UNIVERSIDAD AUTÓNOMA DEL ESTADO

DE HIDALGO

CAMPUS SAHAGÚN

INGENIERÍA INDUSTRIAL

"IMPLEMENTACIÓN DEL MICROCONTROLADOR SX48BD CON MÓDULO BASIC STAMP, CON EJEMPLOS PRÁCTICOS CORRESPONDIENTE A LA MATERIA DE AUTOMATIZACIÓN"

M A T E R I A L D I D Á C T I C O

QUE PARA OBTENER EL TÍTULO DE:

L I C E N C I A D O E N I N G E N I E R Í A

I N D U S T R I A L

P R E S E N T A :

R E Y E S F R A N C O M U Ñ O Z

ASESOR: DR. RODOLFO HERNÁNDEZ SERVIN

SEPTIEMBRE 2007

PÁGINA

ÍNDICE

Dedicatoria

A MI HIJA KATHERINE KETZALLI:

Por que tú presencia angelical ha sido y será el motivo más grande que dios me ha obsequiado; con tu llegada transformaste los días difíciles en los más felices de mi vida; te amo mi niña hermosa.

A MI FAMILIA:

A mi esposa por su cariño y comprensión, A mi madre por su cariño que me ha brindado toda esta vida, a mis hermanos con cariño y respeto.

A MIS MAESTROS:

Al Dr. Rodolfo Hernández Servin, por su comprensión y desinteresada ayuda en la realización de este trabajo; a Ing. Ernesto Gutiérrez R., Lic. María de Jesús Balderas por su amistad y apoyo incondicional en una etapa difícil de mi vida, a todos los catedráticos que en diferentes ocasiones me han brindado su apoyo Gracias.

A MIS COMPAÑEROS Y AMIGOS

Quienes han compartido una etapa de la vida en un objetivo de superación para ser mejores personas.

A Dios por compartir mis triunfos que me hicieron humilde, en mis fracasos que me hicieron fuerte, en momentos de felicidad su dicha y bondad…

INTRODUCCIÓN.

Los Microcontroladores están siendo empleados en multitud de sistemas presentes en nuestra vida diaria, como pueden ser: Medicina, alarmas para casas, en horno microondas, televisores, computadoras, impresoras, el sistema de arranque de los automóviles, etc.; una aplicación típica podría emplear varios Microcontroladores para controlar pequeñas partes del sistema. Estos pequeños controladores podrían comunicarse entre ellos y con un procesador central, compartir la información y coordinar sus acciones. Al estar todos los Microcontroladores integrados en un chip, su estructura fundamental y sus características básicas son muy parecidas. Todos deben disponer de los bloques esenciales procesador, memoria de datos y de instrucciones, líneas de E/S, oscilador de reloj y módulos controladores de periféricos. Sin embargo, cada fabricante intenta enfatizar los recursos más idóneos para las aplicaciones a las que se destinan preferentemente.

El presente desarrollo se enfoca en el manejo del Microcontrolador SX48BD con módulo Basic Stamp, presentando 4 capítulos y que estos se subdividen en 9 capítulos, los cuales a continuación se describen:

Da inicio con un programa que envía señales básicas a través de 4 comandos OUTPUT, PAUSE, GOTO, OUT como dispositivo de salida para empezar a familiarizarse con el software.

El tema 2 muestra como actúa el Microcontrolador al darle una instrucción como dispositivo de entrada a través de 5 comandos INPUT, PAUSE, GOTO, BUTTON, IF-THEN, que servirán como base para detectar algunas variables físicas (temperatura, movimiento).

En el tema 3 nos muestra como controlar una variable física (movimiento), a través de un servomotor como dispositivo de salida, para darnos una idea de ejemplo para la aplicación del movimiento de un brazo mecánico de un taladro en el proceso de barrenado.

En el, tema 4 muestra los principios de control basados en una señal analógica a través del CI LM358 configurado como seguidor de tensión por medio de una resistencia variable (Potenciómetro) para mostrarnos un dato binario interpretado por el microcontrolador.

En el tema 5, se interpreta a la electrónica analógica como una analogía de la naturaleza. Ya que existen variaciones en forma continua en la naturaleza, tales como movimiento, nivel de luz, temperatura y sonido; la finalidad es interpretar los datos para procesarlos como señales digitales.

En el tema 6, Se programara al Micro para que envié un conjunto de niveles de tensión binarios a una red resistiva; para lograr que la red efectué la conversión de un dato digital a un dato analógico; esto con la finalidad de cómo interpreta el micro una señal análoga.

En el tema 7 nos da un ejemplo de control basado en señales digitales al detectar una variable física (temperatura), y dar una respuesta con dispositivos de salida analógicos, en el proceso de control se recibe un dato analógico que en este caso es temperatura, se procesa para ser interpretado digitalmente por el Micro y finalmente se da una respuesta con un dato analógico para controlar la velocidad de un ventilador.

En el tema 8 muestra el uso de dispositivos de salida; display de 7 segmentos, comparando el micro con circuitos de logica común; esto nos da un claro ejemplo de capacidad de respuesta del Micro con solo programarlo evitando el uso de CI de logica común.

En el tema 9 se da un ejemplo para la aplicación del Micro SX48BD retomando algunos temas para el control de una bomba de un despachador de combustible en una estación gasolinera.

JUSTIFICACIÓN.

Las industrias de hoy, que piensan en el futuro, se encuentran provistas de modernos dispositivos electrónicos en sus máquinas y procesos de control. Hoy las industrias automatizadas deben proporcionar en sus sistemas, alta confiabilidad, gran eficiencia y flexibilidad dando como resultado productos de calidad. Una de las bases principales de tales industrias es un dispositivo electrónico llamado Controlador Lógico Programable “PLC”. EL PLC es un sistema electrónico operado digitalmente que usa una memoria programable para el almacenamiento interno de instrucciones las cuales implementan funciones específicas tales como lógicas, secuénciales, de temporización, conteo y aritméticas, para controlar a través de módulos de entrada /salida, digitales y analógicas, varios tipos de máquinas y/o procesos.

Hoy los Controladores Programables son diseñados usando lo ultimo en diseño de Microcontroladores y circuiteria electrónica lo cual proporciona una mayor confiabilidad en su operación en aplicaciones industriales donde existen peligro debido al medio ambiente, altas temperaturas, ruido ambiente o eléctrico, suministro de potencia eléctrica no confiable, vibraciones mecánicas, alta repetibilidad de operaciones.

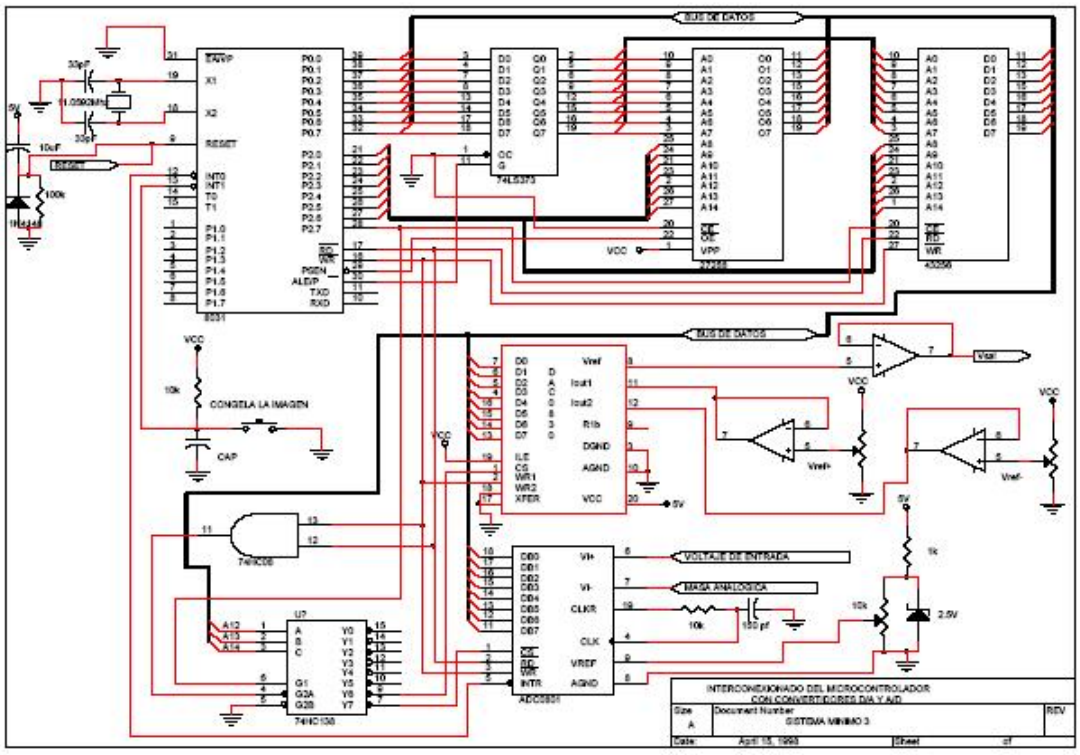
Aunado a esto el presente desarrollo pretende dar a entender como opera el control del PLC por medio de un Microcontrolador; a través de 9 temas con el manejo del Microcontrolador SX48BD con módulo Basic Stamp; aplicándolo en la materia de automatización; para que los alumnos que están por concluir la lic.; en ingeniería Industrial cuenten con una base teórica práctica para la aplicación de sistemas automatizados enfrentando el mundo competitivo de hoy.

PLANTEAMIENTO DEL PROBLEMA.

En el programa curricular del Campús Sahagún, menciona el manejo del Microcontrolador 8051 en un tema, el cual para llegar a entender su funcionamiento, se requiere dominar el lenguaje máquina para su programación, además de armar circuitos auxiliares para interfaz en serie (INTERFASE RS-232C), este tema prácticamente se tendría que ver en un solo semestre; en la actualidad el manejo del PLC en los institutos de educación media y superior es más común, para ofrecer capacitación a los alumnos; en el caso de nuestro campús por el momento no cuenta con un PLC, en su laboratorio; aunado a esto se sugiere el manejo de un microcontrolador que nos ofrezca una idea del control automatizado en corto periodo; realizando una comparación en el mercado el Microcontrolador SX48BD con módulo Basic Stamp, ofrece una excelente ventaja, su programación a través de lenguaje de alto nivel Pbasic que es un lenguaje de programación basado en un Basic estructurado orientado a entrada y salida de señales; el cual cuenta con herramientas de software y hardware para ayudarnos a entender el funcionamiento en cuanto a control de un PLC.

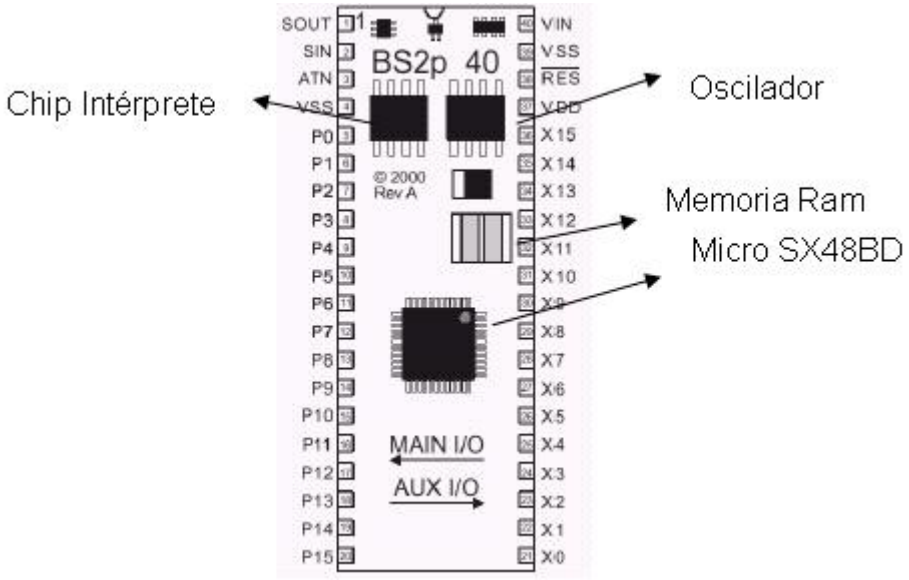
A continuación se da una comparación del Microcontrolador 8051 con el SX48BD

Para la conexión del micro 8051 requiere de circuitos externos, decodificadores, flip flops, para la transmisión de datos en puertos serie y paralelo como se muestra en la figura haciéndolo complejo y tedioso.



El Micro SX48BD con módulo Basic Stamp cuenta con componentes auxiliares integrados en un mismo chip como memoria, chip intérprete, oscilador, entrada para comunicación serial, evitando realizar el conexionado anterior, y lo más importante se iniciaría de inmediato con las prácticas.

Esquema físico del Micro



El Micro 8051 para su programación, se requiere saber del manejo de lenguaje máquina es decir códigos de unos y ceros; a continuación se da un ejemplo de programación de un exhibidor alfanumérico, de cristal líquido de 2 líneas por 16 caracteres, al microcontrolador 8051[1]

1.- Se establece el tipo de interfase a la cual el exhibidor se va a conectar, en este caso, se trata de un microcontrolador con un bus de datos de 8 bits, el cual se conecta directamente. La primera palabra de control que se envía al exhibidor es el número 38H, el cual significa lo siguiente:

Codigo	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
38H	0	0	0	0	1	1	1	0	0	0

Se envía la palabra de control al exhibidor (RS=0 y RW=0), los bits DB5 y DB4 especifican el tamaño del bus, y el bit DB3 el número de líneas del exhibidor. Se espera un lapso de tiempo de 40 ms antes de enviar la siguiente instrucción.

[1] Manual del microcontrolador 8051 Dr. Alejandro Vega Diciembre 1999

2. - Se limpia toda la memoria del exhibidor y se regresa la pantalla del exhibidor a su posición inicial.

Codigo	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
01H	0	0	0	0	0	0	0	0	0	1

Esta instrucción toma un tiempo de 1.64 ms,

```
                PROGRAMA PARA MANEJO DEL EXHIBIDOR ALFANUMÉRICO
        ; ESTE PROGRAMA COMIENZA A PARTIR DE LA LOCALIDAD 4000H POR
        ; SER LA LOCALIDAD DE INICIO DEL EMULADOR, PERO PUEDE COMENZAR
        ; A PARTIR DE CUALQUIER LOCALIDAD.
4000                        ORG 4000H
4000                            ; LAS DIRECCIONES DEL EXHIBIDOR SON
4000                            ; LA 8000H PARA CONTROL DEL EXHIBIDOR
4000                            ; LA 8001H PARA EXHIBICIÓN DEL CARACTER
4000                            ; APUNTADAS POR LOS REGISTROS R0 Y P2.
4000 904075                 EXHIBE: MOV DPTR, #CONTEX; CONTROL DEL
4003 7800                       MOV R0, #00H; EXHIBIDOR
4005 75A050                     MOV P2, #80h
4008 124046                     LCALL XCBDOR
400B                            ; ENVIA LOS CARACTERES DE
                                : CONTROL AL EXHIB
```

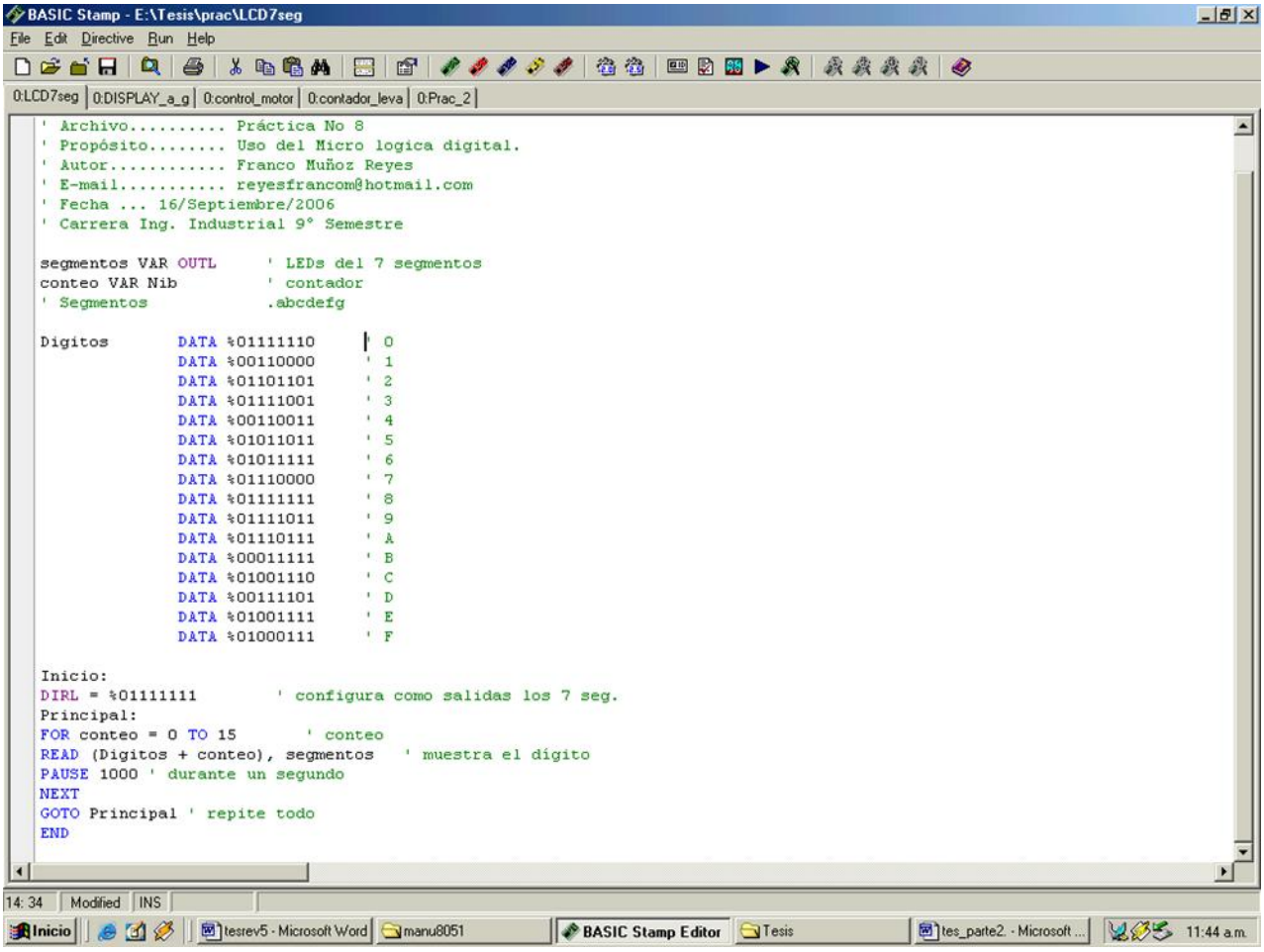
Para no hacerlo tedioso este programa no se presenta por completo; esto es una tercera parte del programa.

El Micro SX48BD[2]. se programa con instrucciones de Pbasic que permiten controlar las líneas de (entrada /salida), realizar temporizaciones, realizar trasmisiones serie asincrónica, programar pantallas LCD, capturar señales analógicas, emitir sonidos, etc. y todo ello en un sencillo entorno de programación en ambiente Windows (98, Mel, xp) que facilita la creación de estructuras condicionales y repetitivas con instrucciones como IF...THEN o FOR...NEXT y la creación de etiquetas de referencia; otra ventaja de los Microcontroladores con módulo Basic Stamp es la facilidad de un puerto abierto de

[2] Diego M. Pulgar G. Manual de programación Basic Stamp Versión 1.1

(entrada / salida), que da la capacidad de evaluación de señales para luego decidir una acción y poder controlar dispositivos externos esto hace que el Microcontrolador sea el cerebro de los equipos.

Ejemplo para la programación de un display de 7 Segmentos.



Por estas razones se propone sustituir el SX48BD con módulo de Basic Stamp por el 8051, dando mejores alternativas en el desarrollo de la materia Automatización.

OBJETIVO GENERAL:

En la actualidad la mayoría de los productos que interactuamos en casa o trabajo, como televisores, en sistemas de iluminación, en los automóviles, en los procesos de producción en las empresas, etc; cuentan en su sistema de control con un microcontrolador donde mucha de las veces se requiere reprogramar y/o detectar fallas en su sistema. Al término del siguiente desarrollo los alumnos de la materia de automatización serán capaces de programar al Microcontrolador SX48BD con módulo Basic Stamp, adquiriendo datos de monitoreo y control, a través de 9 temas; que servirá como base para interpretar la operación de más microcontroladores incluyendo un PLC.

OBJETIVOS ESPECÍFICOS:

- El manejo por parte de los alumnos de Ingeniería Industrial del lenguaje de alto nivel (PBasic), para la implementación en software y hardware de un sistema de monitoreo y control a través del Microcontrolador SX48BD.
- Adquirir los conocimientos básicos en el uso y manejo del Microcontrolador SX48BD para monitorear y/o controlar algunas variables físicas (movimiento, temperatura, luminosidad,), con el fin de implementar un sistema automatizado en el SX48BD.
- Elaborar 9 prácticas, manejando el software y hardware del SX48BD para aplicarlas en un sistema de monitoreo y control en una estación gasolinera.

METODOLOGÍA.

El presente desarrollo es de tipo experimental ya que esta dirigido a responder las causas de los eventos físicos, proporcionando un sentido de entendimiento del fenómeno al que se hace referencia[3]. Las investigaciones experimentales son más estructurales que las demás clases de estudios y de hecho implican estudios de tipo descriptivo y correlación.

Se requiere aplicar sistemas externos al Microcontrolador como (contadores, temporizadores, flip flops, etc.); para realizar dichas tareas que se le asignan al microcontrolador mediante las instrucciones que se le dan a través del programa y observar que variables son las que interfieren directamente.

[3] Sampieri Hernández Roberto 1991

Capítulo 1 Antecedentes históricos

Evolución e historia de los Microprocesadores.

Dentro de los avances de la electrónica esta el año de 1970 como la fecha de invención del microprocesador. Con el microprocesador se inició una nueva era de desarrollo de la industria de las computadoras y de la electrónica, la cual hasta el presente ha ido evolucionando con una velocidad que aún sorprende, nadie en esa época se imaginaba el impacto tan grande que causaría este desarrollo en la vida del hombre moderno. Sin duda la senda la abrió el invento del tubo de vacío por Lee De Forest a comienzos del siglo XIX, basado en el descubrimiento de un fenómeno llamado "Efecto Edison". Este dispositivo hizo posible la radio, la telefonía inalámbrica, etc.; e impulsó el desarrollo comercial e industrial de la electrónica. Inclusive las primeras computadoras eran fabricadas con tubos de vacío. Luego vino la revolución del transistor, desarrollado en los laboratorios de Bell Telephone en 1948 y utilizado a partir de 1950 a escala industrial, con su inclusión en la fabricación de todo tipo de aparatos como: radio, televisión, sonido, computadoras, en la industria Militar, en la industria espacial, etc. El concepto de circuito integrado, empezó a darse de quienes trabajaban en el diseño y fabricación de transistores. El planteamiento fue más o menos el siguiente: si se fabrican transistores en forma individual y luego se tenían que unir siempre de la misma forma entre sí con alambres y con otros componentes, ¿Por qué no fabricar de una vez todo el conjunto de material semiconductor y aislante, interconectado internamente para que cumpliera la misma función del sistema total?, este planteamiento fue desarrollado en la práctica simultáneamente, pero en forma independiente, por dos empresas muy importantes en la historia de la electrónica. Fairchild semiconductor y Texas Instruments.

El diseño del microprocesador se inició en un grupo de trabajo de Intel dirigido por ETD Hoff, un brillante ingeniero egresado de la Universidad de Stanford. Todo empezó cuando Intel firmo un contrato con una compañía japonesa (Busicom Corporation) fabricante de calculadoras. Esta quería que se le fabricara un conjunto de circuitos integrados que reemplazan la gran cantidad de componentes que tenían las calculadoras de ese entonces.

Después de un largo trabajo, se llegó hasta lograr que todo el circuito fuera reemplazado por tres chips, pero estos resultaron ser de un tamaño mayor de acuerdo a los

requerimientos. A Hoff se le ocurrió que debía agrupar toda la parte del proceso aritmético y lógico en un sólo circuito y el resto de la calculadora en los otros dos circuitos. Con la intervención de otro diseñador, Federico Faggin, el proyecto se llevó a cabo con todo éxito. A este circuito, de 2250 elementos integrados en un área de 3 x 4 milímetros, se le llamó microprocesador. También se le dio el nombre de CPU (Central Processing Unit) o MPU (Micro Processing Unit). Aunque este circuito tenía ya muchas de las características de una unidad central de proceso integrada, el primer microprocesador en un solo chip, fabricado como tal, fue el 4004 de Intel, diseñado para reemplazar grandes cantidades de circuitos integrados TTL. El 4004 era un chip muy sencillo que manipulaba datos de cuatro bits. Intel desarrolló muy pronto, en 1972, el 8008, el cual podía procesar datos de ocho bits, pero era muy lento. Para remediar esto, Intel desarrolló un sustituto, el 8080, y posteriormente el 8085 compatible con el primero, con funciones adicionales y menos circuitos de soporte. Un equipo de diseñadores que antes había trabajado para Intel en el 8080 formó la Zilog Inc, y construyó el microprocesador Z-80, el cual incorporaba un conjunto de instrucciones más extensos que el 8080, aunque era compatible con este último. Este microprocesador ha sido uno de los más utilizados en el campo de control. A partir de ese momento, se estableció una guerra técnica y comercial, entre Intel y Motorola, la cual los ha llevado a ser los dos grandes líderes indiscutibles del mercado de microprocesadores. Pero el desarrollo del microprocesador no se quedó ahí. A principios de la década de los 80 empezaron a aparecer los microprocesadores de 16 bits, mucho más potentes. El primero en salir al mercado fue el 8086 de Intel en 1978, el cual fue adoptado por la IBM para la fabricación de su famosa IBM PC. Lo siguieron de cerca el 68000, el 68020, el 68030 y el 68040 de Motorola. Con estos microprocesadores se inició en Apple una nueva familia de microcomputadores: la Macintosh. Luego, en un consorcio entre Apple, IBM y Motorola se desarrolló una nueva familia de microprocesadores: Los Power PC, los cuales se utilizaron en las computadoras Apple e IBM.

Por otra parte en los años 60’s las industrias automotrices usaban sistemas industriales basadas en reveladores, en sus sistemas de manufactura; los cuales eran complejos, esto arrojaba dificultades en cuanto a la instalación de los mismos, los altos costos de los equipos, altos costos de operación, mantenimiento, la poca flexibilidad y confiabilidad de los equipos. Buscando reducir los costos de los sistemas de control por relevadores, General Motor preparo en 1968 ciertas especificaciones detallando un controlador autómata o "Controlador Lógico Programable". Estas especificaciones definían un sistema

de control por relevadores que podían ser asociado no solamente a la industria automotriz, si no prácticamente a cualquier industria de manufactura.
Los PLC's surgen como equipos electrónicos sustitutos de los sistemas de control basados en relevadores, Los primeros PLC's se usaron solamente como reemplazo de relevadores, es decir, su capacidad se reducía exclusivamente al control On −Off (de dos posiciones) en máquinas y procesos industriales. La gran diferencia con los controles por relevador fue su facilidad de instalación, ocupando menor espacio, menor costo, y proporcionan autodiagnósticos sencillos.

Con el avance de la tecnología electrónica antes mencionada cuando surgen los microprocesadores Z-80, 8085, 6800**,** los cuales se agregan a los PLC's, como parte fundamental de control, dando facilidad e inteligencia adicional generando un gran avance y permitiendo un notorio incremento en la capacidad de interfase con el operador; sin embargo se requería de grandes conocimientos de electrónica ya que las memorias y puertos eran componentes exteriores al microprocesador y su manejo requería sofisticadas técnicas de programación en lenguaje de máquina códigos de "unos" y "ceros" ; en los comienzos de los 80's empezaron a aparecer los microcontroladores que tenían toda la estructura exterior del microcomputador incluido en un mismo chip.

La diferencia que existe entre un microprocesador y un Microcontrolador es que un microcontrolador es un circuito integrado de alta integración que incorpora la mayor parte de los elementos que configuran un microprocesador.

El microprocesador es un circuito integrado que contiene la Unidad Central de Proceso (UCP), también llamada procesador, de un computador. La UCP está formada por la Unidad de Control, que interpreta las instrucciones, y el camino de datos, que las ejecuta. Un microprocesador saca al exterior las líneas de sus buses de direcciones, datos y control, para permitir conectarle con la Memoria y los Módulos de E/S y configurar un computador implementado por varios circuitos integrados. Se dice que un microprocesador es un sistema abierto porque su configuración es variable de acuerdo con la aplicación a la que se destine.

El microcontrolador es un sistema cerrado. Todas las partes del computador están contenidas en su interior y sólo salen al exterior las líneas que gobiernan los periféricos.

Si sólo se dispusiese de un modelo de microcontrolador, éste debería tener muy potenciados todos sus recursos para poderse adaptar a las exigencias de las diferentes aplicaciones, sin embargo esto generaría un costo elevado además que no se aprovecharía todo su potencial en algunas aplicaciones. Gracias a esto los fabricantes de microcontroladores ofertan un elevado número de modelos diferentes, desde los más sencillos hasta los más poderosos.

Es posible seleccionar la capacidad de las memorias, el número de líneas de E/S, la cantidad y potencia de los elementos auxiliares, la velocidad de funcionamiento, etc. Por todo ello, un aspecto muy destacado del diseño es la selección del microcontrolador a utilizar.

Es preciso resaltar en este punto que existen innumerables familias de microcontroladores, cada una de las cuales posee un gran número de variantes.

Arquitectura Básica. Aunque inicialmente todos los microcontroladores adoptaron la arquitectura clásica de Von Neumann, en el momento presente se impone la arquitectura Harvard. La arquitectura de Von Neumann se caracteriza por disponer de una sola memoria principal donde se almacenan datos e instrucciones de forma indistinta. A dicha memoria se accede a través de un sistema de buses único (direcciones, datos y control).
La arquitectura Harvard dispone de dos memorias independientes una, que contiene sólo instrucciones y otra, sólo datos. Ambas disponen de sus respectivos sistemas de buses de acceso y es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias.

El procesador o UCP

Es el elemento más importante del microcontrolador y determina sus principales características, tanto a nivel hardware como software. Se encarga de direccionar la memoria de instrucciones, recibir el código de operación de la instrucción en curso, su decodificación y la ejecución de la operación que implica la instrucción, así como la búsqueda de los operándoos y el almacenamiento del resultado[4].
Gracias a los microcontroladores y a su avance; los PLC's se han convertido en sistemas flexibles y versátiles.

[4] José Ma. Angulo Usategui, Susana Romero Yesa E Ignacio Angulo Martínez. 2000

Capítulo 2. Estructura Básica Microcontrolador SX48BD con módulo Basic Stamp

No. DE PRÁCTICA: 1 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Programar y manejar el software y hardware del Micro SX48BD con módulo Basic Stamp mediante cuatro comandos OUTPUT, PAUSE, GOTO, OUT, como actúa el Microcontrolador al darle una instrucción como dispositivo de salida.

INTRODUCCIÓN:

Microcontrolador. Es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales Fig. 1.1, los cuales cumplen una tarea específica; son diseñados principalmente para aplicación de control de máquinas, mas que para interactuar con humanos, una vez dadas las instrucciones especificas para una tarea a realizar, es indispensable reprogramar para cambiarla; estas son básicamente algunas de sus partes:

- Procesador o UCP (Unidad Central de Proceso).
- Memoria RAM para contener los datos.
- Memoria para el programa tipo ROM/PROM/EPROM/EEPROM & FLASH.
- Líneas de (entrada / salida) para comunicarse con el exterior.
- Diversos módulos para el control de periféricos (temporizadores, Puertos Serie y Paralelo, A/D y D/A, etc.).
- Generador de pulsos de reloj que sincronizan el funcionamiento de todo el sistema.

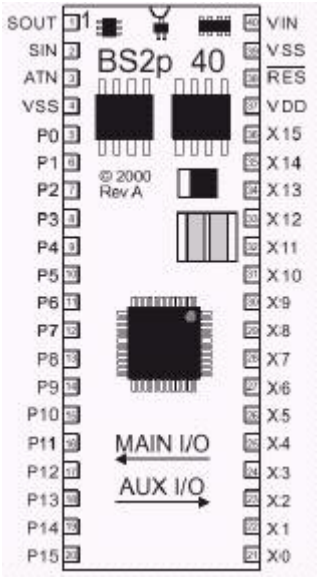


Figura 1.1 Estructura del Módulo Basic Stamp II

El Microcontrolador SX48BD esta programado permanentemente de fábrica con un conjunto de instrucciones predefinidas en lenguaje Basic. Cuando el SX48BD se programa, se le está diciendo que salve las instrucciones compiladas, llamadas fichas de instrucciones hexadecimales, en la memoria EEPROM (24LC16B). Cuando ejecuta el programa, el SX48BD extrae las fichas de instrucciones hexadecimales de la memoria, los interpreta como instrucciones Pbasic, y ejecuta las instrucciones equivalentes.

El SX48BD tiene 48 pines en total vea Fig. 1.2, 16 pines están destinados a entrada / salida (I/O) desde P0 hasta P15, se pueden interconectar con toda la lógica de +5 voltios moderna, de TTL (lógica del transistor-transistor) con CMOS (semiconductor de óxido metálico) que pueden ser conectados directamente a dispositivos digitales o de niveles lógicos, tales como switch N/A o N/C, Leds, altavoces, potenciómetros, y registros de desplazamiento, etc. Además, con unos pocos componentes extras como flip flops, compuertas lógicas, frecuenciometros, etc., estos pines de I/O pueden ser conectados a dispositivos para que ejecuten lo establecido en el programa tales como solenoides, servomotores, y otros dispositivos de alta corriente o potencia a través de interfaces; 16 pines están destinados como auxiliares desde X0 hasta X15 que se pueden configurar como I/O, 4 están destinados a la comunicación serial RS-232, dos se pueden también

utilizar para la comunicación serial asincrónica. Dos más se utilizan para interconectar la memoria EEPROM, 4 para alimentación, 4 para tierra[5].

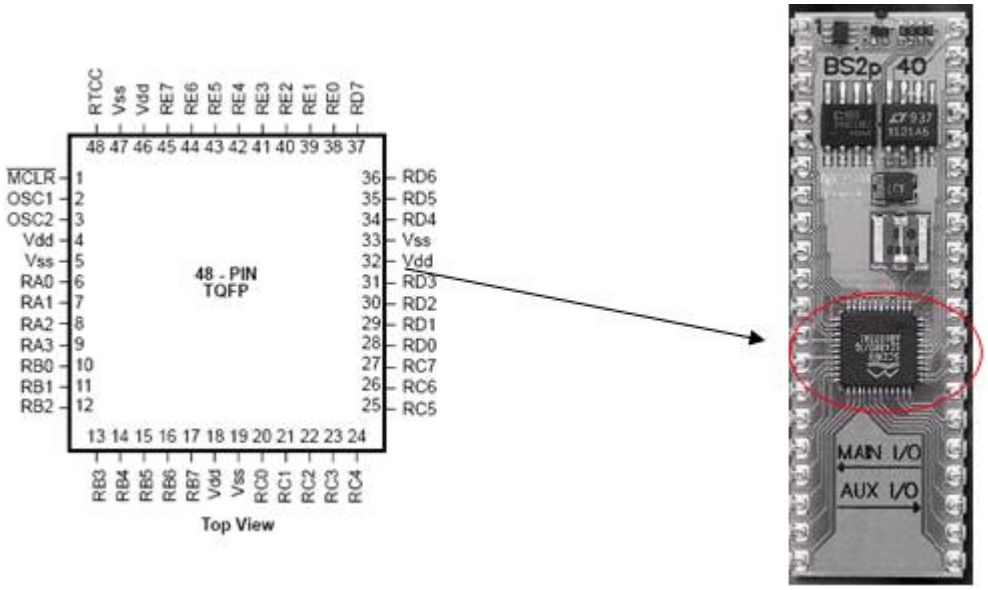


Figura 1.2 Representación esquemático y física del SX48BD

Cualquier sistema micro controlado, consiste en dos componentes primarios: hardware y software (circuito y programa).

El hardware es el componente físico del sistema Fig. 1.2.

El software es la lista de instrucciones que residen dentro del hardware vea ejemplo 1.1
Se requiere crear parte del hardware para el control como (temporizadores, censores), para luego escribir el programa de software y obtener el "control" de las variables asignadas.

Los pines que se usaran en éste experimento son los siguientes vea figura 1.3:

Pin		Nombre de señal
5	Corresponde a	P0
6		P1
37		Vdd (+5 volts)

[5] Diego M. Pulgar G

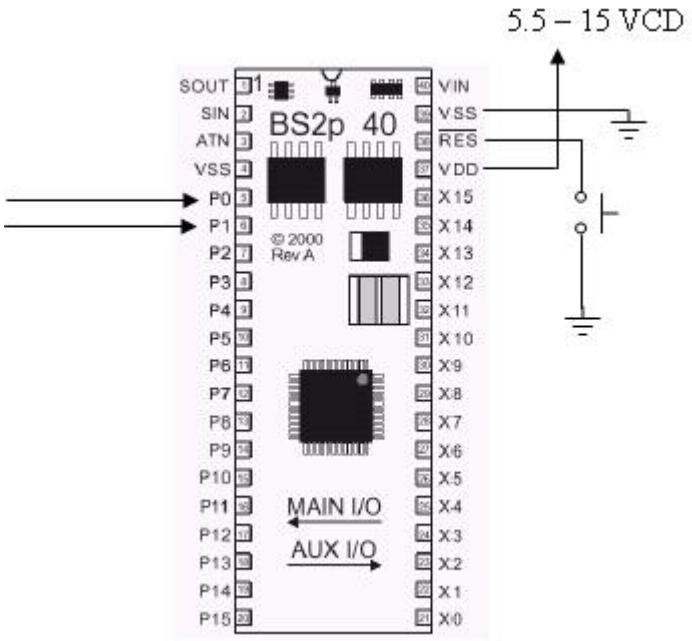


Figura 1.3 Designación de Pines en el módulo

Cuando se programa al Micro, se refiere al nombre de señal, más que al verdadero número de pin es decir P0−P15.

MARCO TEÓRICO:

Los programas se escriben en un lenguaje de programación llamado PBasic, una versión específica del lenguaje de programación Basic “Beginners All-purpose Symbolic Instruction Code” (Código de Instrucciones Simbólicas Multi-propósito para Principiantes) desarrollada por Parallax. El Basic es muy popular debido a su simplicidad y a su sintaxis similar al Inglés.

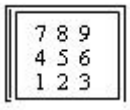
El lenguaje Pbasic para el SX48BD esta conformado por 37 comandos, 24 funciones matemáticas. Instrucciones para definiciones de variables, constantes y etiquetas de

referencia. La combinación de estos comandos con las referencias de direcciones (etiquetas), conformara el programa de Pbasic[6].

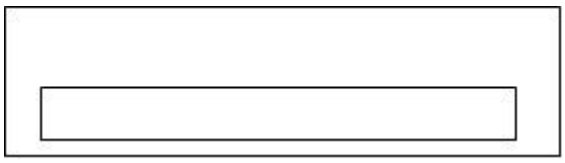
Un programa puede ser tan simple como una lista de instrucciones. Algo así: En la compra de galletas o refrescos en una máquina que ofrece estos artículos.

- Depositar las monedas de acuerdo al precio del producto.

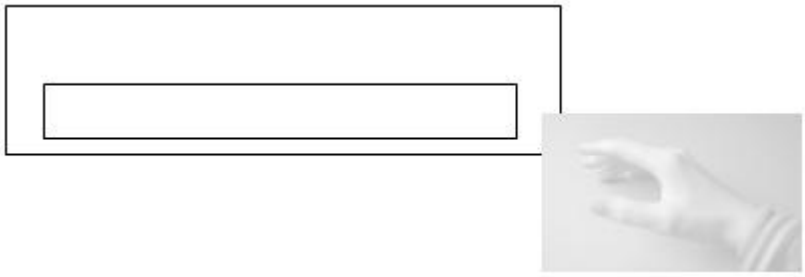
- Ingresar el código del producto elegido



- Esperar que suelte el producto



- Tomar el producto del depósito



- Fin

El método de programación es lineal, es decir se ejecuta un comando a la vez, por lo general se programa de una forma en que se repita las instrucciones en un ciclo cerrado. Los programas de Pbasic contienen: variables de memoria, constantes, direccionamiento de puertos, instrucciones y sub-rutinas. El flujo del programa es a menudo redirigido con saltos, bucles y subrutinas, intercalando con pequeñas secciones lineales. Los requerimientos del flujo del programa son determinados por el objetivo del programa y las condiciones bajo las cuales se ejecutará.

[6] Manual de programación Basic Stamp Versión 1.1

Saltos – Redireccionando el Flujo de un Programa

Un comando de salto es el que causa que el flujo del programa se desvíe de su trayectoria lineal. En otras palabras, cuando el programa encuentra un comando de salto, no ejecutará en la mayoría de los casos, el siguiente código de línea. El programa normalmente, se dirigirá a otro sector. Hay dos categorías de comandos de salto: incondicional y condicional. El PBasic tiene dos comandos, GOTO y GOSUB que realizan saltos incondicionales.

Este es un ejemplo de salto incondicional usando GOTO: Aplicamos el procedimiento anterior cuando requerimos más de dos productos o el producto se ha agotado.

Inicio:

- Depositar las monedas de acuerdo al precio del producto.
- Ingresar el código del producto elegido
- Esperar que suelte el producto
- Tomar el producto del depósito

  GOTO Inicio Regresa a inicio

Se le llama salto incondicional debido a que siempre se ejecuta. GOTO redirige el programa hacia otro lugar en este caso a Inicio. La ubicación se especifica como parte del comando GOTO Inicio y se denomina dirección. Hay que tener presente que las direcciones están al principio de una línea de código y son seguidas por dos puntos (:). Se verá frecuentemente a GOTO al final del cuerpo principal del código, haciendo que las instrucciones del programa se ejecuten nuevamente. Un uso común para GOTO es crear bucles infinitos o cerrados; programas que repiten un grupo de instrucciones.

El salto condicional hará que el flujo del programa se modifique bajo circunstancias específicas. El salto condicional más simple se logra con la función IF-THEN. La función de PBASIC IF-THEN es diferente al programa de Basic original. En PBASIC, THEN siempre es seguido por una dirección de programa (etiqueta), mientras que en el Basic estándar se permite colocar instrucciones a continuación del THEN. Si la condición que se

evalúa es verdadera, el programa saltará a la dirección especificada. Caso contrario, continuará en la siguiente línea de código después del IF... THEN.

Las comparaciones lógicas se efectúan basándose en los operadores los cuales son:

Operador	Descripción
=	Igual
<>	No igual
<	Menor
>	Mayor
<=	Menor o igual
>=	Mayor o igual

Las comparaciones se escriben de la siguiente manera: 0 < 5, los valores a comparar pueden ser entre una variable y una constante, o entre dos variables. El siguiente ejemplo nos muestra una comparación con la sentencia IF... THEN**:**

Ejemplo 1.1

```
Inicio:
IF IN0=0 THEN Enciende_Led      Si P0=0 envía un estado lógico de 0 V, a P1
                                Ve a etiqueta Enciende_Led
GOTO Inicio                     Si no regresa a inicio de programa
Enciende_Led:
High 1                          Envía un estado lógico de +5 V, a P1
PAUSE 1000                      Espera 1 segundo
GOTO Inicio                     Regresa a inicio de programa
Fin
```

Se ejecutan las instrucciones y se verifica la condición. Si se evalúa como verdadera, el programa saltará a Enciende_Led. Si la condición se evalúa como falsa, el programa continuará por la línea que se encuentre a continuación de la instrucción IF-THEN.

A medida que los programas se vuelven más complicados, se podría necesitar que el programa salte a un gran número de direcciones, dependiendo de cierta condición. La primera aproximación es usar múltiples instrucciones IF-THEN, aunque existe un comando especial, BRANCH, que permite que un programa salte a distintas direcciones dependiendo del valor de una variable. Esto es muy útil debido a que las condiciones que a menudo proceden de una variable de control. BRANCH es un poco más sofisticado en su funcionamiento, pero muy poderoso debido a que puede reemplazar múltiples instrucciones IF-THEN. BRANCH requiere una variable de control y una lista de direcciones o etiquetas.

En el caso de una única variable de control, el listado anterior puede ser reemplazado por una línea de código:

BRANCH Índice, [Etiqueta0, Etiqueta1,... EtiquetaN].

La ejecución comienza en la etiqueta especificada. Por ejemplo, si (índice) vale 0, el programa salta a la primera etiqueta especificada en la lista, si (índice) es 1, salta a la segunda y así sucesivamente. Si (índice) es mayor ó igual al número de etiquetas, no se toma ninguna acción y la ejecución continúa con la declaración siguiente al BRANCH. Se pueden usar hasta 256 etiquetas en una instrucción BRANCH[7].

Los bucles condicionales se ejecutarán bajo ciertas circunstancias. La programación condicional es la que actúa para que el microcontrolador tome decisiones establecidas dentro del programa y controle las variables físicas por medio de dispositivos eléctricos, mecánicos a través de su mando.

[7] Manual de programación Basic Stamp Versión 1.1

MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD	con módulo de Basic Stamp.
1	Cable serial RS-232	
1	Protoboard	
2	Leds rojos	
2	Resistores de 470Ω, ¼ watt	Vea apéndice A
−	Cable telefónico	
−	Caimanes	
1	Interfaz transistor	
1	PC con Windows 98, Me, Xp	

MEDIDAS DE SEGURIDAD:

Hay dos cosas muy importantes de recordar cuando se conectan Leds al SX48BD. Lo primero es que siempre se debe asegurar que haya una resistencia conectada, como muestra la figura 1.4 En este experimento el resistor debe tener un valor de 470Ω, ¼ watt. Segundo, esté seguro que la polaridad del Led es la correcta. Hay una zona lisa (Cátodo) en un costado del Led que debería ser conectada como en la Figura 1.5. Si la polaridad es invertida, el Led no encenderá. El lado liso también tiene la pata más corta del Led.
Utilice la interfaz con transistor Fig. 1.6, para interactuar el Micro con una señal externa que requiere de más potencia o voltaje mayor.

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme el circuito que se muestra en la Fig. 1.4 En la Protoboard de acuerdo al esquema que se presenta.

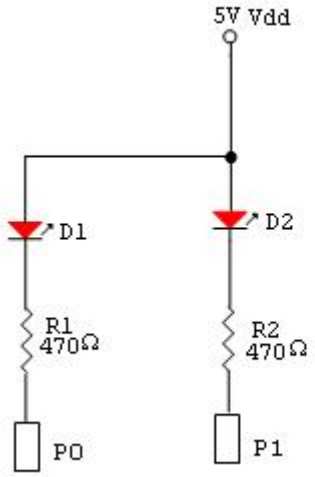


Fig. 1.4a Esquema Eléctrico

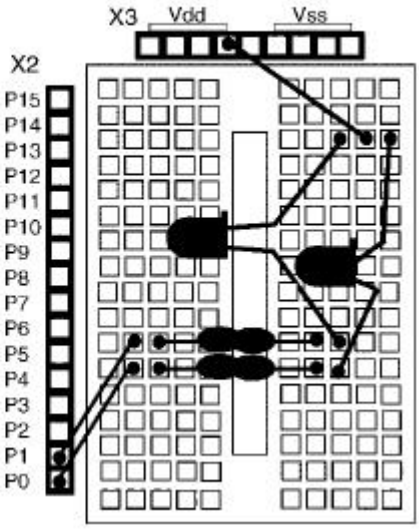


Fig. 1.4b Esquema Físico

Conexión de componentes. (Leds y resistores).

Debe cerciorarse que en la entrada del Microcontrolador Puerto P0 y P1 coloque el resistor de 470 Ω al Cátodo del Led como se muestra en la Fig. 1.5

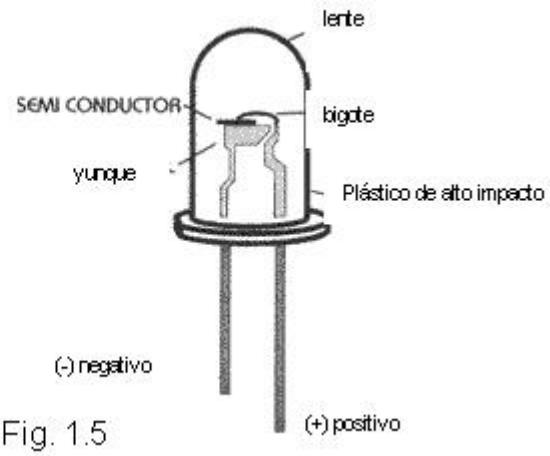


Fig. 1.5

Interfaz con transistor Tip41c, interfaz con Triac

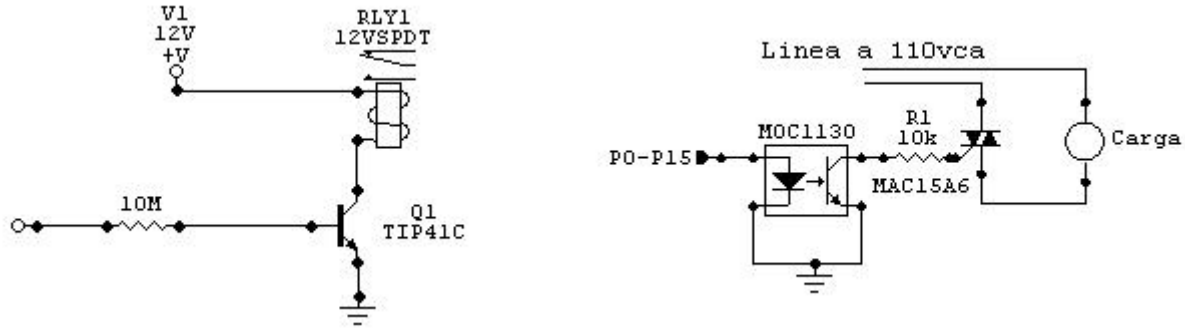
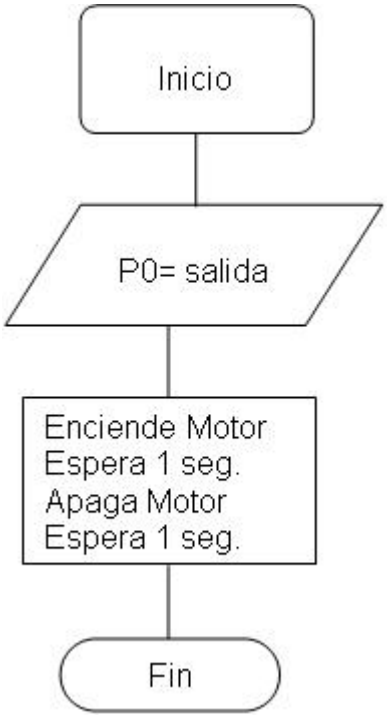


Fig. 1.6

Hasta el momento se ha trabajado para armar el hardware ahora se pasara a la segunda parte, el software.

Las salidas de un microcontrolador pueden ser usadas para controlar el estado de dispositivos de campo de salida. Los dispositivos de salida son aquellos que realizan el trabajo en una aplicación de control de proceso. Ellos entregan la energía al proceso bajo control. Algunos ejemplos incluyen motores, calentadores, solenoides, válvulas y lámparas. La salida de baja potencia del SX48BD (o cualquier otro microcontrolador) no le permite proveer la energía requerida por estas cargas. Con un acondicionamiento de señal apropiado, el Micro puede controlar transistores de potencia, triacs y relevadores Fig. 1.6. Estos son los dispositivos que pueden entregar las tensiones y corrientes que demandan los dispositivos de campo. En algunas aplicaciones, puede usar la salida del SX48BD para comunicarse con otro microcontrolador o circuito electrónico. Podría haber problemas de compatibilidad entre diferentes familias lógicas, fuentes de alimentación separadas, o masas no comunes que requieren consideraciones es de suma importancia tener esto presente.

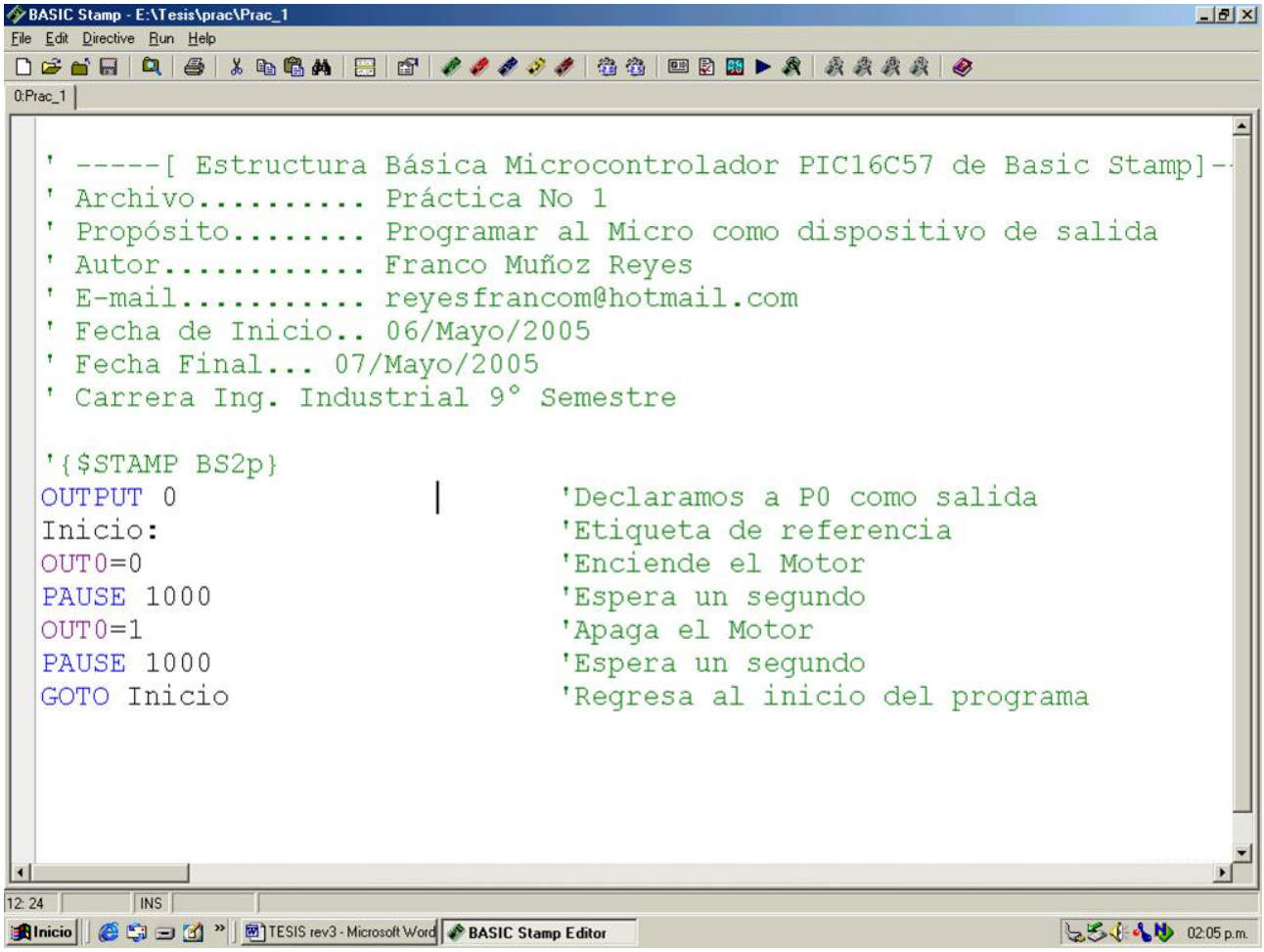
Como primer experimento simularemos a través del Led un motor de corriente alterna para un ventilador con diferentes periodos de encendido y apagado como dispositivo de salida; transcriba y analice el diagrama de flujo correspondiente seguido del programa.



SX48BD

```
' -----[ Estructura Básica Microcontrolador SX48BD de Basic Stamp]---------
' Archivo..... Práctica No 1
' Propósito..... Programar al Micro como dispositivo de salida
' Carrera Ing. Industrial

'{$STAMP BS2p}
OUTPUT 0      u          'Declaramos a P0 como salida
Inicio:                  'Etiqueta de referencia
OUT0=0                   'Enciende el Motor
PAUSE 1000               'Espera un segundo
OUT0=1                   'Apaga el Motor
PAUSE 1000               'Espera un segundos
END                      'Fin
```



Conecte él módulo Basic Stamp a su PC.

- Conecte el cable serial SR-232 de la plaqueta del módulo.
- Conecte la fuente del módulo a 127 CCA.
- Presiona ctrl. + T para verificar si tiene errores en el programa.
- Presione ctrl. + R para correr el programa.

Descripción del Programa

El primer comando usado es “OUTPUT”. Cada señal desde P0 hasta P15 puede ser ajustada como “entrada” o “salida”. La mayoría de los comandos de salidas de Micro, direcciona el puerto automáticamente como salida, por ejemplo los comandos PULSOUT y SEROUT. Direccionando la variable DIRS=$ffff establecemos el puerto completo como salida. O utilizando un pin especifico como por ejemplo DIR8=1, establece al pin 8 como salida. El comando OUTPUT 8, es equivalente a: DIR8=1. Cuando se utiliza el comando OUTPUT se establece automáticamente el pin como salida.

La siguiente línea en el programa “Inicio:”. Es sólo una etiqueta, una marca en cierto punto del programa. Ahora, el pin 5 o P0, es una salida. En el micro, el voltaje en éstos pines, pueden ser tanto “altos” o “bajos”, que significa que puede tener un voltaje de + 5 volts “alto“, o 0 volts “bajo“. Otra forma de referirnos a alto y bajo es “1 y 0” binarios.

Si se requiere encender el Led es necesario hacer que P0 vaya a nivel bajo (que tenga un 0). P0 está actuando como un interruptor, que puede ser cambiado a encendido o apagado bajo un control de programa.

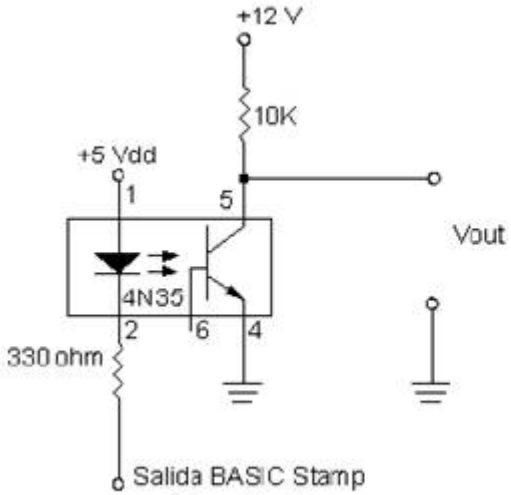
El propósito para el segundo comando: “Out0=0”, causará que P0 vaya a nivel bajo, lo que hace que el Led se encienda. Hay que tener en cuenta que los microcontroladores ejecutan su programa muy rápidamente. En realidad, el SX48BD ejecutará alrededor de 4000 instrucciones por segundo, esto es imposible de visualizar, por lo tanto; si se apagara el Led en el siguiente comando, esto pasaría demasiado rápido para que se pudiera ver. Entonces se necesita retardar el programa, de ésta forma se podrá ver si está operando correctamente o no, y se hace a través del siguiente comando: “Pause 1000”. Este comando hace que el programa espere por 1000 milisegundos, o sea 1 segundo.
El siguiente comando es”out0=1”. Este comando hace que P0 vaya a nivel alto y apague el Led, debido a que no hay flujo de corriente. A continuación hacemos una pausa con “pause 1000” (un segundo). El Led está aún apagado.

Preguntas y desafío:

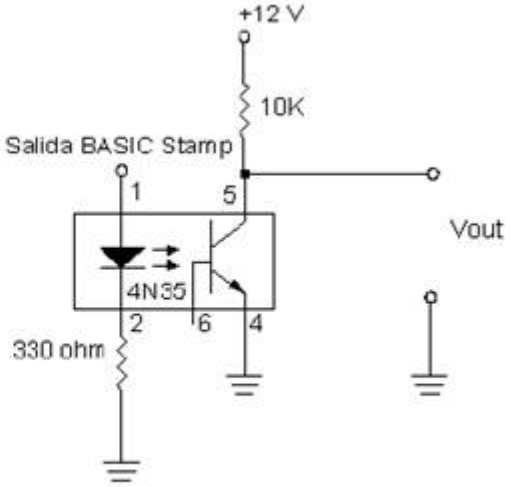
1. Muestre varias posibilidades de interconexión eléctrica que podría necesitar para manejar mayor potencia para el Microcontrolador.

2. Estudie los tres circuitos que se muestran abajo. Escriba qué nivel lógico que utilizaría en la salida del Micro para mantener una tensión de salida (Out) de 12 Volts.

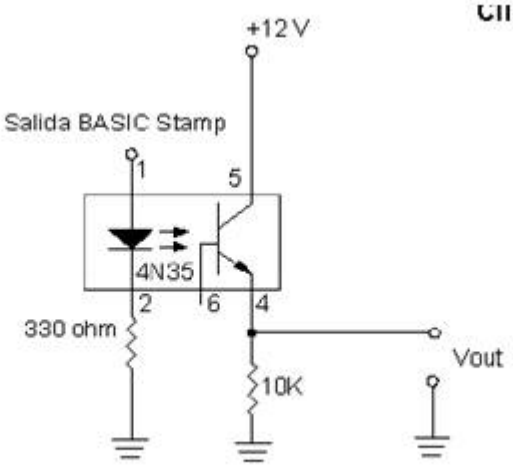
Circuito A _____
Circuito B _____
Circuito C _____



Circuito A



Circuito B



Circuito C

3. Haga que ambos Led se enciendan y apaguen parpadeando al mismo tiempo.

4. Haga encender y apagar los Leds alternativamente; en otras palabras, mientras un Led está encendido el otro está apagado, y viceversa.

5. Encienda el primer Led por 2 segundos, luego apáguelo. Espere 5 segundos y encienda el segundo Led por 1 segundo y luego apáguelo. Espere 3 segundos y repita.

6. Encienda el primer Led por 1.5 segundos, luego apáguelo. Espere 2 segundos y luego encienda el segundo Led por 1.5 segundos, Luego apáguelo. Espere 2 segundos, luego encienda ambos Led por 0.5 segundos y apáguelos por 2 segundos. Repita ésta última acción de 0.5 segundos encendidos y 2 segundos apagados.

Capítulo 2.1 "Adquisición de una señal digital al Microcontrolador."

No. DE PRÁCTICA: 2 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Programar y manejar el software y hardware del Micro SX48BD de Basic Stamp al darle una instrucción como dispositivo de entrada, mediante los comandos INPUT, PAUSE, GOTO, BUTTON, IF-THEN.

INTRODUCCIÓN:

El control de procesos depende de la obtención de información de entrada, su evaluación y la ejecución de la acción correspondiente. En control industrial, la mayoría de las veces la información de entrada involucra el monitoreo de dispositivos de campo que admiten dos estados posibles. Un interruptor es un ejemplo común de dispositivo de dos estados; o está abierto o cerrado. Los interruptores pueden controlar una operación de tres formas.

- ✓ Si se conecta directamente una carga al interruptor, controlando toda la corriente y la tensión de la carga.
- ✓ Conectarlo a la entrada de un relevador. En este caso, el interruptor controla con poca potencia el circuito de entrada del relevador, mientras que la potencia es controlada por el circuito de salida.
- ✓ El estado encendido / apagado (on/off) del interruptor también podría proveer una señal digital a la entrada de un controlador programable.

A continuación se da una representación esquemática de varios Interruptores Industriales

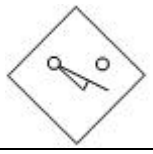
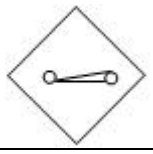
	Pulsador	Limite Mecánico	Interruptor de Proximidad	Relevador
Normalmente Abierto				
Normalmente cerrado				

Fig. 2.1

Los pines de entrada del Microcontrolador no detectan “cambios de resistencia” entre los contactos del interruptor. Estas entradas esperan niveles de tensión apropiados que representen un estado lógico alto o bajo. Idealmente, estas tensiones deberían ser +5 Volts para un nivel lógico alto (1) y 0 Volts para un nivel lógico bajo (0) para convertir los dos estados resistivos del interruptor en entradas aceptables, se coloca en serie un resistor de 10KΩ conectado a la fuente de tensión de +5 Volts del Microcontrolador. Esto forma un circuito divisor de tensión en el cual el estado resistivo del interruptor se compara con el valor del resistor de referencia. La Figura 2.2 muestra dos posibilidades de conexión del interruptor pulsador N.O. (normalmente abierto).

La Figura 2.2a entregará +5 Volts en el pin de entrada cuando es presionado. Cuando el interruptor está abierto, no tiene continuidad; por lo que no circula corriente a través del resistor de 10KΩ y el pin de entrada queda conectado a tierra.

En la Figura 2.2b, al cerrar el interruptor se pone a 0 volts el pin de entrada, es un nivel lógico bajo. Cuando el interruptor se abre, no hay caída de tensión sobre el resistor de 10KΩ y la tensión en la entrada es +5, un nivel lógico alto. Los circuitos son esencialmente iguales, aunque los resultados obtenidos al operar sobre el interruptor son opuestos.

Desde el punto de vista de la programación, es importante saber qué configuración se está utilizando, para saber los estados lógicos uno o cero que entregara a la salida.

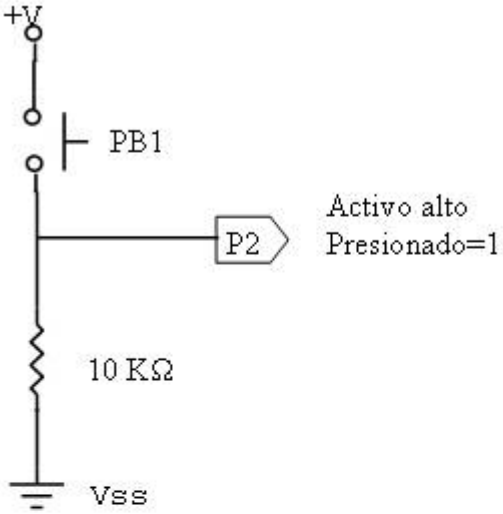


Figura 2.2 a

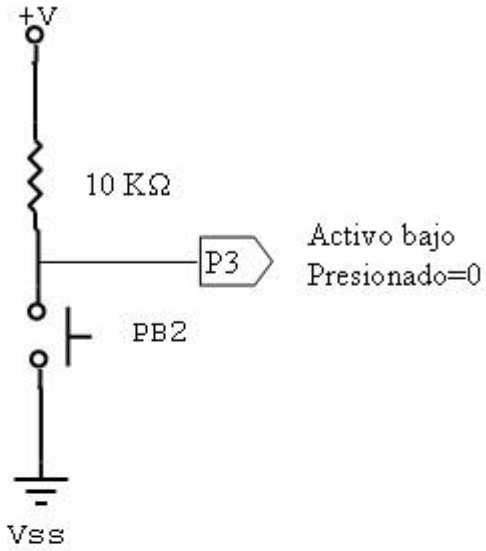
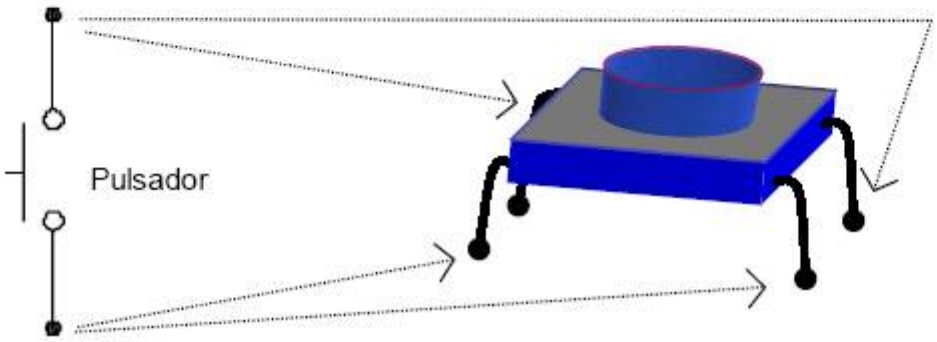


Figura 2.2 b



MARCO TEÓRICO:

Rebote de Interruptores y Rutinas Anti-Rebote.

Cuando se desea presionar rápidamente un botón para lograr que algo suceda solamente una vez surgen dos inconvenientes.
El primero es: Qué tan rápido puede presionar y soltar un botón. Debe hacerlo en menos tiempo de lo que tarda en ejecutarse un ciclo de programa.
El segundo problema es lidiar con el rebote del interruptor.
Se le llama rebote de interruptor a la tendencia que tienen estos dispositivos a realizar varias acciones rápidas de encendido / apagado, en el instante que se activan o desactivan.

Disparo por Flanco.
Las rutinas de conteo presentan problemas adicionales en la programación de entradas digitales. En aplicaciones industriales tales como la cuenta de productos que se desplazan por una banda transportadora. No solamente se trata de que el interruptor posee un rebote inherente a su construcción, sino que el producto a contar en sí mismo puede tener forma irregular, tambalearse, o detenerse durante algún tiempo mientras activa el interruptor. Podría haber solamente un producto, pero el interruptor podría abrirse y cerrarse varias veces. Además, si el producto permanece en contacto con el interruptor durante varios ciclos del bucle, el programa debería registrarlo una sola vez y no continuamente.

Comando BUTTON: Rutina Anti Rebote de PBASIC
Eliminar el rebote de los interruptores es una tarea común en programación. PBASIC cuenta con un comando específico para trabajar con la detección de señal de entrada digital. El comando se llama BUTTON. La sintaxis del comando se muestra a continuación.

BUTTON pin, estado activo, retardo, repetir, byte variable, estado salto, dirección
Pin: (0-15) Número de pin de la entrada.

- ✓ Estado activo: (0 o 1) Especifica el estado lógico que ocurre al activar el interruptor.
- ✓ Retardo: (0-255) Especifica el tiempo de establecimiento del interruptor. Entre 0 y 255 son casos especiales. Si retardo es 0, Button no elimina el rebote ni ejecuta auto repetición. Si es 255, Button elimina el rebote pero no ejecuta auto repetición.
- ✓ Repetir: (0-255) Especifica la cantidad de ciclos por repetición.
- ✓ Byte variable: Nombre de una variable tipo byte que sirve como espacio de trabajo para la instrucción BUTTON.
- ✓ Estado salto: Estado del pin en el que se pretende que salte.
- ✓ Dirección: Etiqueta a la que saltará el programa cuando se cumplan las condiciones.

Interruptor Electrónico.

Los interruptores electrónicos que proveen detección “sin contacto” esto significa que no hay que usar contactos eléctricos ni partes móviles. Son muy populares en aplicaciones industriales como los que se mencionan a continuación y los cuales utilizan uno de tres principios.

- ✓ Los interruptores de proximidad inductivos miden el cambio en el rendimiento de un oscilador, cuando objetos de metal se acercan. A menudo los objetos de metal absorben energía mediante corrientes parásitas, lo que hace que el oscilador se detenga.
- ✓ Los interruptores de proximidad capacitivos miden un incremento en la capacidad cuando se les acerca cualquier tipo de material. Cuando el incremento es suficientemente grande, causa que el oscilador interno comience a oscilar. El circuito se dispara y el estado de la salida cambia.
- ✓ Los interruptores ópticos detectan la presencia o ausencia de un angosto haz de luz, a menudo en el rango infrarrojo. En interruptores ópticos reflexivos, el haz de luz podría reflejarse en un objeto móvil hacia el censor óptico del interruptor. Otros interruptores ópticos se configuran dé forma que un objeto bloquee el haz de luz que va desde la fuente de luz hasta el receptor.

Un pulsador tipo resorte serviría en una aplicación donde se le presionara varios miles de veces. Sin embargo, en algún momento su resorte se gastará, o sus contactos de desgastarán u oxidarán, hasta el punto que deje de ser conveniente usarlo.

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Módulo completo
1	Cable serial RS-232	
1	Protoboard	
3	Leds	Color rojo, amarillo, verde.
2	Resistores de 470Ω, ¼ watt	Vea apéndice A.
2	Resistores de 10 KΩ, ¼ watt	Vea apéndice A.
−	Cable telefónico	Varios colores.
−	Caimanes	
1	PC	

MEDIDAS DE SEGURIDAD:

Verifique que la polaridad del Led sea la correcta. Hay una zona lisa en un costado del Led que deberá ser considerada como cátodo. Si la polaridad es invertida, el Led no encenderá.

Asegurarse que el resistor de 10KΩ este conectado al extremo del interruptor, posteriormente al socket correcto.

Para verificar el valor de las resistencias vea al apéndice A. código de colores.

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Una cinta transportadora está moviendo piezas a través de una estación de perforado. Cuando se detecta una pieza, la cinta transportadora se detiene. Luego de dos segundos se activa para sujetar la pieza; otra pausa corta y el taladro baja hacia la pieza. Un interruptor de proximidad controla la profundidad del agujero taladrado. Cuando se activa el detector de profundidad, el comando “perforar” se detiene y se eleva la posición del taladro. Luego de darle un tiempo para que el taladro se retraiga, el solenoide libera la pieza y se activa la cinta transportadora. La pieza procesada es retirada de este sector y la cinta transportadora continua moviéndose hasta que se detecte otra pieza. Cuando se detecta otra pieza, el proceso secuencia se repite.

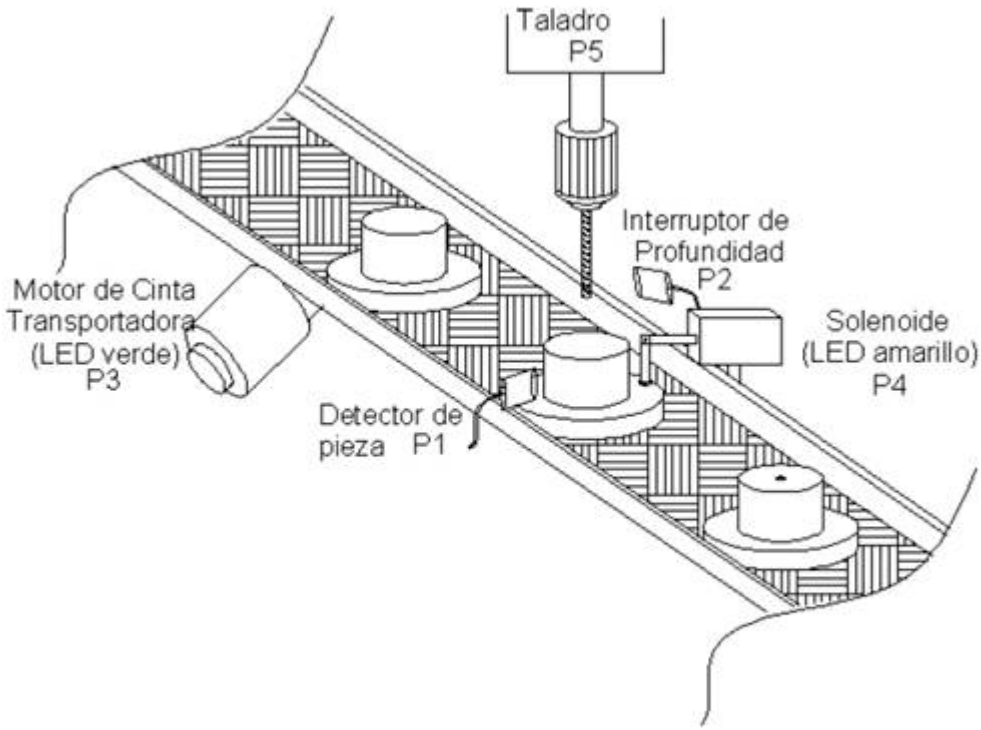


Fig. 2.3

Compare el diagrama de flujo de la Figura 2.4 con la descripción del proceso anterior.

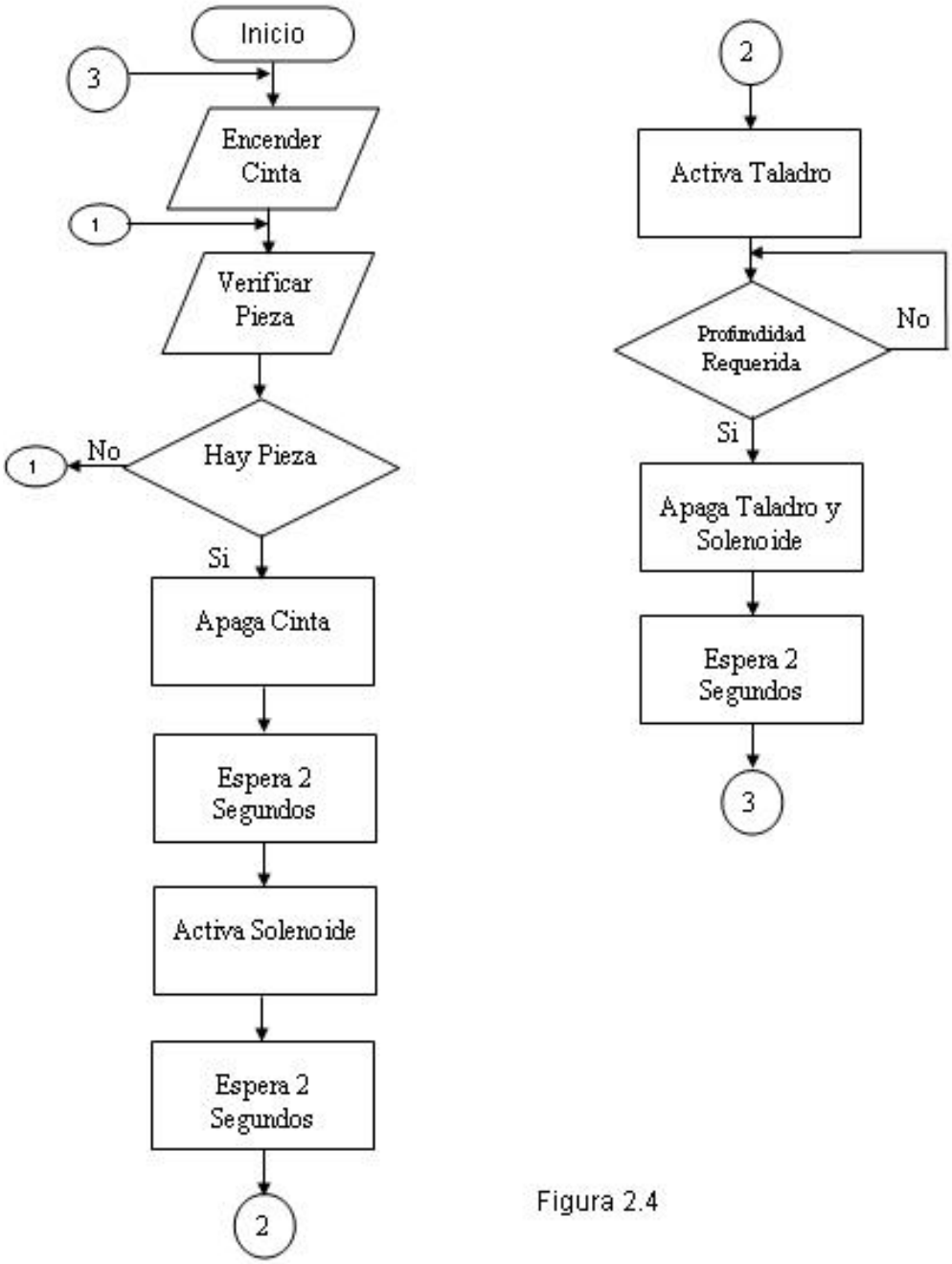


Figura 2.4

Acondicionamiento de señal apropiado. Los Leds simularan el comportamiento de los relevadores que controlarían la cinta transportadora, el solenoide y el taladro. Dos pulsadores S1, S2, simularán una pieza que llega a su posición y la indicación de profundidad de perforado. Arme los circuitos de la Figura 2.5 en la protoboard. Para facilitar las cosas, se sugiere utilizar un Led verde para la cinta transportadora, el amarillo para el solenoide y el rojo para el taladro.

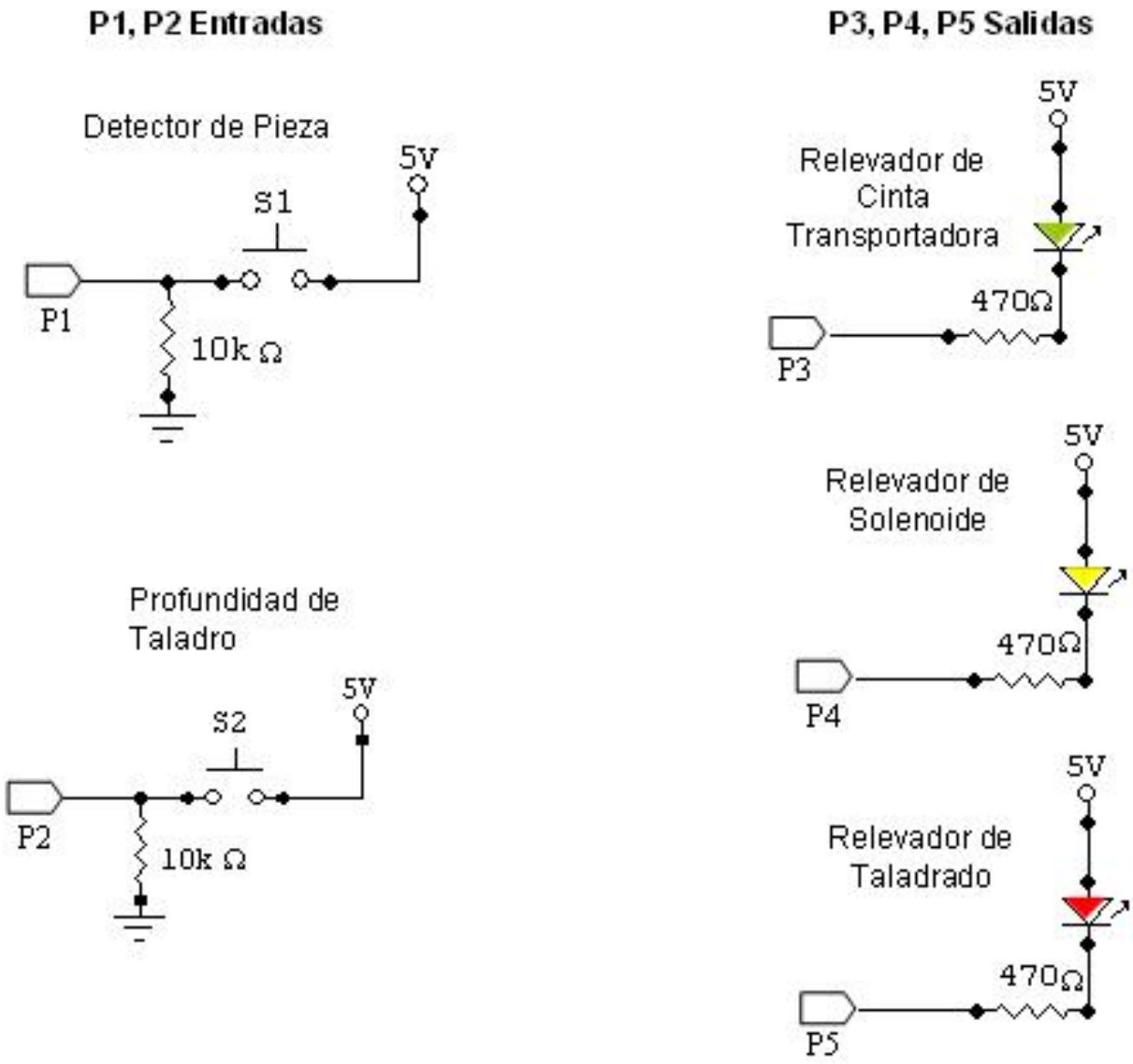


Figura 2.5

Cuando el programa se inicia, el Led verde se enciende. Esto representa la activación de la cinta transportadora. Para simular una pieza entrando en el área de perforado, debe presionar y retener a S1. El Led de la cinta se apagará instantáneamente y el Led amarillo

que representa el relevador del solenoide se encenderá luego de una pausa. Luego de que el solenoide tuvo tiempo suficiente para sujetar la pieza, el taladro se activará y comenzará a perforar la pieza, como lo indica el Led rojo. En este momento presione a S2 para indicar que el agujero es suficientemente profundo. El pulsador S2 representa un interruptor de proximidad que detecta que la profundidad de la perforación es la correcta. Al presionar S2 se apagará el Led rojo, indicando que el taladro se detuvo. Deje de presionar S2 para indicar que el taladro se está alejando de la pieza, a medida que se contrae a su posición original durante los dos segundos que el programa queda detenido. Luego de esta pausa, el solenoide liberará la pieza (luz amarilla se apaga) y la cinta transportadora se activa nuevamente. La pieza está terminada y deja la zona de perforado. Desde este punto la secuencia comienza otra vez.

A continuación se presenta el programa

```
' -----[ Adquisición de una señal digital al  Microcontrolador]--------------------------------
' Archivo..... Práctica No 2
' Propósito..... Programar al Micro como dispositivo de entrada y salida
' Carrera Ing. Industrial

  '{$STAMP BS2p}
INPUT  1                          ' Detector de pieza
INPUT  2                           ' Profundidad de perforación
OUTPUT 3                           ' Relevador de cinta transportadora (Led verde)
OUTPUT 4                           ' Relevador de solenoide (Led amarillo)
OUTPUT 5                          ' Relevador de taladro (Led rojo)
OFF CON 1                         ' Lógica negativa
ON CON 0
OUT3 = OFF                         ' Inicializa salidas en off (apagadas)
OUT4 = OFF
OUT5 = OFF
Inicio:

OUT3 = ON                          ' Enciende cinta transportadora
IF IN1 = 1 THEN Proceso           ' Si se presiona, inicia el "Proceso"
GOTO Inicio
Proceso:                          ' Se inicia el Proceso
OUT3 = OFF                        ' Detiene cinta transportadora
PAUSE 1000
OUT4 = ON                         ' Sujeta la pieza en su lugar
PAUSE 2000                         ' Espera 2 segundos para encender el taladro
Taladrar:
OUT5 = ON                                 ' Enciende el taladro y comienza a bajar
IF IN2 = 1 THEN Levanta _ taladro          ' Si la profundidad es suficiente, levanta el taladro
Levanta _ taladro:
OUT5 = OFF                                 ' Apaga el taladro y lo levanta
IF IN2 = 0 THEN Taladro _ subiendo     ' Indica que el taladro está subiendo
GOTO Levanta _ taladro
Taladro _ subiendo:
PAUSE 2000                                ' Continua subiendo el taladro por 2 segundos
```

```
Suelta:
OUT4 = OFF                                  ' Abre el solenoide para liberar la pieza
PAUSE 1000                                  ' Espera 1 segundo
OUT3 = ON                                   ' Enciende la cinta transportadora
IF IN1 = 0 THEN Siguiente _ pieza           ' Pieza terminada deja el área de Proceso
GOTO Suelta
Siguiente _ pieza:
PAUSE 1000                                   ' Espera 1 segundo
DEBUG "Pieza liberada. Comenzando el siguiente ciclo", CR
GOTO Inicio
```

Conecte el módulo del Micro a su PC.

- Conecte el cable serial RS-232 de la plaqueta del módulo.
- Conecte la fuente del módulo a 127 CCA.
- Presiona ctrl. + T para verificar, si tiene errores en el programa.
- Presione ctrl. + R para correr el programa.

Descripción del programa

El propósito de este programa es ejecutar cierto código de acuerdo al estado (presionado o no) de los pulsadores. Este ejercicio introduce varias consideraciones a tener en cuenta cuando se trabaja con entradas digitales, se programan varias instrucciones if-then, o se usan algunos operadores lógicos de PBasic. Primero, in1 e in2 simplemente contienen el valor lógico de los pines de entrada: +5 V = 1 lógico. Los pulsadores se encuentran en activo alto entregan un 1 si están presionados. El programa controla el estado "lógico" de las entradas; cuando considera las configuraciones de los botones, las acciones de los controladores industriales a menudo dependen del estado de muchos interruptores y contactos. Una revisión de los operadores lógicos de PBasic, incluyendo AND, OR, XOR y NOT, puede ser conveniente para cumplir estos requerimientos usando un Micro con módulo de Basic Stamp. Otro aspecto a observar en el programa es el flujo del programa. Las estructuras IF-THEN verifican una condición y si esta condición se cumple, entonces la ejecución del programa salta hacia la etiqueta. En aplicaciones industriales, esta porción del programa podría causar una acción de salida apropiada. Dado que la última línea de cada subrutina es GOTO Bucle, la ejecución del programa salta al principio del bucle principal y cualquier parte de código que se encuentre por debajo de la instrucción IF-THEN que causó la división se ignora. El diagrama de flujo muestra como se ejecuta el programa.

Preguntas y desafío:

1.- Del problema anterior de proceso de maquinado anexe al programa para que realice lotes de 70 piezas.

2.- Desafío de Programación: Operación de Mezcla Secuencial

Una secuencia de Mezclado se muestra en la Figura 2.6. En este proceso, un operario presiona momentáneamente un botón para abrir una válvula y comenzar a llenar un depósito. Un flotador mecánico sube con el nivel del líquido y acciona un interruptor cuando el recipiente está lleno. En este momento, el solenoide de llenado se apaga y un agitador mezcla el contenido durante 15 segundos. Después del período de mezcla, un solenoide en la parte inferior del recipiente se abre para vaciar el tanque. El flotador mecánico desciende, abriendo el interruptor cuando el recipiente se vacía. En este punto, el solenoide de vaciado se apaga, cerrando la válvula. El proceso está listo para ser inicializado nuevamente por el operador.

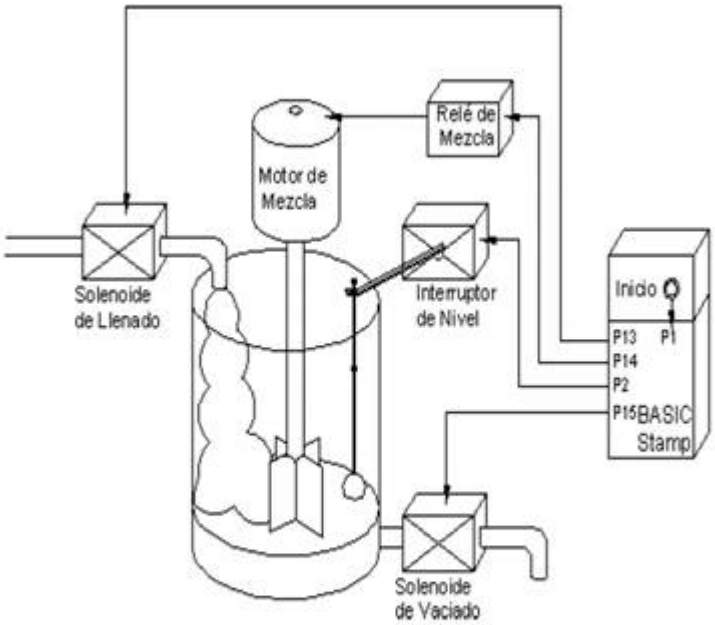


Figura 2.6: Proceso de Control de Mezclado Secuencial

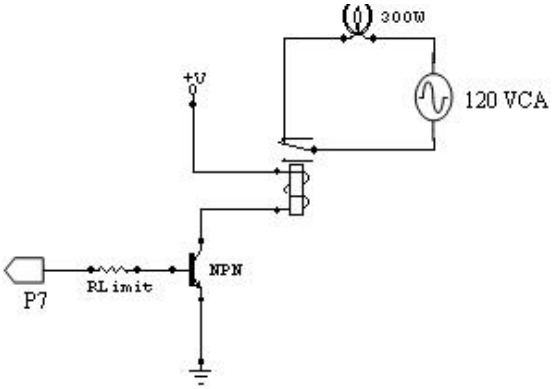
Asigne las siguientes entradas y salidas del Micro para simular la operación.

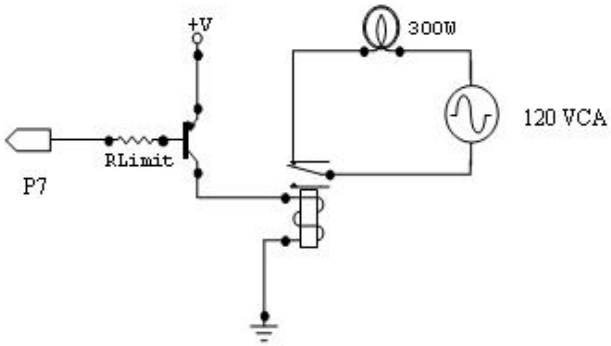
Botón de Inicio Entrada P1 (N.O.)
Interruptor de nivel Entrada P2 (N.O.)
Solenoide de llenado Salida P13 (LED rojo)
Solenoide de mezcla Salida P14 (LED amarillo)
Solenoide de vaciado Salida P15 (Led verde)

Realice el diagrama de flujo y programe la operación.

3.- Cuando la corriente de salida del Micro no es suficiente para encender el dispositivo de control, un _____ de salida podría usarse para incrementar la potencia.

4.- Considere los circuitos A y B de abajo. Escriba una línea de código de Basic Stamp para encender la lámpara en cada circuito.





Capítulo 2.2 "Enviando y recibiendo señales digitales controlando un servomotor"

No. DE PRÁCTICA: 3 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Diseñar un sistema microcontrolado con un dispositivo de salida a través de un servomotor.

INTRODUCCIÓN:

Un servo es un motor de corriente continua que tiene un"circuito de control" construido en su interior. En lugar de rotar continuamente, como los motores comunes, un servo es "posicionable". La posición del engranaje de salida del servo es determinada por el ancho de un pulso de control. Se puede controlar, enviando las señales apropiadas con el comando PULSOUT de PBasic; este comando es ideal para controlar los servos y hacer rotar al servo a un punto específico, para mantener esta posición el servo debe ser actualizado constantemente; la frecuencia de actualización típica es de aproximadamente 50 veces por segundo. El pulso de control normalmente tiene un ancho de entre uno y dos milisegundos. El servo se centrará cuando la señal de control es de 1,5 milisegundos.

Un servo, necesita tener el tren de pulsos (en el cable de "control" blanco o amarillo), que varíen su longitud entre 1 y 2 milisegundos. Con el tren de pulsos que dan una longitud constante de 1 milisegundo, el servo se posicionará en un extremo de su rotación. A medida que el ancho del pulso se incrementa, (1,1ms, 1,2ms, 1,3ms… etc.), el servo cambia su posición. Cuando el ancho del pulso alcanza los 2,0 ms, el servo está en el otro extremo de su rotación. Estos pulsos ocurren a intervalos de aproximadamente 10 ms. La Figura 3.1 es un diagrama de tiempo de los pulsos que el servo necesita.

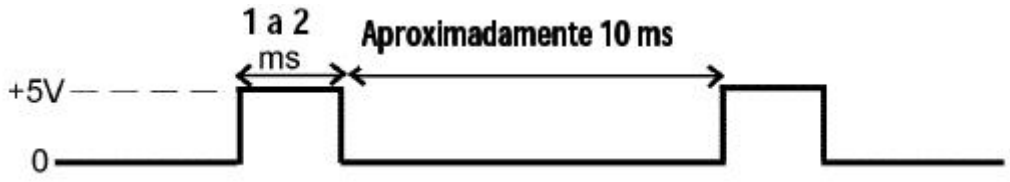


Figura 3.1: Secuencia de pulsos de un servo típico. Diagrama de tiempo de los pulsos que necesita el servo.

MARCO TEÓRICO:

Las computadoras funcionan con una serie de pulsos, normalmente entre 0 y 5 volts. Un diagrama de tiempo es simplemente una forma de visualizar los pulsos. Un diagrama de tiempo se lee de izquierda a derecha. En el diagrama de ejemplo Fig. 3.1, se nota que el voltaje, (en el pin de salida P1) comienza a 0 volts después de un corto período de tiempo se puede notar que P1 pasa a estado alto por una duración de 1 a 2 milisegundos y luego regresa a 0 volt después de aproximadamente 10 milisegundos, P1 vuelve a subir. Salvo que se indique en el diagrama, se puede asumir que el proceso se repite indefinidamente, es decir que cuando llega al extremo derecho del diagrama, regresa al extremo izquierdo, y comienza otra vez.

Los sistemas de microcontroladores y computadoras, operan a velocidades muy altas. Como seres humanos, estamos acostumbrados a usar para mediciones de tiempo, el rango de segundos, o en el caso de competencias atléticas, décimas o centésimas de segundos. Un milisegundo es 1 / 1000 de segundo, es decir hay 1000 milisegundos en 1 segundo. Esto se ve como una pequeña cantidad de tiempo, pero es realmente bastante largo en el mundo de la microelectrónica de las computadoras ya que operan en millonésimas de segundos.

MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Módulo completo
1	Cable serial RS-232	
1	PC	
1	Led	Color rojo
1	Resistor de 470Ω, ¼ watt	Vea apéndice A.
–	Cable telefónico	Varios colores.
–	Caimanes	
1	Servo RC	5 Vcc
1	Capacitor electrolítico	3000 μf

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Conecte en el Protoboard el siguiente circuito

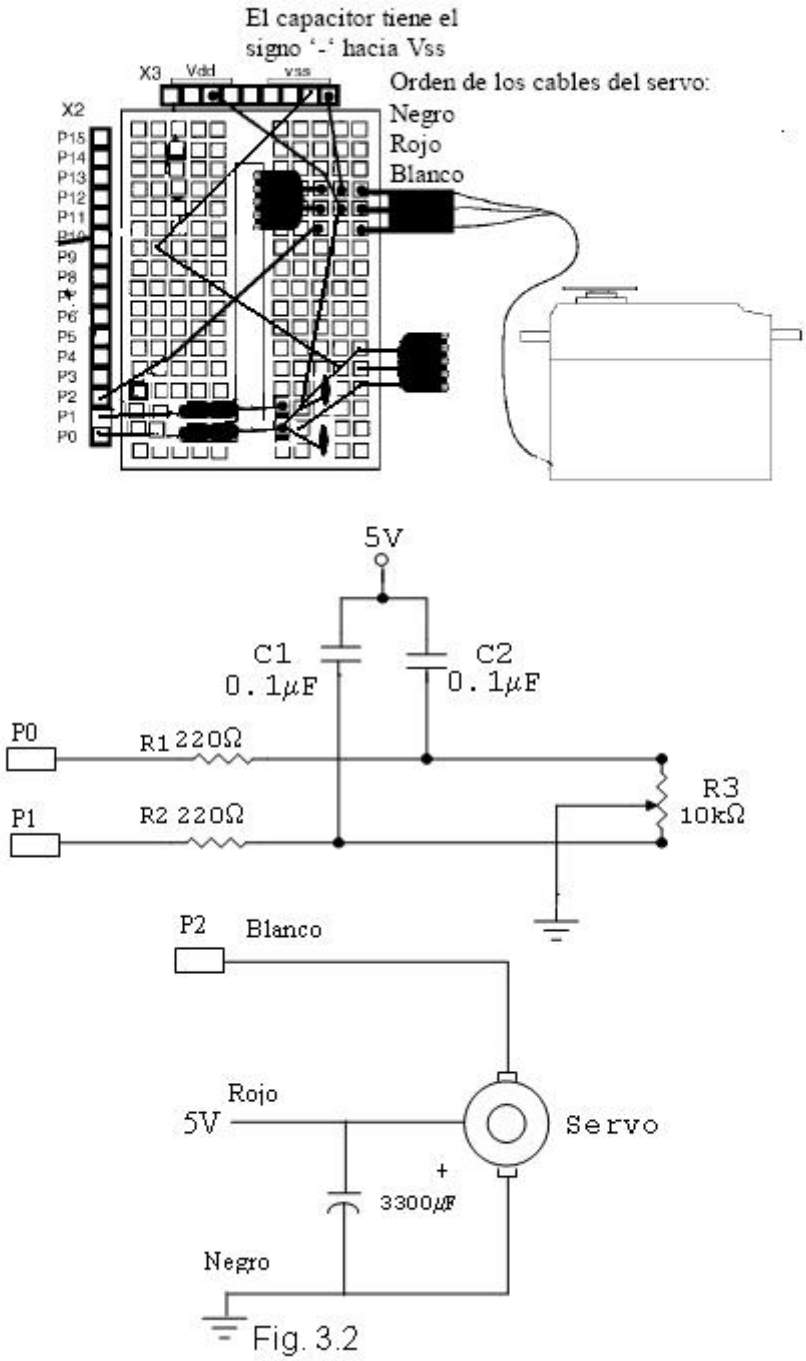


Fig. 3.2

De acuerdo al problema de la práctica anterior considere al servo para realizar el movimiento de un brazo mecánico de un taladro en el proceso de barrenado. Con el siguiente diagrama de flujo elabore el programa correspondiente.

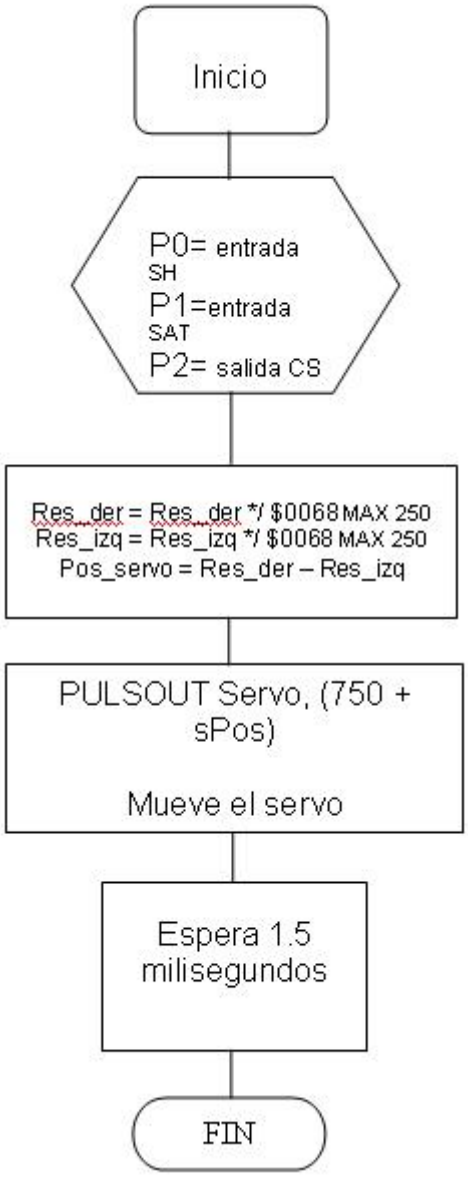


Fig. 3.3 Diagrama de flujo

A continuación se presenta el programa

```
' -----[ Movimiento Microcontrolado ]-----------------------------------
' Archivo..... Práctica No 3
' Propósito..... Controlar la posición deseada a través de un dispositivo de salida servomotor.
' Carrera Ing. Industrial

'{$STAMP BS2p}

Pos_der      CON 0        ' entrada posición sentido horario
Pos_izq      CON 1        ' entrada posición sentido anti-horario
Servo        CON 2        ' Control de servo

Res_der      VAR Word     ' lectura de resistencia lado derecho
Res_izq      VAR Word     ' lectura de resistencia lado izquierdo
diff         VAR Word     ' diferencia entre lecturas
Pos_servo    VAR Word     ' posición del servo
' ---------------------------------------------------------------------
Principal:
HIGH Pos_der                                  ' descarga capacitores
HIGH Pos_izq
PAUSE 1
RCTIME Pos_der,1, Res_der                     ' lee sentido horario
RCTIME Pos_izq  ,1, Res_izq                   ' lee sentido anti-horario
Res_der = Res_der  */ $0068 MAX 250           ' escala RCTIME de 0-250
Res_izq = Res_izq   */ $0068 MAX 250
Pos_servo = Res_der - Res_izq                 ' calcula posición (-250 a 250)

PULSOUT Servo, (750 + Pos_servo)              ' mueve el servo
PAUSE 15
GOTO Principal
END
```

Descripción del Programa.

Recuerde que para que el Micro sepa qué variables están siendo usadas, es necesario "declararlas" en el programa. Este comando le dice al Micro que se utilizara una variable por ejemplo "Res_der", y que tendrá el tamaño de una palabra "word". Una variable "word" de 16 bits puede tener un valor entre 0 y 65,536 en el sistema numérico decimal.

En este experimento, dos circuitos RCTIME se construyen sobre un potenciómetro de 10KΩ. Este circuito y el código pueden ser usados para determinar la posición relativa del potenciómetro. Las lecturas de cada lado del potenciómetro se multiplican por un factor de escala para obtener un rango de 0 a 250 con los operadores */ [1] y MAX[8] . Restando un lado del otro, se obtiene un valor de posición del servo entre –250 y +250.

Este valor se agrega a la posición central de 750. Observe que para rotar el servo a una posición en particular, sólo debe cambiar el valor del ancho del pulso a través del resistor variable. Es bueno recordar que PULSOUT trabaja en unidades de 2 microsegundos, así que un valor de PULSOUT de 750 creará un pulso de 1.5 milisegundos de ancho, haciendo que el servo se centre. Cuando la posición del servo es –250, el valor de PULSOUT es 500 (750-250) de 1,0 milisegundos. Con un valor Pos_servo de +250, el valor de PULSOUT es 1000 (750+250), creando un pulso de control de 2,0 milisegundos haciendo que el servo rote a su extremo opuesto.

[8] Para aplicar la Multiplicación de Fracciones (*/) se deben convertir los valores decimales a Hexadecimales.

Desafío:

- Reemplace el potenciómetro con dos foto resistores y actualice el código para hacer que el servo se centre hacia la fuente de luz más brillante.
- Elabore un diagrama eléctrico para que el servo se mueva a control remoto con una resistencia digital (CIX9313).
- De 5 ejemplos de aplicaciones de este dispositivo

Capítulo 3.1 "Cambios de nivel de tensión analógica aplicada a un circuito."

No. DE PRÁCTICA: 4 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Interpretar una señal analógica a través del circuito integrado (CI) LM358 configurado como seguidor de tensión para mostrarnos un dato binario interpretado por el micro SX48BD.

INTRODUCCIÓN:

Analógico es un "valor que varía en forma continua". También se puede interpretar a la electrónica analógica como una analogía de la naturaleza. Hay muchos valores que varían en forma continua en la naturaleza, tales como movimiento, nivel de luz, temperatura y sonido.

En esta práctica, construiremos un circuito que produce una tensión analógica en su salida. Recuerde que la tensión analógica varía en forma continua. El circuito tendrá salida regulable entre 0 y 5 Volts. También se construirá un circuito llamado seguidor de tensión que emplea esta tensión analógica para alimentar el circuito de un Led. La tensión analógica también estará conectada a uno de los pines de E/S del Micro, configurado como entrada. Esta entrada binaria puede ser usada para medir las variaciones de la tensión analógica. El lenguaje PBasic se usará para programar al Micro de forma que controle un circuito de Led binario, que indicará cuándo se detecten variaciones en la entrada.

MARCO TEÓRICO:

El Amplificador Operacional LM358.

Un op-amp (amplificador operacional) es un bloque de construcción usado comúnmente en circuitos analógicos. La Figura 4.1 muestra el símbolo esquemático y el diagrama en bloques del amplificador operacional LM358 usado en este experimento. El circuito usado se denomina seguidor de tensión, debido a que la tensión de salida es igual a la tensión de entrada. En otras palabras, la tensión de salida "sigue" a la tensión de entrada. La razón para usar el seguidor de tensión es separar eléctricamente el circuito del potenciómetro del circuito del Led.

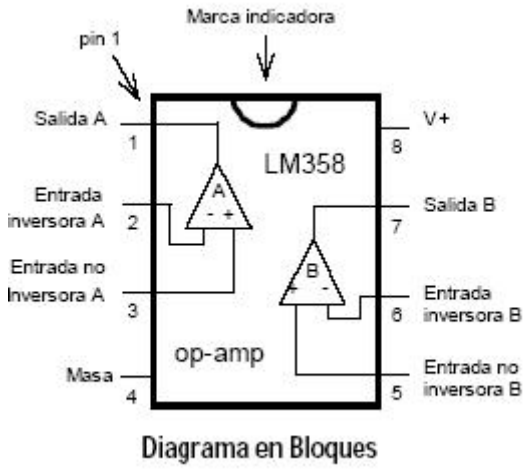


Fig. 4.1

Comandos utilizados en esta práctica

Comando DEBUG

DEBUG es la manera más conveniente de visualizar cualquier variable o desplegar cualquier mensaje. La ventaja que nos ofrece es que podemos tener acceso al interior de cualquier variable o registro de entrada y salida o direccionamiento del puerto del Micro.

DEBUG Outputdata {, Outputdata ...}

Función

Visualiza variables y mensajes por la pantalla de la PC en combinación con el editor del MIcro. Este comando es utilizado para visualizar textos y números en varios formatos.

- OutputData salida de datos pueden ser una variable/constante/expresión, del rango comprendido entre (0– 65535) especifica la salida de datos. La salida de datos puede estar en caracteres ASCII (Texto entre comillas " ", y caracteres de control), los números decimales (0–65535), los números hexadecimales ($0000-$FFFF), y los números binarios (%0000000000000000-%1111111111111111). La data numérica puede ser modificada con formatos como se explica en la tabla 4.1.

Formato	Descripción
?	Muestra el "Nombre de la variable = x" mas un retorno de carro (CR), donde (x), representa al valor numérico decimal.
ASC ?	Muestra el "Nombre de la variable = x" mas un retorno de carro (CR), donde (x), representa al valor ASCII.
DEC{1..5}	Numero decimal, opcional de 1 a 5 dígitos.
SDEC{1..5}	Numero decimal con signo, opcional de 1 a 5 dígitos.
HEX{1..4}	Numero hexadecimal, opcional de 1 a 4 dígitos.
SHEX{1..4}	Numero hexadecimal con signo, opcional de 1 a 4 dígitos.
IHEX{1..4}	Numero hexadecimal con el prefijo ($, ejemplo $FF08) opcional de 1 a 4 dígitos.
ISHEX{1..4}	Numero hexadecimal con signo y el prefijo ($, ejemplo -$FF08) opcional de 1 a 4 dígitos.
BIN{1..16}	Numero binario, opcional de 1 a 16 dígitos.
SBIN{1..16}	Numero binario con signo, opcional de 1 a 16 dígitos.
IBIN{1..16}	Numero binario con el prefijo (%, ejemplo %1001001) opcional de 1 a 16 dígitos.
ISBIN{1..16}	Numero binario con signo y el prefijo (%, ejemplo -%1001001) opcional de 1 a 16 dígitos.
STR Arreglos	Cadena ASCII desde un arreglo completo hasta que encuentre un byte = 0.
STR Arreglos\n	Cadena ASCII desde un arreglo hasta un numero (n) especificado.
REP Byte\n	Muestra un carácter ASCII (n) veces.

Ejemplo 4.1 de programa para comando Debug

```
numero VAR BYTE ‘ Declaración de una variable tipo Byte
numero = 65              ‘ Fijando el valor 65 en numero
DEBUG DEC numero         ‘Visualizar el valor “65” por pantalla
```

Para los valores con signos se sigue el mismo formato:

```
numero VAR WORD          ‘ Declaración de una variable tipo Byte
numero = -65             ‘ Fijando el valor 65 en numero
DEBUG “Signos.....: “, SDEC numero,” “, SHEX numero,“ “,SBIN numero,13
DEBUG “Sin Signos.: “, DEC numero,” “, HEX numero, “ “,BIN numero,13
```

Este código genera los siguientes resultados:

```
Signos.....: -65 -41 -1000001
Sin Signos.: 65471 FFBF 1111111110111111
```

MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Módulo completo
1		Cable serial RS-232
1	PC	
2	Led	Color rojo
2	Resistores de 470Ω, ¼ watt	Vea apéndice A.
–	Cable telefónico	Varios colores.
1	Amplificador Operacional LM358	Vea anexo 1**.**
1	Potenciómetro de 100 kΩ.	

MEDIDAS DE SEGURIDAD:

Asegúrese de identificar correctamente la ubicación del pin 1 y la marca indicadora (Mueca), como se muestra en la figura 4.2. Cuando coloque el LM358 en la protoboard. Un conexionado incorrecto podría dañar el circuito integrado.

Figura 4.2

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme el siguiente circuito de acuerdo a la figura 4.3

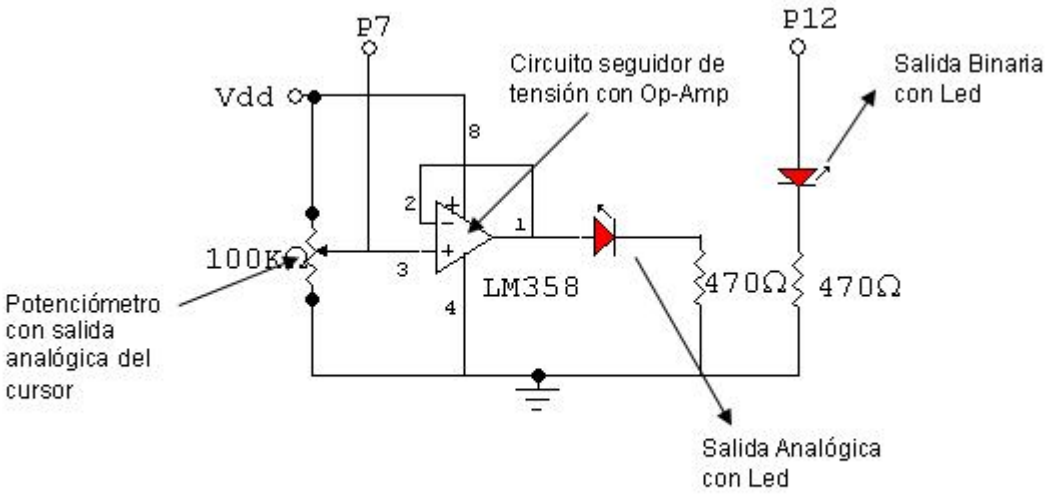


Figura 4.3

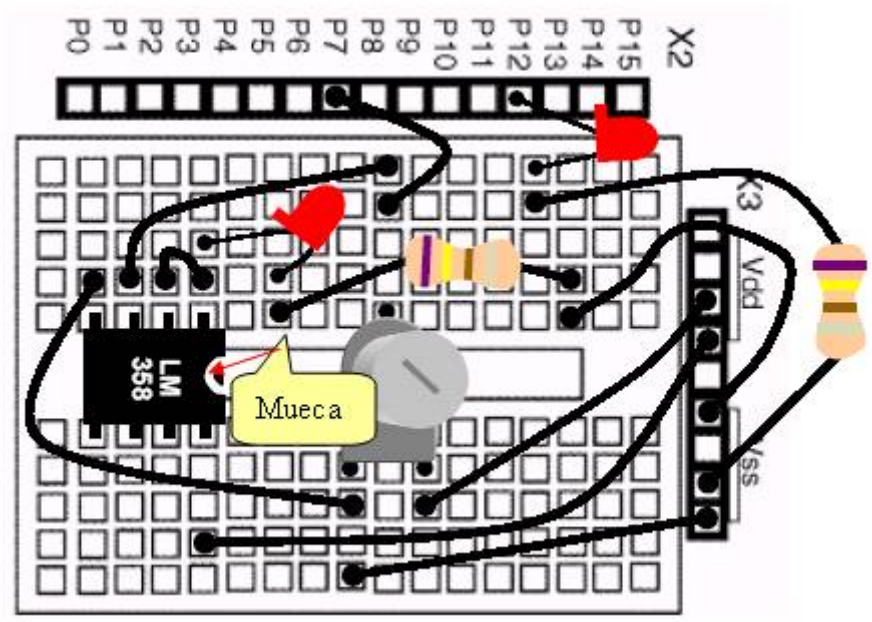


Figura 4.4 Esquema físico de conexionado

Diagrama de flujo

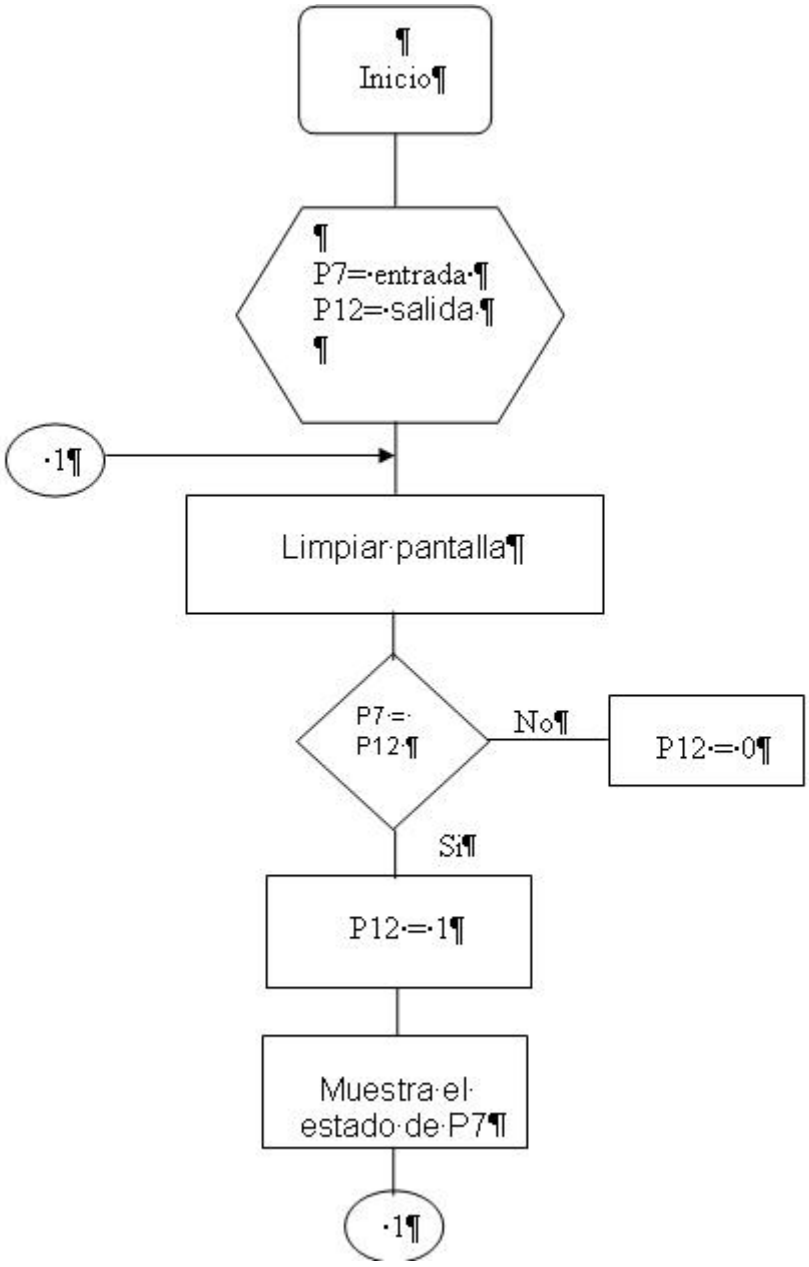


Figura 4.5 Diagrama de flujo

```
' -----[ Tensión analógica]-------------------------------------
' Archivo..... Práctica No 4
' Propósito..... Uso del Micro en aplicaciones analógicas.
' Carrera Ing. Industrial

'{$STAMP BS2p}
DEBUG CLS
INPUT 7
OUTPUT 12
bucle:
OUT12 = IN7
DEBUG HOME, "El estado de P7 es ", BIN IN7
GOTO bucle
```

Descripción del Programa.

Es bueno inicializar la ventana DEBUG y limpiarla antes de mostrar datos. De esta forma se evita mostrar datos erróneos procedentes de programas anteriores en la memoria del Micro. La ventana DEBUG se abre automáticamente la primera vez que encuentra el comando DEBUG en un programa en PBASIC. Este comando DEBUG limpia la ventana luego de abrirla: DEBUG cls

El resto del programa debería repetirse una y otra vez, así que este es un buen lugar para introducir una etiqueta. Un rótulo o etiqueta finaliza con dos puntos. Puede elegir cualquier palabra, a excepción de las palabras reservadas. Usemos la palabra bucle:

Más adelante en el programa, aparecerá la instrucción GOTO bucle. Cada vez que el programa encuentra el comando GOTO bucle, regresa a la etiqueta bucle: y comienza a ejecutar nuevamente las instrucciones.

La siguiente tarea es lograr que el Led conectado al pin P12 se encienda cuando la tensión en P7 es suficientemente alta como para ser considerada como señal binaria de estado alto. En otras palabras, si el valor de entrada medido en P7 es un 1 binario, la salida en P12 deberá ser un 1 binario. Aunque hay varias formas de llevar esto a cabo, la más simple es igualar el valor de salida binario del pin P12 al valor de entrada binario del pin P7.

```
OUT12 = IN7
```

Se pueden usar comandos DEBUG para mostrar los niveles de señal recibidos por un pin de E/S que esté funcionando como entrada, en la ventana de DEBUG. El comando DEBUG de abajo imprime tres datos diferentes. Cuando se imprime más de un dato con el comando DEBUG, éstos deben separarse con comas.

```
DEBUG HOME, "El estado de P7 es ", bin in7
```

Lo primero que imprime el comando DEBUG anterior es HOME. Esto envía el cursor a la esquina superior izquierda (conocida como "HOME" o “inicio”) de la ventana DEBUG. Note que HOME está seguida por una coma para separar el siguiente dato a imprimir.

El siguiente dato está entre comillas:

```
"El estado de pin P7 es".
```

Cada vez que se requiera mostrar un mensaje de texto en la ventana DEBUG, se utilizan comillas. El tercer dato bin in7, le dice a la ventana DEBUG que muestre el valor de entrada binario del pin P7.

Se requiere que el Micro siga controlando el valor en P7 una y otra vez. También que actualice automáticamente el Led y la ventana DEBUG con la información obtenida de P7. Esto se lleva a cabo repitiendo indefinidamente el programa desde la etiqueta bucle, que se creo anteriormente.

Desafío:

1. Agregue un segundo Led al circuito de la Protoboard y use el pin P11 para alimentarlo en sentido inverso. En otras palabras, cuando un Led esté encendido, el otro estará apagado.

2. Modifique el código del Programa para que el Led parpadee mientras la salida del potenciómetro esté por encima de la tensión umbral del pin de entrada del Micro.

Capítulo 3.2 "Interpretando una señal analógica como entrada y darle una salida digital."

No. DE PRÁCTICA: 5 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

La finalidad es interpretar datos analógicos generados por variables físicas, para procesarlos como señales digitales a través del Microcontrolador.

INTRODUCCIÓN:

Calentar o enfriar objetos a cierta temperatura, mantener una presión constante en un ducto de vapor, o fijar un valor de flujo de material a un recipiente para mantener un nivel constante de líquido, son ejemplos de variables analógicas. La condición que queremos controlar se denomina "variable de proceso" temperatura, presión, flujo y nivel del líquido son las variables de proceso de estos ejemplos. Los dispositivos de salida industrial son los elementos de control, motores, válvulas, calentadores y solenoides son ejemplos de dispositivos usados para controlar la energía que determina las salidas de los procesos.

La acción de control tomada se basa en la relación dinámica entre la configuración del dispositivo de salida y su efecto sobre el proceso.

La temperatura es la variable de proceso más común que se encontrará. Desde el control de temperatura del metal fundido en una metalúrgica hasta el control del nitrógeno líquido en un laboratorio, la medición, evaluación, y control de la temperatura son críticos para la industria.

MARCO TEÓRICO:

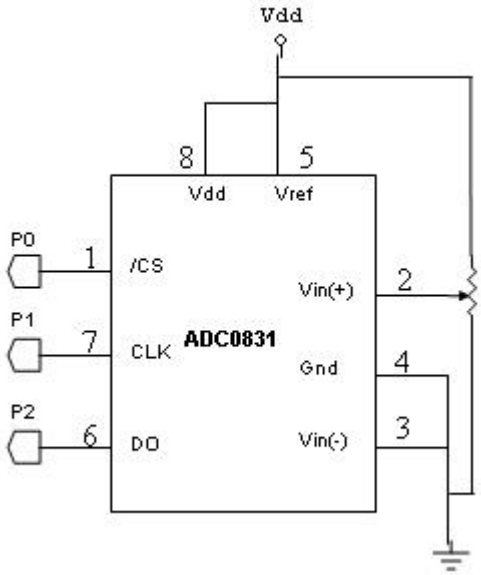
Si bien la información sobre la tensión analógica puede ser eficientemente procesada por dispositivos binarios, la tensión primero debe ser muestreada para poder describirla usando números binarios. El ADC0831 es un circuito integrado común que realiza esta tarea. Describe la información analógica con números binarios para dispositivos que procesan información binaria, como el Micro SX48BD. En este experimento, se construirá un circuito con el ADC0831 al cual se le conectará un potenciómetro; se ajustará para producir la tensión de salida analógica.

MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Módulo completo
1	PC	
1	Cable serial RS-232	
2	Led	Color rojo
2	Resistores de 470Ω, ¼ watt	Vea apéndice A.
1	Resistor de 47Ω ½ watt	Vea apéndice A
–	Cable telefónico	Varios colores.
1	CI Convertidor ADC0831	Vea anexo 1**.**
1	Potenciómetro de 100 kΩ.	
1	Transistor 2N3901	Vea anexo 1**.**
1	CI LM34DZ	Vea anexo 1**.**
1	Protoboard	

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme el siguiente circuito de acuerdo a la figura 5.1



```
' -----[ Conversión Analógica a Digital]-------------------------------------
' Archivo..... Práctica No 5
' Propósito..... Uso del Micro manejo de señales analógicas.
' Carrera Ing. Industrial

'{$STAMP BS2p}
adc0831 VAR Byte
v VAR Byte
R VAR Byte
v2 VAR Byte
v3 VAR Byte
CS CON 0
CLK CON 1
D0 CON 2
DEBUG CLS
Proceso_de_conversionA_D:
GOSUB Datos_ADC0831
GOSUB CALC_Tension
GOSUB MOSTRAR
GOTO Proceso_de_conversionA_D:
Datos_ADC0831:
HIGH CS
LOW CS
LOW CLK
PULSOUT CLK, 210
SHIFTIN D0, CLK, MSBPOST, [adc0831\8]
RETURN
CALC_Tension:
v = 5*adc0831/255
R=5*adc0831//255
v2=100*R/255
v3=100*R//255
v3=10*v3/255
IF v3<5 THEN opcion_2
v2=v2+1
opcion_2:
IF v2<100 THEN sal_de_redondeo
v=v+1
v2=0
```

```
sal_de_redondeo:
RETURN
MOSTRAR:

DEBUG HOME
DEBUG "valor binario de 8-bits: ", BIN8 adc0831
DEBUG CR, CR, "Valor decimal: ", DEC3 adc0831
DEBUG CR, CR, "Tensión: ", DEC3 v, " Volts"
DEBUG DEC1 v, ".", DEC2 v2, " Volts"
RETURN
```

Diagrama de flujo

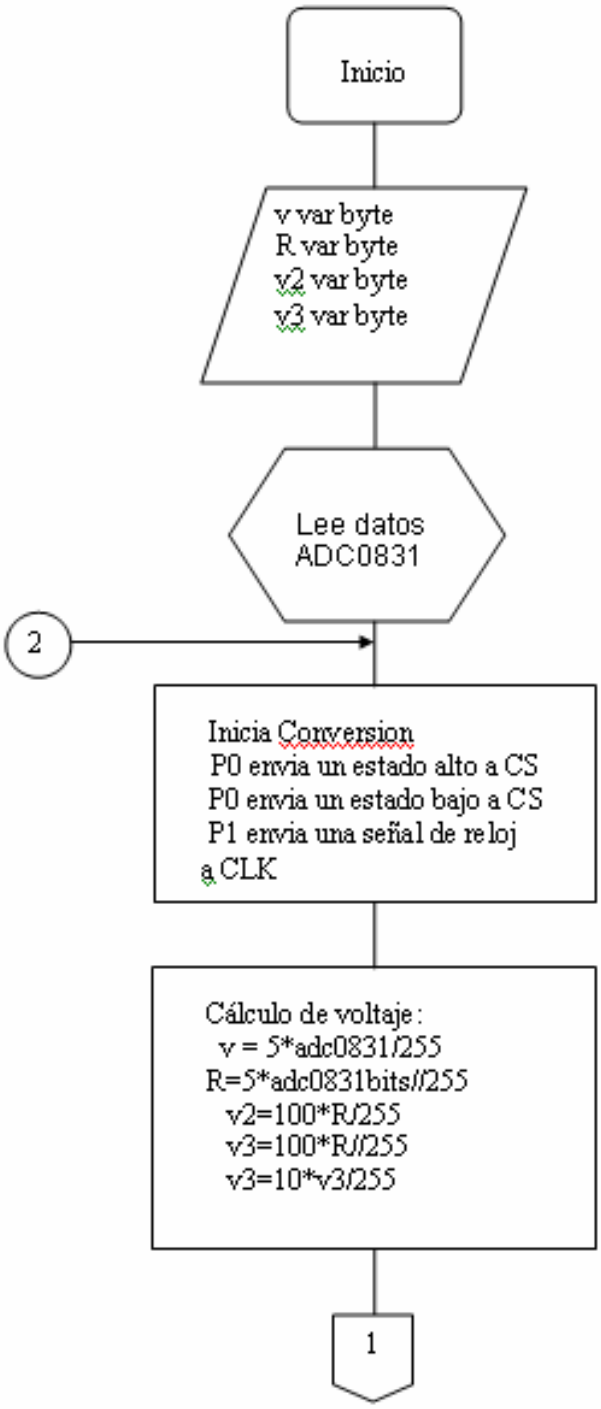
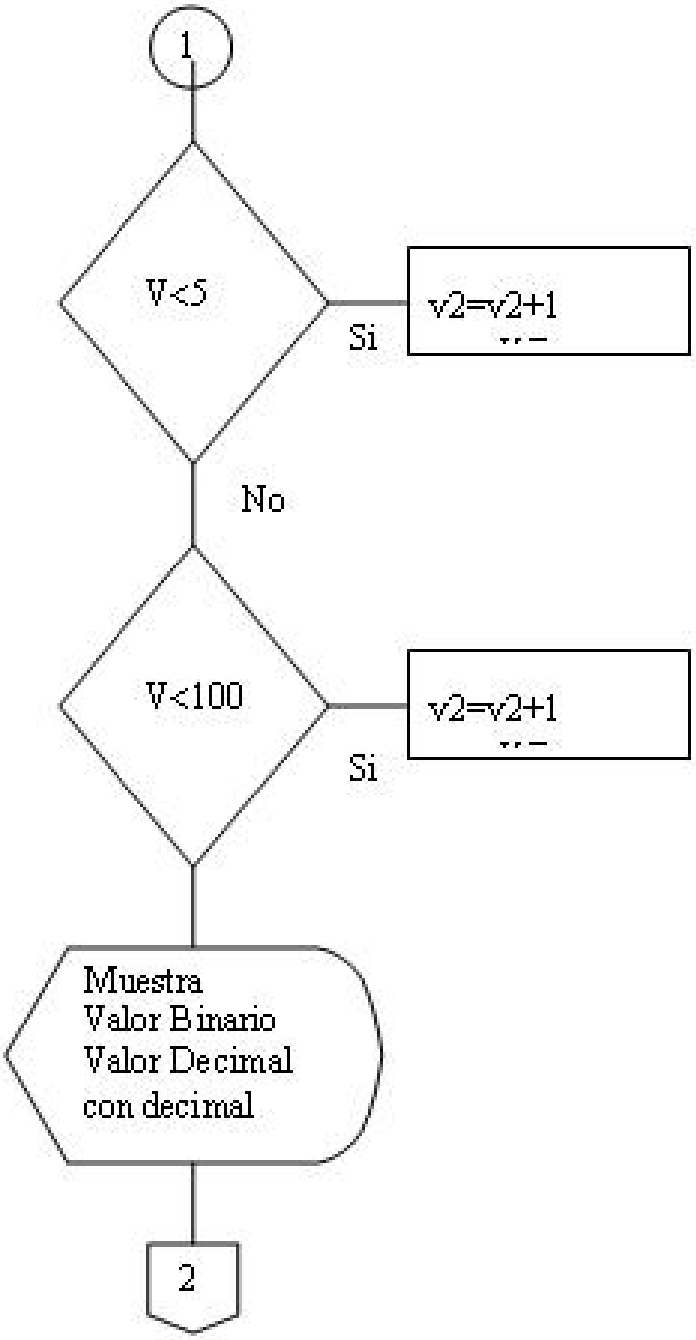


Figura 5.2 Diagrama de flujo


1
V<5
Si
v2=v2+1
No
V<100
Si
v2=v2+1
Muestra
Valor Binario
Valor Decimal
con decimal
2

Descripción del Programa.

La rutina Proceso de conversión A_D ejecuta 3 subrutinas diferentes indefinidamente. Las subrutinas se llaman Datos_ADC0831, CALC_VOLTS, y MOSTRAR.

Gosub Datos_ADC0831 da la orden de ir a la subrutina rotulada A Datos_ADC0831: y volver al finalizar su ejecución. El programa salta a la etiqueta Datos_ADC0831: y comienza a ejecutar comandos. Tan pronto como se encuentra con el comando return, el programa regresa a la línea inmediatamente posterior a gosub Datos_ADC0831. En este caso, el siguiente comando es otra instrucción gosub, más específicamente: gosub CALC_Tension:

La subrutina Datos_ADC0831: envía señales de control y recibe datos del ADC0831. P0 en el Micro está conectado al pin /CS del ADC0831. De la misma forma, los pines P1 y P2 están conectados a CLK y D0. Cuando se envían señales al pin /CS, podemos ingresar un comando como high CS en lugar de high 0. El comando high CS envía una señal de estado alto al pin /CS del ADC0831. Para iniciar la conversión, necesitamos enviar un estado alto (5 Volts), seguido por un estado bajo (0 Volts) en la entrada /CS del conversor de voltaje usando low CS. La señal enviada a la entrada /CS del ADC0831 debe permanecer en estado bajo durante la conversión.

high CS
low CS

El comando low CLK es necesario para que los pulsos de reloj tengan la forma correcta. Al usar este comando se garantiza que al usar la siguiente instrucción (pulsout) se enviará un pulso de reloj con la forma correcta, bajo-alto-bajo; los comandos high y low son para generar estados altos y bajos.

El comando pulsout CLK,210 envía un pulso de reloj a la entrada CLK del ADC0831. Este es el primer pulso de reloj y todo lo que hace es avisar al ADC0831 que inicie la conversión en el siguiente pulso de reloj. Por este motivo, es necesario controlar el estado de la entrada D0 con anterioridad a este pulso. pulsout CLK,210

Dado que se puso el reloj en estado bajo justo antes de este comando, pulsout envía la señal bajo-alto-bajo deseada. La duración en estado alto es dos veces el número especificado en el comando pulsout, en microsegundos (µs). 1 µs = 1/1.000.000 segundos. Por lo tanto, la duración de este pulso es de 2 µs × 210 = 420 µs.

El comando shiftin D0,CLK, msbpost, [adc0831\8] es una instrucción que se encarga de realizar toda la comunicación serial sincrónica, así que no se tendrá que programar manualmente, este comando envía las señales de control a la entrada CLK del ADC0831 y lee los bits de la salida D0 del ADC0831. Este comando también carga cada uno de los bits de salida del ADC 0831 en el byte adc0831.

shiftin D0,CLK,msbpost,[adc0831\8]

En forma general el comando shiftin contiene el siguiente formato

shiftin pin de datos, pin de reloj, modo, [variable \ bits]

shiftin D0, CLK, msbpost, [adc0831\8]

En este caso, el pin de datos es D0, con una constante igual a 2. esta constante es usada como referencia al pin de E/S P2 del Micro en este programa. Del mismo modo, el pin de reloj es CLK, que es una constante igual a 1 y hace referencia al pin de E/S P1. El modo en este caso es msbpost que es uno de los cuatro modos de transmisión que pueden ser usados con este comando. Indica que se transmite primero el MSB (bit más significativo) y que los bits de salida del ADC0831 están listos después del flanco descendente del reloj (post). Los parámetros entre corchetes [adc0831\8], indican que los bits se desplazan dentro de la variable adc0831 y se esperan 8-bits.

Salida del adc0831.

Al mover el cursor del potenciometro el adc0831 mide la tensión analógica que se presenta en su entrada; luego envía al Micro un número binario que describe el valor medido, para esto se tomo una escala de tensión que comienza en 0 Volts y termina en 5 Volts con un número binario de 8-bits, se toma este número de acuerdo a la fórmula:

$$\text{No de combinaciones} = 2^{bits}$$

Esto significa que la cantidad de combinaciones es igual a dos elevado a la cantidad de bits. Para 8-bits, la cantidad de combinaciones es $2^8 = 256$.

Traducido a números decimales, es lo mismo que contar de 0 a 255. Cuando se aplica a una escala de 5 Volts que comienza en 0 Volts, es lo mismo que contar de 0 a 5 Volts usando 255 pasos de tensión; cuando el ADC0831 mide 0 Volts, se obtiene 00000000. Cuando mide 5 Volts, la salida es 11111111. se puede calcular que la salida de 10110100 es igual al número decimal 180. Esto corresponde a una tensión medida de 3.53 Volts.

Para convertir un número de binario a decimal se necesitan dos pasos[9].

Convertir el número Binario 10110100 a decimal

- El primero es multiplicar cada bit por su potencia dos de acuerdo al bit menos significativo (Derecha) en este caso 0 al más significativo 1 (Izquierda).

Mas Significativo Menos Significativo

$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
1	0	1	1	0	1	0	0

128 x 1 = 128
64 x 0 = 0
32 x 1 = 32
16 x 1 = 16
8 x 0 = 0
4 x 1 = 4
2 x 0 = 0
1 x 0 = 0

Segundo, sumar los valores decimales:

$$\begin{array}{rr} & 128 \\ + & 32 \\ & 16 \\ & 4 \\ \hline & 180 \end{array}$$

Cálculo de la Tensión

Ahora se sabe que el equivalente decimal de la salida binaria del ADC0831, se podra realizar unos cálculos para obtener la tensión medida. Para averiguar a que tensión corresponde el número decimal, se debe calcular a que punto del rango de tensión corresponde este número.

[9] Sistemas digitales principios y aplicaciones Ronald Tocci 1999

Éste es un método para razonar el problema.

- Se sabe que la tensión está en el rango de 0 a 5 Volts y también se sabe que la salida del ADC0831 está en el rango de 0 a 255.
- En otras palabras, la tensión es a 5 como la salida del conversor es a 255.

Esto se traduce en la ecuación:

$$\frac{Tensión}{5} = \frac{Salida\,Decimal\ A/D}{255}$$

Despejando esta ecuación para calcular la tensión:

$$Tensión = \frac{5\,(Salida\,Decimal\ A/D)}{255}$$

Ahora si se multiplica por 5 y se divide por 255 se obtendra una escala de 5 Volts con 256 niveles. Se podra calcular la tensión donde la salida del ADC0831 es 10110100 = 180. La tensión medida es:

$$Tensión = \frac{5\,(180)}{255} = 3.52$$

Para calcular esta tensión, se debe ser expresada en código PBASIC

$$V = 5 * adc0831/255$$

Se cálculo que 180 correspondería a una tensión de entrada de 3,52 Volts. Los 003 Volts que aparecen en el ejemplo tienen redondeo a valores enteros. Esto se debe a los comandos de PBASIC ya que trabajan solamente con aritmética entera. Los enteros son los números que se usaron. El entero más grande que puede procesar el Micro SX48BD es 65535.

```
Valor binario de 8-bits: 10110100
Valor decimal: 180
Tensión: 003 Volts
```

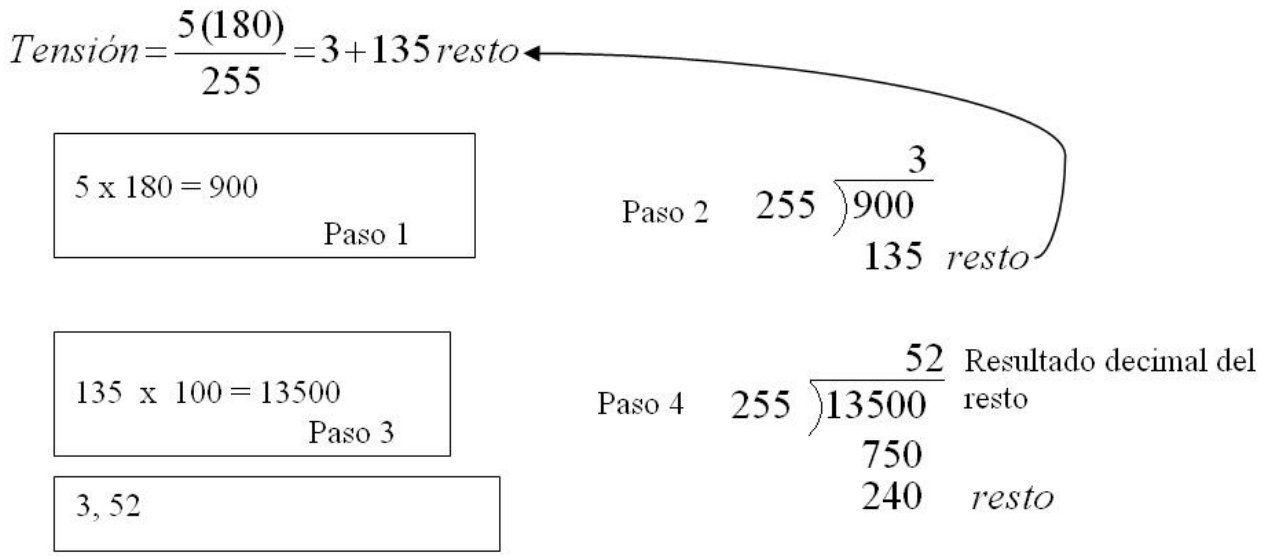
Sin embargo se puede usar aritmética entera para encontrar los valores fraccionarios.

En el ejemplo de cálculo de tensión es 5 x180 = 900 Dado que 900 es un entero que es menor que 65535. El inconveniente se presenta al dividir 900 entre 255.A continuación se da un ejemplo:

Cómo calcular la respuesta para este problema de división.

$$Tensión = \frac{\mathbf{5}(Salida\,Decimal)}{\mathbf{255}} = \mathbf{3} + un\ resto$$

El proceso para calcular la parte fraccionaria de la división es repetitivo. Se multiplica el resto por 10, luego se divide entre 255, obteniendo otro resto que se multiplica por 10 y divide entre 255 otra vez, etc. Para calcular directamente los dos primeros decimales podemos tomar el resto y multiplicarlo por 100, luego dividimos por 255.



La respuesta usando este algoritmo es el entero 3 a la izquierda de la coma y el entero 52 a la derecha. Dado que solamente se uso aritmética entera en este cálculo, debe funcionar con PBASIC en el Micro.

Sin embargo otra caracteristica positiva del lenguaje PBASIC es que cuenta con el operador mátematico División (//) Residuo para calcular el resto entero de una división. El operador para la división es / y el del resto es //. Ahora se pasara al algoritmo en código PBASIC para que realice esta operación.

Los pasos de la división que se mostraron, corresponden a los comandos PBASIC que se usaran.

5 x adc0831= v

$$255\overline{)v}\quad dec\ 1$$
$$\text{Re}\ sto$$

R x 100 = v2

$$255\overline{)v2}\quad dec\ 2$$
$$\text{Re}\ sto$$

R = 5 x adc0831//255

dec1 , dec 2

Para visualizar el valor fraccionario en la pantalla, se soluciona imprimiendo una coma "**,**" entre ambos valores.

debug dec1 v, "**,**", dec2 v2, " Volts"

Para el redondeo correcto de las centésimas, se requiere conocer el dígito de las milésimas. Usando las reglas de división entera, para esto se crea una variable v3; se realiza el mismo cálculo de v2, y dividir nuevamente entre 255.

$$255\overline{)(100x\ 240)} = \quad 255\overline{)24000}\ \begin{matrix}94\\1050\\030\end{matrix}$$

v3=100*R//255

Tesión = 3.594 94 corresponde a las milesimas

v3=10*v3/255

Para evitar el uso de otra variable, v3 es redefinida. El valor de v3 a la derecha del signo (*) corresponde al mismo valor que se cálculo en la primer línea. El valor de v3 a la izquierda del igual es el valor redefinido, que es diez veces el anterior v3, dividido por 255.

v3 = (10x94)/255 = 36

Una vez que se conoce el valor del dígito de las milésimas, se aplican las siguientes reglas de redondeo:

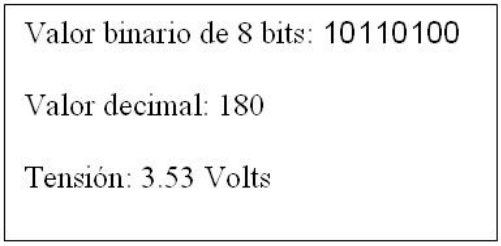
- Si el dígito de las milésimas es mayor a 5, agregar 1 a las centésimas.
- Caso contrario, respetar centésimas.
- En todos los casos, truncar los valores más allá de las centésimas.

Dado que el valor v2 ya está truncado, solamente necesitamos el código para decidir si le sumamos 1 o no a las centésimas.

```
if v3<5 then opción_2
v2=v2+1
opción_2:
```

Dado que el valor de las unidades está almacenado en otra variable, necesitamos verificar si al agregar uno a las centésimas es necesario o no, incrementar ese valor. Sin este código, 3,996 se redondearía a 3,00 en lugar de 4,00.

```
if v2<100 then saltea_todo
opción_2:
if v2<100 then sal_de_redondeo
v=v+1
v2=0
sal_de_redondeo:
return
```



Desafío:

Simule el control de temperatura de un horno, un fermentador de cerveza, elabore diagrama de flujo, comente 3 ejemplos de procesos que se pueden generar a parte de este.

Como ya se comento el conversor ADC0831 tomará una tensión de entrada y la convertirá en un dato digital de 8-bits con un rango de 256 representaciones binarias posibles. La tensión en Vin(-) Pin 3 determina el valor cero. La tensión Vref en el Pin 5 determina el rango de tensión por encima de Vin(-) sobre el que se aplica la resolución de 255 pasos. Fijando Vin(-) a 0,7 limitará el rango a 70 grados. Vref en 0,5 Volts da como resultado un rango de 50 grados. Estas tensiones se utilizan para enfocar el rango del dispositivo A/D de forma que cubra la tensión analógica que se requiere convertir. En este caso se dara aplicación a un sensor de temperatura LM34DZ[10] funcionará desde temperatura ambiente (70^{o}) hasta 120^{o}F. Este rango de temperatura equivale a una salida de tensión del LM34 de 0,70V a 1,20V. Para maximizar la resolución, el rango de interés es 0,5 V (0.7V a 1,2v); y la referencia, llamada offset, existe 0,7 V, cada paso binario representa aproximadamente 0,2 grados. Esto permite cálcular la temperatura en forma muy precisa. La tensión de salida del sensor cambia 10 mV por grado Fahrenheit y está referenciada a 0 grados. Utilizando el Voltimetro se podra visualizar la tensión de salida del LM34 de 0,01 Volts por grado F. Simplemente se desplaza el punto decimal de la lectura dos lugares a la derecha para convertir el valor a temperatura. Por ejemplo; 0,75 V representa 75^{o}F; 0,825 V representa $82,5^{o}$F.

Dado que el conversor A/D de 8- bits tiene una resolución de 255, en un rango de 50 grados puede detectar cambios de 0,31 grados. La conversión generará un número binario igual a [(Vin – 0,7) /0,5] x 255. Por ejemplo la temperatura es 98,6 F, que resulta en una salida del LM34 de 0,986 Volts.

$$\left(\frac{0,986-0,7}{0,5}\right)(255)=145,86$$

Si se trunca el valor se obtendra el entero 145.

El valor binario sería 1001 0010.

Este valor se retendrá y estará listo para la transferencia.

[10] Vea Anexo 1

Las instrucciones LOW CLK y pulsout CLK,10 le dicen al conversor A/D que haga una conversión de la tensión en Vin(+) en este momento. El ADC0831 es un conversor de aproximaciones sucesivas de 8-bits. Sus 256 combinaciones digitales posibles se distribuyen uniformemente por el rango de tensión determinado por los potenciales en los pines V in(-) y Vref. Vin(-) define la tensión a la cual 0000 0000 se obtendrá de la conversión. Vref define el rango de tensiones de entrada por encima de este punto en el que se distribuirán las otras 255 combinaciones digitales. La Figura 5.3 muestra los valores de Cero y Rango para esta aplicación.

Decimal	Binario	Voltaje
255	1111 1111	1,20
224	1110 0000	1,198
192	1100 0000	1,076
160	1010 0000	1,014
128	1000 0000	0,951
127	0111 1111	0,949
96	0110 0000	0,888
64	0100 0000	0,825
32	0010 0000	0,763
0	0000 0000	0,700

Fig. 5.3

Con esta configuración, el ADC0831 se enfoca en un rango de temperatura de 70 a 120 grados. Hay una cantidad infinita de valores de temperatura dentro del rango de salida de 0,7 a 1,2 Volts del LM34. Sin embargo, solamente pueden obtenerse unos pocos valores representativos. Dado que el conversor A/D de 8-bits tiene una resolución de 255, en un rango de 50 grados puede detectar cambios de 0,31 grados.

La instrucción SHIFTIN está diseñada para realizar una comunicación sincrónica entre el Micro y dispositivos seriales como el ADC0831.
A continuación se presenta el programa.

```
' -----[ Conversión Analógica a Digital]-------------------------------------
' Archivo..... Práctica No 5
' Propósito..... Uso del Micro manejo de señales analógicas.
' Carrera Ing. Industrial
'{$STAMP BS2p}

CS CON 3                    ' Chip select del 0831 activo bajo en P3 del Micro
CLK CON 4                   ' Clock del Micro (P4) al 0831
Dout CON 5                  ' Salida serial de datos del 0831 al Micro (P5)
adc0831\in VAR Byte         ' Variable para almacenar el número entrante (0 a 255)
Temp VAR Word               ' Almacena valor convertido que representa la Temp.
TempRango VAR Word          ' Rango completo en décimas de grado.
TempRango = 5000            ' Define el rango para Vref = 0,50V
Offset VAR Word             ' Temperatura mínima. Offset, ADC = 0
Offset = 700                ' Declara Temp. mínima para Vin(-) = 0,7 V
EspTrabajo1 VAR Byte        ' Espacio de trabajo para el comando BUTTON de PB1
EspTrabajo1 = 0             ' Limpia el espacio de trabajo antes de usar BUTTON
LOW 8                       ' inicializa el calentador apagado
Principal:
GOSUB Obtenerdato
GOSUB Calc_Temp
GOSUB Control
GOSUB Mostrar
GOTO Principal
Obtenerdato:          ' Adquiere la conversión del 0831
LOW CS                      ' Selecciona el chip
LOW CLK                     ' Alista la línea clock.
PULSOUT CLK,10              ' Envía un pulso de 10 uS al clock del 0831
SHIFTIN Dout, CLK, MSBPOST,[ adc0831\8]              ' Desplaza los datos al BS2
HIGH CS                     ' Detiene conversión
RETURN
Calc_Temp:                  ' Convierte el valor digital a temperatura
Temp = TempRango/255 * adc0831/10 + Offset
RETURN
Control:                    ' Control manual del calentador
BUTTON 1,1,255,0,EspTrabajo1,1,Invertir
RETURN
Invertir:
TOGGLE 8
RETURN
Mostrar:                    ' Grafica Temp, ADC binaria y estado de Temp
DEBUG DEC Temp,CR
DEBUG IBIN OUT8,CR
DEBUG "!USRS Temp.= ", DEC Temp," ADC Data in = %", BIN Datain, " Decimal", DEC Datain,
CR
RETURN
```

Arme el siguiente diagrama como se muestra a continuación

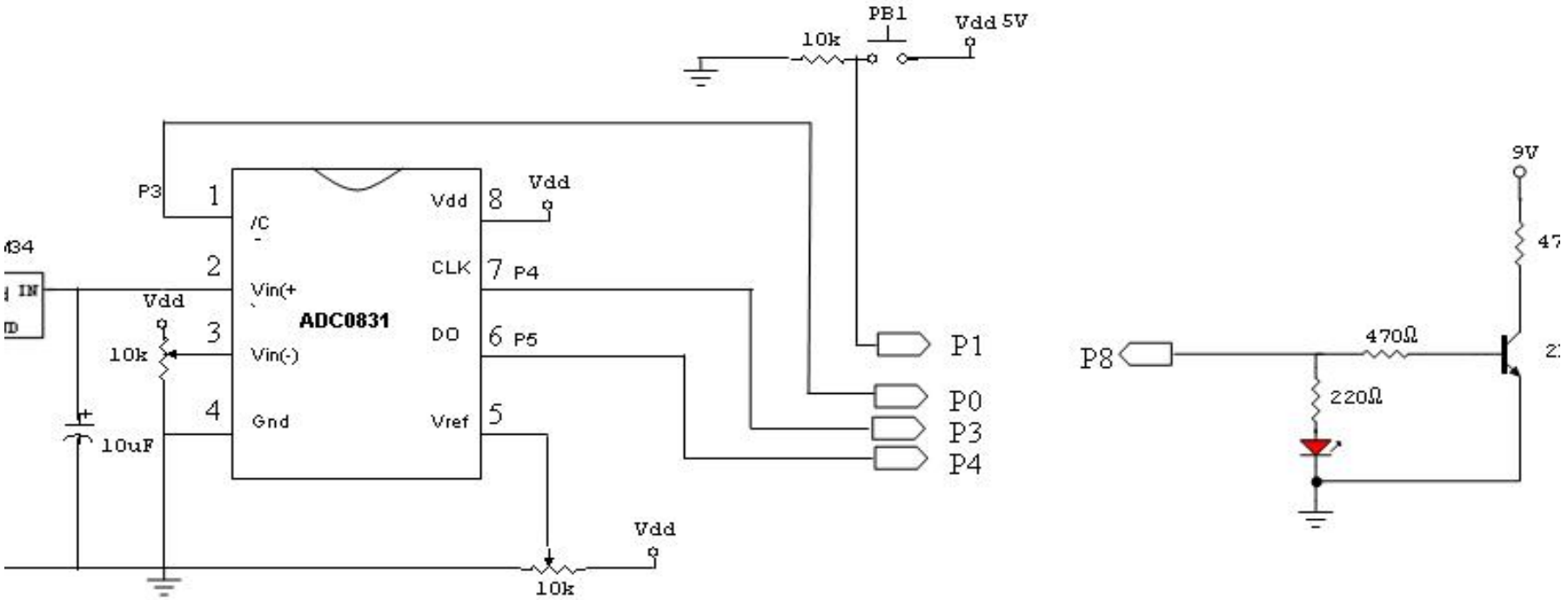


Fig. 5.4

Capítulo 3.3 "Muestreo de tensión Sumador de Red resistiva."

No. DE PRÁCTICA: 6 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Programar al Micro para que envié niveles de tensión binarios a una red resistiva; para lograr que la red efectué la conversión de un dato digital a un dato analógico.

INTRODUCCIÓN:

En este experimento, se construira un conversor D/A usando resistores. Este circuito se llama red resistiva en escalera y se le pueden quitar o agregar resistores para modificar la resolución. Con una red resistiva en escalera, si comienza con un conversor de 4-bits y quiere aumentar la resolución en 1-bit, solamente se debe agregar dos resistores a la red.
La cantidad de niveles de tensión que un conversor D/A puede producir, está determinada por la cantidad de bits binarios que puede manejar, lo que está expresado en la resolución. Podemos usar nuevamente la ecuación de combinaciones para obtener la cantidad.

$$\text{combinaciones} = 2^{\text{bits}}$$

El conversor D/A que usaremos en este experimento tiene una resolución de 4-bits, así que la cantidad de niveles de tensión de salida será:

$\text{combinaciones} = 2^{\text{bits}} = 2^4 = 16$

MARCO TEÓRICO:

Se programara al Micro para lograr que la red efectúe la conversión D/A. El PBasic se usará para programar al SX48BD para que envíe a la red un conjunto de niveles de tensión binarios o señales digitales.

Estas son variables eléctricas con dos niveles diferenciados que se alternan en el tiempo transmitiendo información según un código previamente acordado. Cada nivel eléctrico representa uno de dos símbolos: 0 ó 1. Los niveles específicos dependen del tipo de dispositivos utilizado. Por ejemplo si se emplean componentes de la familia lógica TTL (transistor-transistor-logic) los niveles son 0 V y 5 V, aunque cualquier valor por debajo de 0,8 V es correctamente interpretado como un 0 y cualquier valor por encima de 2 V es interpretado como un 1 (los niveles de salida están por debajo de 0,4 V y por encima de 2,4 V respectivamente). En el caso de la familia CMOS (complementary metal-oxide-semiconductor), los valores dependen de la alimentación. Para alimentación de +5 V, los valores ideales son también 0 V y 5 V, pero se reconoce un 0 hasta 2,25 V y un 1 a partir de 2,75 V. Estos ejemplos muestran uno de los principales atractivos de las señales digitales: su gran inmunidad al ruido.

MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Módulo completo
1	Computadora	
1	Cable serial RS-232	
1	Led	Color rojo
6	Resistores de 2 KΩ,	Vea apéndice A.
1	Resistor de 1 KΩ	Vea apéndice A
1	Resistor de 270 Ω	Vea apéndice A
–	Cable telefónico	Varios colores.
1	CI Convertidor ADC0831	Vea anexo 1.

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme el circuito que se muestra en la Figura 6.1

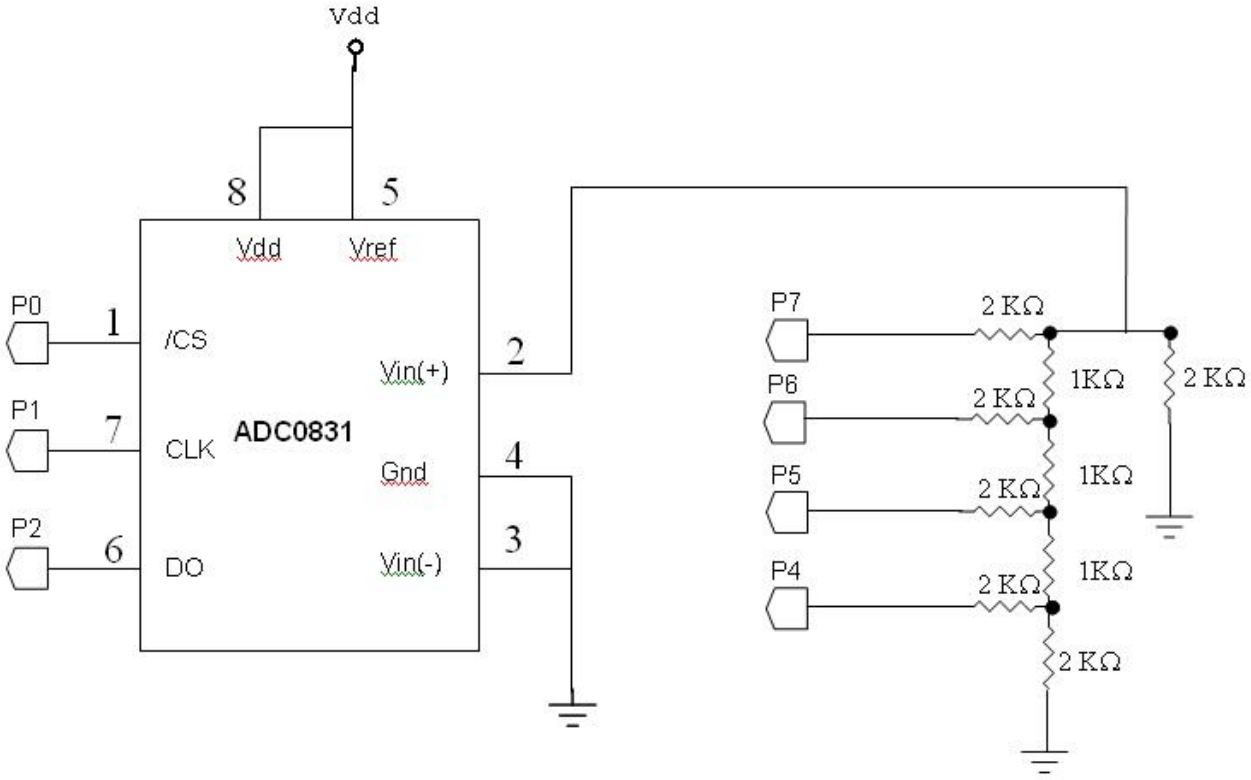


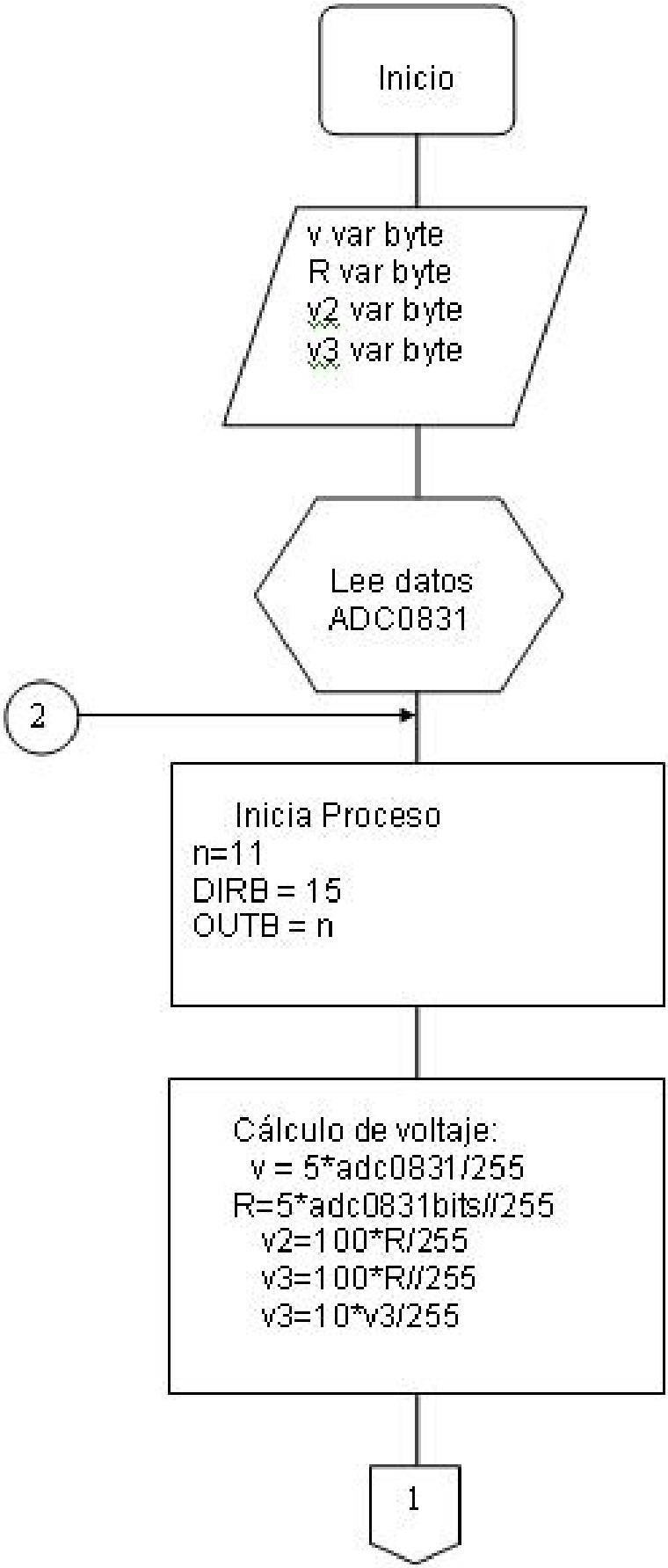
Fig. 6.1

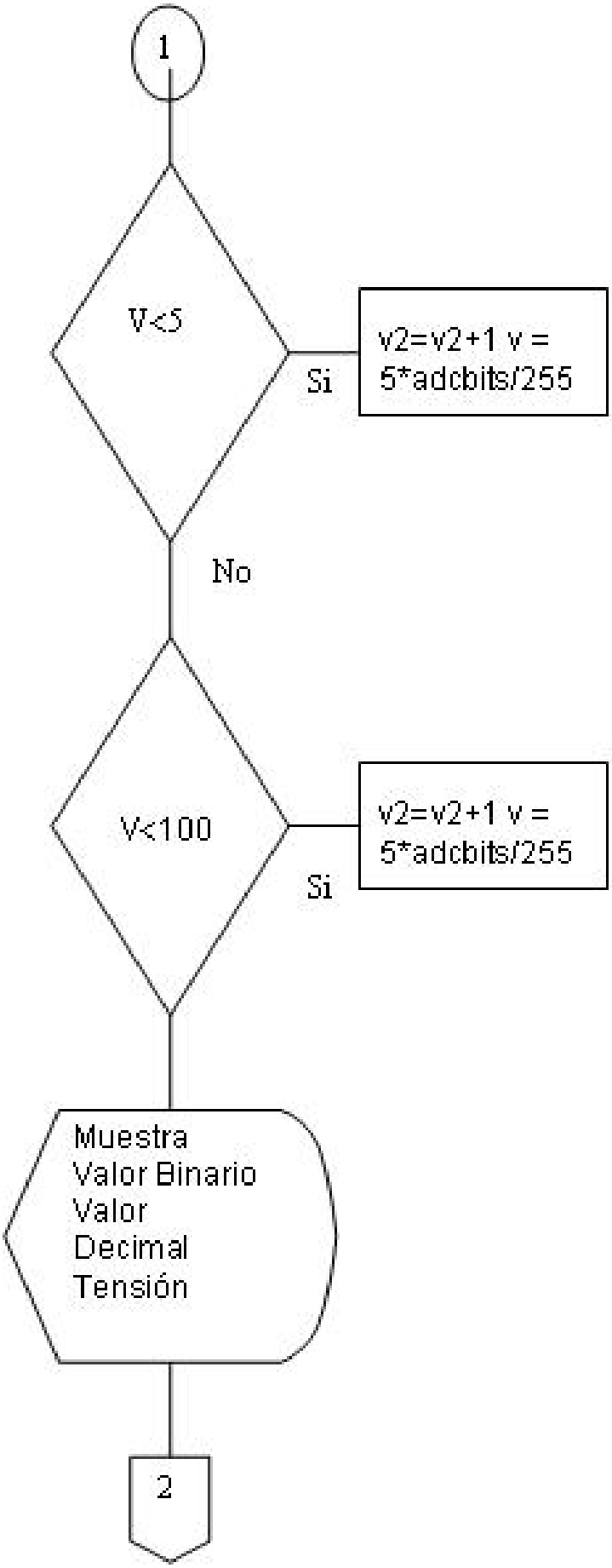
```
' -----[ Conversión Digital a Analógica]-------------------------------------
' Archivo..... Práctica No 6
' Propósito..... Uso del Micro en conversión Digitales.
' Carrera Ing. Industrial
 '{$STAMP BS2p}
adc0831 VAR Byte
v VAR Byte
R VAR Byte
v2 VAR Byte
v3 VAR Byte
n VAR Nib
CS CON 0
CLK CON 1
D0 CON 2
DEBUG CLS
Proceso_de_conversionD_A:
GOSUB CDA
GOSUB Datos_ADC0831
GOSUB CALC_Tension
GOSUB MOSTRAR
GOTO Proceso_de_conversionD_A:
CDA:
n=11
DIRB = 15
OUTB = n
RETURN
Datos_ADC0831:
HIGH CS
LOW CS
LOW CLK
PULSOUT CLK,210
SHIFTIN D0,CLK,MSBPOST,[adc0831\8]
RETURN
CALC_Tension:
v = 5*adc0831/255
R=5*adc0831//255
v2 =100*R/255
v3 =100*R//255
v3 =10*v3/255
```

```
IF v3<5 THEN opcion_2
v2=v2+1
 opcion_2:
IF v2<100 THEN sal_de_redondeo
v=v+1
v2=0

sal_de_redondeo:
RETURN
MOSTRAR:

DEBUG HOME
DEBUG "valor binario de 8-bits: ", BIN8 adc0831
DEBUG CR, CR, "Valor decimal: ", DEC3 adc0831
DEBUG CR, CR, "Tensión: ", DEC3 v, " Volts"
DEBUG DEC1 v, ".", DEC2 v2, " Volts"
RETURN
```


Inicio
v var byte
R var byte
v2 var byte
v3 var byte
Lee datos
ADC0831
2
Inicia Proceso
n=11
DIRB = 15
OUTB = n
Cálculo de voltaje:
v = 5*adc0831/255
R=5*adc0831bits//255
v2=100*R/255
v3=100*R//255
v3=10*v3/255
1


1
V<5
Si
v2=v2+1 v =
5*adcbits/255
No
V<100
Si
v2=v2+1 v =
5*adcbits/255
Muestra
Valor Binario
Valor
Decimal
Tensión
2

Descripción del programa.

Ciertos comandos de PBasic permiten direccionar directamente los pines de E/S como una word (16 bits individuales), dos bytes (dos juegos de 8 bits individuales) o como 4 nibbles (cuatro juegos de cuatro bits individuales). En este caso se requiere direccionar un nibble a la vez, para controlar los cuatro bits que están usando P4-P7.

Direccionamiento del puerto P0-P15

Una vez se direcciona un pin o un puerto este permanecerá indefinidamente en ese estado o hasta que se le indique otra dirección. Para direccionar un pin como salida bastara con:

• DIR0 = 1 Direcciona el Pin 0 como salida

Para direccionar un pin como entrada bastara con:

• DIR0 = 0 Direcciona el Pin 0 como entrada

Un uno (1) direcciona un pin como salida, mientras que un cero (0) direcciona un pin como entrada. El direccionamiento se coloca por lo general al principio del programa. Si se quiere direccionar el puerto completo como salida el formato será (DIRS = %1111111111111111), el registro DIRS contiene el puerto completo.

En la siguiente tabla se tiene que (DIRD = %0000), (DIRC = %1111), (DIRB = %1101) Y (DIRA = %0001).

DIRD				DIRC				DIRB				DIRA			
0	0	0	0	1	1	1	1	1	1	0	1	0	0	0	1
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0

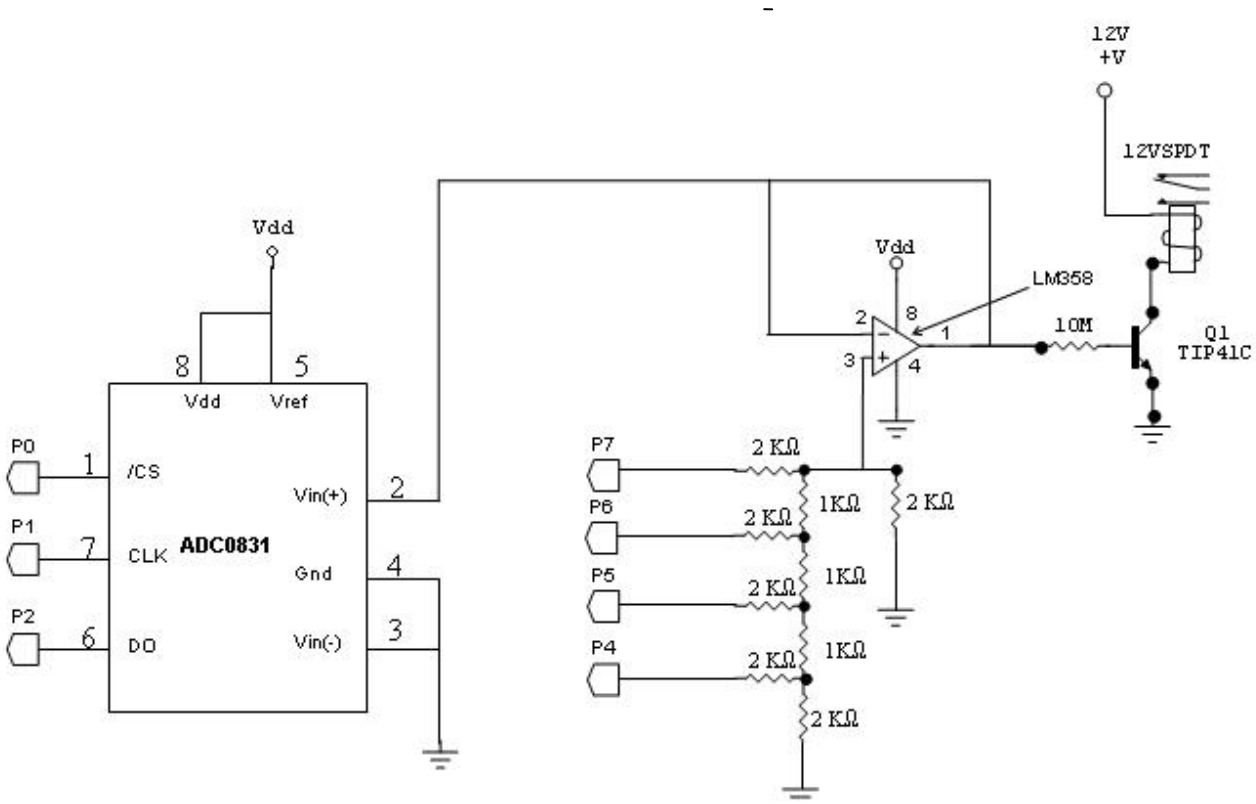
Esto quiere decir que el puerto D, esta definido como entrada, el puerto C, esta definido como salida, el puerto B, contiene 3 salidas y una entrada y el puerto A, contiene 3 entradas y una salida. Nótese en la tabla anterior el orden de los pines de (P15 – P0). Este es el orden que siempre debe llevar para mantener la secuencia. Del BIT más significativo y el menos significativo.

Esta definición también seria equivalente a (DIRS = %0000111111010001) o equivalente a (DIRH = %00001111) y (DIRL = %11010001). Este formato es equivalente para los registros INS y OUTS.

El Seguidor de Tensión

Se utiliza el barrido de tensión para analizar lo que sucede cuando la salida del conversor D/A es conectado a otro circuito para controlar un sistema. En este caso simulemos la salida del conversor D/A para alimentar un relevador.

Arma el circuito que muestra a continuación Fig. 6.2



Desafío:

- Explica por que motivo se utilizo un buffer en el circuito anterior para alimentar al rele y cite otros ejemplos
- Cite por lo menos tres metodos mas para la conversión de digital a analógico por medio de red resistivas

Capítulo 3.4 "Conversión de digital a analógico.."

No. DE PRÁCTICA: 7 No. DE SESIONES:

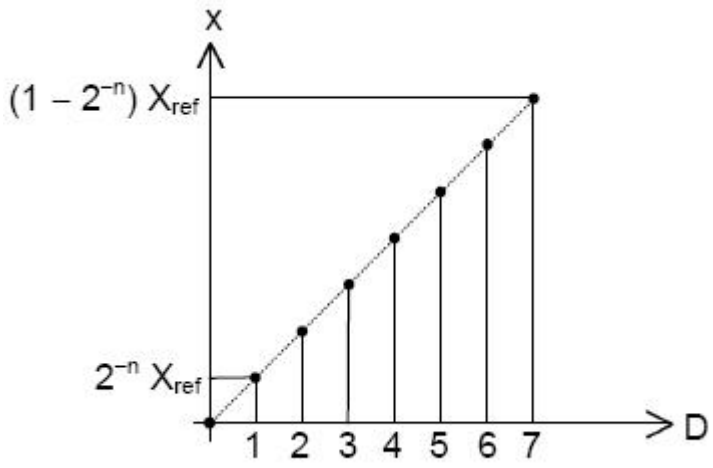
No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Mostrar los principios del control basados señales digitales en el microcontrolador e instruirlo en el conexionado con dispositivos de salida analógicos.

INTRODUCCIÓN:

Para una conversión de Digital a Analógico se parte de una señal digital $D = d_n d_{n-1} \ldots d_1$ en paralelo que responde a la codificación binaria natural y una referencia X_{ref} (podría ser una tensión o una corriente) y pretendemos obtener una señal analógica x que varíe en saltos iguales a $X_{ref}/2^n$ entre 0 y $(2^n - 1)\, X_{ref}/2^n = X_{ref}(1 - 2^{-n})$, como se muestra en la figura 7.1



La estructura genérica de este tipo de conversores es la que se indica en la figura 7.2

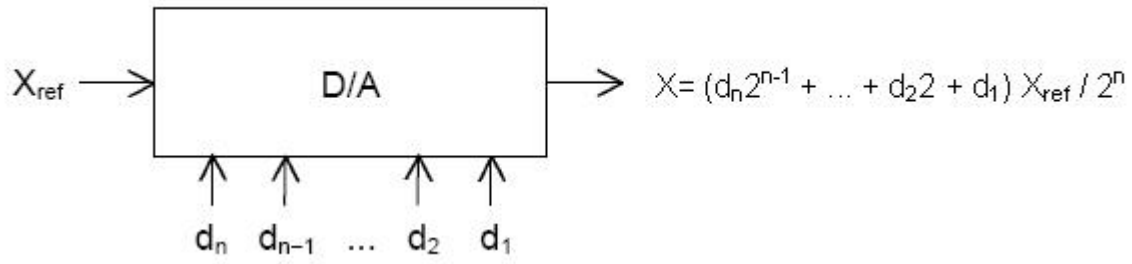


Fig. 7.2

Para esto existen varios métodos como:

- Método de conmutación de corrientes ponderadas
- Redes escalera
- Método de la red escalera R-2R
- Conversor Multiplicativo

Siguiendo una de las ventajas del Micro; puede ser programado, para que de forma directa controle una conversión digital a analógico; el lenguje PBASIC cuenta con un comando PWM (modulación por ancho de pulso).

El comando de modulación por ancho de pulso (PWM) Convierte la señal digital en análoga permite al Micro generar una tensión promedio para controlar el dispositivo analógico. La sintaxis del comando PWM se muestra a continuación.

Convierte la señal digital en análoga por modulación de pulsos.

PWM Pin, Trabajo, Ciclos.

- Pin: especifica el pin que generará la tensión de salida.
- Trabajo: es un valor entre 0 y 255 que expresa la salida analógica promedio entre 0 y 5 Volts.
- Ciclos: es un valor entre 0 y 255 que especifica la duración del PWM en milisegundos.

MARCO TEÓRICO:

Para una aplicación efectiva de los conversores digital-analógicos es preciso conocer y saber interpretar las especificaciones de los mismos, ya que ponen de manifiesto las limitaciones así como las verdaderas prestaciones, que en muchos casos difieren considerablemente de la idealidad. A continuación se presenta una descripción de las especificaciones más importantes.

- Resolución: Es la cantidad de bits o dígitos binarios que acepta en su entrada.
- Exactitud: Es la máxima desviación respecto a la línea recta que une el mínimo y el máximo valores ideales
- Error de escala: Es el error que se obtiene a fondo de escala con respecto al valor ideal
- Error de offset: Es el valor de salida obtenido cuando la entrada es nula.
- No linealidad: El valor ideal es 0 LSB.
- No linealidad diferencial: Es la máxima diferencia entre un salto a la salida debido a un cambio de 1 LSB y el salto ideal.
- Monotonía: Es la cualidad de generar valores analógicos crecientes ante códigos digitales de entrada crecientes.
- Tiempo de establecimiento: Es el máximo tiempo transcurrido luego de un cambio de código de entrada arbitrario para alcanzar el valor analógico correspondiente con un error de ± 0,5 LSB.

MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Módulo completo
1	Computadora	
1	Cable serial RS-232	
1	Led	Color rojo
1	Resistor de 47 Ω ½ watt	Vea apéndice A
1	Resistor de 220 Ω	Vea apéndice A
1	Resistor de 470 Ω	Vea apéndice A
3	Resistores de 10 KΩ,	Vea apéndice A.
2	Preset de 10KΩ	
1	Capacitor ,68μf	Cerámico
1	Capacitor 10μf	Electrolítico
1	CI LM34DZ	Vea anexo 1
1	CI LM358	Vea anexo 1
1	CI ADC0831	Vea anexo 1
2	Trans. 2N3904	Vea anexo 1
1	Ventilador	9 - 12v CC
1	Interruptor de presión	Normalmente abierto
–	Cable telefónico	Varios colores.

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme los siguientes circuitos fig. 7.3a, 7.3b, 7.3c, 7.3d, en la Protoboard.

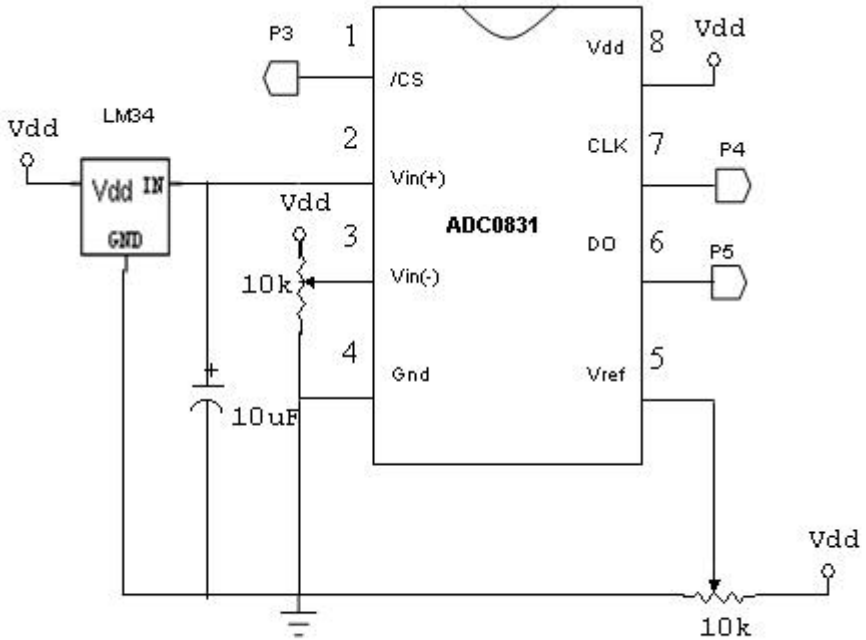


Fig. 7.3a Convertidor A/D

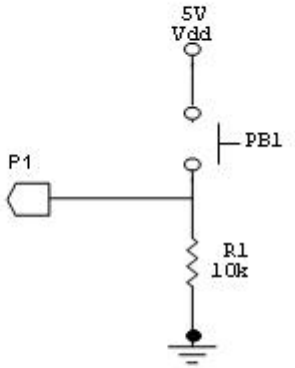
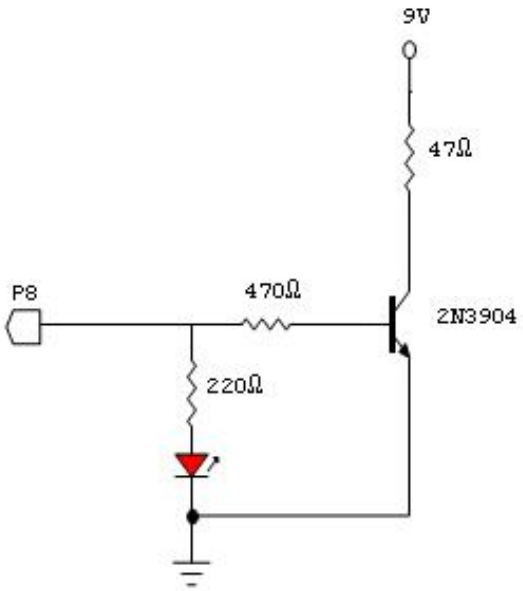


Fig. 7.3b Pulsador de Inicio



- Fig. 7.3c Calentador

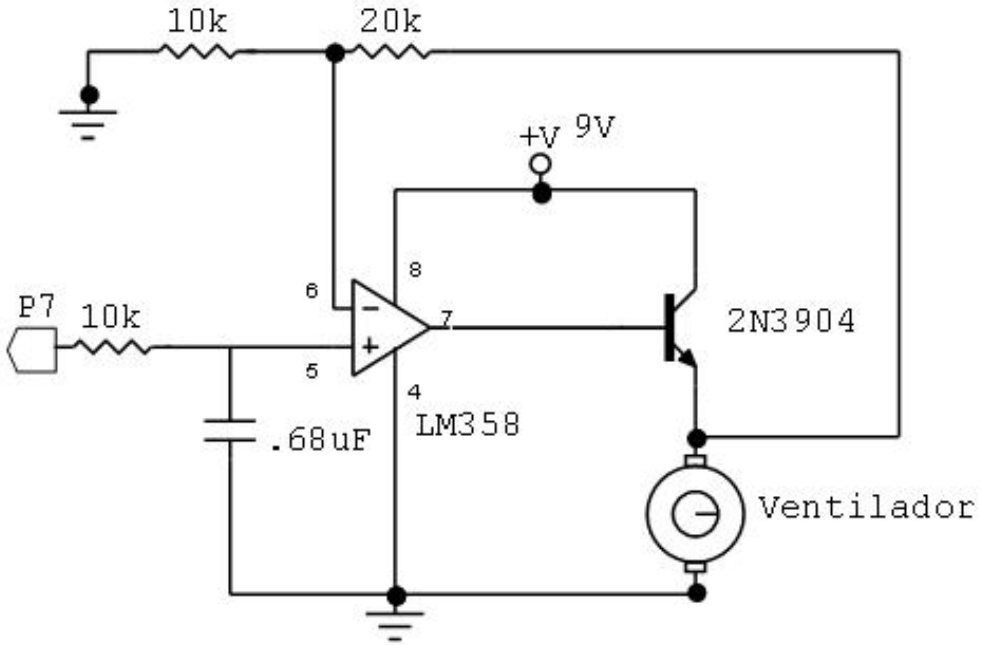


Fig. 7.3d Control del ventilador

```
' -----[ Conversión Digital a Analógica]-------------------------------------
' Archivo..... Práctica No 7
' Propósito..... Uso del Micro en conversión Digitales.
' Carrera Ing. Industrial 9

 '{$STAMP BS2p}
CS CON 3                ' 0831 chip select activo bajo (P3)
CLK CON 4               ' Clock del Micro (P4) al 0831
Dout CON 5              ' Salida serial del 0831 al  Micro (P5)
Datain VAR Byte         ' Variable para almacenar número (0 to 255)
Temp VAR Word           ' Almacena valor convertido en temperatura
TempRango VAR Word      ' Rango máximo en décimas de grado.
TempRango = 5000        ' Declara rango para Vref 0,50V
Offset VAR Word         ' Temp. mínima. @Offset, ADC = 0
Offset = 700            ' Declara el Cero para Vin(-) 0,7 Con estos
                        ' valores, la salida del ADC0831 será de 0 - 255
                        ' para temps de 700 a 1200 décimas de grado.
LOW 8                   ' Inicializa calentador apagado
contar VAR Word         ' Variable para llevar la cuenta
RPM VAR Word            ' Variable para RPM calculadas
Contar = 0              ' Limpia Contar
OUTPUT 7                ' Declara el pin para PWM.
x VAR Word              ' Variable Trabajo
y VAR Byte              ' Repeticiones por valor de Trabajo
Tvolts VAR Word
                        ' Obtiene y muestra valores iniciales.
GOSUB Obtenerdato
GOSUB Calc_Temp
Proceso:                ' Bucle principal
PAUSE 10
```

```
GOSUB Control
GOTO Proceso
Obtenerdato:                              ' Adquiere conversión del 0831
LOW CS                                    ' Selecciona el chip
LOW CLK                                   ' Prepara la línea clock.
PULSOUT CLK,10                            ' Envía un pulso de 10 uS al 0831
SHIFTIN Dout, CLK, MSBPOST,[Datain\8]     ' Recibe los datos
HIGH CS                                   ' Detiene la conversión
RETURN
Calc_Temp:                                ' Convierte el valor digital a temperatura
Temp = TempRango/255 * Datain/10 + Offset ' basándose en las variables Rango
y
RETURN                                    ' Offset.
Control :                                 ' Prueba el sistema a otro % ciclo de trabajo
FOR x = 70 TO 210 ' Variable Trabajo
FOR y = 0 TO 5 ' Prueba valor de Trabajo por 5 segundos
PWM 7, x, 50 ' Genera PWM a un Ciclo de Trabajo de x
Tvolts = 50 * x / 255 * 3 ' Calcula tensión en decenas de Volts.
COUNT 3,1000, Contar ' Contar los pulsos del pin 3 por 1 segundo
RPM = Contar * 60 ' Ajusta escala de RPM
NEXT
x = x + 9
NEXT
END
```

Descripción del programa.

PWM aplica pulsos por un período de tiempo definido por su parámetro duración. Durante el tiempo que se ejecuta el resto del programa no se aplica ninguna salida a la carga. Como resultado, la tensión promedio para un ciclo de trabajo del 100% (trabajo=255) generará valores menores que los esperados con tensión máxima. Cuanto más tiempo demore su ciclo de programa, mayor será la diferencia.

El programa anterior está desarrollado para estudiar la relación entre la modulación PWM sobre el calentador y la temperatura resultante. El programa aplicará niveles PWM en incrementos del 10%. Cada incremento durará aproximadamente cuatro minutos. El programa terminará luego de aplicar un nivel del 100%.

Desafío:

Debido a la respuesta rápida del ventilador a las fluctuaciones de tensión, el circuito de Sample and Hold es necesario para controlar efectivamente la velocidad, usando la instrucción PWM. Construya este circuito para que sea capaz de variar la velocidad del ventilador. Conecte el calentador directamente a la fuente +Vin. Con el ventilador apuntando directamente al envase, experimente con diferentes valores de PWM en el ventilador. Y determine que nivel es necesario para enfriar el envase a 101^{o}F.

Capítulo 4 "Dispositivos de salida, Display de 7 Segmentos cátodo común"

No. DE PRÁCTICA: 8 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

El propósito de este experimento es demostrar el uso de dispositivos de salida; display de 7 segmentos, creando un contador decimal simple; comparandolo con circuitos de lógica común.

INTRODUCCIÓN:

Los mapas de Karnaugh es uno de los métodos más prácticos, cuando el número de variables de entrada es menor o igual a seis. En general, el mapa de Karnaugh se considera como la forma gráfica de una tabla de verdad o como una extensión del diagrama de Venn. El explicar como se utiliza el mapa de Karnaugh en la minimización de funciones, Esto nace de la representación geométrica de los números binarios. Un número binario de n bits, puede representarse por lo que se denomina un punto en un espacio N. Para entender lo que se quiere decir con esto, considérese el conjunto de los números binarios de un bit, es decir 0 o 1.

Procedimiento para minimizar una función por mapas Karnaugh, los pasos a seguir serán los mismos para cualquier mapa, no importa cual sea el número de variables.

1. De la definición del problema y de la tabla funcional se obtiene la función canónica.
2. Los mini términos o maxi términos de la función canónica se trasladan al mapa K. Se coloca un 1 si es mini término y 0 si es maxi término.

3. Se realizan los enlaces abarcando el mayor número de términos bajo los siguientes criterios:
   a. El número de términos que se enlazan (agrupan), deben seguir la regla de formación binaria, es decir, de 1 en 1, de 2 en 2, de 4 en 4, de 8 en 8, etc.
   b. Al agrupar los términos, se debe cuidar la simetría con los ejes centrales y secundarios.
4. El hecho de que se haya tomado un término para un enlace no quiere decir que éste mismo no pueda utilizarse para otros enlaces.
5. La función reducida tendrá tantos términos como enlaces se hayan realizado.

Marco Teórico:

Ejemplo de diagrama lógico para cada uno de los elementos del display de 7 segmentos de acuerdo a su tabla de verdad tabla. 8.1 para el Circuito integrado DM7448

Decimal	Binario	Salidas						
	ABCD	a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
4	0100	0	1	1	0	0	1	1
5	0101	1	0	1	1	0	1	1
6	0110	0	0	1	1	1	1	1
7	0111	1	1	1	0	0	0	0
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	0	0	1	1
10	1010	0	0	0	1	1	0	1
11	1011	0	0	1	1	0	0	1
12	1100	0	1	0	0	0	1	1
13	1101	1	0	0	1	0	1	1
14	1110	0	0	0	1	1	1	1
15	1111	0	0	0	0	0	0	0

Tabla 8.1

Ejercicios de los mapas de Karnaugh

Seg. a

DC \ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	0
11	0	1	0	0
10	1	1	0	0

$$a = \overline{D}\,\overline{C}\,\overline{A} + B\,A\,\overline{D} + \overline{B}AC + D\,\overline{C}\,\overline{B}$$
$$a = \overline{D}\,(\overline{C}\,\overline{A} + BA) + B(AC + D\,\overline{C})$$
$$a = \overline{D}(B + \overline{C}) + \overline{B}(A + D)$$

Seg. b

DC \ BA	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	1	0	0	0
10	1	1	0	0

$$b = \overline{D}\overline{C}\overline{A} + \overline{D}\overline{C}A + BA\overline{D} + D\overline{C}B + \overline{B}\overline{A}$$
$$b = \overline{C}(\overline{D}\,\overline{A} + D\overline{B} + \overline{D}A) + BA\overline{D} + \overline{B}\overline{A}$$
$$b = \overline{D}\,\overline{C} + BA\overline{D} + \overline{B}\overline{A}$$

Seg. c

DC \ BA	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	0

$$c = \overline{C}\,\overline{B} + \overline{C}A + \overline{D}C$$
$$c = \overline{C}(\overline{B} + A) + \overline{D}C$$

Seg. d

DC \ BA	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	0	1	0	1
10	1	0	1	1

$$d = \overline{C}\,\overline{B}\,\overline{A} + C\overline{B}A + \overline{C}B + B\overline{A}$$
$$d = \overline{B}(\overline{C}\,\overline{A} + CA) + B(\overline{A} + \overline{C})$$
$$d = \overline{B}(\overline{C \oplus A}) + B(\overline{A} + \overline{C})$$

Seg. e

DC \ BA	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	0	0	0	1
10	1	0	0	1

$$e = \overline{C}\,\overline{B}\,\overline{A} + \overline{C}\,B\,\overline{A}$$

$$e = \overline{C}\,\overline{A}(B + \overline{B})$$

$$e = \overline{C}\,\overline{A}$$

Seg. f

AB \ CD	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	1	1	0	1
10	1	1	0	0

$$f = \overline{B}\,\overline{C}\,\overline{D} + B\overline{C} + A\overline{B}\,\overline{C} + BC\,\overline{D}$$

$$f = \overline{D}(\overline{B}\,\overline{C} + BC) + \overline{C}(B + A\overline{B})$$

$$f = \overline{D} + \overline{C}A$$

Seg. g

DC \ BA	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	1	1	0	1
10	1	1	1	1

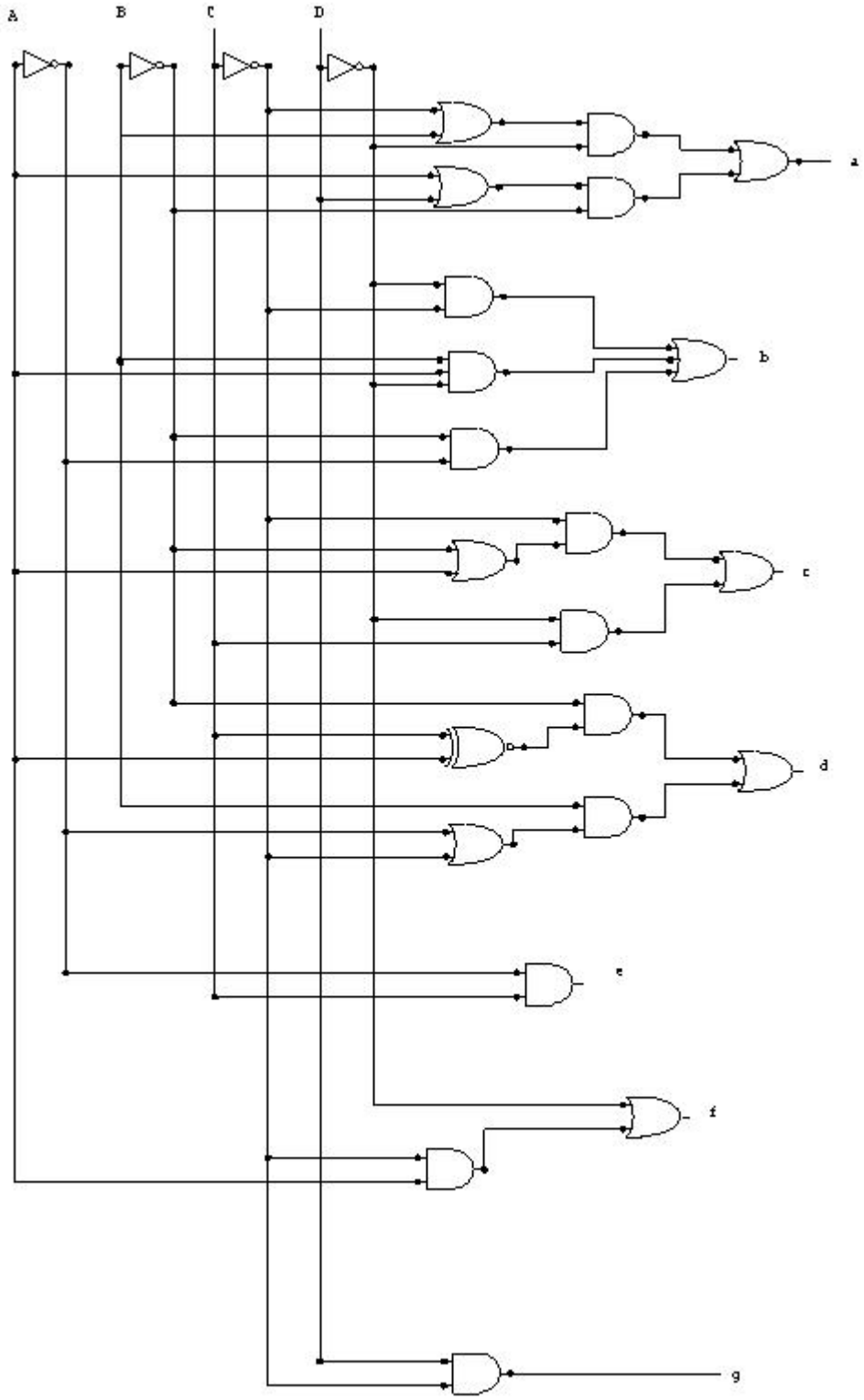
$$g = C\overline{B} + D\overline{C} + B\overline{A} + \overline{C}BA$$

$$g = C\overline{B} + D\overline{C} + B(\overline{A} + \overline{C}A)$$

$$g = C\overline{B} + D\overline{C} + B\overline{C}$$

$$g = D\,\overline{C}$$

A continuación se presenta el diagrama lógico Figura 8.1



MATERIAL:

CANTIDAD	MATERIAL Y EQUIPO	ESPECIFICACIONES
1	Microcontrolador SX48BD.	Con Módulo de Basic Stamp
1	Computadora	
1	Cable serial RS-232	
2	Display de 7 Segmentos	
1	CI DM7448	Ver anexo B
1	Módulo de interfaz	Buffer

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme el siguiente circuito como se muestra a continuación

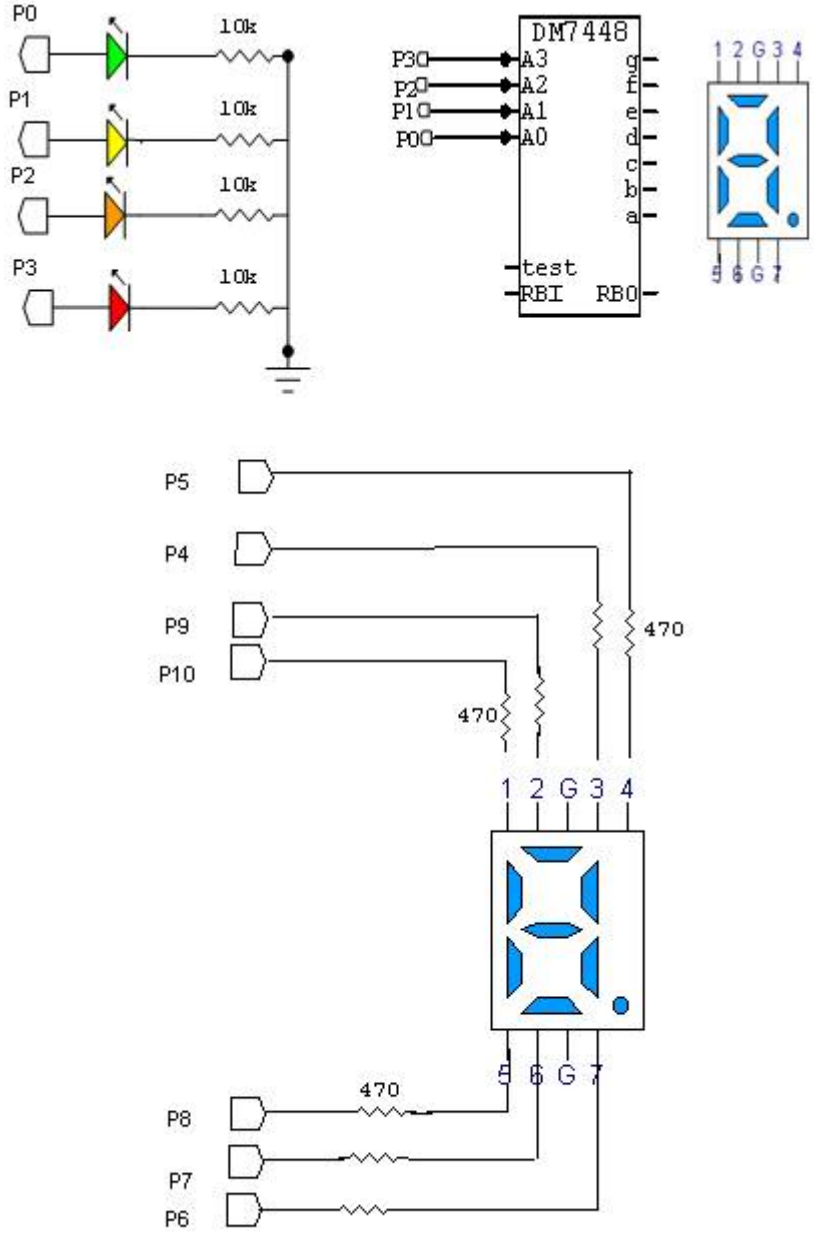


Fig. 8.2

Programa al Micro para mostrar cada segmento de acuerdo a la lógica establecida en la tabla de verdad del circuito integrado

```
'----- [Display de 7 segmentos] ----------------------------------
'Archivo..... Práctica No 8
'Propósito..... Uso del Micro lógica digital.
'Carrera Ing. Industrial

'{$STAMP BS2p}

Cero   CON  %0000
Uno    CON  %0001
Dos    CON  %0010
Tres   CON  %0011
Cuatro CON  %0100
Cinco  CON  %0101
Seis   CON  %0110
Siete  CON  %0111
Ocho   CON  %1000
Nueve  CON  %1001
Diez   CON  %1010
Once   CON  %1011
Doce   CON  %1100
Trece  CON  %1101
Catorce CON  %1110
Quince  CON  %1111
DIRA=%1111
Conteo VAR Byte
Tiempo CON 42

Ciclo:
'
'segmento a del display
FOR conteo= 1 TO 2
PAUSE tiempo
OUTA = cero
 HIGH 4
PAUSE tiempo
OUTA = uno
 LOW 4
PAUSE tiempo
OUTA = dos
HIGH 4
PAUSE tiempo
OUTA = tres
HIGH 4
HIGH 4
PAUSE tiempo
OUTA = cuatro
LOW 4
PAUSE tiempo
OUTA = cinco
HIGH 4
PAUSE tiempo
OUTA = seis
LOW 4
PAUSE tiempo
OUTA = siete
HIGH 4
PAUSE tiempo
OUTA = ocho
HIGH 4
```

```
PAUSE tiempo
OUTA = nueve
HIGH 4
PAUSE tiempo
OUTA = diez
LOW 4
PAUSE tiempo
OUTA = once
LOW 4
PAUSE tiempo
OUTA = doce
LOW 4
'_____________________
'segmento b del display
OUTA = cero
 HIGH 5
PAUSE tiempo
OUTA = uno
 HIGH 5
PAUSE tiempo
OUTA = dos
HIGH 5
PAUSE tiempo
OUTA = tres
HIGH 5
PAUSE tiempo
OUTA = cuatro
HIGH 5
PAUSE tiempo
OUTA = cinco
LOW 5
PAUSE tiempo
OUTA = seis
LOW 5
OUTA = siete
HIGH 5
PAUSE tiempo
OUTA = ocho
HIGH 5
PAUSE tiempo
PAUSE tiempo
OUTA = trece
HIGH 4
PAUSE tiempo
OUTA = catorce
LOW 4
PAUSE tiempo
OUTA= quince
LOW 4
PAUSE tiempo

OUTA = nueve
HIGH 5
PAUSE tiempo
OUTA = diez
LOW 5
PAUSE tiempo
OUTA = once
LOW 5
PAUSE tiempo
OUTA = doce
HIGH 5
PAUSE tiempo
OUTA = trece
HIGH 5
PAUSE tiempo
OUTA = catorce
LOW 5
PAUSE tiempo
OUTA= quince
LOW 5
PAUSE tiempo
```

```
'_____________________
'segmento c del display
OUTA = cero
 HIGH 6
PAUSE tiempo
OUTA = uno
 HIGH 6
PAUSE tiempo
OUTA = dos
LOW 6
PAUSE tiempo
OUTA = tres
HIGH 6
PAUSE tiempo
OUTA = cuatro
HIGH 6
PAUSE tiempo
OUTA = cinco
HIGH 6
PAUSE tiempo
OUTA = seis
HIGH 6
PAUSE tiempo
OUTA = siete
HIGH 6
PAUSE tiempo
OUTA = ocho
HIGH 6
PAUSE tiempo
OUTA = nueve
HIGH 6
PAUSE tiempo
OUTA = diez
LOW 6
PAUSE tiempo
OUTA = once
HIGH 6
PAUSE tiempo
OUTA = doce
LOW 6
PAUSE tiempo
OUTA = trece
LOW 6
PAUSE tiempo
OUTA = catorce
LOW 6
PAUSE tiempo
OUTA= quince
LOW 6
PAUSE tiempo

'_____________________________
'segmento d del display
OUTA = cero
 HIGH 7
PAUSE tiempo
OUTA = uno
 LOW 7
PAUSE tiempo
OUTA = dos
HIGH 7
PAUSE tiempo
OUTA = tres
HIGH 7
PAUSE tiempo
OUTA = cuatro
LOW 7
PAUSE tiempo
```

```
OUTA = cinco
HIGH 7
PAUSE tiempo
OUTA = seis
HIGH 7
PAUSE tiempo
OUTA = siete
LOW 7
PAUSE tiempo
OUTA = ocho
HIGH 7
PAUSE tiempo
OUTA = nueve
LOW 7
PAUSE tiempo
OUTA = diez
HIGH 7
'_____________________________
'segmento e del display

OUTA = cero
 HIGH 8
PAUSE tiempo
OUTA = uno
 LOW 8
PAUSE tiempo
OUTA = dos
HIGH 8
PAUSE tiempo
OUTA = tres
LOW 8
PAUSE tiempo
OUTA = cuatro
LOW 8
PAUSE tiempo
OUTA = cinco
LOW 8
PAUSE tiempo
OUTA = seis
HIGH 8
PAUSE tiempo
OUTA = siete
LOW 8
PAUSE tiempo
OUTA = ocho
PAUSE tiempo
OUTA = once
HIGH 7
PAUSE tiempo
OUTA = doce
LOW 7
PAUSE tiempo
OUTA = trece
HIGH 7
PAUSE tiempo
OUTA = catorce
HIGH 7
PAUSE tiempo
OUTA= quince
LOW 7
PAUSE tiempo

PAUSE tiempo
OUTA = nueve
LOW 8
PAUSE tiempo
OUTA = diez
HIGH 8
PAUSE tiempo
OUTA = once
LOW 8
PAUSE tiempo
OUTA = doce
LOW 8
PAUSE tiempo
OUTA = trece
LOW 8
PAUSE tiempo
OUTA = catorce
HIGH 8
PAUSE tiempo
OUTA= quince
LOW 8
PAUSE tiempo
```

```
'_____________________________
'segmento f del display
OUTA = cero
HIGH 9
PAUSE tiempo
OUTA = uno
LOW 9
PAUSE tiempo
OUTA = dos
LOW 9
PAUSE tiempo
OUTA = tres
LOW 9
PAUSE tiempo
OUTA = cuatro
HIGH 9
PAUSE tiempo
OUTA = cinco
HIGH 9
PAUSE tiempo
OUTA = seis
HIGH 9
PAUSE tiempo
OUTA = siete
LOW 9
PAUSE tiempo
OUTA = ocho
HIGH 9
PAUSE tiempo
OUTA = nueve
LOW 9
PAUSE tiempo
OUTA = diez
HIGH 9
PAUSE tiempo
OUTA = once
LOW 9
PAUSE tiempo
OUTA = doce
LOW 9
PAUSE tiempo
OUTA = trece
LOW 9
PAUSE tiempo
OUTA = catorce
HIGH 9
PAUSE tiempo
OUTA= quince
LOW 9
PAUSE tiempo
'_____________________________________
'segmento g del display

OUTA = cero
 LOW 10
PAUSE tiempo
OUTA = uno
 LOW 10
PAUSE tiempo
OUTA = dos
HIGH 10
PAUSE tiempo
OUTA = tres
HIGH 10
PAUSE tiempo
OUTA = cuatro
HIGH 10
PAUSE tiempo
OUTA = cinco
HIGH 10
PAUSE tiempo
OUTA = seis
HIGH 10
PAUSE tiempo
OUTA = siete
LOW 10
PAUSE tiempo
```

```
OUTA = ocho
HIGH 10
PAUSE tiempo
OUTA = nueve
HIGH 10
PAUSE tiempo
OUTA = diez
HIGH 10
PAUSE tiempo
OUTA = once
HIGH 10
PAUSE tiempo
OUTA = doce
HIGH 10
PAUSE tiempo
OUTA = trece
HIGH 10
PAUSE tiempo
OUTA = catorce
HIGH 10
PAUSE tiempo
OUTA= quince
LOW 10
PAUSE tiempo
NEXT
  END
GOTO ciclo
```

Descripción del programa.

Se direcciona al puerto A como salida; se declaran los números del 0 al 15 binario y una variable generalmente de tipo Byte o Word utilizada por el comando For Next para el conteo del ciclo.

For Next. Esta función crea un bucle programado entre un rango de valores iniciales y finales, (1-2), el cuerpo del bucle queda comprendido en el medio de FOR y NEXT, El bucle puede incrementar o decrementar la variable Counter acorde con el valor incremento establecido. Si no se establece un valor incremento asume que el incremento será de uno (1). El bucle finaliza cuando la variable Counter llegue al Valor Final establecido que en este caso es 2.

Se declara como ciclo a cada uno de los segmentos del display para que en el display se pueda ver cada uno de los segmrentos encendidos.

Se programa para dar seguimiento a los mapas de Karnough que de acuerdo a la tabla de verdad 8.1 para el circuito integrado DM7448 nos muestra

De todo esto podemos visualizar, la gran ventaja de disponer de un Micro, al inicio de esta práctica se mostro todo el procedimiento para el funcionamiento del circuito DM7448 para solo encender los 7 segmentos del display, sin que nos muestre algun número en especifico; posteriormente con algunas instrucciones de programación se presenta el funcionamiento de un contador simple; el cual sustituye la logica digital que seria muy laborioso de realizar.

Desafío:

Arme el siguiente circuito como se muestra a continuación fig. 8.3

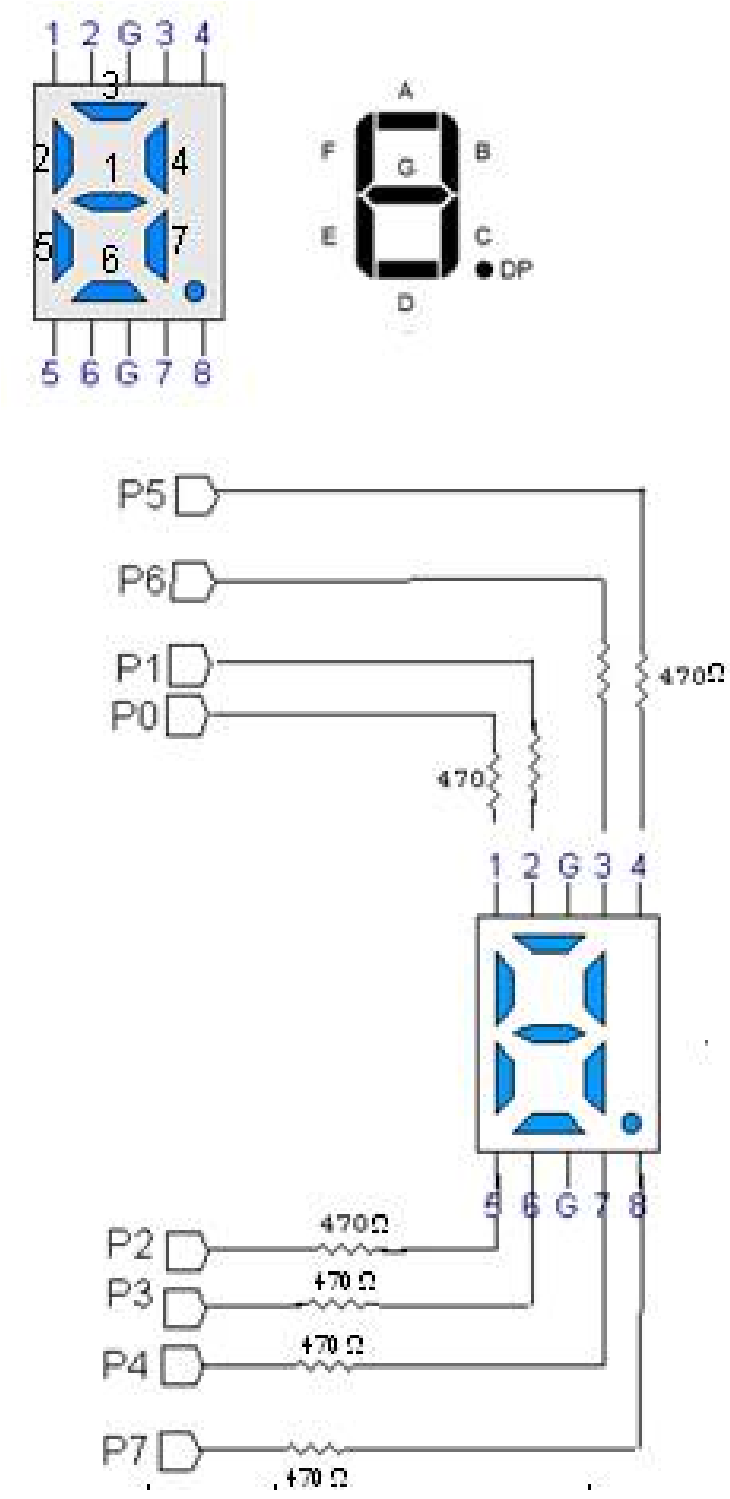
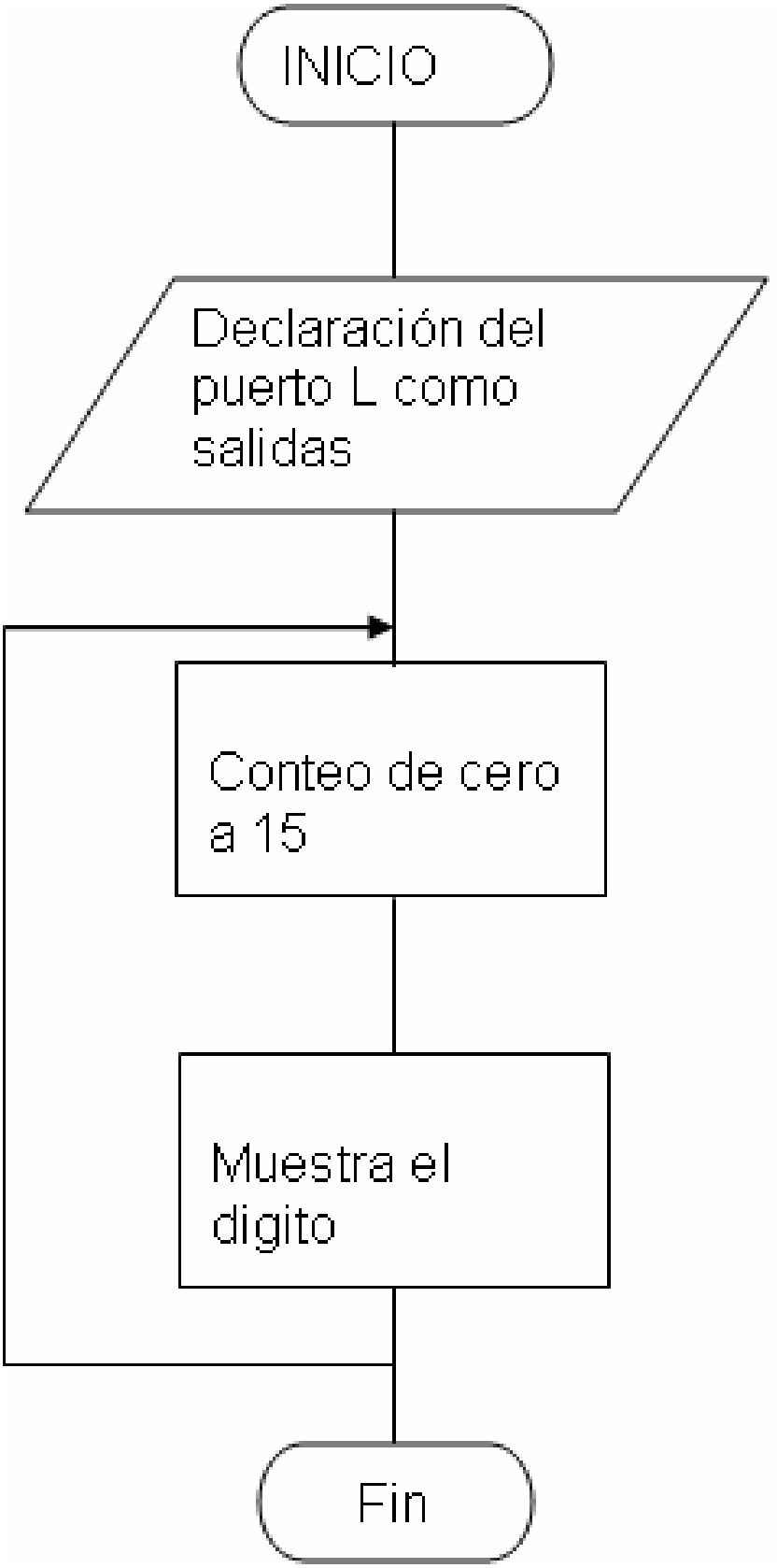


Figura 8.3


INICIO
Declaración del puerto L como salidas
Conteo de cero a 15
Muestra el digito
Fin

```
'-----[ Display de 7 segmentos]----------------------------------------
'Archivo..... Práctica No 8
'Propósito..... Uso del Micro lógica digital.
'Carrera Ing. Industrial

'{$STAMP BS2p}

Segmentos VAR OUTL              'LEDs del 7 segmentos
Conteo VAR Nib                  'contador
Segmentos               .abcdefg

Digitos     DATA %01111110      '0
            DATA %00110000      '1
            DATA %01101101      '2
            DATA %01111001      '3
            DATA %00110011      '4
            DATA %01011011      '5
            DATA %01011111      '6
            DATA %01110000      '7
            DATA %01111111      '8
            DATA %01111011      '9
            DATA %01110111      'A
            DATA %00011111      'B
            DATA %01001110      'C
            DATA %00111101      'D
            DATA %01001111      'E
            DATA %01000111      'F

Inicio:
DIRL = %01111111                'configura como salidas los 7 seg.
Principal:
FOR conteo = 0 TO 15            'conteo
READ (Digitos + conteo), segmentos   'muestra el dígito
PAUSE 1000 ' durante un segundo
NEXT
GOTO Principal ' repite todo
END
```

- De sus conclusiones del programa anterior.
- El siguiente programa es para controlar un display de 2 x 16 segmentos

```
'-----[ Display de 2x16 segmentos]-------------------------------------
'Archivo..... Práctica No 8
'Propósito..... Uso del Micro lógica digital.
'Carrera Ing. Industrial

'{$STAMP BS2p}

LCDpin      CON  0
I2Cpin      CON  8

NoCmd       CON  0
ClrLCD      CON  $01
CrsrHm      CON  $02
CrsrLf      CON  $10
CrsrRt      CON  $14
DispLf      CON  $18
DispRt      CON  $1C

DDRam       CON  $80
CGRam       CON  $40
Line1       CON  $80
Line2       CON  $C0

MaxEE       CON  4095

LCD_Setup:

addr        VAR  Word
addrHi      VAR  addr.HIGHBYTE
addrLo      VAR  addr.LOWBYTE
rVar        VAR  Word
tOut        VAR  Byte
tIn         VAR  Byte
temp        VAR  Word
width       VAR  Nib
pos         VAR  Byte
digits      VAR  Nib

Super2      DATA  %00000
            DATA  %01111
            DATA  %01001
            DATA  %01001
            DATA  %01111
            DATA  %01100
            DATA  %01010
            DATA  %01001
            DATA  %01001


 PAUSE  500
  LCDCMD  LCDpin, %00110000 : PAUSE 5
  LCDCMD  LCDpin, %00110000 : PAUSE 0
  LCDCMD  LCDpin, %00110000 : PAUSE 0
  LCDCMD  LCDpin, %00100000
  LCDCMD  LCDpin, %00101000
  LCDCMD  LCDpin, %00001100
  LCDCMD  LCDpin, %00000110
```

```
LCDCMD LCDpin, CGRam

 FOR addr = Super2 TO (Super2 + 7)
 READ addr, temp
 LCDOUT LCDpin, NoCmd,[temp]
NEXT

Splash:
   LCDOUT LCDpin, ClrLCD,["Micro SX48BD <-> I", 0, "C"]
   LCDOUT LCDpin, Line2, [" AUTOMATIZACION"]
   PAUSE 9000

Main:
  LCDOUT LCDpin, ClrLCD,["I", 0,"C:     Out="]
  LCDOUT LCDpin, (Line2+5),["In="]


  FOR addr = 0 TO MaxEE STEP 1000
   RANDOM rVar
   tOut = rVar.HIGHBYTE

AUXIO
I2COUT I2Cpin, $A0, addrHi\addrLo, [tOut]
PAUSE 100
I2CIN I2Cpin, $A0, addrHi\addrLo, [tIn]


MAINIO

LCDOUT LCDpin, (Line1 + 4), [DEC addr]
temp = tOut : width = 3 : pos = Line1 + 13

GOSUB RJ_Print
temp = tIn : width = 3 : pos = Line2 + 13
GOSUB RJ_Print

PAUSE 250

NEXT


PAUSE 2000
LCDOUT LCDpin, ClrLCD, ["**GRACIAS
POR**" ]
LCDOUT LCDpin,(Line2+1), ["SU ATENCION
!"]

END

RJ_Print:
  digits = width
  LOOKDOWN temp, <=[0, 10, 100, 1000,
65535], digits
  LCDOUT LCDpin, pos, [REP " "\(width -
digits), DEC temp]
  RETURN
```

CAPÍTULO 4.1 "Aplicación de la lógica de control en una estación gasolinera"

No. DE PRÁCTICA: 9 No. DE SESIONES:

No. DE INTEGRANTES POR EQUIPO*:*

OBJETIVO:

Dar una aplicación con los temas vistos al Micro SX48BD para el control de una bomba de un despachador de combustible en una estación gasolinera.

INTRODUCCIÓN:

En este caso simulamos el despachador de combustible a través de una leva la cual realiza el conteo de litros de combustible; también el programa ofrece un censor de contraincendios, el cual activa una bomba de agua e inmediatamete hace el paro de la leva que simula la bomba de combustible; se activa una alarma sonora para aviso de peligro; al activarse esta en automatico realiza una marcación teléfonica de auxilio a bomberos; en caso de que hubiese algún incendio. También cuanta con un display de 7 segmentos para visualizar el conteo, éste tiene una limitante que solo puede mostrarar de 1 a 9 litros y de la letra A a la F en Hexadecimal; pero el programa también muestra en pantalla de la PC, el conteo de litros de combustible aquí no existe alguna limitante muestra la cantidad que uno desee.

Para realizar esta práctica se tomaron algunos ejemplos vistos en clase como el dispositivo de contra incendios, el conteo de combustible con circuitos de lógica común, al igual se retomaron algunos ejemplos de este trabajo como el contador del display, comparador y lógica de programación.

PROCEDIMIENTO PARA EL DESARROLLO DE LA PRÁCTICA:

Arme los siguientes circuitos en el Protoboard Figura 9.1

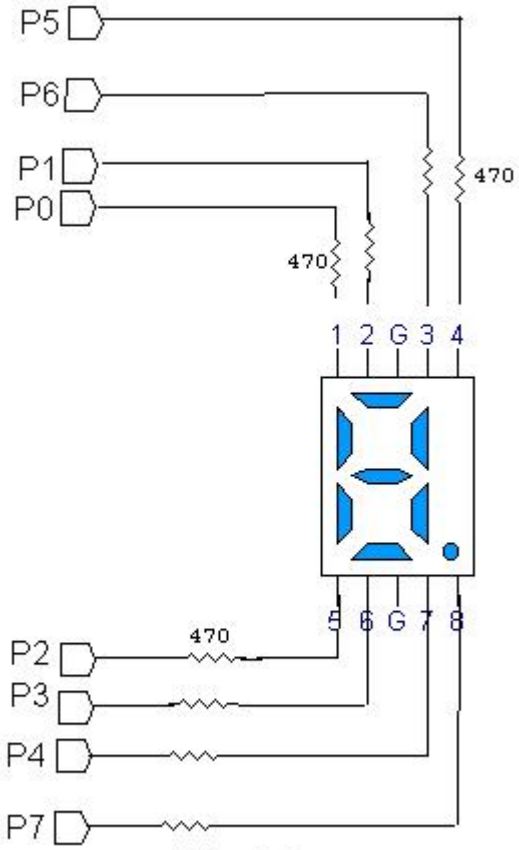
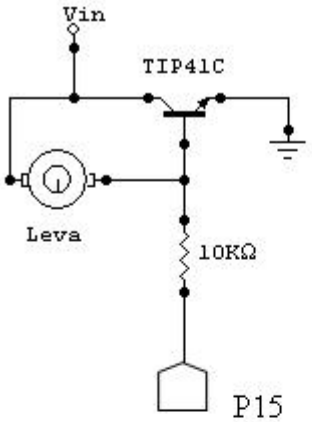
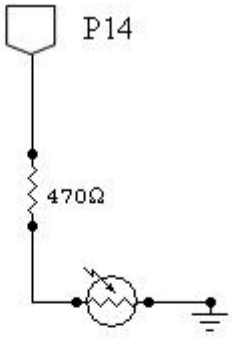


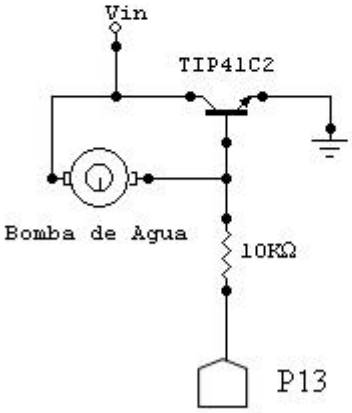
Fig. 9.1



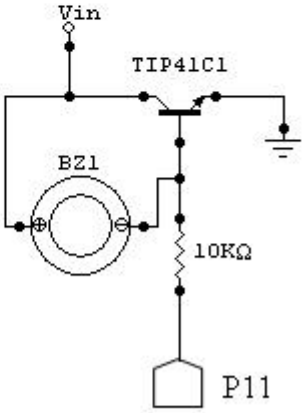
Bomba de Gasolina



Sensor de Humo



Bomba Contra incendió



Alarma Sonora

Realice el siguiente programa

```
'{$STAMP BS2p}
' ---- [Monitoreo y control en gasolinera]----------------------------
'Archivo..... Práctica No 9
'Propósito..... Uso del Micro en un ejemplo práctico.
'Autor..... Franco Muñoz Reyes
'E-mail..... francorey@hotmail.com
'Fecha... 12/Noviembre/2006
'Carrera Ing. Industrial

segmentos VAR OUTL
alarma VAR Byte
Cantidad _ litros VAR Word   'variable para introducir la cantidad     de
                              combustible deseado
litros VAR Word              ' variable para guardar la cantidad
                             'Recibe señal de alarma de alarma de contra
                             incendio;
INPUT 14                     'Recibe señal de alarma de contra incendio;
                              Dispositivo de entrada
OUTPUT 13                    'Activa bomba para agua; Dispositivo de salida
t VAR Word                   'variable para tiempo de bomba y display
t=300
C2 CON 1336
C3 CON 1477
F2 CON 770
F3 CON 852
On_Time CON 350
Off_Time CON 100
'_____________________________________________

Cantidad_litros = 15  '¿¿¿¿¿¿ introducir la cantidad de litros deseados !!!!!!
'_____________________________________________

Inicio:
DIRL = %0111111
IF litros > Cantidad_litros THEN paro_bomba        ' Condición para que no
                                                    exceda lo requerido
FOR litros = 1 TO Cantidad_litros          'Operardor para realizar el conteo
READ (Display + litros), segmentos          ' muestra el dígito en display
PAUSE t
DEBUG CR, CLS, "Litros:  ",DEC litros       'Muestra en pantalla el conteo de
                                            los litros
IF IN14=1 THEN paro_bomba                   'Condición: Si el censor de
                                            contra incendió  se activa se para
                                            bomba
la
LOW 13
HIGH 15
PAUSE t
LOW 15
```

```
PAUSE t
Display  DATA %01111110    '0
         DATA %00110000    '1
         DATA %01101101    '2
         DATA %01111001    '3
         DATA %00110011    '4
         DATA %01011011    '5
         DATA %01011111    '6
         DATA %01110000    '7
         DATA %01111111    '8
         DATA %01111011    '9
         DATA %01110111    'A
         DATA %00011111    'B
         DATA %01001110    'C
         DATA %00111101    'D
         DATA %01001111    'E
         DATA %01000111    'F
NEXT
END
paro_bomba:
LOW 15
HIGH 13
PAUSE 500 ' Espera 1/2 segundo
FOR alarma = 1 TO 30 'Alarma sensora que avisa cuando hay
peligro
FREQOUT 11,1000,440,480
PAUSE 500 ' Espera 1/2 segundo
NEXT
' Marcar el numero 068 estación de bomberos
FREQOUT 11, On_Time, F2,C2 ' Genera el tono
PAUSE Off_Time
FREQOUT 11, On_Time, F2,C3 ' Genera el tono 6
PAUSE Off_Time
FREQOUT 11, On_Time, F3,C2 ' Genera el tono 8
PAUSE Off_Time
PAUSE Off_Time
DEBUG CLS
GOTO Inicio
```

Descripción del programa

El programa realiza el conteo de conbustible por medio de la sentencia For, que a la vez esta condiciodo por medio de IF cantidad de litros, cuando el programa corre, le pedimos que monitoré el pin 14 si se acitva el censor de fuego es decir si recibe un estado lógico uno para el conteo y reinicia; también envia u uno al pin 13 para activar por medio de un interfaz transistor una bomba de agua para apagar el fuego; a la vez envia al pin 11 que genere un tono sonoro de alarma de aviso, que a su vez después de 30 timbres genera 3 tonos que simulan los números 0 6 8 de un teléfono para dar aviso a bomberos. El programa muestra por medio de la ventana Debug el conteo de combustible; también lo muestra por medio de un display de 7 segmentos pero solo por 15 litros incluyendo números hexadecimales de éste ya se dió la explicación de cómo funciona en una práctica anterior, no se explica en forma muy explicita ya que en prácticas anteriores ya se dieron explicación de cada uno de estos comandos.

Desafío:

- Realice el diagrama de flujo
- Modifique el programa para que realice 2 llamadas de auxilio a estación de bomberos.
- Haga comentarios finales.

Conclusiones

Las 9 prácticas presentadas ofrecen información para la operación del Microcontrolador SX48BD con módulo Basic Stamp, para que sirva como base en el manejo de otros microcontroladores, y a su vez, entender la parte principal de un PLC (control lógico programable), su control de mando, en la actualidad en todo equipo o sistema tecnológico es indispensable contar la presencia de un micro como parte central de control.

El microcontrolador SX48BD con módulo Basic Stamp, cuenta con un software en ambiente Windows 98, Me, Xp, que facilita la programación, ofreciendo una ventaja para que podamos aprender más, dando aplicaciones reales; una de las partes fundamentales para las aplicaciones del micro es el interfaz, de ello depende la fuerza de aplicación de equipos que se deseen manejar.

En la práctica 8 se da un claro ejemplo de cómo el microcontrolador reduce en tiempo para generar los mapas de Karnaugh, el conexionado entre circuitos, ofreciendo aplicaciones de mayor complejidad como display de cristal liquido, con tan solo generar un programa y que sin necesidad de anexar o cambiar circuitos, se le pueden dar una infinidad de aplicaciones totalmente ajenas a los display.

La explicación y aplicación de cada una de las prácticas, llevan un seguimiento, para poder enfrentar cada desafío que se da al término de la práctica; algunas prácticas se han tomado de diferentes manuales, de experiencia académica, y laboral que da como resultado la recopilación de las prácticas y que esta pudieran dar auge para proponer una infinidad de aplicaciones y así hacer más desafiante la materia de automatización.

Bibliografía.

Alejandro Vega

Manual del microcontrolador 8051 Diciembre 1999

Claus Kunhel klaus Zahnert

Basic Stamp 2p Commands, Features and Projects

Copyright©2000 by parallax

BASIC Stamp ® Programming manual Version 2.0b

Diego M. Pulgar G.

Manual de programación Basic Stamp Versión 1.1

I-Four de Grass Valley

Qué es un Microcontrolador, Guía del estudiante versión 1.1, derechos reservados por parallax 1999.

José Ma. Angulo Usategui, Susana Romero Yesa E Ignacio Angulo Martínez.

Microcontroladores PIC: Diseño Práctico De Aplicaciones, USA, editorial McGraw-Hill 2000

Ronald Tocci

Sistemas digitales principios y aplicaciones, México, editorial Prentice Hall, 1999

Sampieri Hernández Roberto

Metodología de la investigación, México, editorial Mc Graw Hill, 1991

Cybergrafía.

http://www.parallax.inc.com

http://www.infoplc.net/

http://www.alldatasheet.com/

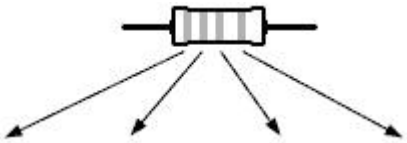
http://www.domotica.net/ir'www.comunidadelectronicos.com/

http://www.ero-pic.com

APÉNDICE A

CODIGO DE COLORES.

Para determinar el valor de la resistencia se comienza por determinar la banda de la tolerancia: oro, plata, rojo, marrón, o ningún color. Si las bandas son de color oro o plata, está claro que son las correspondientes a la tolerancia y debemos comenzar la lectura por el extremo contrario. Si son de color rojo o marrón, suelen estar separadas de las otras tres o cuatro bandas, y así comenzaremos la lectura por el extremo opuesto, 1ª cifra, 2ª cifra, número de ceros o factor multiplicador y tolerancia, aunque en algunos casos existe una tercera cifra significativa.En caso de existir sólo tres bandas con color, la tolerancia será de +/- 20%. La falta de esta banda dejará un hueco grande en uno de los extremos y se empezará la lectura por el contrario. Suele ser característico que la separación entre la banda de tolerancia y el factor multiplicativo sea mayor que la que existe entre las demás bandas.



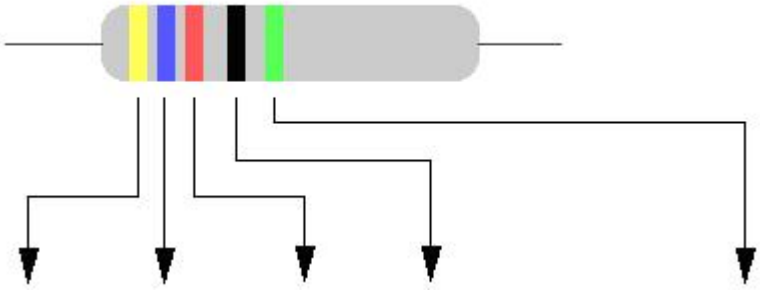
Colores	1ª Cifra	2ª Cifra	Multiplicador	Tolerancia
Negro		0	0	
Café	1	1	x 10	±1%
Rojo	2	2	x 10^2	±2%
Naranja	3	3	x 10^3	
Amarillo	4	4	x 10^4	
Verde	5	5	x 10^5	±0.5%
Azul	6	6	x 10^6	
Violeta	7	7	x 10^7	
Gris	8	8	x 10^8	
Blanco	9	9	x 10^9	
Oro			x 10^{-1}	±5%
Plata			x 10^{-2}	±10%
Sin color				±20%

Ejemplo:

Si los colores son: (Café - **Negro** - **Rojo** - Oro) su valor en ohmios es:

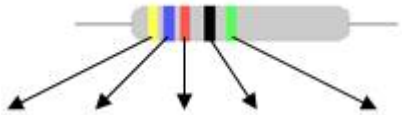
1 **0** **x 100** ,5 %

= **1000**Ω = 1KΩ Tolerancia de ±**5%**



Colores	1ª Cifra	2ª Cifra	3ª Cifra	Multiplicador	Tolerancia
Negro		0	0	0	
Café	1	1	1	x 10	±1%
Rojo	2	2	2	x 10^2	±2%
Naranja	3	3	3	x 10^3	
Amarillo	4	4	4	x 10^4	
Verde	5	5	5	x 10^5	±0.5%
Azul	6	6	6	x 10^6	
Violeta	7	7	7	x 10^7	
Gris	8	8	8	x 10^8	
Blanco	9	9	9	x 10^9	
Oro				x 10^{-1}	±5%
Plata				x 10^{-2}	±10%
Sin color					±20%

Ejemplo:



Si los colores son: (Amarillo - **Azul** - **Rojo** -**Negro** **Verde)** su valor en ohmios es:

4 6 3 0 5%

= 4630Ω Tolerancia de ±5%

Como valor nominal se puede encontrar tres, cuatro, o cinco caracteres formados por la combinación de dos, tres, o cuatro números y una letra, de acuerdo con las cifras significativas del valor nominal. La letra del código sustituye a la coma decimal, y representa el coeficiente multiplicador según la siguiente correspondencia:

LETRA CÓDIGO	R	K	M	G	T
COEFICIENTE MULTIPLICADOR	x1	$x10^3$	$x10^6$	$x10^9$	$x10^{12}$

Algunos ejemplos

Valor de la resistencia en ohmios	**Código de marcas**	**Valor de la resistencia en ohmios**	**Código de marcas**
0,1	R10	10K	10K
3,32	3R32	2,2M	2M2
59,04	59R04	1G	1G
590,4	590R4	2,2T	2T2
5,90K	5K9	10T	10T

ANEXO 1 HOJA DE DATOS

24LC168

MICROCHIP

24LC16B

16K 2.5V I²C™ Serial EEPROM

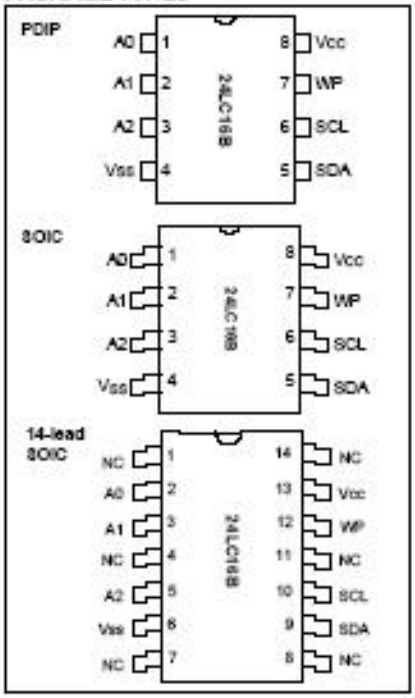
FEATURES

- Single supply with operation down to 2.5V
- Low power CMOS technology
  - 1 mA active current typical
  - 10 µA standby current typical at 5.5V
  - 5 µA standby current typical at 3.0V
- Organized as 8 blocks of 256 bytes (8 x 256 x 8)
- 2-wire serial interface bus, I²C™ compatible
- Schmitt trigger inputs for noise suppression
- Output slope control to eliminate ground bounce
- 100 kHz (2.5V) and 400 kHz (5V) compatibility
- Self-timed write cycle (including auto-erase)
- Page-write buffer for up to 16 bytes
- 2 ms typical write cycle time for page-write
- Hardware write protect for entire memory
- Can be operated as a serial ROM
- Factory programming (QTP) available
- ESD protection > 4,000V
- 10,000,000 erase/write cycles guaranteed
- Data retention > 200 years
- 8-pin DIP, 8-lead or 14-lead SOIC packages
- Available for extended temperature ranges
  - Commercial (C): 0°C to +70°C
  - Industrial (I): -40°C to +85°C

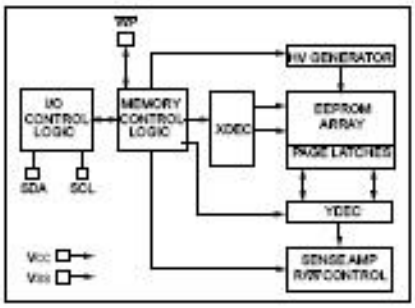
DESCRIPTION

The Microchip Technology Inc. 24LC16B is a 16K bit Electrically Erasable PROM. The device is organized as eight blocks of 256 x 8 bit memory with a 2-wire serial interface. Low voltage design permits operation down to 2.5 volts with standby and active currents of only 5 µA and 1 mA respectively. The 24LC16B also has a page-write capability for up to 16 bytes of data. The 24LC16B is available in the standard 8-pin DIP and both 8-lead and 14-lead surface mount SOIC packages.

PACKAGE TYPES



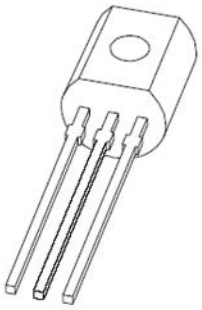
BLOCK DIAGRAM



I²C is a trademark of Philips Corporation.

2N3904

DATA SHEET



2N3904
NPN switching transistor

4N35

VISHAY

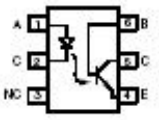
4N35/ 4N36/ 4N37/ 4N38

Vishay Semiconductors

Optocoupler, Phototransistor Output, With Base Connection

Features

- Isolation Test Voltage 5300 V_{RMS}
- Interfaces with common logic families
- Input-output coupling capacitance < 0.5 pF
- Industry Standard Dual-in line 6-pin package
- Lead-free component
- Component in accordance to RoHS 2002/95/EC and WEEE 2002/96/EC



Agency Approvals

- Underwriters Laboratory File #E52744
- DIN EN 60747-5-2 (VDE0884)
  DIN EN 60747-5-5 pending
  Available with Option 1

Applications

AC mains detection

Reed relay driving

Switch mode power supply feedback

Telephone ring detection

Logic ground isolation

Logic coupling with high frequency noise rejection

Description

This data sheet presents five families of Vishay Industry Standard Single Channel Phototransistor Couplers.These families include the 4N35/ 4N36/ 4N37/ 4N38 couplers.

Each optocoupler consists of gallium arsenide infrared LED and a silicon NPN phototransistor.

These couplers are Underwriters Laboratories (UL) listed to comply with a 5300 V_{RMS} isolation test voltage.

This isolation performance is accomplished through Vishay double molding isolation manufacturing process. Comliance to DIN EN 60747-5-2(VDE0884)/ DIN EN 60747-5-5 pending partial discharge isolation specification is available for these families by ordering option 1.

These isolation processes and the Vishay ISO9001 quality program results in the highest isolation performance available for a commecial plastic phototransistor optocoupler.

The devices are available in lead formed configuration suitable for surface mounting and are available either on tape and reel, or in standard tube shipping containers.

Note:
Designing with data sheet is cover in Application Note 45

Order Information

Part	Remarks
4N35	CTR > 100 %, DIP-6
4N36	CTR > 100 %, DIP-6
4N37	CTR > 100 %, DIP-6
4N38	CTR > 20 %, DIP-6
4N35-X006	CTR > 100 %, DIP-6 400 mil (option 6)
4N35-X007	CTR > 100 %, SMD-6 (option 7)
4N35-X009	CTR > 100 %, SMD-6 (option 9)
4N36-X007	CTR > 100 %, SMD-6 (option 7)
4N36-X009	CTR > 100 %, SMD-6 (option 9)
4N37-X006	CTR > 100 %, DIP-6 400 mil (option 6)
4N37-X009	CTR > 100 %, SMD-6 (option 9)

For additional information on the available options refer to Option Information.

ADC0831

National Semiconductor

August 1999

ADC0831/ADC0832/ADC0834/ADC0838
8-Bit Serial I/O A/D Converters with Multiplexer Options

General Description

The ADC0831 series are 8-bit successive approximation A/D converters with a serial I/O and configurable input multiplexers with up to 8 channels. The serial I/O is configured to comply with the NSC MICROWIRE™ serial data exchange standard for easy interface to the COPS™ family of processors, and can interface with standard shift registers or µPs.

The 2-, 4- or 8-channel multiplexers are software configured for single-ended or differential inputs as well as channel assignment.

The differential analog voltage input allows increasing the common-mode rejection and offsetting the analog zero input voltage value. In addition, the voltage reference input can be adjusted to allow encoding any smaller analog voltage span to the full 8 bits of resolution.

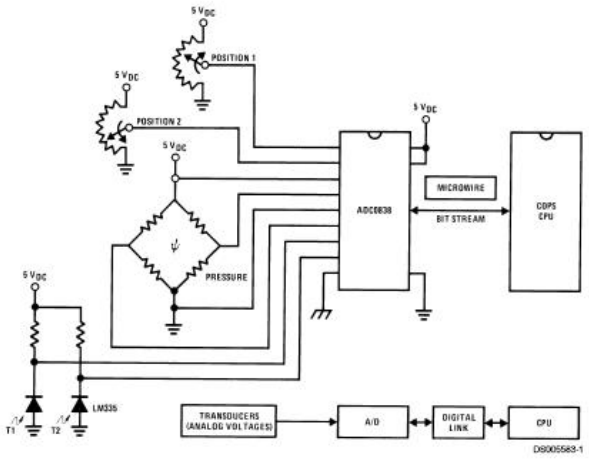
Features

- NSC MICROWIRE compatible — direct interface to COPS family processors
- Easy interface to all microprocessors, or operates "stand-alone"
- Operates ratiometrically or with 5 V_{DC} voltage reference
- No zero or full-scale adjust required
- 2-, 4- or 8-channel multiplexer options with address logic
- Shunt regulator allows operation with high voltage supplies
- 0V to 5V input range with single 5V power supply
- Remote operation with serial digital data link
- TTL/MOS input/output compatible
- 0.3" standard width, 8-, 14- or 20-pin DIP package
- 20 Pin Molded Chip Carrier Package (ADC0838 only)
- Surface-Mount Package

Key Specifications

Resolution	8 Bits
Total Unadjusted Error	±½ LSB and ±1 LSB
Single Supply	5 V_{DC}
Low Power	15 mW
Conversion Time	32 µs

Typical Application



ADC0831/ADC0832/ADC0834/ADC0838 8-Bit Serial I/O A/D Converters with Multiplexer Options

BC547

FAIRCHILD SEMICONDUCTOR™

Discrete POWER & Signal Technologies

BC547
BC547A
BC547B
BC547C

NPN General Purpose Amplifier

This device is designed for use as general purpose amplifiers and switches requiring collector currents to 300 mA. Sourced from Process 10. See PN100A for characteristics.

Absolute Maximum Ratings*

TA = 25°C unless otherwise noted

Symbol	Parameter	Value	Units
V_{CEO}	Collector-Emitter Voltage	45	V
V_{CES}	Collector-Base Voltage	50	V
V_{EBO}	Emitter-Base Voltage	6.0	V
I_C	Collector Current - Continuous	500	mA
T_J, T_{stg}	Operating and Storage Junction Temperature Range	-55 to +150	°C

*These ratings are limiting values above which the serviceability of any semiconductor device may be impaired.

NOTES:
1) These ratings are based on a maximum junction temperature of 150 degrees C.
2) These are steady state limits. The factory should be consulted on applications involving pulsed or low duty cycle operations.

Thermal Characteristics

TA = 25°C unless otherwise noted

Symbol	Characteristic	Max	Units
		BC547 / A / B / C	
P_D	Total Device Dissipation Derate above 25°C	625 5.0	mW mW/°C
$R_{\theta JC}$	Thermal Resistance, Junction to Case	83.3	°C/W
$R_{\theta JA}$	Thermal Resistance, Junction to Ambient	200	°C/W

547ABC, Rev B

LM34DZ

National Semiconductor

July 1999

LM34
Precision Fahrenheit Temperature Sensors

General Description

The LM34 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Fahrenheit temperature. The LM34 thus has an advantage over linear temperature sensors calibrated in degrees Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Fahrenheit scaling. The LM34 does not require any external calibration or trimming to provide typical accuracies of ±½°F at room temperature and ±1½°F over a full −50 to +300°F temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM34's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies or with plus and minus supplies. As it draws only 75 µA from its supply, it has very low self-heating, less than 0.2°F in still air. The LM34 is rated to operate over a −50° to +300°F temperature range, while the LM34C is rated for a −40° to +230°F range (0°F with improved accuracy). The LM34 series is available packaged in hermetic TO-46 transistor packages, while the LM34C, LM34CA and LM34D are also available in the plastic TO-92 transistor package. The LM34D is also available in an 8-lead surface mount small outline package. The LM34 is a complement to the LM35 (Centigrade) temperature sensor.

Features

- Calibrated directly in degrees Fahrenheit
- Linear +10.0 mV/°F scale factor
- 1.0°F accuracy guaranteed (at +77°F)
- Rated for full −50° to +300°F range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 5 to 30 volts
- Less than 90 µA current drain
- Low self-heating, 0.18°F in still air
- Nonlinearity only ±0.5°F typical
- Low-impedance output, 0.4Ω for 1 mA load

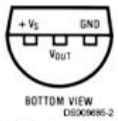
Connection Diagrams

TO-46
Metal Can Package
(Note 1)

DS006685-1

Order Numbers LM34H, LM34AH, LM34CH, LM34CAH or LM34DH
See NS Package Number H03H

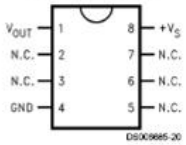
TO-92
Plastic Package



BOTTOM VIEW
DS006685-2

Order Number LM34CZ, LM34CAZ or LM34DZ
See NS Package Number Z03A

SO-8
Small Outline
Molded Package



DS006685-20

N.C. = No Connection

Top View
Order Number LM34DM
See NS Package Number M08A

Note 1: Case is connected to negative pin (GND).

TRI-STATE® is a registered trademark of National Semiconductor Corporation.

LM358

Order this document by LM358/D

MOTOROLA

LM358, LM258, LM2904, LM2904V

Dual Low Power Operational Amplifiers

DUAL DIFFERENTIAL INPUT OPERATIONAL AMPLIFIERS

SEMICONDUCTOR TECHNICAL DATA

Utilizing the circuit designs perfected for recently introduced Quad Operational Amplifiers, these dual operational amplifiers feature 1) low power drain, 2) a common mode input voltage range extending to ground/V_{EE}, 3) single supply or split supply operation and 4) pinouts compatible with the popular MC1558 dual operational amplifier. The LM158 series is equivalent to one–half of an LM124.

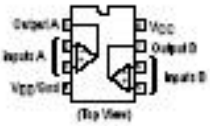
These amplifiers have several distinct advantages over standard operational amplifier types in single supply applications. They can operate at supply voltages as low as 3.0 V or as high as 32 V, with quiescent currents about one–fifth of those associated with the MC1741 (on a per amplifier basis). The common mode input range includes the negative supply, thereby eliminating the necessity for external biasing components in many applications. The output voltage range also includes the negative power supply voltage.

- Short Circuit Protected Outputs
- True Differential Input Stage
- Single Supply Operation: 3.0 V to 32 V
- Low Input Bias Currents
- Internally Compensated
- Common Mode Range Extends to Negative Supply
- Single and Split Supply Operation
- Similar Performance to the Popular MC1558
- ESD Clamps on the Inputs Increase Ruggedness of the Device without Affecting Operation

N SUFFIX
PLASTIC PACKAGE
CASE 626

D SUFFIX
PLASTIC PACKAGE
CASE 751
(SO–8)

PIN CONNECTIONS



(Top View)

MAXIMUM RATINGS (T_A = +25°C, unless otherwise noted.)

Rating	Symbol	LM358 LM358	LM2904 LM2904V	Unit
Power Supply Voltages Single Supply Split Supplies	V_{CC} V_{CC}, V_{EE}	32 ±16	26 ±13	Vdc
Input Differential Voltage Range (Note 1)	V_{IDR}	±32	±26	Vdc
Input Common Mode Voltage Range (Note 2)	V_{ICR}	–0.3 to 32	–0.3 to 26	Vdc
Output Short Circuit Duration	t_{SC}	Continuous		
Junction Temperature	T_J	150		°C
Storage Temperature Range	T_{stg}	–65 to +125		°C
Operating Ambient Temperature Range LM258 LM358 LM2904 LM2904V	T_A	 –25 to +85 0 to +70 – –	 – – –40 to +105 –40 to +125	°C

NOTES: 1. Split Power Supplies.
2. For Supply Voltages less than 32 V for the LM258/358 and 26 V for the LM2904, the absolute maximum input voltage is equal to the supply voltage.

ORDERING INFORMATION

Device	Operating Temperature Range	Package
LM2904D	T_A = –40° to +105°C	SO–8
LM2904N		Plastic DIP
LM2904VD	T_A = –40° to +125°C	SO–8
LM2904VN		Plastic DIP
LM258D	T_A = –25° to +85°C	SO–8
LM258N		Plastic DIP
LM358D	T_A = 0° to +70°C	SO–8
LM358N		Plastic DIP

NE555

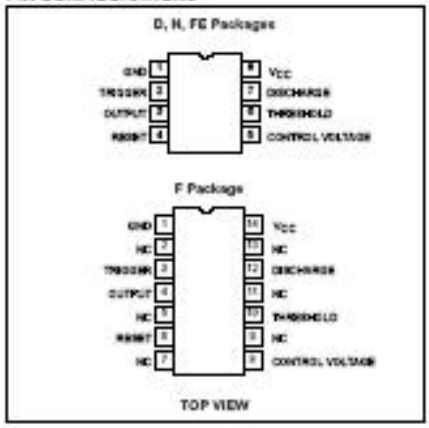
Philips Semiconductors Linear Products — Product specification

Timer — **NE/SA/SE555/SE555C**

DESCRIPTION

The 555 monolithic timing circuit is a highly stable controller capable of producing accurate time delays, or oscillation. In the time delay mode of operation, the time is precisely controlled by one external resistor and capacitor. For a stable operation as an oscillator, the free running frequency and the duty cycle are both accurately controlled with two external resistors and one capacitor. The circuit may be triggered and reset on falling waveforms, and the output structure can source or sink up to 200mA.

PIN CONFIGURATIONS



FEATURES

- Turn-off time less than 2µs
- Max. operating frequency greater than 500kHz
- Timing from microseconds to hours
- Operates in both astable and monostable modes
- High output current
- Adjustable duty cycle
- TTL compatible
- Temperature stability of 0.005% per °C

APPLICATIONS

- Precision timing
- Pulse generation
- Sequential timing
- Time delay generation
- Pulse width modulation

ORDERING INFORMATION

DESCRIPTION	TEMPERATURE RANGE	ORDER CODE	DWG #
8-Pin Plastic Small Outline (SO) Package	0 to +70°C	NE555D	0174C
8-Pin Plastic Dual In-Line Package (DIP)	0 to +70°C	NE555N	0404B
8-Pin Plastic Dual In-Line Package (DIP)	-40°C to +85°C	SA555N	0404B
8-Pin Plastic Small Outline (SO) Package	-40°C to +85°C	SA555D	0174C
8-Pin Hermetic Ceramic Dual In-Line Package (CERDIP)	-55°C to +125°C	SE555CFE	
8-Pin Plastic Dual In-Line Package (DIP)	-55°C to +125°C	SE555CN	0404B
14-Pin Plastic Dual In-Line Package (DIP)	-55°C to +125°C	SE555N	0405B
8-Pin Hermetic Cerdip	-55°C to +125°C	SE555FE	
14-Pin Ceramic Dual In-Line Package (CERDIP)	0 to +70°C	NE555F	0581B
14-Pin Ceramic Dual In-Line Package (CERDIP)	-55°C to +125°C	SE555F	0581B
14-Pin Ceramic Dual In-Line Package (CERDIP)	-55°C to +125°C	SE555CF	0581B

SX48BD

MICROCHIP

PIC16C5X

EPROM/ROM-Based 8-Bit CMOS Microcontroller Series

Devices Included in this Data Sheet:

- PIC16C52
- PIC16C54s
- PIC16CR54s
- PIC16C55s
- PIC16C56s
- PIC16CR56s
- PIC16C57s
- PIC16CR57s
- PIC16C58s
- PIC16CR58s

Note: The letter "s" used following the part numbers throughout this document indicate plural, meaning there is more than one part variety for the indicated device.

High-Performance RISC CPU:

- Only 33 single word instructions to learn
- All instructions are single cycle (200 ns) except for program branches which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle

Device	Pins	I/O	EPROM/ ROM	RAM
PIC16C52	18	12	384	25
PIC16C54	18	12	512	25
PIC16C54A	18	12	512	25
PIC16C54B	18	12	512	25
PIC16C54C	18	12	512	25
PIC16CR54A	18	12	512	25
PIC16CR54B	18	12	512	25
PIC16CR54C	18	12	512	25
PIC16C55	28	20	512	24
PIC16C55A	28	20	512	24
PIC16C56	18	12	1K	25
PIC16C56A	18	12	1K	25
PIC16CR56A	18	12	1K	25
PIC16C57	28	20	2K	72
PIC16C57C	28	20	2K	72
PIC16CR57B	28	20	2K	72
PIC16CR57C	28	20	2K	72
PIC16C58A	18	12	2K	73
PIC16C58B	18	12	2K	73
PIC16CR58A	18	12	2K	73
PIC16CR58B	18	12	2K	73

- 12-bit wide instructions
- 8-bit wide data path
- Seven or eight special function hardware registers
- Two-level deep hardware stack
- Direct, indirect and relative addressing modes for data and instructions

Peripheral Features:

- 8-bit real time clock/counter (TMR0) with 8-bit programmable prescaler
- Power-On Reset (POR)
- Device Reset Timer (DRT)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code-protection
- Power saving SLEEP mode
- Selectable oscillator options:
  - RC: Low-cost RC oscillator
  - XT: Standard crystal/resonator
  - HS: High-speed crystal/resonator
  - LP: Power saving, low-frequency crystal

CMOS Technology:

- Low-power, high-speed CMOS EPROM/ROM technology
- Fully static design
- Wide-operating voltage and temperature range:
  - EPROM Commercial/Industrial 2.0V to 6.25V
  - ROM Commercial/Industrial 2.0V to 6.25V
  - EPROM Extended 2.5V to 6.0V
  - ROM Extended 2.5V to 6.0V
- Low-power consumption
  - < 2 mA typical @ 5V, 4 MHz
  - 15 µA typical @ 3V, 32 kHz
  - < 0.6 µA typical standby current (with WDT disabled) @ 3V, 0°C to 70°C

Note: In this document, figure and table titles refer to all varieties of the part number indicated, (i.e., The title "Figure 14-1: Load Conditions - PIC16C64A", also refers to PIC16LC64A and PIC16LV64A parts).

X9313TP

APPLICATION NOTES
AVAILABLE
AN42 • AN44–48 • AN50 • AN52 • AN53 • AN71 • AN73

Terminal Voltage ±5V, 32 Taps

Xicor®

X9313

E²POT™ Nonvolatile Digital Potentiometer

FEATURES

- Low Power CMOS
  - —V_{CC} = 3V to 5.5V
  - —Active Current, 3mA Max
  - —Standby Current, 500µA Max
- 31 Resistive Elements
  - —Temperature Compensated
  - —±20% End to End Resistance Range
  - —–5V to +5V Range
- 32 Wiper Tap Points
  - —Wiper Positioned via Three-Wire Interface
  - —Similar to TTL Up/Down Counter
  - —Wiper Position Stored in Nonvolatile Memory and Recalled on Power-Up
- 100 Year Wiper Position Data Retention
- X9313Z = 1KΩ
- X9313W = 10KΩ
- Packages
  - —8-Lead MSOP
  - —8-Lead PDIP
  - —8-Lead SOIC

DESCRIPTION

The Xicor X9313 is a solid state nonvolatile potentiometer and is ideal for digitally controlled resistance trimming.

The X9313 is a resistor array composed of 31 resistive elements. Between each element and at either end are tap points accessible to the wiper element. The position of the wiper element is controlled by the $\overline{CS}$, U/$\overline{D}$, and $\overline{INC}$ inputs. The position of the wiper can be stored in nonvolatile memory and then be recalled upon a subsequent power-up operation.

The resolution of the X9313 is equal to the maximum resistance value divided by 31. As an example, for the X9313W (10KΩ) each tap point represents 323Ω.

All Xicor nonvolatile memories are designed and tested for applications requiring extended endurance and data retention.

FUNCTIONAL DIAGRAM