de la IADPC:

```
datent  equ $b6
numfant equ $07
tesby1com equ $b7
dirmon      equ $fc80
bycomsaltoamon equ $f0
```

Variables auxiliares del convertidor analógico-digital:

```
ncanal equ $b8
adscr equ $3c
adr   equ $3d
adclk equ $3e
```

Aquí inicia el código para inicializar sistemas. Al ser el programa de base en la IADPC es necesario guardarlo en la memoria FLASH:

```
org $8000 ;      Guardando en memoria FLASH
lda #$60  ;      reloj del CAD de 1MH
sta adclk   ;    1MH→adclk (reloj del CAD)
bset 0,config
```

Inicializa al puerto D como salida en los bits 0,1 y 2 y a “C” como salida en el bit 5:

```
mov #$07,ddrd          ;ptd es salida en bits 0,1 y 2
mov #numfant,ptd       ;selecciona numfant
mov #$16,ddrc
```

Inicializa apuntador de bufer de recepción serie en RAM:

```
ldhx #dirinibufresci
sthx aprecepsci
```

Inicializa apuntador de dirección de siguiente byte por leer:

```
ldhx #dirinibufresci
sthx apbyporleer
```

Inicialización del puerto serie para que se genere una interrupción al recibirse un byte:

```
bset 5,scc2
```

Al iniciar la IADPC todos los puertos, tanto digitales como analógicos, deben estar en “cero. Por esta razón se invoca la rutina “ini0portsal”, encargada de colocar un nivel bajo en todos los puertos de salida digital, incluyendo aquellos que están acoplados a un convertidor digital analógico (Salidas analógicas). Dicha rutina se detalla más adelante.

```
jsr ini0portsal

cli                ; Habilita las interrupciones. (Necesario para la recepción de comandos)
```

4.2 Lazo principal (Administración del Buffer y rutina de interrupción).

Durante la operación de la IADPC es posible que el envío de comandos desde la PC, sea más rápido que la capacidad del MCU de la IADPC para procesarlos. Por otra parte la IADPC y la PC no están coordinadas para la transmisión de comandos, por lo que para la IADPC, la recepción de comandos desde la PC es aleatorio.

Por estas razones se diseñó una rutina que se ejecuta cada que el puerto serie recibe un comando, mediante una interrupción de los procesos de la IADPC. Cada vez que esta rutina de interrupción se ejecuta, se almacena el comando recibido en localidades de memoria del MCU destinadas para esta función. A este conjunto de localidades de memoria se le llamará Buffer.

La IADPC esta programada para que verificar cíclicamente la existencia de comandos en el Buffer de recepción, mediante la ejecución del Lazo principal. Para que esta rutina de interrupción funcione, es necesario habilitar una interrupción cada vez que se reciba un dato por el puerto serie.

La forma de operar de la rutina de interrupción y la administración del buffer que almacena los comandos (el lazo principal), se describen a continuación.

4.2.1 El estado del Buffer al iniciar la IADPC.

El bloque de recepción de comandos utiliza dos apuntadores de dirección, los cuales son:

aprecepsci: Este apuntador contiene la dirección del último comando enviado por la PC, almacenado en el buffer. Al iniciar la IADPC apunta al principio del buffer. Ver figura 4.1

apbyporleer: Este apuntador contiene la dirección del comando que se pretende utilizar en el ciclo actual, es decir, el comando que se va a enviar al bloque dedicado a la interpretación de comandos. Al iniciar la IADPC apunta al principio del buffer. Ver figura 4.3

Inicialmente no hay comandos provenientes de la PC en el Buffer, por lo que ambos apuntadores apuntan a la misma dirección, como lo muestra la figura 4.3.

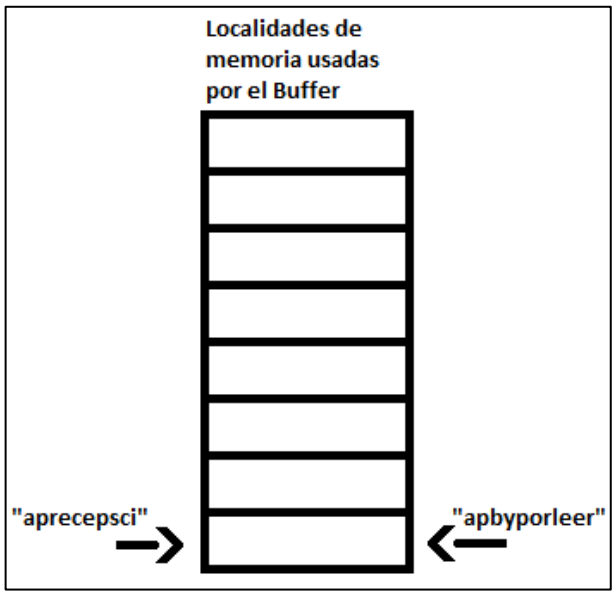


FIGURA 4.3: Estado inicial del buffer que contiene los comandos recibidos desde la PC

4.2.2 El estado del buffer después de una interrupción por recepción de comando

El MCU esta programado para que al recibir un comando desde la PC, se active una interrupción. La rutina de interrupción almacena el comando recibido en la dirección que actualmente apunta “aprecepsci”. Finalmente “aprecepsci” se incrementa en uno, de manera que ahora apunta a la siguiente dirección de memoria disponible en el Buffer. En otras palabras “aprecepsci” se incrementa cada vez que ocurre una interrupción. Ver figura 4.4.

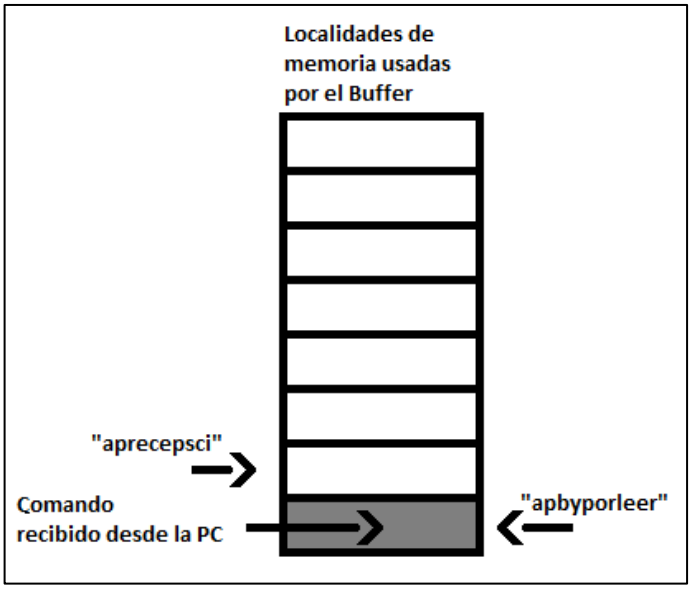


FIGURA 4.4: *Estado del Buffer después de una interrupción por recepción de comando desde la PC*

4.2.3 El estado del buffer durante la ejecución del Lazo principal.

El Lazo principal verifica el estado del Buffer comparando el valor del apuntador “aprecepsci” con el valor de “apbyporleer”. Si el primero es mayor que el segundo, significa que se ha producido previamente una interrupción por recepción de comando.

Consecuentemente se carga el comando y se invoca al código que se encargará de interpretarlo. Posteriormente se ejecuta las rutinas requeridas para la ejecución de la orden enviada desde la PC (ver 4.4).

Finalmente “apbyporleer” incrementa su valor en uno, es decir que ahora apunta a la siguiente dirección de memoria (donde se encuentra el siguiente comando a ejecutar). De esta manera “apbyporleer” incrementa su valor en uno cada vez que se ejecuta el Lazo principal. Ver figura 4.5.

Después de esto se pasa nuevamente a verificar si existe en el buffer de recepción algún comando pendiente de ejecución, iniciando de nueva cuenta la ejecución del Lazo principal.

Debido a que la PC envía comandos de forma aleatoria para la IADPC, durante un solo ciclo de ejecución del Lazo principal, pueden ocurrir varias interrupciones por recepción de comando, por lo que puede que “aprecepsci” se encuentre varias localidades de memoria adelantado, con respecto a “apbyporleer”.

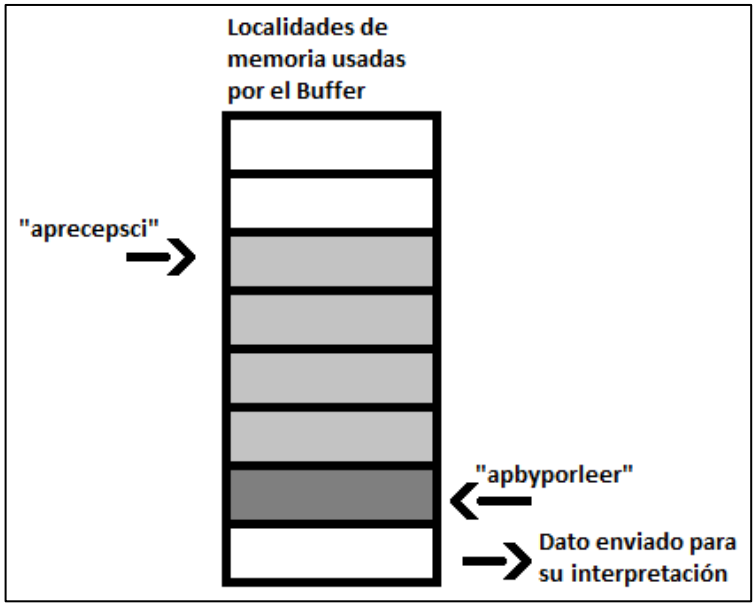


FIGURA 4.5: *Estado del buffer después del ciclo de ejecución del bloque de recepción de comandos*

Durante toda la operación de la IADPC, “apbyporleer” y “aprecepsci” se acercaran cada vez que se ejecute el Lazo principal y se alejaran cada vez que la PC envíe un comando. si “apbyporleer” y “aprecepsci” se encuentran apuntando a la misma dirección, se considera que todos los comandos en espera se han procesado; en dicho estado ambos apuntadores reinician a su estado inicial (véase 4.2.1).

En el caso de que el buffer se llene por completo (lo cual sucedería si “apbyporleer” y “aprecepsci” nunca se “alcanzaran”), el bloque receptor de comandos detendrá la ejecución de la IADPC. Para que el usuario tenga conocimiento de que la IADPC ha fallado a causa de la saturación del Buffer, se utiliza brillo de un LED testigo denominado TESSAT, ver 4.4.12.

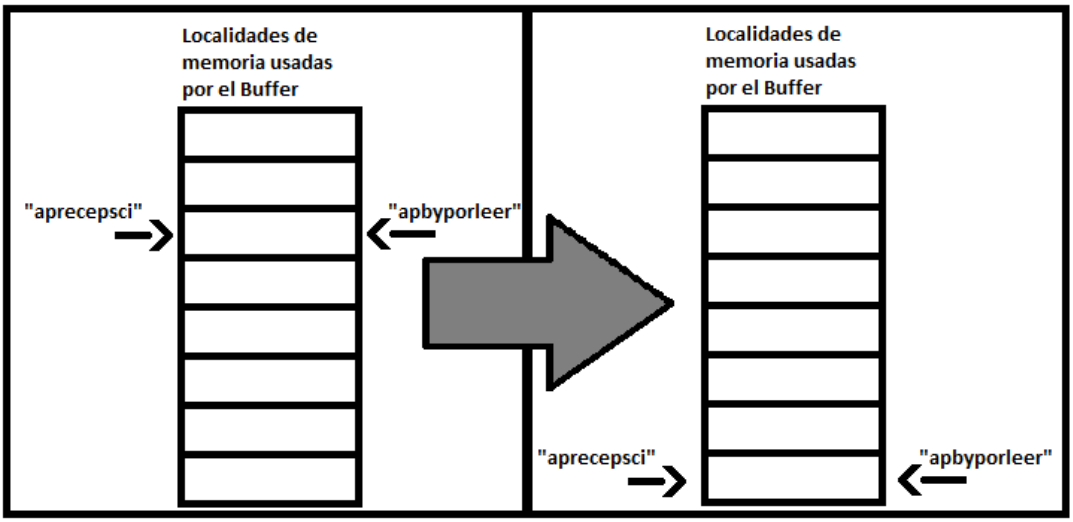


Figura 4.6: *Reinicio del buffer tras haber terminado de enviar todos los comandos, para su interpretación*

A continuación se muestra al código del Lazo principal:

```
esperarxsci:    ldhx aprecepsci
                cphx #topbufrec ;       Compara al apuntador con el tope del Buffer.
                bls bufnolleno  ;       De tener espacio en el buffer, salta a la etiqueta: bufnolleno
:                                       De estar lleno el buffer, desactiva las interrupciones y
                jsr error;              detiene la ejecución del programa

bufnolleno:     cphx apbyporleer ;      aprecepsci>apbyporleer
          bhi haybytesporleer ;         De cumplirse la condición, es sinónimo de que hay bytes por
 ;                                      leer en el buffer
```

'dirinibufresci' es una variable donde se guarda la dirección de inicio del buffer. Si ambos apuntadores son iguales y a su vez, ambos son diferentes de 'dirinibufresci' (ya no hay comandos que leer del buffer), se deben reiniciar ambos a la dirección 'dirinibufresci'(la dirección de memoria inicial en el buffer).

```
apunsiguales:   cphx #dirinibufresci ;  Revisa si han llegado comandos, de no ser asi
          beq esperarxsci       ;       regresa a esperar comando de la PC

          ldhx #dirinibufresci ;
          sthx aprecepsci ;             Reiniciando apuntador
          sthx apbyporleer ;            Reiniciando apuntador
          bra esperarxsci ;

haybytesporleer: ldhx apbyporleer       cargando el comando del buffer
          lda ,x ;                      comando→a
          sta tesby1com   ;             comando → tesbycom (se guarda el comando)
          jsr ret1050us
          bsr procesacomando    ;       comando →se utiliza en “procesacomando”
;.....  Actualiza apuntador de bytes comando recibidos
          ldhx apbyporleer    ;
          aix #$01
          sthx apbyporleer;             apbyporleer  apunta a la siguiente dirección
;.....................................................

          bra esperarxsci ;             Regresar a esperar comando de la PC
```

NOTA: La recepción de comandos se realiza mediante una rutina de interrupción asociada con el evento de recepción de un byte por parte del puerto serie del SCI. Ésta se mostrará más adelante en la sección 4.4.13.

4.3 Subrutina para la interpretación de comandos

Como se menciona con anterioridad, cada comando es un byte en el que cuatro bits son utilizados para caracterizar la acción a ejecutar, mientras que los otros cuatro bits se utilizan para declarar en que puerto se ejecutara dicha acción (ver la figura 4.7). El número de puerto se asignó como lo muestra la tabla 4.2

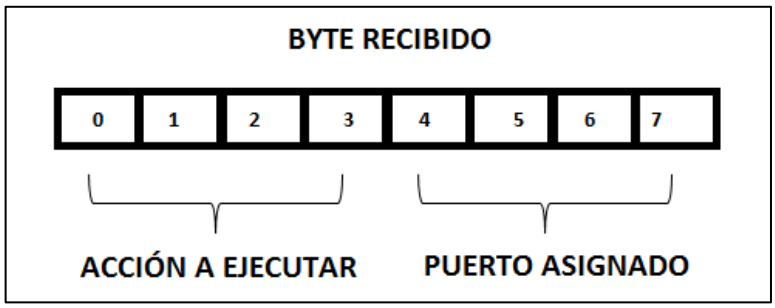


FIGURA 4.7:*Formato del comando recibido desde la PC.*

Las acciones posibles para la IADPC se enlistaron y enumeraron utilizando los cuatro bits. En la tabla 4.1 es posible observar el código de 4 bits, asignado a cada acción.

TABLA 4.1: *Muestra la relación entre las acciones posibles para la IADPC y el número binario asignado.*

Tipo de acción	Código en binario de la acción a ejecutar
Lectura analógica (8 bits)	0000
Lectura digital (8 bits)	0001
N/A	0010
N/A	0011
N/A	0100
N/A	0101
N/A	0110
N/A	0111
Escritura analógica (8 bits)	1000
Escritura digital (8 bits)	1001
N/A	1010
Forzar un nivel alto en un bit	1011
Forzar un nivel bajo en un bit	1100
N/A	1101
N/A	1110
Salto a monitor	1111

Tabla 4.2: *Relación entre los puertos de la IADPC y su número binario asignado*

Número de puerto	Denominación	Número binario asignado
Puerto 0	Entrada digital	0000
Puerto 1	Entrada digital	0001
Puerto 2	Salida Analógica	0010
Puerto 3	Salida Analógica	0011
Puerto 4	Salida digital	0100
Puerto 5	Salida digital	0101

4.3.1 Identificación de la acción a realizar

La interpretación del comando se realiza en primer lugar analizando la acción a ejecutar, para después, durante la subrutina que corresponda a dicha acción, utilizar la variable que contiene el número de puerto recibido en el comando. A continuación se muestra el segmento de código en el que se encarga de identificar al tipo de acción a ejecutar. Estas acciones las realiza la rutina denominada “procesacomando”, cuyo código se muestra a continuación:

```
procesacomando:  pshh  ;      a, hx→ Stock
                 pshx
                 psha

; Identifica tipo de acción a efectuar

        lda tesby1com ;   Se invoca el comando recibido
        and #$f0 ;        Enmascara los 4 bits identificadores de acción (ver figura 4.5)
```

Las siguientes líneas se diseñaron para identificar cual es el tipo de acción a realizar y ejecutar el código adecuado.

```
        cmp #$00  ;
        beq fuelecanal8 ;        Lectura analógica de 8 bits

        cmp #$10
        beq fuelecdig8 ;         Lectura digital de 8 bits

        cmp #$80
        beq fueescanal8  ;       Escritura analógica de 8 bits

        cmp #$90
        beq fueescdig8  ;        Escritura digital de ocho bits

        cmp #$B0
        beq fuebset;             Forzar un nivel alto en un bit


        cmp #$C0
        beq fuebclr  ;           Forzar un nivel bajo en un bit

        cmp #bycomsaltoamon
        beq fuesaltoamon ;       Salto a monitor (reset)

        bra error ;   En caso de no identificar el tipo de acción, se activa el estado de error
```

Direccionamientos a los segmentos de código asignados a cada acción:

```
fuelecanal8:   bsr rutlecanal8
```

```
            bra salprocom
fuelecdig8:     bsr rutlecdig8
            bra salprocom
fueescanal8:    bsr rutescanal8
            bra salprocom
fueescdig8:     bsr rutescdig8
            bra salprocom
fuebset:        bsr rutbset
            bra salprocom
fuebclr:        jsr rutbclr
            bra salprocom

fuesaltoamon:   jsr saltoamon
            bra salprocom


salprocom:      pula    ;   Stock → hx , a
                pulx
                pulh
            rts         ;   Regresa al Lazo principal.
```

Como se mencionó anteriormente, el análisis del número de puerto en que la acción se ejecuta, se realiza en las rutinas y subrutinas de control, descritas a continuación.

4.4 Rutinas y subrutinas de control

Las rutinas y subrutinas de control son aquellas que realizan las acciones finales sobre los puertos de entrada y salida de la IADPC. Cada rutina y subrutina de control es especifica para cada acción solicitada por la PC y son invocadas por la subrutina para la interpretación de comandos. Por otra parte también se encuentran las subrutinas que son auxiliares en los procesos de la IADPC.A continuación se describen cada una de ellas.

4.4.1 Subrutina rutlecanal8

Esta subrutina se diseñó para realizar la lectura de uno de los canales del CAD embebido en el MCU y enviarlo a la PC. El bloque de código de esta subrutina se describe a continuación:

```
rutlecanal8:    lda tesby1com ;             Invoca comando
            and #$0f ;a← número de puerto digital a leer
            sta ncanal ; ncanal ←número de puerto digital a leer
            lda ncanal
            sta adscr  ;            Inicia conversión
checoco:        brclr 7,adscr,checoco ;         Espera a que finalice la conversión
            mov adr,datent  ;
txsci1analog:   brclr 6,scs1,txsci1analog   ; Espera a que el puerto este disponible
```

```
        mov datent,scdr          Envía el resultado de la conversión
        jsr ret1050us            Retraso
        rts
```

4.4.2 Subrutina rutlecdig8

Esta subrutina tiene como propósito leer un byte en un puerto de entrada digital, para ello utiliza las subrutinas “leeport” y “ret1050us”.

```
rutlecdig8:   lda tesby1com
        and #$0f ;                    a← Número de puerto digital a leer
        sta numport ;                 Numport←Número de puerto digital a leer


;..... Actualiza apuntador de comandos a procesar
        ldhx apbyporleer
        aix #$01
        sthx apbyporleer
;.....................................................

        jsr leeport              Salto a rutina de lectura
        jsr ret1050us            Salto a rutina de retraso
        rts
```

4.4.3 Subrutina rutescanal8

Esta subrutina se diseñó para enviar un número binario a un puerto digital de salida, acoplado a un CDA. Para lograr este propósito utiliza las subrutinas “escport” y “ret1050us”

```
rutescanal8:   lda tesby1com ;        Se invoca al comando
        and #$0f ;                    a← Número de puerto digital a escribir
        sta numport ;                 Numport← número de puerto digital a escribir

;..... Actualiza apuntador de comandos a procesar ....
        ldhx apbyporleer
        aix #$01                      incrementar en uno a apbyporleer
        sthx apbyporleer
;.....................................................

        lda ,x ;                      a<-- Byte dato a escribir en puerto
        sta datsal
        jsr escport ;   Salto a rutina de escritura
        jsr ret1050us       ;  Salto a rutina de retraso
        rts
```

4.4.4 Subrutina rutescdig8

Esta subrutina esta diseñada para enviar a un puerto digital un número de ocho bits a un puerto digital acoplado a relevadores. Esta subrutina se auxilia de las subrutinas “escport” y “ret1050us”.

```
rutescanal8:   lda tesby1com ;        Se invoca a la variable que contiene al comando y
                                      el dato a escribir
               and #$0f ;             a← Número de puerto digital a escribir
               sta numport ;          Numport← número de puerto digital a escribir

;..... Actualiza apuntador de comandos a procesar

               ldhx apbyporleer
               aix #$01               incrementar en uno a apbyporleer
               sthx apbyporleer
;.....

               lda ,x ;               a<-- Byte dato a escribir en puerto
               sta datsal
               jsr escport ;  Salto a rutina de escritura
               jsr ret1050us  ;  Salto a rutina de retraso
               rts
```

4.4.5 Subrutina rutbset

Esta subrutina se diseñó para forzar a un nivel alto en uno o más bits, en un puerto digital, sin afectar a los demás bits del puerto. Para lograr su propósito se auxilia de las subrutinas “mibset” y “ret1050us”.

```
rutbset:    lda tesby1com  ;       Invocación del comando
            and #$0f  ;            a←  Número de puerto digital a escribir
            sta numport  ;         numport← Nnúmero de puerto digital a escribir

;..... Actualiza apuntador de comandos a procesar ....
            ldhx apbyporleer
            aix #$01               incrementar en uno a apbyporleer
            sthx apbyporleer
            lda ,x ;               a← bit dato a escribir en puerto
            sta datsal ;           Dato a escribir→ datsal

            jsr mibset ;
            jsr ret1050us ;
            rts
```

4.4.5 Subrutina rutbclr

Esta subrutina se diseñó para forzar a un nivel bajo en uno o más bits, en un puerto digital, sin afectar a los demás bits del puerto. Para lograr su propósito se auxilia de las subrutinas “mibclr” y “ret1050us”.

```
rutbclr:    lda tesby1com ;        Invocación del comando
            and #$0f ;             a<-- número de puerto digital a modificar
            sta numport ;          numport<-- número de puerto digital a modificar

;..... Actualiza apuntador de comandos a procesar ....
            ldhx apbyporleer
            aix #$01               incrementar en uno a apbyporleer
            sthx apbyporleer
            lda ,x ;               a<--Bit dato a borrar en puerto
            sta datsal ;           datsal ← Bit dato a borrar en puerto

            jsr mibclr
            jsr ret1050us
            rts
```

4.4.6 Subrutina saltoamon

Subrutina diseñada para reiniciar la ejecución del monitor de base del MCU

```
saltoamon:  bclr 5,scc2
            sei
            pula
            pula
            ldhx #dirmon
            pshx
            pshh
            rts
```

4.4.7 Subrutina escport

La subrutina “escport” escribe un byte dato en puerto de salida de la Interfaz analógica y digital para PC (IADPC). Antes de invocar se debe ingresar en las variables:
(datsal) ← btye a escribir
(numport) ← Número de puerto de salida en (IADPC)
Al retornar ya se habrá escrito el dato en el puerto implicado.

```
escport:    pshh
            pshx
            psha
            mov #$ff,ddra ;        pta modifica su operación para funcionar como salida
            lda numport
            add #numport ;         Dir de tesssalx en a.
```

```
psha
pulx ;                  a→x
clrh ;                  h:x←dir en página cero de tessalx.
mov datsal,x+     ;     datsal→tessalx.
mov datsal,pta ;        escribe dato al bus externo
mov numport,ptd ;       Selecciona puerto de salida externo
nop
mov #numfant,ptd ;      Se genera flanco de subida

mov #$00,ddra ;         pta regresa a su estado de entrada
pula
pulx
pulh
rts
```

4.4.8 Subrutina Leeport

La subrutina leeport esta diseñada para enviar información a la PC sobre el estado de un puerto digital de entrada. Antes de invocar es necesario ingresar en la variable “numport”, el número del puerto que se desea leer. Al retornar, la subrutina ya habrá ingresado el bite leído en la variable “datent”. El código descrito se muestra a continuación:

```
leeport:  mov numport,ptd ;     Selección del puerto a leer
          mov pta,datent ;      (datent)← byte leído
txsci1:   brclr 6,scs1,txsci1 ;  Enviando bite leído
          mov datent,scdr

          mov #numfant,ptd ;    Regresa al puerto fantasma
          rts
```

4.4.9 Subrutinas mibset y mibclr

Las rutinas “mibset” y ”mibset” son una modificación de la rutina “escport” descrita en el apartado 4.4.7. El objetivo de la rutina “mibset” es colocar un nivel alto en uno o varios bits seleccionados de un puerto digital de salida, sin modificar el estado los demás bits. A continuación se muestra el código de la subrutina “mibitset”:

```
mibset:   pshh
          pshx
          psha
          mov #$ff,ddra ;       pta es salida
          lda numport
          add #numport ;        Dir de tesssalx en a.
          psha
          pulx ;                a-->x
          clrh ;                h:x<--dir en página cero de tessalx.
          lda datsal
          ora ,x
          sta datsal
```

```
        mov datsal,x+   ;          datsal-->tessalx.
        mov datsal,pta ;           escribe dato al bus externo
        mov numport,ptd ;          Selecciona puerto de salida externo
        nop
        mov #numfant,ptd ;         Se genera flanco de subida en línea
;                                  de selección del puerto.

        mov #$00,ddra ;            pta es entrada
        pula
        pulx
        pulh
        rts
```

La subrutina “mibclr” tiene como objetivo colocar un nivel bajo en uno o varios bits seleccionados de un puerto digital de salida, sin modificar el estado los demás bits. A continuación se muestra el código de la subrutina “mibclr”:

```
mibclr:   pshh
        pshx
        psha
        mov #$ff,ddra ;        pta es salida
        lda numport
        add #numport ;         Dir de tesssalx en a.
        psha
        pulx ;                 a →x
        clrh ;                 h:x← dir en página cero de tessalx.
        lda datsal
        coma
        and ,x
        sta datsal
        mov datsal,x+                datsal→tessalx.
        mov datsal,pta ;             Escribe dato al bus externo
        mov numport,ptd ;            Selecciona puerto de salida externo
        nop
        mov #numfant,ptd ;           Se genera flanco de subida en línea
;
        mov #$00,ddra ;              pta se configura como entrada
        pula
        pulx
        pulh
        rts
```

4.4.10 Subrutina ini0portsal

Justo después de energizar a la IADPC sus puertos de salida se encuentran en un estado aleatorio. Ésta subrutina tiene como objetivo inicializar todos los puertos de salida, tanto digitales como analógicos, a un nivel bajo. Este proceso de inicialización se presenta únicamente durante el inicio de todo el sistema de la IADPC. La subrutina “escport” se utiliza para tener acceso a los cuatro puertos de salida. A continuación se muestra el código de la rutina mencionada:

```
ini0portsal: mov #$00,datsal
        mov #$02,numport
        jsr escport
        mov #$03,numport
        jsr escport
        mov #$04,numport
        jsr escport
        mov #$05,numport
        jsr escport
        rts
```

4.4.10 Subrutina ret250ms

Esta subrutina tiene por objetivo realizar un retraso por un periodo de 250 milisegundos.

```
ret250ms:   pshh
        pshx
        ldhx #$c34e
vuelta:     nop
        nop
        aix  #$ff
        cphx #$0000
        bne vuelta
        pulx
        pulh
        rts
```

4.4.11 Subrutina ret1050us

Esta subrutina tiene por objetivo realizar un retraso por un periodo de 1050 microsegundos

```
ret1050us:  pshh
        pshx
        ldhx #$0832
vv:     nop
        nop
        aix #$ff
        cphx #$0000
        bne vv
```

```
        pulx
        pulh
        rts
```

4.4.12 Subrutina error

Esta subrutina se activa en caso de que La IADPC presente un error por saturación del Buffer de recepción. También puede activarse al recibir un comando que no este registrado en la tabla 4.1. La función de esta subrutina es notificar mediante el parpadeo de un LED testigo, denominado TESSAT, que alguno de estos errores se ha presentado. A continuación el código de la presente subrutina:

```
error:  sei ;              Deshabilita las interrupciones
errc:   mov #$16,ptc ;     El LED va a parpadear hasta que se reinicie el MCU manualmente
        jsr ret250ms
        jsr ret1050us
        mov #$00,ptc
        jsr ret1050us

        mov #$16,ptc
        jsr ret250ms
        jsr ret1050us
        mov #$00,ptc
        jsr ret1050us

        mov #$16,ptc
        jsr ret250ms
        jsr ret250ms
        mov #$00,ptc
        jsr ret250ms
        jsr ret250ms

        bra errc
```

4.4.13 Rutina servscirec

El servicio para la interrupción por el evento de recepción del puerto serie, se presenta cada vez que PC envía un comando, sin importar que parte del código se encuentre ejecutando el MCU. Durante la ejecución de la rutina de servicio se carga el comando recibido en la dirección del buffer especificada por el apuntador “aprecepsci”. A continuación se muestra el código de ésta rutina de servicio de interrupción:

```
servscirec:  pshh
             lda scs1
             lda scdr ;        Se carga en al comando recibido
             ldhx aprecepsci
             sta ,x;           El comando en A es almacenado en el buffer
             aix #$01
```

```
sthx aprecepsci      ;  Se incrementa en uno el valor del apuntador aprecepcsi
pulh
rti
org $d7e4
dw servscirec
```

Como podrá verse en los capítulos anteriores, las rutinas de control se basaron en las pruebas de software, realizadas para observar la compatibilidad del hardware periférico de la IADPC. El previo conocimiento de los puertos periféricos de la IADPC y del MCU y sus subsistemas, contribuyeron al desarrollo del software de base.

Con este capítulo, se termina por completo la descripción del sistema de la IADPC. Hasta este punto, la IADPC puede ser controlada desde la PC (mediante los comandos establecidos en la sección 4.3). Las acciones, tanto para la adquisición de datos, como para control, se hacen mediante los módulos de hardware descritos anteriormente.

CAPÍTULO 5

Desarrollo de módulos de software ejecutables en la PC.

En el capítulo anterior se revisó que utilizando los comandos establecidos en la tabla 4.1, se puede modificar el estado de los puertos de salida, o bien adquirir los datos que se encuentran en los puertos de entrada. Gracias a esto es posible un sistema de control o automatización, en el que la IADPC se comporte como un elemento seguidor, mientras que la PC como un elemento maestro.

Una de las características de la IADPC es que puede controlarse desde una PC y una aplicación programada en cualquier lenguaje programación que pueda enviar mediante el puerto serie a los comandos antes mencionados. Esta característica le proporciona al usuario la capacidad de poder desarrollar software de control en la plataforma que considere más adecuada para la aplicación.

Desde este enfoque, el usuario tiene más herramientas de desarrollo de software pues no esta limitado a lógica booleana, como en el caso de los lenguajes de escalera de los PLC. Además adquiere la capacidad de desarrollar un sistema de control capaz de incluir algoritmos complejos, una Interfaz Hombre-Máquina, supervisión remota, etc.

En el presente capítulo se abordará el análisis del panel de prueba del a IADPC, desarrollado en Visual Basic. El objetivo de esta aplicación es evidenciar, de una forma rápida y fácil, el correcto funcionamiento de la comunicación serie y de los puertos de entrada y salida.

Además de los comandos que se explicaron en el capítulo anterior, el panel de prueba de la IADPC, hace uso de algunos de los comandos del “Protocolo PUMMA” que proporcionan el control sobre el software monitor del MCU de la IADPC. Gracias a esto último es posible ejecutar un programa almacenado en la memoria FLASH del MCU sin utilizar del software manejador PUMMA_08+.

Las funciones que se revisan a continuación, se presentan como una guía adecuada para desarrollar la programación de cualquier proyecto, que pretenda el uso de la IADPC como parte de un sistema de control, adquisición de datos o automatización.

5.1 Panel de prueba de la IADPC

Para fines de verificar la funcionalidad de los comandos básicos de la IADPC mostrados en la tabla 4.1, se desarrolló el panel de prueba que se aprecia en la figura 5.1. Mediante éste se puede tener acceso y control sobre todos los puertos de la IADPC, e incluso a algunas de las funciones del monitor del MCU. El panel de prueba se diseñó empleando Visual Basic y tiene por objetivo presentar de una manera ordenada el estado de los puertos de entrada la IADPC y controlar los puertos de salidas.

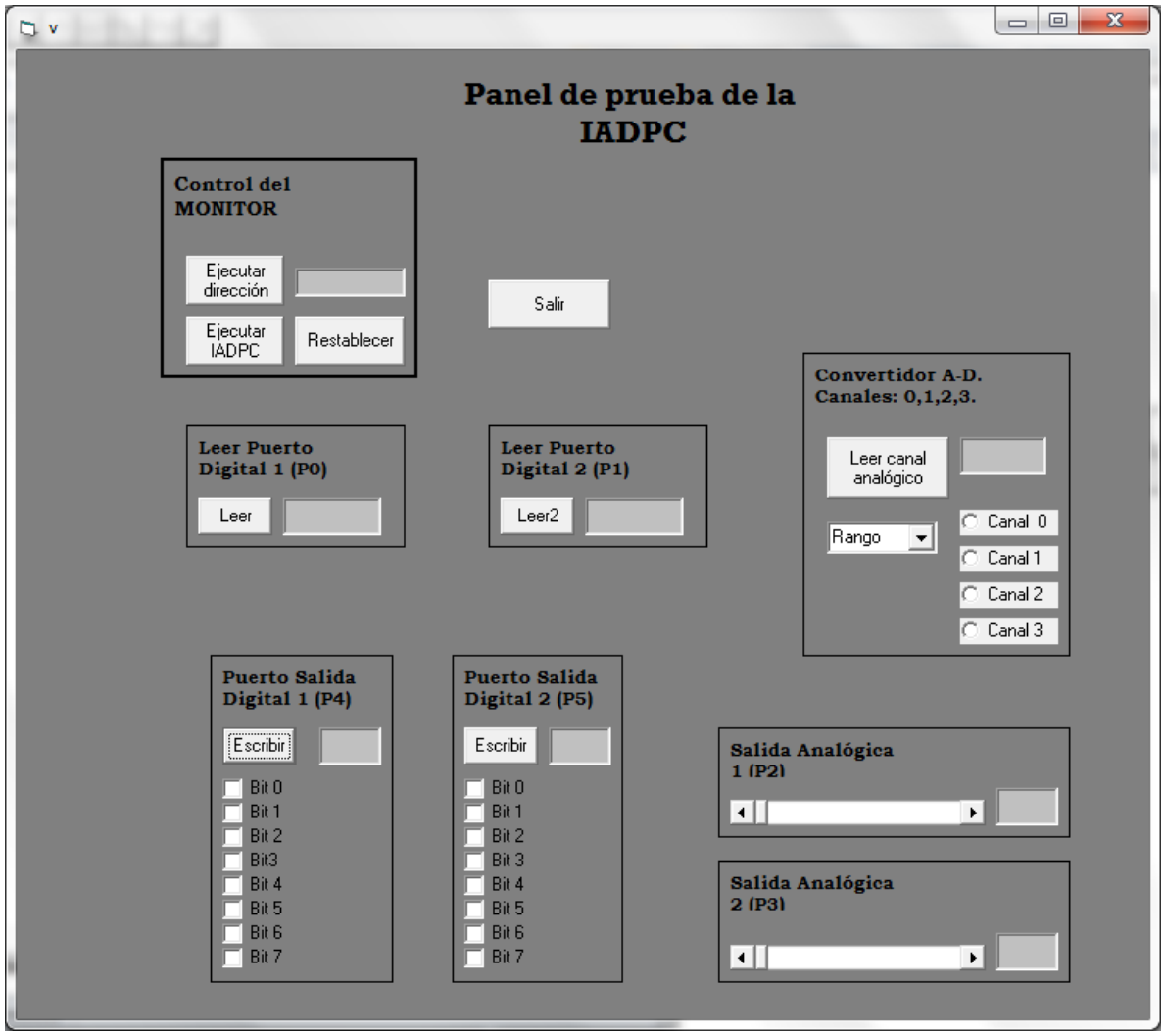


Figura 5.1: *Panel de prueba de la IADPC, en él se concentran los controles de los puertos de entrada y salida*

El ambiente de programación de visual Basic provee de herramientas que facilitan la creación del panel de prueba, tales como controles intuitivos y ventanas de datos, adecuadas para presentar resultados congruentes al tipo de variable que monitorean o controlan. Entre estos controles destacan, por su utilidad en este proyecto:

- Botones de control (control button).
- Controles de selección (Check box).
- Barras de desplazamiento (Scroll bar).
- Cuadros de texto (Text box).
- Botones de opción (Option button).
- Menús (Combo box).
- Control del puerto serie(MS Comm).

La principal función del panel de prueba es proveer de una herramienta al usuario, para que antes de iniciar el proyecto de control se pueda comprobar el funcionamiento de la IADPC, y su relación con los elementos finales de control que se encuentran conectados a sus puertos. También sirve como referencia al momento de calibrar un sensor, pues se puede consultar el estado de los canales analógicos desde ésta aplicación.

A continuación se revisaran las funciones programadas en la PC para el manejo de comandos. Estas funciones están diseñadas para realizar las operaciones de recepción de datos y envío de comandos.

5.2 Las funciones generales de transmisión y recepción.

5.2.1 El control del puerto serie

El control del puerto serie es una herramienta que permite la transmisión y recepción de un byte por medio de un puerto serie. Para transmitir es necesario asignar un número entero decimal, en su propiedad de salida (output). El control del puerto realiza un cambio de base para transformar dicho numero en binario y enviarlo por el puerto serie. Al recibir un número binario desde la IADPC, por el puerto serie, el control del puerto serie se encarga de transformarlo en un número entero decimal y se almacena en la propiedad de entrada (input).

El número entero recibido es de ocho bits, por lo que el resultado de la transformación siempre se encuentra entre 0 y 255. De la misma forma cuando se pretende enviar un Byte a la IADPC, es necesario ingresar al control del puerto serie, un número entero entre 0 y 255.

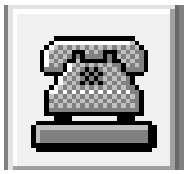


Figura 5.2: *Icono del control del puerto serie. Este icono desaparece mientras la aplicación se ejecuta, por esta razón no es posible verlo en la figura 5.1*

5.2.2 Procedimiento general de transmisión “txps(x,y)”

Este procedimiento se diseñó para transmitir a la IADPC un byte comando y si es el caso el byte dato implicado. Antes de invocar este procedimiento, en la variable “y” se debe precargar el byte comando asociado en un momento dado; en la variable “x” se debe precargar el byte dato que se desea sea colocado en un puerto de salida binario; o bien, en un puerto asociado con uno de los convertidores analógico-digital de la IADPC. Por ejemplo, supóngase que deseamos escribir al puerto de salida digital número cuatro el byte $AB. Es claro que esto requerirá la línea de código en Visual Basic mostrada a continuación:

txps 171,148

Para entender como se obtuvo el valor del byte comando pueden verse las tablas 4.1, 4.2, así como la figura 4.7.

A continuación se analizarán las líneas de código del procedimiento “txps(x, y)”

Public Sub txps(x, y)

El procedimiento se utiliza de la siguiente manera: **txps (Dato a enviar, Comando)**

Se envía el primer Byte, es decir, el comando:

MSComm1.Output = Chr(y)

Se envía el segundo byte, es decir, el dato:

MSComm1.Output = Chr(x)
End If

End Sub

Para los comandos que implican lectura de un dato presente en la IADPC, se utiliza al procedimiento

“txp1(y)” revisado a continuación.

5.2.2 Procedimiento de transmisión de comandos “txp1(y)”

Este procedimiento tiene como objetivo el envío de bytes comando. A diferencia del procedimiento general de transmisión, éste no envía bytes dato. Este procedimiento auxilia operaciones de lectura y de control del monitor del MCU. A continuación se analizará las líneas de código de esta función.

Public Sub txp1(y)
MSComm1.Output = Chr(y) Transmite el byte comando en “y”
End Sub

5.2.3 La función general de recepción “rxps(y)”.

Esta función sirve para enviar un comando de lectura, ya sea de uno de los puertos digitales de entrada, o bien del convertidor analógico-digital. La función retorna el valor en decimal del valor leído. Si por alguna causa falla la comunicación entre la IADPC y la PC, el Panel de prueba de la IADPC despliega la caja de mensaje mostrada en la figura 5.2.

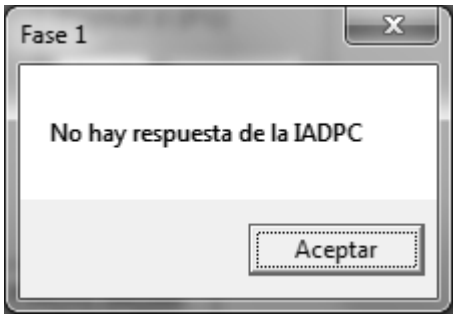


Figura 5.2: *Caja de mensaje que se despliega al ocurrir un evento de error en la comunicación entre la IADPC y la PC.*

A continuación se analizarán las líneas de código de la función:

Public Function rxps (y) As integer

Primero se introduce el número de caracteres que se espera recibir, después el comando de lectura.

Se realiza la transmisión del comando de lectura. Para ello se utiliza la función **"txp1 (y)"**

txp1 (y)

Aquí finaliza la transmisión del comando y se procede a esperar por el dato que la IADPC envía. Se declara la variable del contador.
Dim ii As Integer

Ciclo que espera la recepción de un solo Byte, si la cuanta llega a 30000, se determina que se ha perdido el enlace con la IADPC.

```
MSComm1.InputLen = 1
Do
ii = ii + 1
If ii = 30000 Then
GoTo finn
End If
Loop Until MSComm1.InBufferCount >= 1
```

rxps = Asc(MSComm1.Input) La función devuelve un número entero entre 0 y 255.
MSComm1.InBufferCount = 0
GoTo finfel

El final que a continuación se presenta, se da en caso de que se termine el tiempo de espera y la IADPC no presente todos los bits esperados:

finn:

Mensaje de error:

MsgBox " No hay respuesta de la IADPC"
MSComm1.InBufferCount = 0

En caso de que todo se lleve a cabo con normalidad la función termina.

finfel:
End Function

5.3 Controles asignados al monitor del MCU

Estos controles hacen uso de los comandos del protocolo PUMMA, para enviarle al monitor del MCU la orden de ejecutar un programa almacenado en memoria FLASH. Para lograr esto último es necesario ingresar en el cuadro de texto la dirección de memoria donde el programa inicia. Ver la figura 5.3.

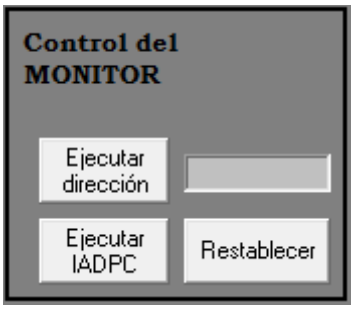


Figura 5.3: *Controles asignados al Monitor del MCU.*

Este mismo proceso se utiliza el botón de control “Ejecuta Interfaz” el cual le da la orden al monitor del MCU de ejecutar la dirección de memoria donde se encuentra el software de base de la IADPC, visto en el capítulo 3. En el caso del botón restablecer, se envía el comando asignado para ejecutar la rutina de reinicio (Rutina de nombre “saltoamon” vista en la sección 4.4.6) utilizando la función general de transmisión visto en la sección 5.2.1.

5.3.1 Botón “Ejecutar Dirección”

El objetivo de este botón es indicar a la IADPC la ejecución del programa almacenado en la localidad de memoria, que el usuario coloca en el cuadro de texto que tiene a su derecha (ver la figura 5.3). A continuación se muestra el código del procedimiento asociado a esto control.

Private Sub Command1_Click()

Dim By1 As Long
Dim BY2 As Long

Transformación del valor ASCCI en un número entero.

A = Val(DirecEjec.Text)

By1 = A / 256
BY2 = A - (By1 * 256)

Envío del protocolo PUMMA para la ejecución del programa

txp1 (14)
txp1 (0)
txp1 (192)
txp1 (0)
txp1 (194)
txp1 (204)

A continuación se envía la dirección a ejecutar.
txp1 (By1)
txp1 (BY2)

End Sub

5.3.2 Botón Ejecutar IADPC

Este procedimiento indica al MCU de la IADPC ejecutar desde la dirección donde inicia el software de base de la IADPC. Como a continuación se podrá ver a continuación, este botón utiliza el mismo proceso del botón visto en la sección 5.3.1, a diferencia es que se ha precargado la dirección del software de base de la IADPC.

Private Sub IADPC_Click()

Dim By1 As Long
Dim BY2 As Long

Dirección donde inicia el Software de base de la IADPC en “A”

A = 32768

By1 = A / 256
BY2 = A - (By1 * 256)

Se envían los comandos del protocolo PUMMA

txp1 (14)
txp1 (0)
txp1 (192)

txp1 (0)
txp1 (194)
txp1 (204)

Envío de la dirección de la interfaz:

txp1 (By1)
txp1 (BY2)

End Sub

5.3.3 Botón restablecer

Este botón envía el comando para restablecer el estado del monitor del MCU. A continuación se muestra el código de esta función.

Private Sub Reset_Click()

Uso de la función general de transmisión para enviar el comando de "restablecer"

txp1 (255)
End Sub

5.4 Controles asignados a Entradas y salidas digitales

Los controles de las salidas digitales se basan en las funciones generales de transmisión y recepción, su función es leer y modificar los puertos digitales de la interfaz mediante el envío de los comandos de control establecidos en la tabla 4.1 del capítulo anterior. La figura 5.4 muestra a los controles asociados a entradas digitales, mientras que la figura 5.5 muestra los controles asociados a los puertos de salida digital.

Figura 5.4: *Controles asignados a los puertos de entrada digital.*

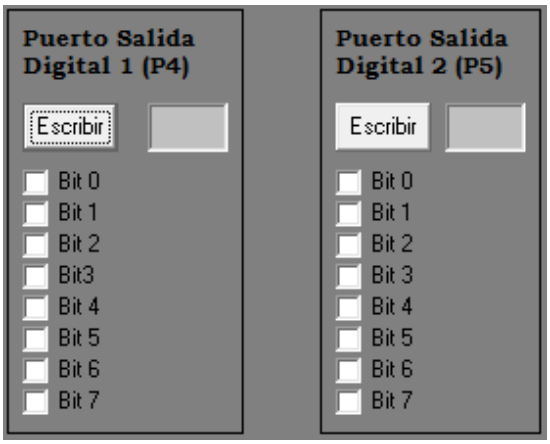


Figura 5.5: *Controles asignados a los puertos de salida digital.*

5.4.1 Botón de control para realizar una lectura de puerto

Los botones para realizar la lectura de un puerto digital utilizan la función general de recepción. A continuación se muestran las líneas de código, propias del procedimiento asociado a este botón.

Private Sub Cmd3_Click()

Envío del resultado de la lectura realizada se despliega en el cuadro de texto asociado al botón de control. Como se vio anteriormente es necesario ingresar el comando, compuesto por la dirección del puerto y la acción a ejecutar.

Lec1Txt = (rxps(16))

End Sub

5.4.2 Botón de control para realizar una escritura digital

Este botón hace uso de la función general de transmisión, para enviar un byte al puerto digital de salida que tiene asignado. A continuación se revisarán las líneas de código propias de la función asignada a este control.

Private Sub Cmd1_Click()

Se almacena en la variable “Data” al valor numérico de los caracteres introducidos en el cuadro de texto asociado.

Data = Val(EscP4Txt)

El valor es enviado, junto con el comando de escribir.

txps (Data), (148)
End Sub

5.4.3 Controles de selección de los bits propios de los puertos digitales de salida

Estos controles actúan en forma paralela a los de la sección 5.4.2, la diferencia es que modifican al puerto digital, únicamente en el bit que tienen asociado. En este caso también se hace uso de la función general de transmisión, para enviar a través del puerto serie los comandos dedicados a forzar los bits en un nivel alto o bajo. Ver la sección 4.4.9.

Cada control de selección tiene asociado un bit de uno de los puertos de salida digital. Por cada control se tiene el siguiente código:

Private Sub P4b0_Click()

Si se encuentra verificado entonces se envía el comando “Forzar en alto en un bit” junto con la dirección asociada.

If P4b0 = 1 Then
txps (1), (180)

Al modificar al control de selección, quitando la “paloma”, se envía el comando para forzar un nivel Bajo en un bit:

Else
txps (1), (196)
End If
End Sub

Esta función se ejecuta solamente cuando el control de selección sufre un cambio, por lo que la IADPC mantendrá el estado del bit mientras el usuario no modifique el estado de este control.

5.5 Controles asignados a las salidas analógicas y al Convertidor A-D

Las funciones asociadas a los controles asignados a las salidas y entradas analógicas, que se analizarán a continuación, incluyen un proceso de conversión para los datos que envían o reciben, con el objetivo de desplegar resultados útiles para el usuario. Los controles asociados a los puertos analógicos de salida aparecen en la figura 5.6 mientras que el control para el convertidor analógico–digital se muestra en la figura 5.7

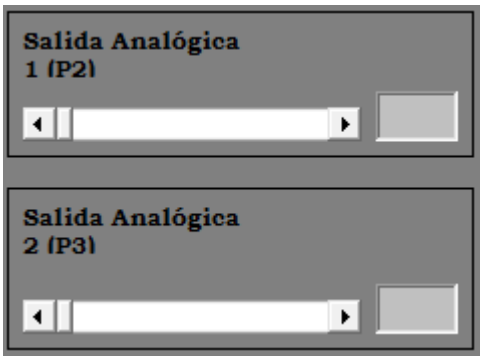


Figura 5.6: *Controles asignados a los puertos de salida analógica*

Figura 5.7: *Control asignado al convertidor analógico-digital*

5.5.1 Control asignado al convertidor analógico-digital

Antes de hacer uso de este control, es necesario que el usuario ajuste el rango de entrada del adecuador de entrada, del convertidor CAD en la IADPC o (ver 3.5.5). Para que el valor que se despliega en la interfaz de usuario sea el congruente es necesario que en el menú de opciones seleccione el rango de voltaje para el cual se ajustó la IADPC y el canal que se desea leer. De igual manera que en los casos anteriores, en este también se hace uso de la función general de recepción.

En cuanto se obtiene el número se convierte en un valor en volts, útil para el usuario, considerando siempre el rango de voltaje seleccionado. A continuación se analizarán las líneas de código de la función de recepción analógica.

Private Sub Command2_Click()

Selección del canal analógico de entrada

Dim canal As Integer
Dim ADConv As Double

If Option1 = True Then

canal = 0
End If

If Option2 = True Then
canal = 1
End If

If Option3 = True Then
canal = 2
End If

If Option4 = True Then
canal = 3
End If

Aquí se envía el comando para realizar una lectura analógica, como respuesta se obtiene un número real de 0 a 255 (como se analizó con anterioridad, el comando para solicitar a la IADPC la lectura analógica es 0X, donde X se sustituye por el número asignado al canal analógico que se desea leer):

ADConv = (rxps(canal))

Aquí se selecciona el rango de entrada para el ADC, el mismo seleccionado desde la IADPC. Para cada rango existe una conversión, necesaria para que en la interfaz de usuario se despliegue un valor congruente. Las formulas de las conversiones son las siguientes:

- Para rangos bipolares

$$VD=\left((C1)\frac{255}{R}\right)-\frac{R}{2}$$

Donde:

VD es el valor desplegado por la Interfaz de usuario
C1 es el número entero de 0 a 255 que se recibe del control del puerto serie
R es el rango total seleccionado, por ejemplo en el rango de -10 volts a 10 volts el rango total es 20 V.

- Para rangos unipolares

$$VD=\left((C1)\frac{255}{R}\right)$$

Donde
VD es el valor desplegado por la Interfaz de usuario
C1 es el número entero de 0 a 255 que se recibe del control del puerto serie
R es el rango total seleccionado, por ejemplo en el rango de 0 volts a 10 volts el rango total es 10 V.

A continuación se muestra al código correspondiente a cada uno de los casos:

Rango de -10 a 10:

If Combo1.ListIndex = 0 Then

LecAtext1 = (ADConv * (20 / 255)) - 10
Text1.Text = ADConv

End If
Rango de -5 a 5

If Combo1.ListIndex = 1 Then

LecAtext1 = (ADConv * (10 / 255)) - 5
End If
Rango de -1 a 1
If Combo1.ListIndex = 2 Then

LecAtext1 = (ADConv * (2 / 255)) - 1
End If

Rango de 0 a 10:
If Combo1.ListIndex = 3 Then

LecAtext1 = (ADConv * (10 / 255))
End If

Rango de 0 a 5:
If Combo1.ListIndex = 4 Then

LecAtext1 = (ADConv * (5 / 255))
End If
Rango de 0 a 1
If Combo1.ListIndex = 5 Then

LecAtext1 = (ADConv * (1 / 255))
End If

End Sub

5.5.2 Controles asignados a las salidas analógicas.

Los controles de las salidas analógicas hacen uso de la función general de transmisión para enviar a la IADPC el comando y valor de voltaje que se requiere en la salida.

La barra de desplazamiento que se utiliza como control, devuelve un número entero de 0 a 250 cada vez que se modifica su estado. Cuando se encuentra en el extremo izquierdo devuelve el número 0, y 255 cuando se encuentra en su extremo derecho. Para que la interfaz de usuario despliegue el valor real en volts, que se encuentra en el puerto de salida analógica de la IADPC, es necesario transformar el valor a un rango de voltaje entre -10 y 10 volts. A continuación se revisarán las líneas de código propias de esta función.

Private Sub SalAnalog2_Change()

Se asigna el valor de la barra de desplazamiento a la variable “outan2”

outan2 = SalAnalog2.Value

El valor es enviado a la IADPC, mediante la función general de transmisión.

txps (outan2), (131)

Se convierte el número entre 0 y 255, para desplegar un valor en voltaje en el rango entre -10 y 10 Volts.

AnaSal2txt.Text = outan2 * (20 / 255) - 10
End Sub

Con este capítulo se concluye a la IADPC como un producto terminado. En la próxima sección se desarrollará una aplicación para la IADPC, en la que se pueda ejemplificar el desarrollo de un sistema de control basado en este dispositivo.

CAPÍTULO 6

Ejemplo de aplicación de la IADPC

6.1 Introducción

En el presente capítulo se utilizan las características de la IADPC para implementar un ejemplo de control automático mediante el software y hardware vistos en capítulos anteriores, en combinación con los recursos de la PC. Para lograr este objetivo se ha planteado el problema de desarrollar un sintetizador de soluciones, como una aplicación idónea que requiere el uso de restricciones lógicas, temporizadores, lógica combinacional y una interfaz de usuario.

El sintetizador de soluciones es un dispositivo utilizado por farmacéuticas, laboratorios y procesos que impliquen la manipulación de sustancias, muchas veces peligrosas o inestables o que requieren de automatizarse. Como consecuencia de las características del proceso, la combinación de las substancias no puede realizarse directamente por el personal, ya sea por cuestiones de seguridad o por que no es rentable hacerlo.

A continuación se explicará a detalle la estrategia que se sigue para el desarrollo de la aplicación teniendo como objetivo comprobar la funcionalidad y compatibilidad de la IADPC como un instrumento de control altamente adaptable. El código fuente de daa uno de los objetos procedimiento y boques lógicos que se utilizan en la presente aplicación, pueden consultarse en el Apéndice B.

6.2 El sintetizador de soluciones

Un sintetizador de soluciones es un sistema que realiza una mezcla a partir de reactivos primarios, administrándolos en la proporción correcta para crear una solución final. Las características de la solución dependen de los reactivos que se suministren y la cantidad de los mismos.

Esta clase de dispositivos ya existe en el mercado y la razón de utilizarlos es aumentar la productividad y eficacia en sistemas de producción que involucre la síntesis de soluciones, o bien para evitar que el personal entre en contacto directo con las substancias primarias. Entre varios de sus beneficios destacan los siguientes:

- Mejorar la productividad mediante automatización.
- Reduce la probabilidad de ocurrencia de errores humanos.
- Ofrece un nivel de riesgo aceptable en el manejo de sustancias peligrosas.
- Minimiza la interacción con los reactivos.
- Proporciona repetitividad en los procesos.
- Fácil de utilizar y no requiere un entrenamiento elaborado o capacitación cuando se utiliza el método automático.

6.3 Planteamiento del problema

Se desea realizar la automatización de un sintetizador de soluciones mediante una PC y utilizando como interfaz a la IADPC. El objetivo del sintetizador de soluciones es mezclar cuatro sustancias primarias, configurando la proporción entre estas y la cantidad de la solución final.

El sistema a automatizar se constituye por cinco tanques, de los cuales cuatro son utilizados para contener sustancias primarias, y uno para realizar la mezcla entre ellas. Para fines de este ejemplo de aplicación, a los tanques que almacenan las substancias primarias se les conocerá como “tanques contenedores”, mientras que al tanque que contiene la solución final se le conocerá como “tanque solución”.

El líquido de los tanques contenedores es llevado al tanque solución por medio de bombas, una por cada tanque contenedor. El nivel de cada tanque contenedor se monitorea mediante sensores de nivel. Cada uno de los componentes del sistema se describe detalladamente más adelante.

6.3.1 Requisitos del sistema de control

Se desea automatizar el control del llenado del tanque solución mediante la programación de una PC y utilizando a la IADPC como interfaz entre la PC y las bombas y sensores de los tanques.

Esta automatización debe considerar los siguientes factores:

a) El nivel de agua en cualquiera de los tanques de suministro no puede estar por debajo de las bombas, ya que estas no deben operar sin líquido.

b) En caso de que el nivel del agua en un tanque de suministro disminuya por debajo de lo permitido el sistema debe detener el proceso. En cuanto el tanque de suministro de llenado nuevamente, el sistema debe reiniciar automáticamente el proceso.

c) Se debe impedir el desbordamiento del tanque solución resultado, por lo que en cuanto este tanque se llene se debe detener automáticamente el proceso de llenado. Más adelante en la sección 6.3.2 se detalla la configuración del proceso de llenado

d) Tanto el estado de los tanques como cualquier eventualidad relevante en el proceso de llenado debe informarse en la interfaz de usuario.

e) Adicionar en la interfaz de usuario un botón paro de emergencia de todo el sistema.

6.3.2 Controles requeridos

Se requiere que la interfaz de usuario incluya los siguientes controles:

a) Cantidad de la solución final.

Para el proceso de llenado se requiere un control para que el usuario pueda configurar la cantidad total de la solución final, limitándose únicamente por la capacidad del tanque solución.

b) Proporciones de las sustancias primarias.

Por otra parte, el usuario debe ingresar la proporción en porcentaje, entre cada sustancia primaria y la solución final, por ejemplo, si para una sustancia final no se requiere el uso de alguna sustancia primaria, se ingresa la proporción de ésta como 0 %.

c) Botón de paro.

Es necesario el paro de emergencia del sistema por parte del usuario, por lo que se incluye un botón de paro, que además de interrumpir el proceso, regresa a las condiciones y variables del proceso a su estado inicial.

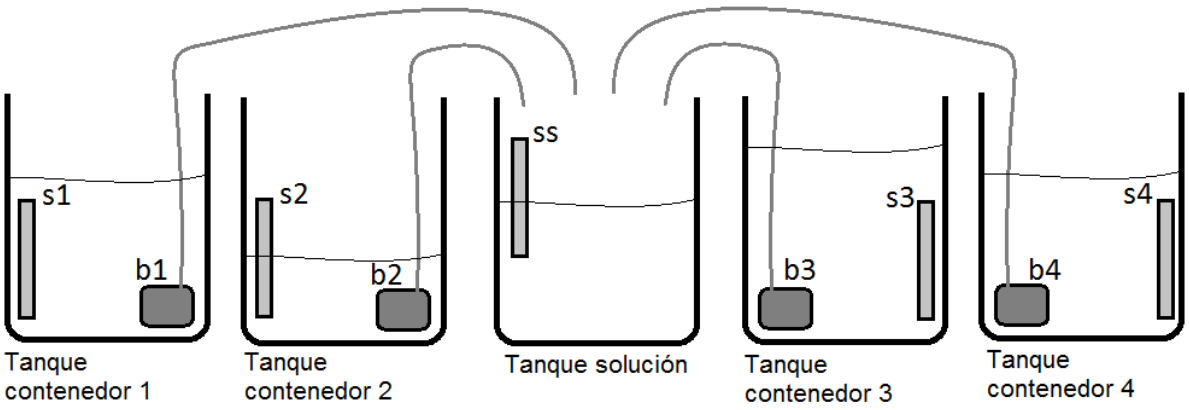
d) Guardar o cargar un “Perfil de solución”.

Se le llama Perfil de solución al conjunto de parámetros que definen las características de una solución, es decir la cantidad de solución final y la proporción de las sustancias primarias utilizadas.

Se solicita un control para poder guardar opcionalmente los parámetros de un perfil de solución, en un archivo de texto. Por otra parte es necesario un control para cargar este perfil de carga en el sintetizador de soluciones.

6.4 Hardware disponible del sintetizador de soluciones

En este apartado se describen cada uno de los dispositivos del hardware que se encuentran disponibles para realizar la automatización. La figura 6.1 representa al hardware disponible para el sintetizador de soluciones.



b1	Bomba 1, asociada con el tanque 1
s1	Sensor de nivel 1, asociado con el tanque 1
b2	Bomba 2, asociada con el tanque 2
s2	Sensor de nivel 2, asociado con el tanque 2
b3	Bomba 3, asociada con el tanque 3
s3	Sensor de nivel 3, asociado con el tanque 3
b4	Bomba 4, asociada con el tanque 4
s4	Sensor de nivel 4, asociado con el tanque 4
ss	Sensor de nivel del tanque solución.

Figura 6.1: *Diagrama del hardware disponible para el sintetizador de soluciones.*

6.4.1 Sensores de nivel

Los sensores se nivel están inmersos en el líquido del tanque contenedor y detectan si el nivel del líquido es adecuado o no, para el funcionamiento de la bomba.

Los sensores están constituidos por un “reed switch” y un imán adherido a un flotador. Dichos elementos están contenidos en un tubo que permite la flotación del imán hacia arriba y hacia abajo según el nivel del líquido, véase la figura 6.3.

Un reed switch es un dispositivo que establece continuidad eléctrica entre sus terminales, cuando se encuentra en presencia de un campo magnético y la pierde cuando el campo magnético no es suficientemente denso, véase la figura 6.2.

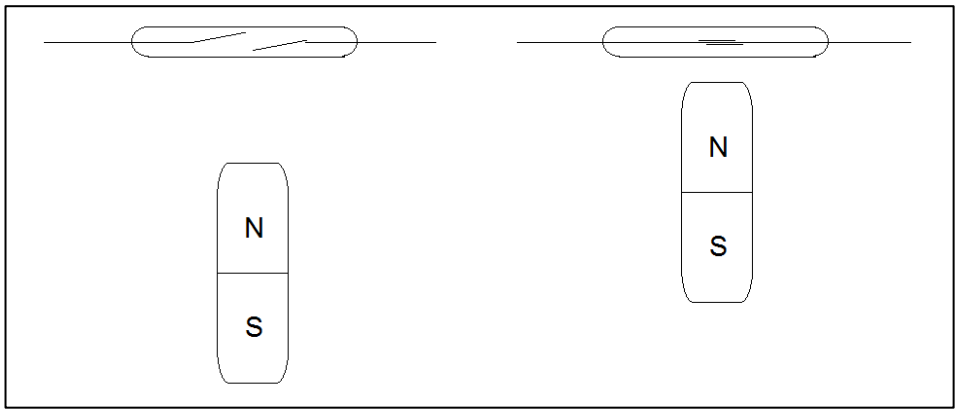
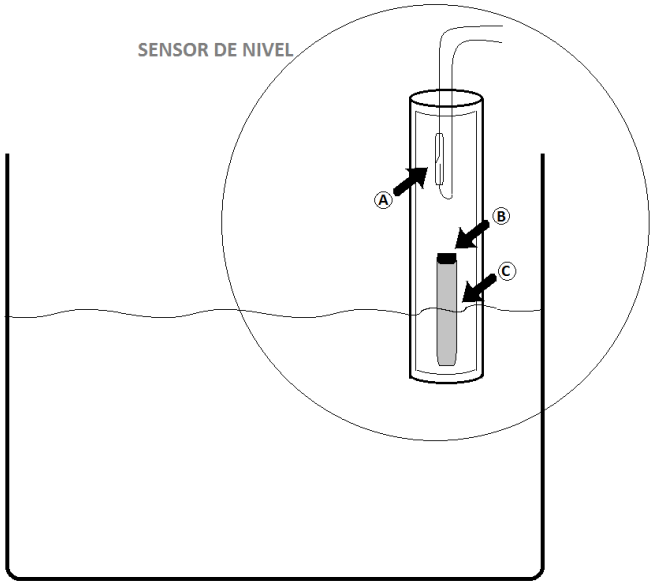


Figura 6.2: *Esquema que ejemplifica la funcionalidad de un reed switch*



A	es el Reed switch
B	es el Imán
C	es el flotador

Figura 6.3: Sensor de nivel del tanque solución.

Cuando el nivel del líquido esta por debajo de un nivel aceptable, el imán adherido al flotador se encuentra suficientemente lejos como para que el reed switch pierda continuidad, mientras que si se tiene un nivel aceptable de líquido, el imán adherido al flotador se encuentra suficientemente cerca del reed switch como para permitirle restablecer la continuidad. Este comportamiento se utiliza para posicionar al sensor de nivel a la altura requerida del líquido en un tanque.

6.4.2 Tanques contenedores

En cada tanque contenedor se almacena una substancia primaria diferente. A modo de ejemplo y para que la combinación entre substancias primarias resulte evidente, estas son representadas por cuatro colores primarios sustractivos.

La combinación de las substancias primarias (colores primarios sustractivos) se realiza en el tanque solución y dependiendo de cuales se utilicen y la proporción entre ellas se tiene por resultado la síntesis de colores.

6.4.2.1 Sustancias primarias

Las sustancias primarias se representan por colores, esto se hace con el objetivo de esquematizar el desempeño del sintetizador y evidenciar el efecto de la combinación de sustancias y su proporción en la solución final. Los cuatro colores primarios substractivos utilizados para representar las cuatro substancias son:

a) La sustancia primaria 1 se representa por el color amarillo.

b) La sustancia primaria 2 se representa por el color azul.

c) La sustancia primaria 3 se representa por el color magenta.

d) La sustancia primaria 4 se representa por el color negro.

El abastecimiento de las sustancias primarias hacia los tanques contenedores no se controla, por lo que solo es necesario detectar si el nivel del líquido es adecuado para que la bomba funcione correctamente. Se asume que el reabastecimiento de la sustancia primaria de cualquier tanque se realiza por otros medios.

6.4.3 Tanque solución

El tanque solución tiene el objetivo de almacenar el resultado final de la mezcla entre las sustancias primarias, este tanque no esta equipado con una bomba, pues el objetivo de esta aplicación no es controlar el vaciado de este tanque para el uso final de la sustancia, sin embargo es necesario detectar cuando el nivel del líquido en el tanque solución esta en peligro de desbordarse.

Por esta razón se adiciona al tanque solución un sensor de nivel que funciona para detectar cuando el líquido en el tanque alcanza el nivel máximo, dicho sensor funciona bajo el mismo principio explicado anteriormente, y simplemente se ha invertido su posición para hacerlo detectar el nivel máximo en lugar del mínimo.

6.4.3.1 Combinación de las sustancias primarias

El resultado final de tales combinaciones puede resumirse con los siguientes patrones, la tonalidad dependerá de la combinación utilizada, la proporción entre ellas y la adición del color negro:

a) Sustancia primaria 1 y sustancia primaria 2 (amarillo y azul) resultan en el color verde.

c) Sustancia primaria 1 y sustancia primaria 3 (amarillo y magenta) resultan en el color rojo.

d) Sustancia primaria 2 y sustancia primaria 3 (azul y magenta) resultan en morado.

e) La sustancia primaria 4 (color negro) puede modificar el tono de una combinación.

6.4.4 Bombas

Cada tanque contenedor esta equipado con una bomba eléctrica que establece un flujo de liquido (sustancia primaria) del tanque contenedor al tanque solución. Las bombas tienen las siguientes características:

a) Alimentación, 120 v 60 Hz.

b) Potencia, 2.5 W.

c) Altura máxima del bombeo, 60 cm.

d) La capacidad de flujo de cada bomba se determina midiendo el tiempo que le toma incrementar 100 mL el volumen del tanque solución, los resultados se expresan en la tabla 6.1.

Tabla 6.1: *Relación volumen-tiempo de una bomba.*

Volumen	Tiempo
100	6.2 s
200	14.8 s
300	22.4 s
400	31 s
500	38 s
600	45.6 s

Las bombas no están diseñadas para funcionar sin líquido, por esta razón el nivel del líquido siempre debe mantenerse sobre las bombas, por lo que el sensor de nivel debe indicar cuando el nivel del líquido en un tanque contenedor ha llegado a un mínimo crítico.

6.5 Propuesta de Automatización (ejecución del proyecto)

6.5.1 Sistema de control Maestro-Seguidor

Para fines de esta automatización, se utilizará una configuración Maestro- Seguidor, donde la PC es el dispositivo maestro, mientras que la IADPC es el dispositivo seguidor. El intercambio de información se realiza mediante el enlace usb- serie de la PC.

De esta manera, en la PC es donde se programa toda la lógica de control, mientras que la IADPC interpreta las señales de los sensores para la PC mediante el puerto 1 de entradas digitales y realiza la activación de las bombas mediante el puerto 2 de salidas digitales. La configuración física de las conexiones de los puertos digitales y el hardware de la IADPC se detalla en la tabla 6.2.

Gracias al tipo de puertos digitales, no se necesito de ninguna adecuación para las señales de control. Los puertos de salida digital al constituirse por relevadores, pueden realizar el control directo de las bombas, mientras que los puertos de salidas digitales al ser del tipo de contacto seco, pueden detectar directamente a la señal de los sensores de nivel (la pérdida o el restablecimiento de la continuidad en el reed switch); véase el capítulo 3.

En la figura 6.4 se muestra la configuración de las conexiones eléctricas entre los puertos de salida digital y las bombas, así como la conexión de los sensores de nivel con las entradas digitales de la IADPC. En la tabla 6.2 se hace referencia a la configuración de las conexiones eléctricas entre la IADPC y el hardware disponible y su relación con cada uno de los tanques con los que se asocian.

Tabla 6.2: *Configuración de las conexiones entre el hardware disponible del sintetizador de soluciones y los puertos digitales de la IADPC.*

Puerto de la IADPC		Dispositivo	
Puerto 1 de entrada digital	Bit 0	Tanque contenedor 1	Sensor de nivel mínimo 1
	Bit 1	Tanque contenedor 2	Sensor de nivel mínimo 2
	Bit 2	Tanque contenedor 3	Sensor de nivel mínimo 3
	Bit 3	Tanque contenedor 4	Sensor de nivel mínimo 4
	Bit 4	Tanque solución	Sensor de nivel máximo
Puerto 2 de salida digital	Bit 0	Tanque contenedor 1	Bomba 1
	Bit 1	Tanque contenedor 2	Bomba 2
	Bit 2	Tanque contenedor 3	Bomba 3
	Bit 3	Tanque contenedor 4	Bomba 4

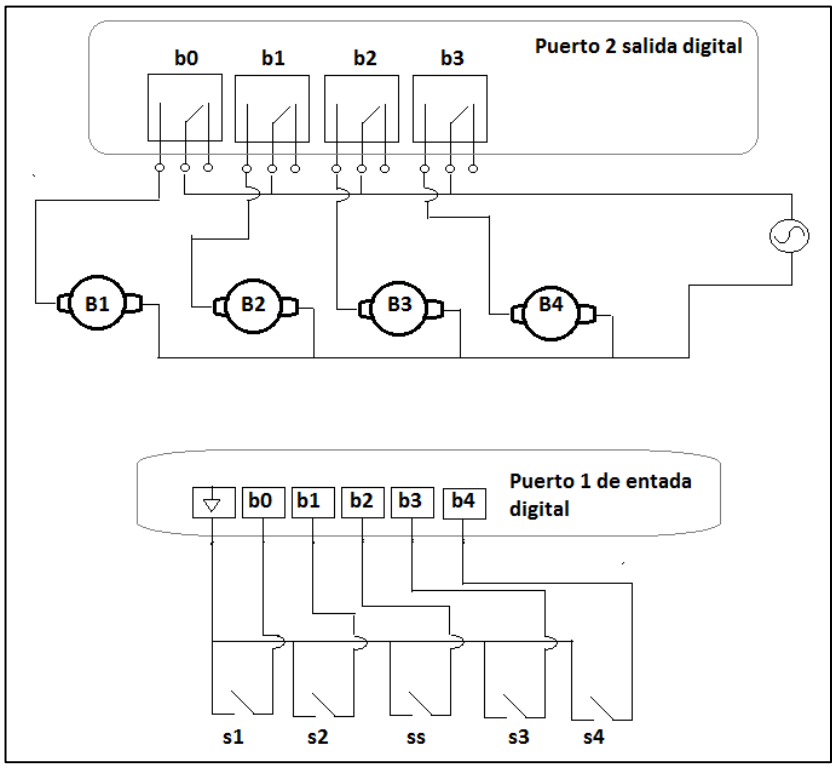


Figura 6.4: *Diagrama de conexiones entre la IADPC y el hardware disponible del sintetizador de soluciones.*

6.6 Software del sistema sintetizador de soluciones

El sintetizador de soluciones se controla mediante una interfaz de usuario, tal como se establece en el planteamiento del problema. Esta interfaz se diseñó con elementos de control e indicadores, fáciles de utilizar e intuitivos. Se incluyen medios para ingresar los datos y arrancar o parar el proceso, además de la visualización del estado de los tanques y sus bombas.

En primer lugar es necesario que el usuario introduzca al programa las proporciones requeridas de cada substancia, de manera que el programa pueda calcular el tiempo de operación de cada bomba.

El volumen que la bomba de uno de los tanques contenedores aporta al tanque solución, esta en función del tiempo y para lograr que al tanque solución llegue la proporción correcta de las sustancias primarias, es necesario controlar el tiempo de operación de las bombas.

El tiempo de operación se controla mediante un bloque de código que se ejecuta cada cierto tiempo, es decir un temporizador. Cada vez que se ejecuta este temporizador verifica si el tiempo de operación de cada bomba ya ha llegado a su fin, por otra parte también comprueba el estado de los tanques, es decir que el líquido de los tanque no este por debajo del nivel mínimo que se permite para el funcionamiento de las bombas.

El tiempo de operación de cada bomba, se calcula en base a la caracterización de la capacidad de la bomba que se revisó en la tabla 6.1 y asumiendo un comportamiento lineal de la bombas. El software del sintetizador de soluciones puede resumirse en tres bloques principales, véase la figura 6.5:

a) Adquisición de datos (requerimientos del usuario y estado de los sensores).

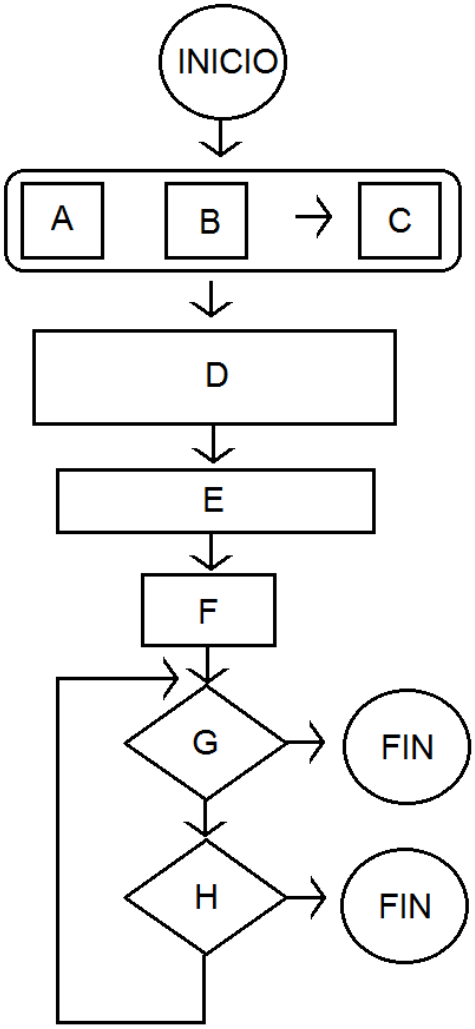
b) Calculo del tiempo de operación.

c) Ciclo principal (temporizador).

6.6.1 Adquisición de los parámetros de control

6.6.1.1 Objetivo

El objetivo del presente bloque de código es permitirle al usuario ingresar los parámetros de control que después se utilizaran para realizar el proceso de llenado, considerando las limitaciones lógicas y físicas del sistema.



A	Parámetros de control ingresados por el usuario.
B	Parámetros de control ingresados desde un archivo de texto.
C	Parámetros de control guardados en un archivo de texto.
D	Parámetros de control: - Cantidad total de sustancia. - % de sustancia primaria 1. - % de sustancia primaria 2. - % de sustancia primaria 3. - % de sustancia primaria 4.
E	Cálculo del tiempo de operación de cada bomba, en función del porcentaje requerido: - Tiempo de operación de la bomba 1. - Tiempo de operación de la bomba 2. - Tiempo de operación de la bomba 3. - Tiempo de operación de la bomba 4.
F	Asignación del tiempo establecido para cada bomba.
G	Interrupción del ciclo principal debida al estado de los tanques: -Fin del tiempo de funcionamiento. -Tanque lleno. -Tanque vacío.
H	Interrupción de ciclo principal debida al accionamiento del botón de paro de emergencia.

Figura 6.5: *Diagrama que muestra los bloques de código principales que componen el software del sintetizador de soluciones*

6.6.1.2 Modos de adquisición

Los parámetros de control pueden ingresarse directamente en la interfaz de usuario o desde un archivo de texto.

a) Directamente desde la interfaz de usuario: Los parámetros de control se pueden ingresar directamente en los cuadros de texto designados para ello, la figura 6.6 muestra la sección de la interfaz de usuario que realiza esta función. Una vez que el usuario ingresa los parámetros de control, es decir el perfil de la solución (véase el inciso d) de 6.3.2) se puede guardar en un archivo de texto mediante el botón guardar.

b) Cargar un perfil de solución desde un archivo de texto: Después de guardar el perfil de la solución, es posible volverlo a utilizar mediante el botón “Abrir solución previa” y localizando el archivo de texto en el lugar donde se guardó previamente. Véase la figura 6.6.

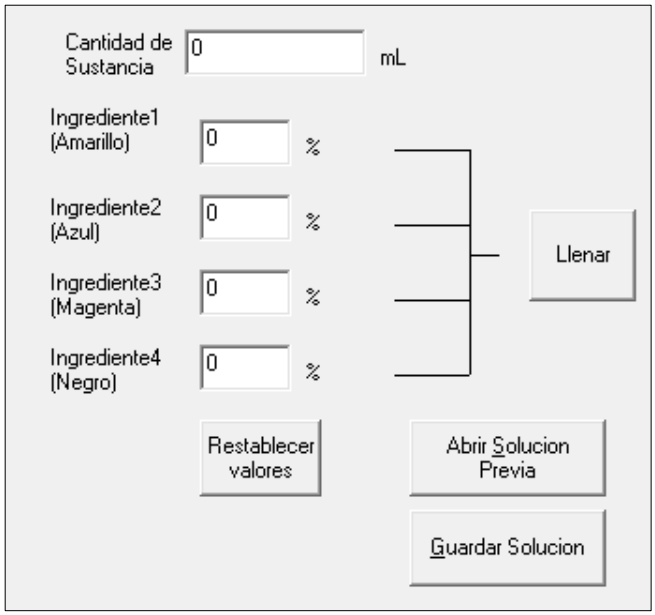


Figura 6.6: *Cuadros de texto utilizados para ingresar los parámetros de control*

6.6.1.3 Restricciones para el ingreso de datos

Al ingresar el perfil de solución se deben tener en cuenta las siguientes restricciones:

a) La cantidad de solución total debe ser mayor a cero y no debe sobrepasar los 600 ml, en caso de que el parámetro de control no cumpla con el requisito, la IADPC devuelve el mensaje de error que se muestra en la figura 6.7.

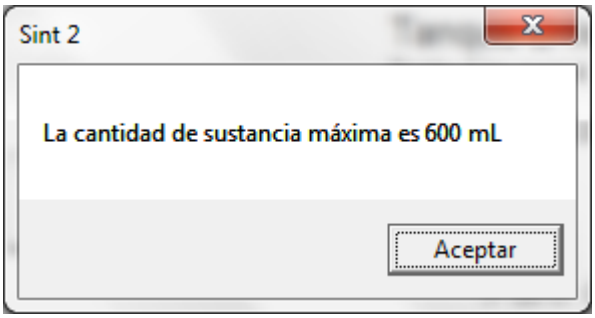


Figura 6.7: *Mensaje de error de la IADPC por ingresar un valor fuera de rango para la cantidad de solución total*

b) La suma de todos los porcentajes de todas las substancias primarias debe ser menor o igual que el 100%. Si la suma sobrepasa este valor, la IADPC devuelve el mensaje de error que se muestra en la figura 6.8.

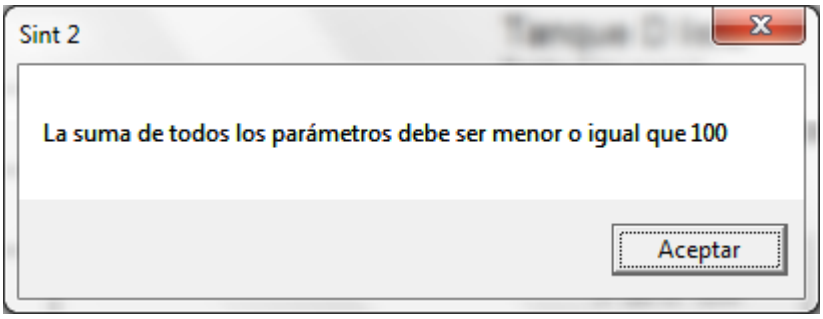


Figura 6.8: *Mensaje de error de la IADPC al ingresar una combinación de proporciones cuya suma sobrepasa el 100%*

6.6.2 Cálculo del tiempo de operación.

6.6.2.1 Objetivo

En el capítulo anterior se determino que es el usuario quien determina la cantidad de volumen que se desea de cada sustancia primaria (en proporción al volumen total). Sin embargo el control del volumen de una solución primaria que una bomba entrega al tanque solución se realiza mediante el control del tiempo de operación de dicha bomba, pues la cantidad de líquido que entrega una bomba es proporcional al tiempo de operación de la misma (se asume el comportamiento lineal de la misma).

Para controlar el tiempo de operación se utiliza un objeto de Visual Basic llamado temporizador, cuya función principal es ejecutar el código asociado con él, una vez cada cierto tiempo (esta ejecución cíclica se describirá detalladamente en 6.5.3). Para este proyecto el temporizador se va a ejecutar cada 200 ms.

Mediante una variable contadora que se incrementa en uno cada vez que el temporizador se ejecuta es posible controlar el tiempo de operación de una bomba, por ejemplo en un minuto el temporizador puede ejecutarse trecientas veces y por lo tanto el valor de la variable contadora será 300. Si se determina que una bomba debe operar durante un minuto, esta se debe activar hasta que la variable contadora tenga un valor de 300.

El objetivo del presente bloque de código es asignar a cada bomba el número de cuentas (ciclos) que va a operar y de esta forma controlar el tiempo de operación de la bomba y por lo tanto el volumen entregado al tanque solución.

6.6.2.2 El desempeño aproximadamente lineal de una bomba

La capacidad de flujo de una bomba se caracterizó en la tabla 6.1 del inciso 6.4.4, midiendo el tiempo que le toma a una bomba entregar 600 mL y registrando una medición cada 100 ml. Considerando que cada 200 ms se ejecuta el temporizador es posible calcular el número de cuentas que le toma a una bomba para incrementar el volumen de líquido del tanque solución en pasos de 100 mL, véase la tabla 6.4. y la figura 6.9.

TABLA 6.4: *Relación entre volumen y el número de cuentas*

Volumen (mL)	Cuentas
100	31
200	74
300	112
400	155
500	190
600	228

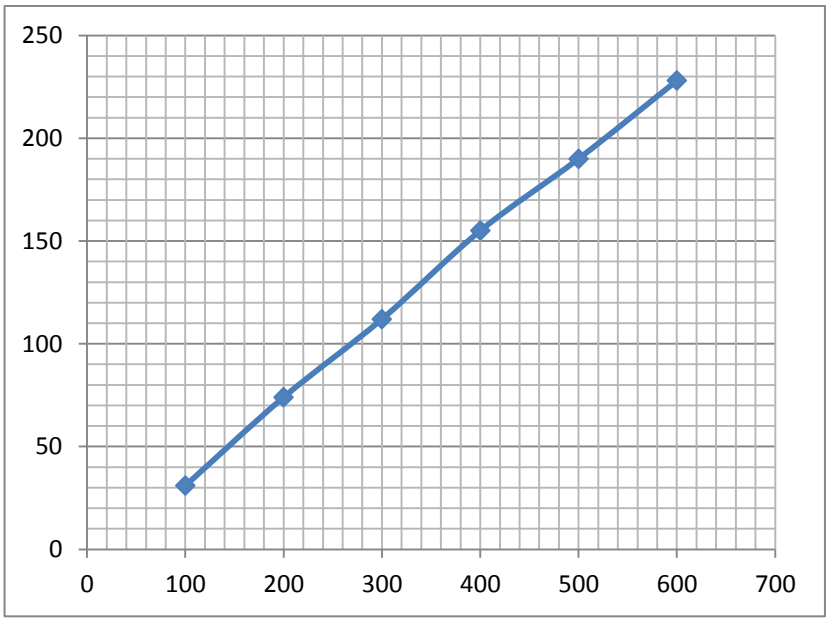


Figura 6.9: *Gráfica que muestra la relación volumen por número de cuentas, el eje de las abscisas corresponde al número de cuentas, mientras que el eje de las ordenadas corresponde al volumen en mL.*

Como se puede apreciar el la figura 6.9, el desempeño de la bomba es aproximadamente lineal, sin embargo, se necesita una función lineal que aproxime la relación entre el volumen solicitado y el número de cuentas.

Mediante la función que se obtenga será posible obtener a partir del volumen solicitado, el número de cuentas que se necesita que una bomba opere.

6.6.2.2.1 Aproximación lineal

El método de regresión lineal es un modelo con forma de polinomio también conocido como estimación por mínimos cuadrados (véase la referencia [5] de la bibliografía).

$$\hat{Y}=b_0+b_1X \qquad 6.1$$

Donde $\hat{Y}$ corresponde el valor estimado de Y para una X dada, cuando se determinan b_0 y b_1.

Para calcular el valor de b_0 y b_1 se utilizan las siguientes fórmulas:

$$b_1=\frac{\sum X_iY_I-(\sum X_i)(\sum Y_i)/n}{\sum X_i^2-(\sum X_i)^2/n} \qquad 6.2$$

$$b_0=\bar{Y}-b_1\bar{X} \qquad 6.3$$

$$\bar{Y}=\left(\sum Y_1\right)/n \qquad 6.4$$

$$\bar{X}=\left(\sum X_1\right)/n \qquad 6.5$$

Donde X es la variable independiente, en este caso el volumen entregado en mL y Y la variable dependiente que corresponde al número de cuentas. La variable “n” representa el numero de muestras o mediciones, en este caso igual que 6.

A continuación en la tabla se muestra el resultado de los cálculos que se requieren para encontrar el valor de b_1 y b_0.

TABLA 6.5: *Resultado de los cálculos requeridos para realizar la regresión lineal*

i	Cuentas (Y)	mL (X)	X^2	X*Y
1	31	100	10 000	3 100
2	74	200	40 000	14 800
3	112	300	90 000	33 600
4	155	400	160 000	62 000
5	190	500	250 000	95 000
6	228	600	360 000	136 800

$\sum X_i=2100$

$\bar{X}=350$

$\bar{Y}=131.66667$

$\sum X_i Y_i = 345300$

$\sum X_i^2 = 910000$

Sustituyendo en la ecuación 6.2:

$$b_1=\frac{68800}{175000}=0.393142857$$

Sustituyendo en la ecuación 6.3

$$b_0=-5.933333$$

Como resultado se obtiene la ecuación 6.6, cuya gráfica se muestra en la figura 6.10; como puede apreciarse es una recta muy aproximada a los valores obtenidos en la tabla 6.4.

$$Y=0.393142857*X - 5.93333 \qquad 6.6$$

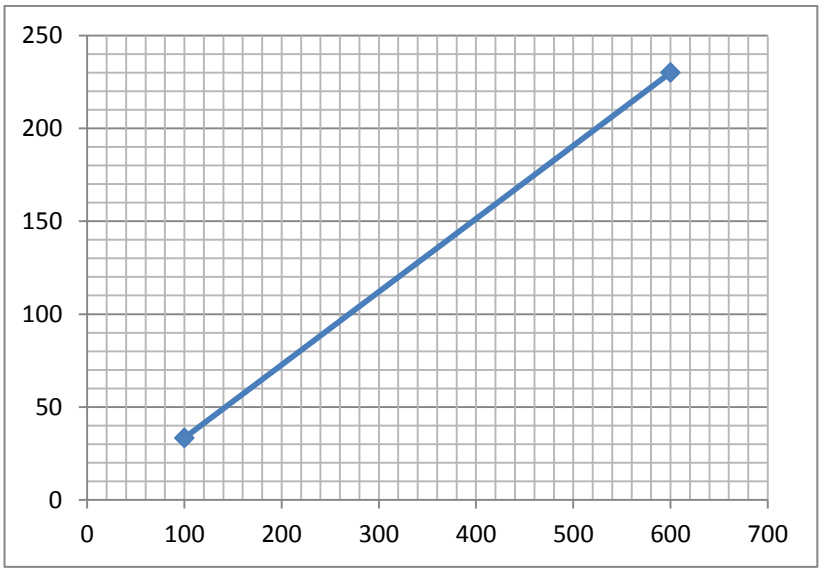


Figura 6.10: *Se muestra la gráfica resultante de la regresión lineal realizada para aproximar la relación volumen por número de cuentas de una bomba, corresponde a la ecuación 6.6*

De esta forma es posible realizar la aproximación lineal de la relación entre volumen y el número de cuentas, mediante la ecuación 6.6.

Como restricciones lógicas, este sistema no puede aceptar números fraccionarios (no se puede realizar la fracción de un ciclo) ni números negativos, por lo que las siguientes condiciones se implementan cuando se asigna el tiempo de operación de una bomba:

a) El valor de Y resultante de la ecuación 6.6 se redondea.

b) El valor de X debe ser mayor o igual que 15 mL para que Y no resulte en un número negativo, por lo que cualquier valor menor que 15 mL se iguala a cero.

Una vez asignado el tiempo (traducido en un número de cuentas) que una bomba debe permanecer en operación, la bomba se activa.

Es función del ciclo principal determinar el momento de apagar la bomba, es decir determinar en que momento la bomba ha terminado su tiempo de operación.

6.6.3 Ciclo principal

6.6.3.1 Objetivo

El objetivo de este bloque de código es determinar cuando una bomba debe parar. El ciclo principal tiene la función de comparar el valor de la variable contadora con el número de cuentas asignado a cada bomba, el paro de una bomba sucede cuando el número de cuentas asignadas a la misma es igual que el valor de la variable contadora. Esto último significa que la bomba ha agotado su tiempo de operación y le ha proporcionado al tanque solución el volumen determinado por el usuario.

Tanto el incremento de la variable contadora, como la comparación del valor de la misma con el número de cuentas asignado a una bomba, se realizan de manera cíclica cada 200 ms.

Sin embargo el paro de las bombas no se produce únicamente por el fin del proceso de llenado, pues según las condiciones de operación puede ser necesario que las bombas realicen un paro extraordinario, mismo que se explica a continuación.

6.6.3.2 Paro extraordinario de las bombas

El paro extraordinario de las bombas es la desactivación de las bombas por otra causa que no sea el fin del proceso de llenado, a lo que se le llama paro normal. Su objetivo es proporcionar la desactivación de las bombas bajo las siguientes condiciones críticas en el proceso de llenado:

a) Cuando uno de los tanques contenedores llega a un nivel peligroso para el funcionamiento de una bomba, el sistema pone en pausa a todo el sistema, este paro no es definitivo y en cuanto el nivel del tanque se restablece a un punto aceptable, el sistema reinicia en su último estado. En la figura 6.12 es posible apreciar a los indicadores de la interfaz desplegando el estado de los tanques contenedores y sus bombas asociadas.

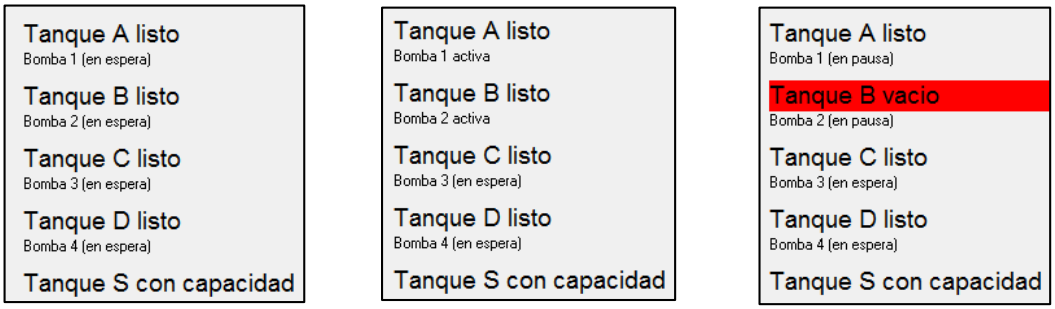


Figura 6.11: *Indicadores de estado de los tanques y las bombas asociadas, en la figura A se puede apreciar a los indicadores con todos los tanques en espera de que inicie el proceso de llenado, en la figura B el estado de los indicadores mientras la bomba 1 y 2 se encuentran en operación, finalmente el estado de las bombas 1 y 2 cuando el tanque contenedor B tiene un nivel por debajo del mínimo aceptable.*

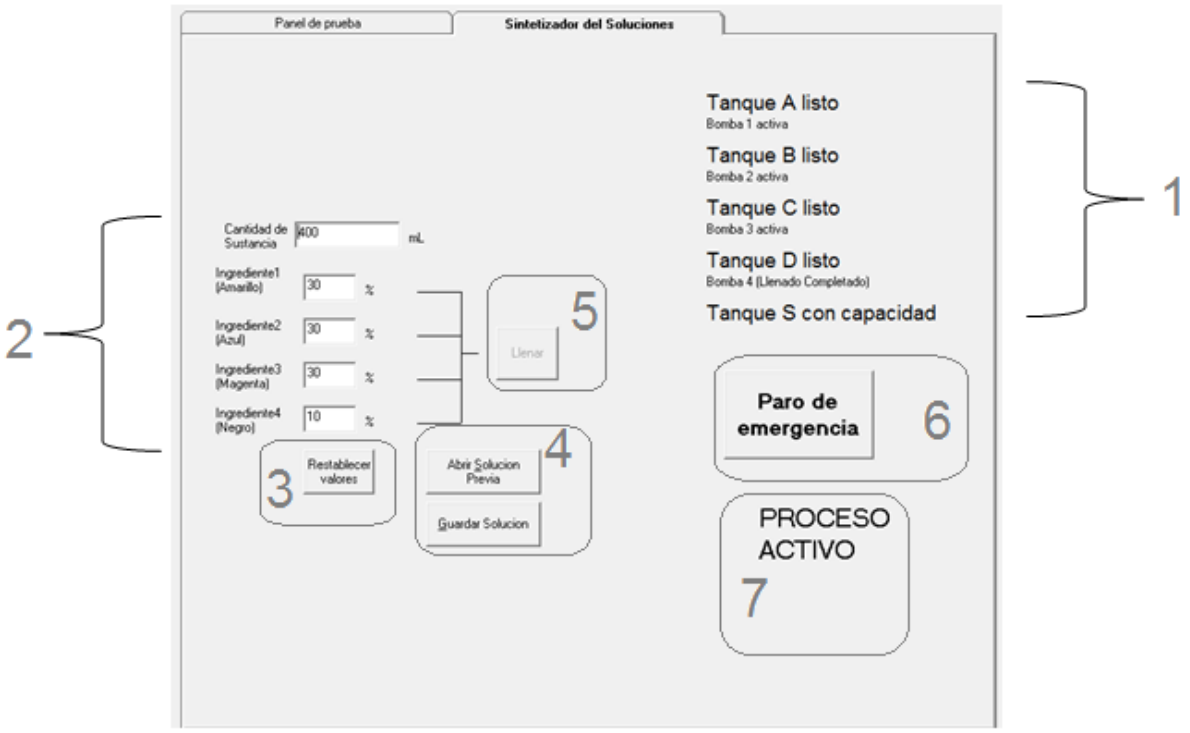
De manera similar ocurre con el tanque solución cuando este alcanza su máximo nivel, para evitar el derrame de la solución final, el sistema se detiene, en cuando el nivel es restablecido a un nivel aceptable, el sistema reinicia.

b) La segunda forma de paro extraordinario es el botón de paro de emergencia en la interfaz de usuario. Este botón cancela definitivamente el proceso de llenado y lo lleva a su estado inicial, es decir, listo para comenzar un nuevo proceso de llenado desde la adquisición de los parámetros de control, el botón de paro puede apreciarse en la figura 6.13.

Figura 6.12: *Botón de paro de emergencia. En la figura A se muestra el estado del botón durante el proceso de llenado (habilitado). En la figura B se muestra el estado del botón de paro cuando se encuentra en espera.*

6.6.4 Indicadores de la interfaz

La interfaz de usuario se diseño para que desde ella se pueda controlar en su totalidad al sintetizador de soluciones, desde un ambiente gráfico e intuitivo, y no se requiere de entrenamiento para poder utilizarla. En la interfaz se encuentran todos los controles e indicadores vistos anteriormente, cumpliendo con los requisitos del sistema. En la figura 6.14 se muestra a la interfaz de usuario en su totalidad.



Número	Descripción
1	Indicadores de estado de las bombas y sensores de nivel asociados a tanques contenedores, designados como tanques A, B,C, D e indicador del estado del sensor de nivel asociado con tanque solución, designado como tanque S.
2	Ingreso de los parámetros de control.
3	Botón que borra el estado actual de los parámetros de control.
4	Controles para cargar o guardar el perfil de una solución
5	Botón para iniciar el proceso de llenado (se deshabilita durante el proceso de llenado).
6	Botón de paro de emergencia.
7	Indicador de estado del proceso de llenado, según las condiciones muestra los estados: - Proceso terminado. - Proceso cancelado. - Proceso activo. - En espera.

Figura 6.14: *Interfaz de usuario*

CONCLUSIONES

El desarrollo de la IADPC fue un proyecto de tesis cuyo objetivo es el de crear una solución para el control automático que fuese intercambiable y compatible con la gran variedad de lenguajes de programación que actualmente existen.

Debido a la complejidad del proyecto se tuvo la oportunidad de aplicar la mayor parte de las habilidades obtenidas durante la carrera, no obstante el enfoque práctico de un proceso de diseño real, retroalimenta y enriquece los conocimientos con experiencia.

Durante el desarrollo de la tesis se presentaron varios retos que fueron paulatinamente superados. Ejemplos de estos retos fueron la falta de equipo para realizar los circuitos impresos y la gran complejidad que conlleva el diseñarlos y armarlos bajo las especificaciones del proyecto. Uno de los mayores desafíos, relacionados con el desarrollo del software de la interfaz de usuario, fue que no se contaba con experiencia en el leguaje de programación Visual Basic y gracias a este proyecto se tuvo la oportunidad de un primer acercamiento a este tipo de programación. El resultado final es la creación de un dispositivo completamente funcional y la documentación de su desarrollo, que representó una acumulación de experiencias y aprendizajes a las que solo se pueden acceder mediante el diseño de un prototipo de esta naturaleza.

Uno de los objetivos alcanzados con esta tesis es el diseño de una interfaz de control que no se limita a un solo software o ambiente de programación y que tiene como ventaja económica que no se necesita el pago de licencias ni derechos, o de equipo de control altamente especializado y, desde el punto de vista funcional, es posible a partir de este diseño inicial modificar a la IADPC para las aplicaciones requeridas, todo esto sin contar que en el transcurso de su desarrollo se proponen soluciones, esquemas y diseños que pueden ser utilizados en otros proyectos.

Los productos que actualmente existen en el mercado son capaces de superar la funcionalidad de la IADPC, tanto en su capacidad de expansión de entradas salidas, como en el desarrollo de sus ambientes de programación, que por lo regular son muy amigables con el usuario.

Sin embargo el hecho de que la IADPC no tenga una interfaz de usuario en particular es, para ciertas aplicaciones, una característica competitiva, tanto del punto de vista económico (no hay pago por licencias ni por cursos de entrenamiento) como en el aspecto de que el desarrollador realiza la programación de la IADPC en el lenguaje que el selecciona en función de la aplicación requerida y conoce enteramente las características de su algoritmo, en otras palabras tiene el control total sobre la interfaz.

Teniendo en cuenta que en el presente trabajo se explica cada uno de los aspectos de diseño de la IADPC, puede suponerse que el usuario de la misma tiene la opción de modificar su diseño y características para ajustarlas a sus propios intereses e incluso mejorar el desempeño de la misma como parte de un proyecto más ambicioso. Tal como se demuestra en el capítulo 6, la IADPC es por si misma un dispositivo cuya funcionalidad y características le permiten ser parte de muchos otros proyectos de automatización y control.

Próximas etapas:

Sin prejuicio de que este proyecto ha llegado a la fase de prototipo es importante hacer mención de algunas de las posibles etapas que este proyecto puede tener, considerando que el presente trabajo puede servir como base de un desarrollo con mayor alcance.

1 - Protocolos de comunicación: El uso de protocoles de comunicación estandarizados permite la expansión de las funcionalidades de la IADPC como para poder emplear distintas filosofías de control y para integrar el dispositivo a sistemas ya existentes.

2 - Tarjetas de expansión: Lograr realizar el diseño para tarjetas de expansión tanto de salidas como para entradas aumenta el desempeño y adaptabilidad de la IADPC.

3 - Producción en masa: El crear sistemas de producción en masa y de control de la calidad permitirán la comercialización de la IADPC.

BIBLIOGRAFÍA

[1]COUGHLIN, F, Robert. DRISOCOLL, F, Frederick. “Amplificadores operacionales y circuitos integrados lineales. Prentice - Hall Hispanoamericano S.A. México. 1993. Cuarta edición. 538pp. (Págs. 411-412, 420-421).

[2]PUCKNELL, A, Douglas. “Fundaments of Digital Logic Design: with VLSI circuit aplications”. Prentice Hall. Australia. 1990. 472pp. (Págs. 383-389).

[3]GUREWICH, Nathan. GUREWICH, Ori. “Teach Yourself. Visual Basics in 21 days” SAMS.Indianapolis.1995.Tercera edición.960pp. (Págs. 1-37).

[4]CEHMIEHEN, P, J. “Como deben emplearse los circuitos integrados: estructuras, funcionamiento y aplicaciones de los principales circuitos integrados”. Paraninto. Madrid. 1985. Tercera edición. 620pp. (Págs. 240-245).

[5]PRAPER, R, Norman. SMITH, Hanry. “Apllied Regresion Analysis”. A Willey-Intersciencia publication. “United States of América. 1998. Tercera edición. 706pp. (Págs. 22-29).

[6]Salvá Calleja Antonio “AIDA 08 Ambiente integrado para desarrollo y aprendizaje con microcontroladores de la familia 68HC908 de FREESCALE (manual de usuario básico), Enero 2011(Págs. 4-86).

[7]Salvá Calleja Antonio, Víctor Manuel Sánchez Esquivel, María Leonor Salcedo Ubilla, Jose Luis Ramírez Gutierrez “Manual de usuario del PLM2” Noviembre 2006 (Págs. 20-30).

[8]Hoja técnica: MC68HC908GP32/H Rev. 6,8/2002

Apéndice A

Módulos de software ejecutables en la IADPC

A continuación se expone el código fuente de los módulos de software ejecutables en la PC. A diferencia del Capítulo 5, este Apéndice únicamente al código comentado organizado de la siguiente manera:

- procedimientos y funciones generales.
- código asociado a los controles del panel de prueba de la IADPC.

Se recomienda contar con conocimientos en la plataforma de programación Visual Basic.

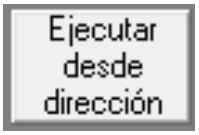
A.1 Procedimientos y funciones generales

```
Public Function rxps(y) As Integer

'transmisión del comando, solicitando  la lectura desde la IADPC

txp1 (y)
'fin de transmicion de comando

Dim ii As Integer

'determinar el tamaño del paquete a recibir
MSComm1.InputLen = 1

'Periodo de espera
Do
ii = ii + 1
If ii = 30000 Then
GoTo finn
End If
Loop Until MSComm1.InBufferCount >= 1

'Si se recibe antes de que se acabe el tiempo de espera, entonces devuelve el valor del puerto
rxps = Asc(MSComm1.Input)
MSComm1.InBufferCount = 0
GoTo finfel


'Si se acaba el tiempo de espera y no hay recepción, manda mensaje de error
finn:
MsgBox " No hay respuesta de la IADPC"
MSComm1.InBufferCount = 0

finfel:
End Function
```

```
Public Sub txp1(x)
'Procedimiento de transmisión simple desde la PC a la IADPC, solo envía un byte
'transmite el valor en decimal en x'

MSComm1.Output = Chr(x)
End Sub


Public Sub txps(x, y)
'Procedimiento de transmisión de comandos de control, envía primero  el puerto y comando de escritura, después el dato a
'escribir:

'Transmite el byte cuyo valor es x
'Número de puerto y comando
repetir:

ii = ii + 1
If ii < 30000 Then GoTo repetir

MSComm1.Output = Chr(y)

'Byte Dato
ii = 0
repetir2:
ii = ii + 1
If ii < 30000 Then GoTo repetir2

MSComm1.Output = Chr(x)

End Sub
```

A.2 Código asociado a los controles del panel de prueba de la IADPC

```
Private Sub Command1_Click()
'Botón ejecutar desde dirección, envía la indicación de ejecutar
'un programa residente en el MCU, antes de activar se debe
'ingresar la dirección en el cuadro de texto contiguo.

Dim By1 As Long
Dim BY2 As Long

A = Val(DirecEjec.Text)
By1 = A / 256

BY2 = A - (By1 * 256)

txp1 (14)
txp1 (0)
txp1 (192)
txp1 (0)
txp1 (194)
txp1 (204)
'txp1 (128)
'txp1 (0)
txp1 (By1)
txp1 (BY2)

End Sub
```



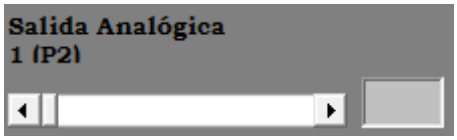
```
Private Sub IADPC_Click()

'Botón ejecutar desde dirección, envía la indicación de ejecutar
'un programa residente en el MCU, tiene precargado la
'dirección del software base del al IADPC

Dim By1 As Long
Dim BY2 As Long

A = 32768
By1 = A / 256
BY2 = A - (By1 * 256)

txp1 (14)
txp1 (0)
txp1 (192)
txp1 (0)
txp1 (194)
txp1 (204)
'envío de la dirección de la interfaz:
txp1 (By1)
txp1 (BY2)

End Sub
```


```
Private Sub Reset_Click()

'Envío del comando para restablecer la IADPC

txp1 (255)
End Sub
```

```
Private Sub Cmd1_Click()
' Función que envía el comando de escritura al puerto de salida
'digital 1, y el dato que el usuario escribe.

'inicializa los testigos
P4b0 = 0
P4b1 = 0
P4b2 = 0
P4b3 = 0
P4b4 = 0
P4b5 = 0
P4b6 = 0
P4b7 = 0
'Lee el dato ingresado en el cuadro de texto.
Data = Val(EscP4Txt)
'Verifica si el dato se encuentra en el rango de 0 a 255
If Data > 255 Then
'De no estar en el rango:
MsgBox "ERROR.          El dato a escribir debe estar entre 0 y
255"
'En caso contrario envía comando de escritura.
Else
txps (Data), (148)
'testificado en controles bit por bit, cada vez que se envíe un
'dato por medio del boton "escribir" escribira el mismo dato en
'binario en los controles 'por bit
aux = Data And 1
If aux = 1 Then
P4b0 = 1
End If

aux = Data And 2
If aux = 2 Then
P4b1 = 1
End If

aux = Data And 4
If aux = 4 Then
P4b2 = 1
End If

aux = Data And 8
If aux = 8 Then
P4b3 = 1
End If

aux = Data And 16
If aux = 16 Then
P4b4 = 1
End If

aux = Data And 32
If aux = 32 Then
P4b5 = 1
End If

aux = Data And 64
If aux = 64 Then
P4b6 = 1
End If

aux = Data And 128
If aux = 128 Then
P4b7 = 1
End If
End If

End Sub
```

```
Private Sub Command2_Click()


'control del convertidor analógico-digital
'Seleccion del canal Analógico de entrada
Dim canal As Integer
Dim ADConv As Double

If Option1 = True Then
canal = 0
End If

If Option2 = True Then
canal = 1
End If

If Option3 = True Then
canal = 2
End If

If Option4 = True Then
canal = 3
End If

'Lectura analógica
ADConv = (rxps(canal))

'Seleccion del modo para el ADC

'10 a +10
If Combo1.ListIndex = 0 Then
LecAtext1 = (ADConv * (20 / 255)) - 10
Text1.Text = ADConv
End If
'de -5 a 5
If Combo1.ListIndex = 1 Then
LecAtext1 = (ADConv * (10 / 255)) - 5
End If
'de -1 a 1
If Combo1.ListIndex = 2 Then
LecAtext1 = (ADConv * (2 / 255)) - 1
End If
'de 0 a 10
If Combo1.ListIndex = 3 Then
LecAtext1 = (ADConv * (10 / 255))
End If
'de 0 a 5
If Combo1.ListIndex = 4 Then
LecAtext1 = (ADConv * (5 / 255))
End If
'de 0 a 1
If Combo1.ListIndex = 5 Then
LecAtext1 = (ADConv * (1 / 255))
End If
End Sub
```

```
Private Sub SalAnalog2_Change()
'Control del convertidor Digital-Analógico 1
a la IADPC el valor de la barra de desplazamiento
outan2 = SalAnalog2.Value
txps (outan2), (131)

'Indicar el valor en el texto contiguo

AnaSal2txt.Text = outan2 * (20 / 255) - 10
End Sub
```




```
Private Sub Form_Load()

'Cuando el programa inicia se activa e  puerto serie

MSComm1.PortOpen = True
 Dim canal As Integer
End Sub
```

Apéndice B

Módulos de software del sintetizador de soluciones

A continuación se expone el código fuente de los módulos de software ejecutables del ejemplo de aplicación de la IADPC, el sintetizador de soluciones. A diferencia del Capítulo 6, este Apéndice únicamente al código comentado organizado de la siguiente manera:

- declaraciones generales
- procedimientos y funciones generales.
- código asociado a los controles del sintetizador de soluciones.

Considerar que las funciones de transmisión del “Panel de prueba de la IADPC” se utilizan por el sintetizador de soluciones, por lo que las funciones y procedimientos del inciso A.1 también son utilizados por la presente aplicación. Se recomienda contar con conocimientos en la plataforma de programación Visual Basic.

B.1 Declaraciones generales

```
'Variables globales de estado de los sensores
Public TA, TB, TC, TD, TS As Integer

'Variables globales, numero de cuentas por bomba
Public Nctas1, Nctas2, Nctas3, Nctas4 As Integer

'Variables globales de control de los sensores
Public BT1, BT2, BT3, BT4, CTot As Integer

'Variables globales de temporizadores
Public CONT As Integer

'Variables globales s auxiliares por bomba

Public aux1, aux2, aux3, aux4 As Integer

'Variables globales de temporizador
Public enc1, enc2, enc3, enc4, esp As Boolean

'Variable global, botón de paro
Public Cancelacion As Boolean
```

B.2 Procedimientos y funciones generales

```
Public Function Calcula(Ca) As Integer

'Función resultado de la aproximación por mínimos cuadrados la relación volumen-número de cuentas,
‘ para comenzar ingresar el volumen requerido.

Public Function Calcula(Ca) As Integer

If Ca < 15.1 Then

 Calcula = 0


Else

Calcula = 0.3931 * Ca - 5.9333

End If

End Function
```

```
Public Sub EjecIADPC()

'Procedimiento que envía los comandos necesarios para
indicar la ejecución del software base en el 'MCU, para
el funcionamiento de la IADPC


Dim By1 As Long
Dim BY2 As Long

a = 32768

By1 = a / 256

BY2 = a - (By1 * 256)

txp1 (14)
txp1 (0)
txp1 (192)
txp1 (0)
txp1 (194)
txp1 (204)
'envío de la dirección:
txp1 (By1)
txp1 (BY2)
End Sub
```

```
Public Sub Encender()

'Asignar el numero de cuentas (tiempo de ejecución) para cada bomba,
'se utiliza la función “Calcula ( )”

Label14.Caption = "PROCESO ACTIVO"

 cuetot = CTot

If BT1 > 0 Then
   a1 = Calcula((cuetot * BT1) / 100)

    Nctas1 = Abs(a1)
    txps (1), (181)
    enc1 = True
    Label17.Caption = "Bomba 1 activa"
End If

If BT2 > 0 Then
b1 = Calcula((BT2 * cuetot) / 100)
     Nctas2 = Abs(b1)
     txps (2), (181)
     enc2 = True
     Label18.Caption = "Bomba 2 activa"
End If


If BT3 > 0 Then
c1 = Calcula((BT3 * cuetot) / 100)
     Nctas3 = Abs(c1)
     txps (4), (181)
     enc3 = True
     Label19.Caption = "Bomba 3 activa"
End If


If BT4 > 0 Then


d1 = Calcula((cuetot * BT4) / 100)


    Nctas4 = Abs(d1)
    Label20.Caption = "Bomba 4 activa"

     txps (8), (181)
     enc4 = True


End If

' en caso de que se necesite llamar de nuevo a la
' funcion encender ya no se reasigna el valor a NctasX
BT1 = 0
 BT2 = 0
 BT3 = 0
  BT4 = 0

'deshabilitar los comandos de llenado
'se habilitaran cuando el llenado termine
MezcManual.Enabled = False

Proceso.Enabled = True
Cancelar.Enabled = True

End Sub
```

```
Public Sub Evaluar()
Call status
Dim a, b, c, d As Integer
a = b = c = d= 1
Label14.Caption = "Proceso en pausa, revisar el estado de los tanques o cancelar"
'Verificando tanque 1 solo si éste se encontraba activo
If enc1 = True Then
Label17.Caption = "Bomba 1 (en pausa)"
   If aux1 = 0 Then
   a = 1
   Else
   a = 0
   End If
End If
'Verificando tanque 2 solo si éste se encontraba activo
If enc2 = True Then
Label18.Caption = "Bomba 2 (en pausa)"
   If aux2 = 0 Then
   b = 1
   Else
   b = 0
   End If
End If
'Verificando tanque 3 solo si éste se encontraba activo
If enc3 = True Then
Label19.Caption = "Bomba 3 (en pausa)"
   If aux3 = 0 Then
   c = 1
   Else
   c = 0
   End If
End If
'Verificando tanque 4 solo si éste se encontraba activo
If enc4 = True Then
Label20.Caption = "Bomba 4 (en pausa)"
   If aux4 = 0 Then
   d = 1
   Else
   d = 0
   End If
End If
End Sub
```

```
andozos = a
andozos = andozos And b
andozos = andozos And c
andozos = andozos And d
Label16.Caption = Val(andozos)
If andozos = 1 Then
   If enc1 = True Then
   Label17.Caption = "Bomba 1 (activa)"
   txps (1), (181)
   txps (1), (181)
   End If
   If enc2 = True Then
   Label18.Caption = "Bomba 2 (activa)"
   txps (2), (181)
   txps (2), (181)
   End If
   If enc3 = True Then
   Label19.Caption = "Bomba 3 (activa)"
   txps (4), (181)
   txps (4), (181)
   End If
   If enc4 = True Then
   Label20.Caption = "Bomba 4 (activa)"
   txps (8), (181)
   txps (8), (181)
   End If
esp = False
End If
```

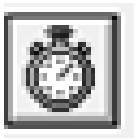
```
Public Sub status()
'funcion que reporta el estado de los cinco tanques
Dim X, Y As Integer
'leer el puerto 2;
X = (rxps(17))
   'estado del tanque a
   Y = X And 1
   If Y = 1 Then
        TA = 1
        LabTA.Caption = "Tanque A vacio"
        LabTA.BackColor = &HFF&
        Else
        TA = 0
        LabTA.Caption = "Tanque A listo"
        LabTA.BackColor = &H8000000F
              End If
  'estado del tanque b
   Y = X And 2
   If Y = 2 Then
        TB = 1
        LabTB.Caption = "Tanque B vacio"
        LabTB.BackColor = &HFF&
        Else
        TB = 0
        LabTB.Caption = "Tanque B listo"
        LabTB.BackColor = &H8000000F
        End If
  'estado del tanque c
   Y = X And 4
   If Y = 4 Then
        TC = 1
        LabTC.Caption = "Tanque C vacio"
        LabTC.BackColor = &HFF&
        Else
        TC = 0
        LabTC.Caption = "Tanque C listo"
        LabTC.BackColor = &H8000000F
        End If
 'estado del tanque D
   Y = X And 8
   If Y = 8 Then
        TD = 1
        LabTD.Caption = "Tanque D vacio"
        LabTD.BackColor = &HFF&
        Else
        TD = 0
        LabTD.Caption = "Tanque D listo"
        LabTD.BackColor = &H8000000F
        End If
```

```
  'estado del tanque s
   Y = X And 16
   If Y = 16 Then
        TS = 1
        LabTS.Caption = "Tanque S lleno"
        LabTS.BackColor = &HFF&
        Else
        TS = 0
        LabTS.Caption = "Tanque S con capacidad"
        LabTS.BackColor = &H8000000F
        End If


'auxiliares:
      aux1 = TA Or TS
      aux2 = TB Or TS
      aux3 = TC Or TS
      aux4 = TD Or TS
      End Sub
```

B.3 Código asociado a los controles del sintetizador de soluciones

```
Private Sub MezcManual_Click()

Dim O, P, Q, R, S As Integer

'Este boton activa el método del llenado manual
'que consiste en tomar los valores que el ususario
'ingresa en los cuadros de texto Text, Text2
'Text3 y Text4


'estado inicial de las variables

BT1 = BT2 = BT3 = BT4 = 0

'adquisición de los datos en los cuadros de texto

O = Val(Text1.Text)
P = Val(Text2.Text)
Q = Val(Text3.Text)
R = Val(Text4.Text)
S = Val(cantidad.Text)

'En caso de que la suma sea mayor que 100%
'enviar mensaje de error

suma = O + P + Q + R
If suma > 100 Then
MsgBox "La suma de todos los parámetros debe ser menor o igual que 100"

Else

'En caso de la suma de los valores asignados sea menor
'o igual que 100%, asigna los valores e inicia la secuencia
'Llenado

BT1 = O
BT2 = P
BT3 = Q
BT4 = R

'iniciar el llenado si corresponde al máximo
'aceptable

   If S <= 600 Then

   CTot = S
   Call Encender
   Else

   MsgBox "La cantidad de sustancia máxima es 600 mL"

   End If

End If

End Sub
```

```
Private Sub Proceso_Timer()
'verifica si el sistema esta en el periodo de espera,
'de ser así evita el proceso de conteo y llenado
If esp = True Then GoTo ESPERA
Call status
'LLENADO CON TANQUE 1
      If CONT >= Nctas1 Then
         If enc1 = True Then
         enc1 = False
         Label17.Caption = "Bomba 1 (Llenado Completado)"
         txps (1), (197)
         txps (1), (197)
         End If
      Else
      If aux1 > 0 Then
     'sistema en pausa
      GoTo ESPERA
      End If
      End If
'LLENADO CON TANQUE 2
      If CONT >= Nctas2 Then
         If enc2 = True Then
         enc2 = False
         txps (2), (197)
         txps (2), (197)
         Label18.Caption = "Bomba 2 (Llenado Completado)"
         End If
      Else
         If aux2 > 0 Then
         txps (2), (197)
         txps (2), (197)
         'insertar stand by 1
         GoTo ESPERA
         End If
      End If
'LLENADO CON TANQUE 3
      If CONT >= Nctas3 Then
         If enc3 = True Then
         enc3 = False
         Label19.Caption = "Bomba 3 (Llenado Completado)"
         txps (4), (197)
         txps (4), (197)
        End If
      Else
         If aux3 > 0 Then
         txps (2), (197)
         'insertar stand by 1
         GoTo ESPERA
         End If
      End If
'LLENADO CON TANQUE 4
      If CONT >= Nctas4 Then
         If enc4 = True Then
         enc4 = False
         Label20.Caption = "Bomba 4 (Llenado Completado)"
         txps (8), (197)
         txps (8), (197)
        End If
      Else
         If aux4 > 0 Then
         txps (8), (197)
         'insertar stand by 1
         GoTo ESPERA
         End If
      End If
 'incrementa al contador
  CONT = CONT + 1
      'Verifica si todas las bombas ya han terminado su ciclo
      funcionando = enc1 Or enc2 Or enc3 Or enc4
      'secuencia del final del proceso
      If funcionando = False Then
      CONT = 0
      'hailita al boton de mando para el llenado
      MezcManual.Enabled = True
      Cancelar.Enabled = False
      'apagar proceso y reestablecer las condiciones para otro
'proceso
      Proceso.Enabled = False
      Label14 = "Proceso Terminado"
         End If
GoTo final
ESPERA:
txps (0), (149)
txps (0), (149)
esp = True
         Call Evaluar
final:
End Sub
```



```
Private Sub Command3_Click()
'Boton para reestablecer los valores de
'de entrada del  los parámetros de control

Text1.Text = 0
Text2.Text = 0
Text3.Text = 0
Text4.Text = 0

cantidad.Text = 0

End Sub
```


```
Private Sub CmdAbrir_Click()
Dim IntFile As Integer
Dim StrLinea As String

IntFile = FreeFile

Me.CommonDialog1.DialogTitle = "Abrir Solución"
Me.CommonDialog1.Filter = "Text Documents (*.txt)|*.txt"
Me.CommonDialog1.FilterIndex = 1
Me.CommonDialog1.ShowOpen

On Error GoTo Escape

Open Me.CommonDialog1.filename For Input As #IntFile

Line Input #IntFile, StrLinea
CTot = CInt(StrLinea)
Line Input #IntFile, StrLinea
BT1 = CInt(StrLinea)
Line Input #IntFile, StrLinea
BT2 = CInt(StrLinea)
Line Input #IntFile, StrLinea
BT3 = CInt(StrLinea)
Line Input #IntFile, StrLinea
BT4 = CInt(StrLinea)

cantidad.Text = CTot

Text1.Text = BT1
Text2.Text = BT2
Text3.Text = BT3
Text4.Text = BT4

Escape:
End Sub
```

```
Private Sub CmdGuardar_Click()
Dim IntFile As Integer

Me.CommonDialog1.DialogTitle = "Guardar Solución"
Me.CommonDialog1.Filter = "Text Documents (*.txt)|*.txt"
Me.CommonDialog1.FilterIndex = 1
Me.CommonDialog1.ShowSave

On Error GoTo Escape
IntFile = FreeFile

Open Me.CommonDialog1.filename For Output As #IntFile
Write #IntFile, Val(cantidad.Text)
Write #IntFile, Val(Text1.Text)
Write #IntFile, Val(Text2.Text)
Write #IntFile, Val(Text3.Text)
Write #IntFile, Val(Text4.Text)
Close #IntFile

Escape:
End Sub
```


```
Private Sub Cancelar_Click()

Proceso.Enabled = False
MezcManual.Enabled = True
txps (0), (149)
Nctas1 = 0
Nctas2 = 0
Nctas3 = 0
Nctas4 = 0
CONT = 0
esp = False
enc1 = False
enc2 = False
enc3 = False
enc4 = False
Label14.Caption = "Proceso cancelado"
Cancelar.Enabled = False
End Sub
```