UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

ING. MECÁNICA ELÉCTRICA ELECTRÓNICA

“PRÁCTICAS PARA EL LABORATORIO DE MICROCONTROLADORES Y MICROPROCESADORES”

TESIS

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO MECÁNICO ELECTRICISTA
ÁREA: ELÉCTRICO-ELECTRÓNICA

P R E S E N T A :
OLIVARES LEYVA JOSÉ DE JESÚS

ASESOR: **MCIC. DÍAZ RANGEL ISMAEL**

MÉXICO 2010

Microcontroladores y Microprocesadores, prácticas.

Laboratorios de Electrónica.

Ingeniería Mecánica Eléctrica.

Profesor de laboratorio:

Alumno:

Grupo:

Horario:

VNIVERSIDAD NACIONAL AVFNOMA DE MEXICO

Facultad de Estudios Superiores Aragón

DIRECCIÓN

JOSE DE JESUS OLIVARES LEYVA
Presente

Con fundamento en el punto 6 y siguientes, del Reglamento para Exámenes Profesionales en esta Facultad, y toda vez que la documentación presentada por usted reúne los requisitos que establece el precitado Reglamento; me permito comunicarle que ha sido aprobado su tema de tesis y asesor.

TÍTULO:

"PRÁCTICAS PARA EL LABORATORIO
DE MICROCONTROLADORES Y MICROPROCESADORES"

ASESOR: MCIC. ISMAEL DÍAZ RANGEL

Aprovecho la ocasión para reiterarle mi distinguida consideración.

Atentamente
"POR MI RAZA HABLARÁ EL ESPÍRITU"
Nezahualcóyotl, Estado de México a 18 de agosto de 2010.

EL DIRECTOR

M. en I. GILBERTO GARCÍA SANTAMARÍA GONZÁLEZ

FACULTAD DE ESTUDIOS SUPERIORES ARAGON U N A M. DIRECCION

C p Secretaria Académica
C p Jefatura de Carrera de Ingeniería Mecánica Eléctrica
C p Asesor de Tesis

GGSG/JGPO/vr

100 UNAM

Agradecimientos

A mi madre, por haber creído siempre en mí, por sus grandes esfuerzos y su apoyo en toda mi carrera, tanto económico como moral, gracias te amo.

A MCIC. Ismael que fue mi guía con sus consejos y ayuda pude concluir mi tesis.

A mis hermanos que con su ejemplo, consejos y apoyo crearon en mí una conciencia de lo importante que es el estudio y tener una carrera profesional.

A mi hermana Perla que sin su apoyo hubiera sido más difícil concluir mis estudios.

A mi hermana Claudia que con su apoyo y consejos me incline al área de la ingeniería.

A Guadalupe que en todo momento soporto la falta de tiempo y estuvo conmigo en todo momento.

Índice

Introducción.

Estas prácticas tienen como objetivo acercar al alumno al manejo de los microcontroladores, no importando el modelo o la marca del microcontrolador ya que las prácticas no están diseñadas para un microcontrolador en particular sino que están orientadas a la programación en lenguaje ensamblador y los periféricos más comunes; considerando que la lógica que se utiliza es esencialmente la misma.

También ayudará al alumno con el entorno del software, que va a depender del microcontrolador que se vaya a utilizar, como también al hardware que se implemente, que en el mismo caso dependerá del microcontrolador elegido.

Este manual ayudará tanto al alumno como al profesor, ya que es un manual flexible. Dependiendo de las necesidades del profesor y alumnos, cada una de las prácticas propone un trabajo de casa para que el trabajo de laboratorio sea un 80% práctico, deja las fichas de consulta sujetas a la consideración del profesor. El trabajo de casa tiene como objetivo que la práctica sea más de retroalimentación y no de conceptos.

Al finalizar las prácticas de esta guía, el alumno habrá adquirido la capacidad de identificar si es necesaria la implementación de un control por medio de un microcontrolador. Creará programas (software) en lenguaje ensamblador, para así, controlar cualquier dispositivo bajo cualquier algoritmo de control que sea necesario implementar.

Estas prácticas están orientadas al área eléctrica electrónica, más específicamente al área de sistemas digitales y comunicaciones, no obstante los sistemas digitales hoy

en día tienen una gran importancia en nuestra vida diaria, desde nuestro televisor, horno de microondas, teléfono de casa y celular, hasta computadoras, sistemas de radiolocalización control de la producción, robots, etc. De hecho en cualquier ámbito existen sistemas diseñados con base a los microcontroladores. Pero no por esto se reduce el campo de acción, de hecho son cada vez más utilizados en diferentes áreas.

Se dice que una señal es digital cuando las magnitudes de la misma se representan mediante valores discretos en lugar de variables continuas. Los sistemas digitales usan lógica de dos estados representados por dos niveles de tensión; un alto, (H) y otro bajo, (L). Por abstracción, dichos estados se sustituyen por ceros y unos, lo que facilita la aplicación de la lógica y aritmética binaria. Si el nivel alto se representa por "1", se habla de lógica positiva y el bajo por "0", se habla de lógica negativa.

También buscamos que el alumno pueda tomar decisiones prácticas a problemas que se le presenten tanto fuera como dentro del salón de clases, ya sea que tenga que implementar un control con un microcontrolador o sólo con unos cuantos botones.

Lo fascinante será que tendrá una opción más para resolver problemas, que como Ingeniero es lo que buscamos diariamente.

Objetivo General.

El objetivo general de estas prácticas es el manejo e implementación de los microcontroladores y microprocesadores.

Estas prácticas están diseñadas para poder realizarse con cualquier microcontrolador, ya que como la materia del mismo nombre esta impartida por diferentes profesores y que cada profesor enseña diferentes microcontroladores, es por eso que estas prácticas pueden ser adaptadas para cualquier microcontrolador.

Mediante la realización de estas prácticas el alumno podrá adquirir el conocimiento y la experiencia en el uso de los diferentes puertos, comunicación, entrada salida, convertidores A/D, etc. Para poder implementar el control de un sistema, por medio de señales de control, por tiempo, por la conversión de una variable física, etc.

Al término de estas prácticas el alumno tendrá los conocimientos mínimos necesarios para diseñar un sistema digital, el cual puede tener una o más señales de control como las antes mencionadas.

Definición de Microcontrolador.

Para definir microcontrolador, es necesario saber el significado de control y micro:

Control: Regulación, manual o automática, sobre un sistema.

Micro: (Del gr. μικρο-) elem. compos. 1. Significa 'muy pequeño'. 2. Significa 'una millonésima (10^{-6}) parte'. Se aplica a nombres de unidades de medida para designar el submúltiplo correspondiente (Símb. μ).

Ya definidos estos 2 conceptos, podemos explicar que, un **microcontrolador** es un circuito de muy alta escala de integración, que incorpora la mayor parte de los elementos que conforman un controlador.

Un microcontrolador, está integrado normalmente de los siguientes componentes:

Unidad Central de Proceso (CPU, Central Processing Unit), es el elemento más importante del microcontrolador. Se encarga de direccionar la memoria de instrucciones, recibe el código de operación de la instrucción en curso. Decodifica y ejecuta de la operación, que implica la instrucción, así como la búsqueda de los operandos y el almacenamiento del resultado.

Memoria RAM, (Random Access Memory) o volátil, sirve para guardar las variables y los datos. (En los microcontroladores, la memoria de instrucciones y datos está dentro del mismo chip)

Memoria para el programa o no volátil, tipo **ROM (Read Only Memory)**, esta memoria guarda el programa o instrucciones que gobiernan la aplicación. Pueden ser de tipo:

1. ROM.
2. PROM. (Programmable Read Only Memory)
3. EPROM. (Erasable Programmable Read Only Memory)
4. EEPROM. (Electrically Erasable Programmable Read Only Memory)
5. FLASH. (Funciona como una ROM y una RAM pero con menor consumo y es más pequeña). Actualmente es la de mayor utilización.

Unidades de E/S: Es la interfaz con el exterior (sirven para comunicarse con el exterior).

Circuito de reloj: Los microcontroladores cuentan con un circuito oscilador que genera una onda cuadrada de alta frecuencia.

Circuito de Reset: Sirve para reiniciar el microcontrolador.

1. El Contador de Programa (PC, Program Counter) se carga con la dirección 0, apuntando a la primera dirección de la memoria de programa, en donde deberá estar situada la primera instrucción del programa de aplicación.
2. La mayoría de los registros de estado y control del procesador, toman un estado conocido y determinado.

Buses, sistema digital que transfiere datos entre componentes está formado por cables o líneas de circuito impreso, existen dos tipos Bus serie y Bus paralelo, en el Bus paralelo los datos son enviados al mismo tiempo y en el serie los datos son enviados bit a bit, en un microcontrolador nos encontramos con este tipo de buses:

1. Direcciones: Selecciona dirección origen o destino.
2. Datos: Realiza la transferencia del dato.
3. Control: reloj, Reset, read/write, interrupción, etc.

Unidades especiales:

1. Temporizadores o Timers.
2. Perro guardián o Watchdog.
3. Protección ante fallo de alimentación o Brownout.
4. Estado de reposo o de bajo consumo.
5. Convertidor A/D.
6. Convertidor D/A.
7. Comparador analógico.
8. Modulador de ancho de impulsos o PWM.
9. Puertos paralelos de E/S digitales.
10. Puertos de comunicación serie, (UART, USART, USB, Bus I^2C, CAN).

Existen tres orientaciones en cuanto a la arquitectura y funcionalidad de los procesadores actuales.

1. Computadores de Juego de Instrucciones Complejo (**CISC, Complex Instruction Set Computer**). Disponen de más de 80 instrucciones.

2. Computadores de Juego de Instrucciones Reducido (**RISC, Reduced Instruction Set Computer**), el PIC fue el primer microcontrolador en usar esta arquitectura.
3. Computadores de Juego de Instrucciones Específico (**SISC, Specific Instruction Set Computer**), el juego de instrucciones es reducido, pues éstas se adaptan a las necesidades de la aplicación prevista.

Sesión Introductoria.

Objetivo General.

Que el alumno conozca el funcionamiento físico y lógico, de un microprocesador y los elementos que lo componen.

Objetivo Particular.

1. Que el alumno se familiarice con algunos de los conceptos más usuales en el estudio de estos dispositivos.
2. Que conozca la estructura interna de un microcontrolador.

Guía de estudio para el alumno.

Después de la explicación del profesor, leer la sección introductoria para así entender los conceptos que aquí se explican.

Introducción.

Un microprocesador es un circuito de alta escala de Integración (LSI, Large Scale Integration), compuesto de muchos circuitos más simples como son los Flip-Flops, contadores, registros, decodificadores, comparadores, etc. Todos ellos en una misma pastilla de silicio, de modo que el microprocesador puede ser considerado un dispositivo lógico de propósito general o universal. Los componentes que llevan a cabo físicamente la lógica y operación del microprocesador se denominan hardware. Además existe una lista de instrucciones (acciones) que puede realizar un microprocesador. Estas constituyen el lenguaje del microprocesador o software.

Ejemplifiquemos, un microprocesador que realice cuatro tareas lógicas: AND, OR, NAND, XOR. Estas cuatro acciones serían el lenguaje del micro y a cada una le corresponderá una combinación binaria de dos dígitos.

Tabla 1.1 En esta tabla podemos ver el funcionamiento del microcontrolador.

CÓDIGO	ACCIÓN
00	AND
10	OR
01	NAND
11	XOR

El hardware quedaría de la siguiente forma:

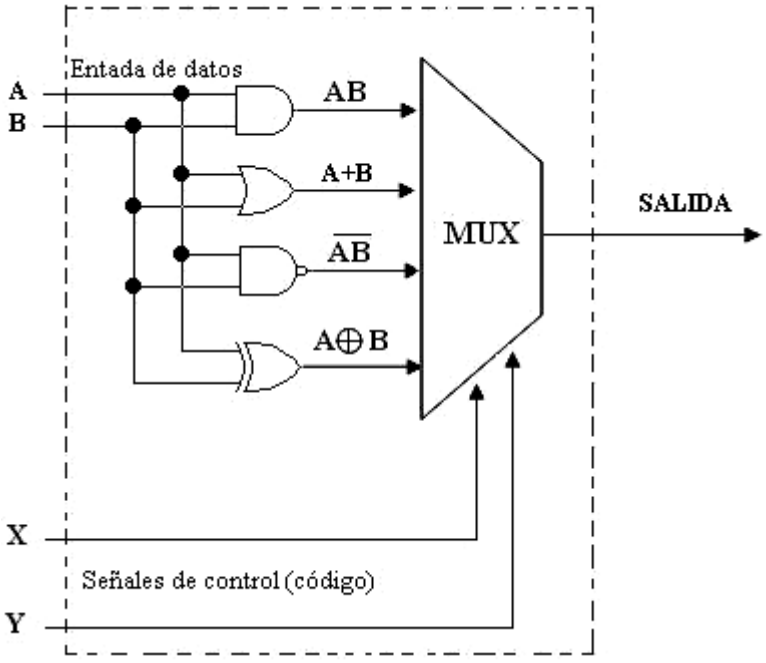


fig. 1.1 Diagrama esquemático del circuito

En este ejemplo, se puede ver claramente lo que es un microprocesador. Las señales de control, son las que seleccionan cada una de las cuatro instrucciones (acción) que el microprocesador puede realizar. Los datos se presentan en las líneas A y B.

Sin embargo, en la actualidad se requiere que un sistema cuente con una unidad de control, unidad aritmético-lógica y algunos registros. La forma en que están conectadas estas unidades se denomina, organización de un microprocesador. A continuación se muestra la estructura general de un microprocesador de 8 bits (fig1.2).

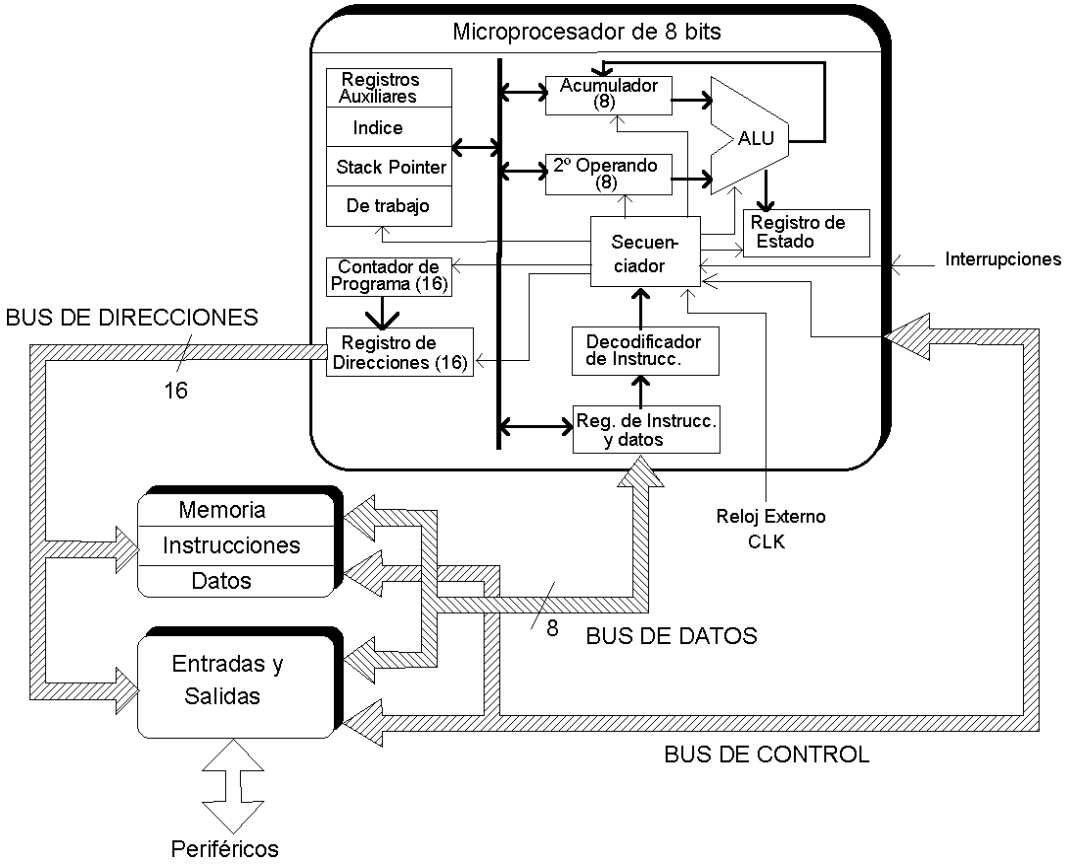


fig. 1.2 Estructura general de un microprocesador de 8 bits

Unidad central de Proceso:

1. **Unidad Aritmética-Lógica (ALU, Arithmetic Logic Unit):** Como su nombre lo indica, realiza las operaciones aritméticas y lógicas indicadas por las instrucciones.
2. **Secuenciador**: Instruido por el código de la instrucción en curso, activa las líneas de selección de la ALU para realizar la operación.
3. **Acumulador**: Proporciona el primer operando y en él se guarda el resultado de la última operación realizada en la ALU.
4. **Registro 2º Operando**: Proporciona el 2º operando, para realizar la instrucción y viene normalmente suministrado por el código de operación de la instrucción a ejecutar, según los diferentes modos de direccionamiento.
5. **Registro de Estado**: Está formado por bits denominados banderas (flags) que se ponen a 1 ó 0 de acuerdo con el resultado obtenido. Algunos bits típicos son:

    a) **Z**, bit zero; se pone a 1 si el resultado fue nulo.
    b) **C**, bit carry; se pone a 1 si hubo acarreo de orden superior.
    c) **V**, bit overflow; se pone a 1 si hubo desbordamiento.
    d) **I**, bit de interrupción; este bit es independiente del resultado. Escribiendo 1 en él, por medio de la instrucción correspondiente, se puede habilitar la interrupción exterior.

6. **Decodificador de Instrucciones**: Selecciona las posiciones que corresponden a esa instrucción en una memoria ROM interna de la CPU. En ella se almacenan las diferentes instrucciones elementales o las que componen esa instrucción.
7. **Registro de Instrucción (IR, Instruction Register)**: Guarda temporalmente la instrucción que habrá de ejecutarse.

8. **Contador de Programa (PC, Program Counter)**: Es el registro que almacena la dirección de la próxima instrucción a ejecutarse.
9. **Registros índices**: Permiten el acceso a memoria RAM de manera indirecta.
10. **Puntero de Pila (SP, Stack Pointer)**: También conocido como, puntero o apuntador de pila, es el registro que apunta hacia la dirección en que se encuentra la pila.
11. **Registros de trabajo y auxiliares**: Son los registros de propósito general. No tienen asignada una tarea específica y pueden ser utilizados como apoyo a otros registros o asignándoles una tarea especial, dentro de un programa.

Sistema Mínimo.

Un microprocesador por sí mismo no es capaz de realizar tarea alguna. Para ello requiere de los siguientes elementos:

1. Una fuente de alimentación
2. Un circuito de reloj
3. Dispositivos de memoria
4. Interfaz o módulo de entrada y salida (E/S)

La implementación de este circuito, constituye lo que se conoce como sistema mínimo. El siguiente diagrama corresponde a un sistema basado en la estructura de Von Newman. Sus bloques básicos son los siguientes.

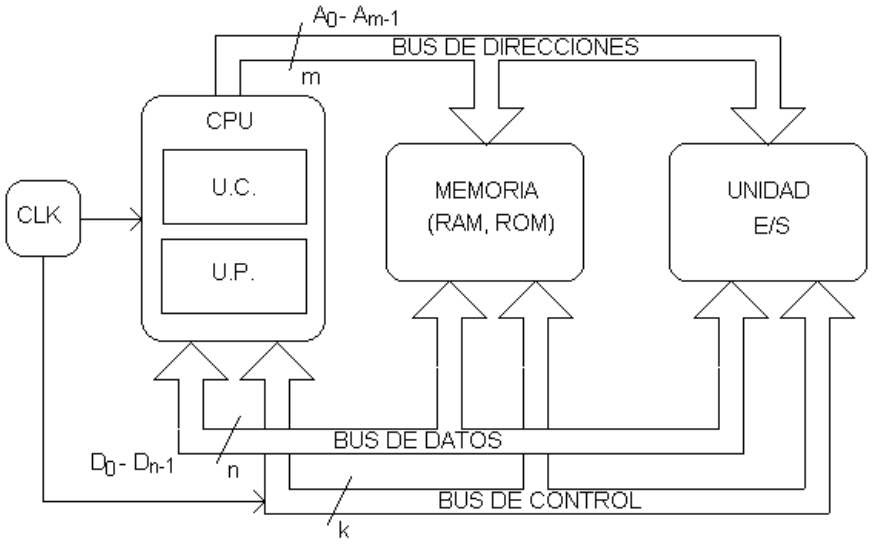


Fig. 1.3 Sistema basado en la estructura de Von Newman.

Microprocesador o CPU está formado por dos bloques:

1. Unidad de Control.
2. Unidad de Proceso.

Memoria:

1. **RAM** o volátil: Sirve para guardar las variables y los datos. (En los microcontroladores la memoria de instrucciones y datos está dentro del mismo chip).
2. **ROM:** En esta memoria se guarda el programa o instrucciones que gobiernan la aplicación.

Unidades E/S, son los elementos encargados de recibir y entregar información al exterior.

Los tres módulos están conectados entre sí, por medio de los Buses de Sistema**.** Cada bus está formado por un conjunto de conductores, por los cuales se transmite la información digital en forma de pulsos eléctricos.

Microcontrolador.

Un microcontrolador está compuesto por la unidad central de proceso (CPU), memoria RAM, Memoria de programa y unidades de entrada-salida. También necesita de circuitos adicionales, como son de Reset, generador de pulsos y alimentación.

Sugerencias para el profesor.

1. Explicar los diferentes módulos que componen a un microprocesador y un microcontrolador.
2. Definir los microcontroladores a utilizar y proponer los circuitos para programarlos, en caso de necesitarse.

Bibliografía.

1. La que el profesor indique, además de los manuales del fabricante y de usuario del dispositivo que se empleará en estas prácticas (cabe señalar que el microcontrolador a manejar es elección del profesor).

PRÁCTICA 1

“LENGUAJE ENSAMBLADOR”

Objetivo General.

Que el alumno conozca las partes que forman el set de instrucciones (juego de instrucciones) y aprenda a utilizarlo, así como los conceptos fundamentales para el manejo de los microcontroladores.

Objetivo Particular.

1. Que el alumno conozca las categorías de instrucciones.
2. Cómo se dividen las instrucciones.
3. Cuál es su función de las instrucciones.
4. Algunos conceptos y definiciones necesarias para el entendimiento de esta práctica.

Guía de estudio para el alumno.

1. Leé la introducción de esta práctica, trata de aprender los conceptos que se mencionan, te serán de gran utilidad.
2. Leé la parte del manual o la bibliografía que el profesor dio en la sesión introductoria donde habla del set de instrucciones. Identifica el tipo de instrucciones en las que se divide y las partes que lo forman.
3. Ejercicio, divide las instrucciones del microcontrolador en las siguientes categorías; aritméticas y lógicas, transferencia de datos, transferencia de control, manipulación de bit y condicionales, y Otras.

4. Si tienes dudas, pídele a tu profesor que te las aclare dentro de la clase de laboratorio.

Introducción.

El lenguaje ensamblador, es de bajo nivel, tiene una correspondencia directa de (1 a 1) con el código máquina, es decir, que a cada instrucción le corresponde una codificación en dígitos, en binario.

Se utiliza el sistema binario, porque en la electrónica digital se trabaja con bits (0 ó 1), es decir, hay voltaje o no, pero como esto es complicado, se decidió trabajar con grupos de 8 bits (byte), y es por eso que se utiliza el sistema hexadecimal.

Cada microcontrolador tiene un set de instrucciones, para comprender las instrucciones debemos entender los siguientes conceptos.

Mnemónicos: Son abreviaturas de palabras en inglés, que describen la acción de cada instrucción. Cada mnemónico es asociado con un opcode.

Ejemplo:

Tabla2.1 Representación de un par de instrucciones con su mnemónico y la acción que realiza

PALABRA	MNEMÓNICO	SIGNIFICADO
MOVE	MOV	MOVER
LOAD	LD	CARGAR

Nota: estos dos mnemónicos hacen lo mismo, sólo que para diferentes microcontroladores.

Código de operación : (**Opcode,** Operation Code) es el código numérico de la instrucción, que representa la operación a realizar por el CPU. El Opcode es un grupo de bits que indican al microcontrolador la operación a realizar.

Ejemplo:

Tabla2.2 Instrucciones y sus OPCODES.

MNEMÓNICO	OPCODE EN BINARIO	OPCODE EN HEXADECIMAL
RETURN[1]	0000 0000 1000	008
CLC[2]	1001 0100 1000 1000	9488
DECA[3]	1000 1011	8B

1. Retorno de subrutina (microcontrolador PIC)
2. Limpiar bandera de acarreo (microcontrolador AVR)
3. Decrementa el acumulador (microcontrolador COP)

Partes del set de instrucciones.

Dentro del set de instrucciones algunos fabricantes manejan una introducción, en la que se especifican los tipos de direccionamiento (se verán en la práctica 2), clasificación de las instrucciones y también conceptos fundamentales para la comprensión del mismo. Cada fabricante maneja un set de instrucciones; por tanto todos son diferentes, en esta sección se pondrá una idea general.

Ejemplo:

Tabla2.3 No todas las columnas aparecen dentro del set de instrucciones, sin embargo deberá contar con al menos cuatro de estas.

MNEMÓNICO	OPERANDOS	DESCRIPCIÓN (CON PALABRAS)	DESCRIPCIÓN (OPERACIONAL)	MODO DE DIRECCIONAMIENTO *	BANDERAS AFECTADAS	CICLOS DE RELOJ
1° COLUMNA. EN ESTA COLUMNA SE ENCUENTRAN LAS INSTRUCCIONES O MNEMÓNICOS.	2° COLUMNA. ENCONTRAMOS LOS REGISTROS ENTRE LOS QUE SE LLEVA A CABO LA OPERACIÓN.	3° COLUMNA. AQUÍ SE DESCRIBE CON PALABRAS LA ACCIÓN QUE REALIZA LA INSTRUCCIÓN.	4° COLUMNA. DESCRIBE QUE PASA CON LOS OPERANDOS Y DONDE QUEDA EL RESULTADO. LO HACE A TRAVÉS DE SÍMBOLOS.	5° COLUMNA. INDICA QUE TIPO DE DIRECCIONAMIENTO PUEDE SER MANEJADO POR ESA INSTRUCCIÓN.	6° COLUMNA. NOS MUESTRA LOS BITS QUE PUEDE SER AFECTADOS POR ESA INSTRUCCIÓN. EJEMPLO, BANDERA DE ACARREO CERO.	7° COLUMNA. NOS MUESTRA LOS CICLOS DE RELOJ QUE CONSUME CADA INSTRUCCIÓN.

* Esta columna es poco común que aparezca.

Categorías de Instrucciones.

El set de instrucciones se divide dependiendo del tipo de la acción que realizan las instrucciones, podríamos agruparlo en cinco grupos que son:

1. **Aritméticas y lógicas**: Como ya sabemos las operaciones aritméticas son la suma, resta, multiplicación y división, Ejemplo de las lógicas pueden ser las compuertas que conocemos: AND, OR, XOR.
2. **Transferencia de datos**: Son instrucciones que tienen como origen o destino (e incluso ambos) un registro del microcontrolador. No realizan operaciones

con la ALU, sólo transfieren un dato de un sitio a otro. Estas instrucciones, junto con las aritméticas, son las más frecuentes.

3. **Transferencia de control** (saltos y subrutinas): Las instrucciones de un programa son ejecutadas usualmente en orden. Sin embargo, las instrucciones de salto pueden ser utilizadas para cambiar la secuencia de ejecución. No realizan operaciones con la ALU.
4. **Manipulación de bit y condicionales**: Con estas instrucciones se puede cambiar el estado de un bit (bandera), es decir, cambiarlo de un 1 a un 0 o viceversa. Las condicionales sirven para evaluar, si el dato es verdadero o falso.
5. **Otras**: Son instrucciones que por su naturaleza, no entran en ninguna de las clasificaciones anteriores, pero no por eso dejan de ser muy importantes.

El orden de los grupos puede cambiar dependiendo de cada microcontrolador, así como, la cantidad de éstos, pero en general todas las instrucciones están dentro de alguna de estas categorías.

Por ejemplo, los microcontrolador PIC manejan la siguiente clasificación para sus instrucciones:

1. Instrucciones que operan con bytes y que involucran algún registro de la memoria interna.
2. Instrucciones que operan sólo sobre el registro W (work o acumulador) y que permiten cargarle una constante implícita o incluida literalmente en la instrucción (literal o dato inmediato).
3. Instrucciones que operan sobre bits individuales de los registros de la memoria interna.
4. Instrucciones de control de flujo del programa, es decir las que permiten alterar la secuencia de ejecución del programa.

5. Por último, se agrupan algunas instrucciones que llamaremos especiales, porque no encajan en ninguna de las clasificaciones anteriores.

Conceptos y definiciones.

Acumulador: Este registro funciona como uno de los operandos y también es en el que se guarda el resultado de la última operación. Algunos microcontroladores tienen más de uno y otros no se basan en acumuladores.

Registros de control: Estos registros sirven para configurar, es decir, para indicar al microcontroladores como debe de funcionar. Un ejemplo de configuración, es cuando le indicamos al celular o al televisor en qué idioma debe mostrar su menú (siempre nos mostrará el mismo idioma, a menos que lo reconfiguremos). De esta misma forma funcionan los registros en los microcontroladores.
Registros de propósito general: Estos registros no tienen funciones específicas y se pueden usar para almacenar datos temporalmente.

Registros índices: Permiten el acceso a memoria RAM de manera indirecta.

PC (contador de programa): Es el registro en el que se almacena la dirección de la próxima instrucción a ejecutarse.

Apuntador: Un apuntador es un registro que contiene una dirección, en la cual se especifica una localidad de memoria. Es decir, un apuntador "apunta" a la localidad de algún dato.

Pila (Stack): La sección de pila es memoria dedicada a almacenar datos y direcciones en forma de pila (consecutivamente). Existen dos tipos LIFO y FIFO.

LIFO (Last In, First Out): Estructura donde el último dato en entrar es el primero en salir.

FIFO (First In, First Out): Estructura donde el primer dato en entrar es el primero en salir.

Puntero de pila (SP, Stack Pointer): Puntero o apuntador de pila, es el registro que apunta hacia la dirección en que se encuentra la pila.

Memoria RAM o volátil: Guardar las variables y los datos. (En los microcontroladores la memoria de instrucciones y datos está dentro del mismo chip)

Memoria para el programa o no volátil: Tipo ROM, esta memoria guarda el programa o instrucciones que gobiernan la aplicación.

1. ROM.
2. PROM.
3. EPROM.
4. EEPROM
5. FLASH (Funciona como una ROM y una RAM pero consume menos y es más pequeña). Actualmente es la de mayor utilización.

BIT: Unidad mínima de información en un sistema binario, abreviación de digito binario puede tener el valor de 0 ó 1.

BYTE: Conjunto ordenado de 8 bits, cada BIT tiene un valor.

NIBLE: Grupo ordenado de 4 bits. Dos nibles forman un byte.

Palabra: Conjunto de bits relacionados con el código máquina de una instrucción.

Bit más significativo: (MSb, Most Significant Bit).

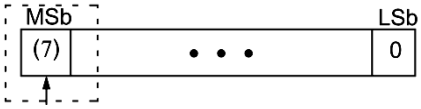


Fig.2.1 Este es el bit más significativo

Bit menos significativo: (LSb, Least Significant Bit).

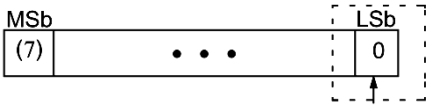


Fig.2.2 Este es el bit menos significativo

Bandera (Flag): Bit que proporciona información específica acerca del último proceso ejecutado.

Fetch: La dirección de memoria de programa que se encuentra almacenada en el PC, es utilizada para capturar la instrucción localizada en esta dirección. La instrucción es copiada al registro de instrucciones (IR, Instruction Register). El registro PC es incrementado para apuntar a la siguiente instrucción disponible.

Etiqueta: Elemento de programación que permite sustituir un valor numérico por un texto o cadena de caracteres, sirve como ayuda para ordenar el programa.

ASCII (American Standard Code for Information Interchange): Codificación estándar para el intercambio de información de los EEUU.

Lazo o loop: Sección de un programa que se ejecuta cíclicamente.

Subrutina: Rutina que se encuentra dentro de un programa, la cual puede ser ejecutada varias veces dentro del programa. Simplifica el programa principal.

Sugerencias para el profesor.

1. Mencionar conceptos fundamentales para facilitarle al alumno la comprensión del set de instrucciones.
2. Explicar las partes del set de instrucciones.
3. Poner ejemplos de cómo se escriben instrucciones, que hacen, dónde queda el resultado y que banderas puede afectar.
4. Hacer énfasis en la significancia (MSb y LSb).

Nota: El trabajo de laboratorio depende del profesor.

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

PRÁCTICA 2

"HERRAMIENTAS DE DESARROLLO (SOFTWARE)"

Objetivo General.

Que el alumno aprenda a utilizar el software de desarrollo.

Objetivo Particular.

1. Que el alumno conozca el entorno del software.
2. Que el alumno comprenda para qué sirve el editor de texto.
3. Que el alumno sepa cómo compilar un programa.
4. Que el alumno aprenda a crear y guardar un proyecto.

Guía de estudio para el alumno.

1. Leé la introducción de esta práctica, trata de aprender los conceptos que se mencionan, te serán de gran utilidad.
2. Leé el manual del usuario y la introducción del software que utiliza el microcontrolador y familiarízate con el entorno de la herramienta.
3. Busca en la página del fabricante del microcontrolador, como crear un proyecto, como guardar un proyecto, como ensamblar o compilar un proyecto y las reglas más usuales del software.
4. Si tienes dudas, pídele a tu profesor que las aclare dentro de la clase de laboratorio.

Introducción.

Hoy en día las herramientas que nos ayudan a la programación de los microcontroladores son variadas, existen desde las de más bajo nivel hasta las que se conocen como de alto nivel, pero cualquiera que sea el microcontrolador, tiene su propia herramienta de desarrollo. No obstante casi todos los microcontroladores se pueden programar con distintos lenguajes de programación como son; Basic, C, C++, ensamblador, etc.

Algunas de las herramientas más populares para los microcontroladores más usuales son: Mplab de Microchip, esta herramienta es muy usada en el nivel académico y en las empresas dedicadas al desarrollo de soluciones con microcontroladores, con sus PIC's.

Otra de estas herramientas es AVR Studio, de fácil uso para la realización de proyectos con los microcontroladores AVR's. Estos sólo son algunos ejemplos de las herramientas de desarrollo que se pueden usar para realizar estas prácticas.

No sólo basta con realizar el programa, si no que ahora necesitamos una herramienta de tipo hardware y software que nos ayude a grabar el microcontrolador. Las herramientas de desarrollo del fabricante nos permiten hacer la grabación del microcontrolador, pero necesitamos el hardware para hacerlo. En algunos casos el fabricante del microcontrolador proporciona el circuito para armarlo y así tenerlo, en algunos casos el fabricante del microcontrolador vende este mismo hardware, para el caso de que el fabricante del microcontrolador no proporcione el hardware y se decida no comprar, el profesor podrá elegir el grabador y el software que considere apto para el microcontrolador con el que se trabaja.

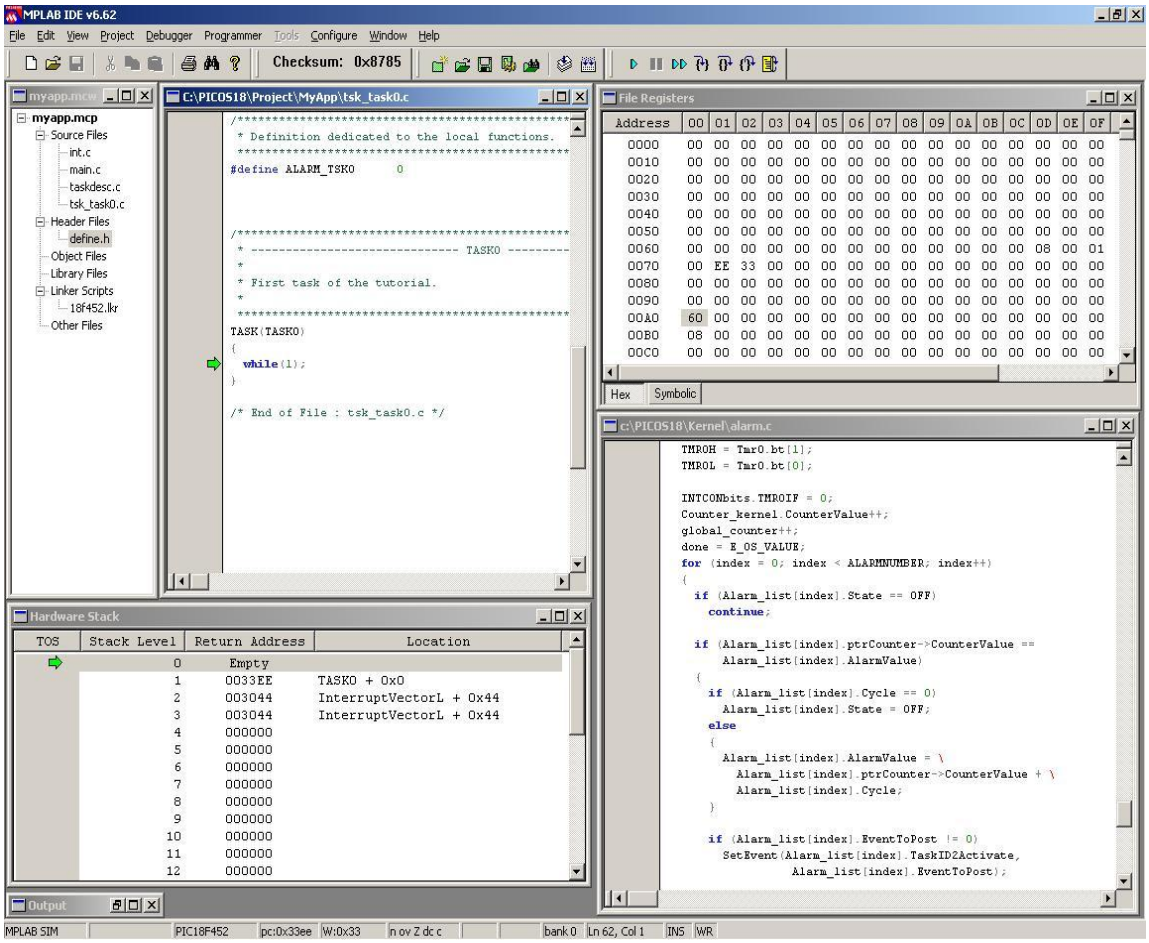


Fig.3.1 Entorno Mplab para microcontroladores Pic

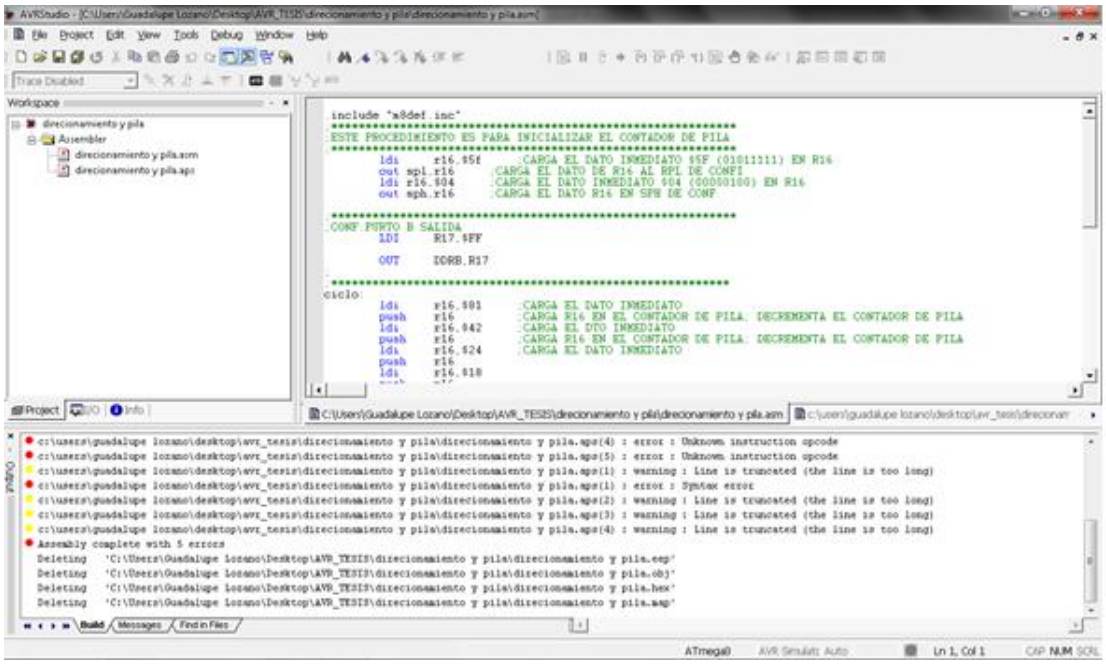


Fig.3.2 Entorno AVR Studio para microcontroladores AVR

Sugerencias para el profesor.

1. Mencionar como crear un proyecto, como guardar un proyecto, como compilar un proyecto.
2. Explicar la ventana principal con los botones rápidos para realizar las acciones antes mencionadas, crear una presentación, para que sea más fácil de entender, donde se explique a detalle las ventanas del software, donde se captura el programa, donde se puede simular el programa, y donde se puede ver la creación de los archivos que contienen el programa que se va a cargar en el microcontrolador.
3. Poner ejemplos de cómo se crea un proyecto, como se guarda y como se ensambla o compila.

Nota: El trabajo de laboratorio depende del profesor.

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del software que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

PRÁCTICA 3

"PROGRAMACIÓN EN ENSAMBLADOR 1"

(Configuración de puertos)

Objetivo General.

Que el alumno conozca el concepto de puerto y como se puede configurar.

Objetivo Particular.

1. Que el alumno sepa que registros están asociados a la configuración de puertos de E/S.
2. Que el alumno conozca las diferentes formas de programación de un puerto E/S.
3. Que el alumno lleve reglas de programación.

Guía de estudio para el alumno.

1. Busca la forma de configurar un puerto como entrada y como salida. En los manuales se encuentra en la parte que se denomina, configuración de puertos **(I/O Port Configuration** o **I/O Ports)**.

Introducción

Los microcontroladores requieren de una interfaz para comunicarse con la circuitería externa. Esta interfaz es denominada comúnmente como puerto. Existen puertos de entrada y salida los cuales permiten que las señales (o datos) sean leídos del exterior o mandados al exterior del microcontrolador. Los puertos tienen implementados pines, (terminales del circuito integrado), los cuales, dependiendo de la aplicación, son conectados a un sin fin de dispositivos como teclados, interruptores, sensores, relevadores, etc.

Puerto: Es un grupo de pines utilizado para mandar o recibir información. Pueden tener únicamente salidas, entradas o incluso una combinación de pines de entradas y salidas. Actualmente la mayoría de los puertos son bidireccionales, es decir pueden ser configurados como pines de entrada o salida dependiendo de los requerimientos del usuario.

Entradas: Un dispositivo externo otorga al microcontrolador una señal en estado alto o bajo. El nivel lógico es leído por el microcontrolador como un bit sencillo de información de entrada.
Salidas: El microcontrolador fuerza uno de sus pines a un estado alto o bajo. El voltaje de salida en el pin, corresponde a un bit sencillo de información.

Usualmente cada puerto (grupo de pines) tiene asignada una dirección como si fuera un registro en memoria. La escritura a una dirección asignada a un puerto, ocasiona que los pines asociados con la dirección del puerto sean forzados a un estado alto o bajo de acuerdo al valor escrito. Si los puertos no son mapeados en memoria, se tendrán instrucciones especiales de Entrada/Salida para accederlos.

Un puerto puede ser configurado como entrada o como salida. Para esto se requiere configurar algunos registros, como pueden ser el registro de configuración, el de datos y de pines, éste último es de sólo lectura y no siempre existe, ya que, algunos dispositivos sólo tienen dos registros asociados a los pines, uno de configuración y otro que agrupa el de datos y pines. El registro de configuración sirve para indicarle al dispositivo si por ese pin van a entrar o salir datos y el otro u otros para sacar información o leerla de algún dispositivo externo.

Dentro del laboratorio.

Como en todo programa estructurado debemos de seguir ciertas reglas de programación para hacer más comprensible el programa, en ensamblador, no importando la herramienta de desarrollo que se utilice es más fácil poder encontrar un error si se sigue una estructura de programación. Los siguientes pasos nos ayudaran para realizar un programa de forma que sea comprensible y que nos ayude a encontrar una falla si es que el programa nos indica que tenemos un error.

Debemos iniciar con el nombre del programa, este nombre deberá ser corto y muy específico ya que el programa en su estructura puede realizar muchas operaciones, pero todas estas están unidas con un fin común, por eso es necesario ser especifico en el nombre del programa, por ejemplo:

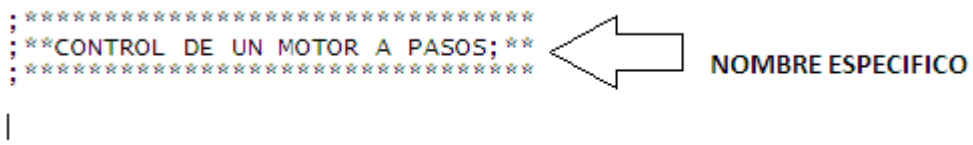


Fig. 4.1 Nombre de un programa en ensamblador.

Al iniciar un programa debemos indicar en donde vamos a comenzar, esto lo logramos indicando en el programa, que vectores serán afectados y dependiendo de él orden de los vectores de interrupción es que debemos indicar donde comenzamos.

```
;*****************************************
.include "m8def.inc"                    ;incluir este archivo atmega8

                rjmp reset                              ;saltar a reset

.*****************************************
```

Fig. 4.2 Ejemplo de un inicio tomando en cuenta el vector Reset

Las etiquetas de retornos deben de estar colocadas en la parte extrema izquierda, es recomendable dar dos espacios con la tecla TAB, esto nos ayuda a tener un orden en la estructura del programa, posteriormente están las instrucciones, al igual que en el espacio anterior es recomendable dejar un, TAB, ahora en la línea de programación colocaremos un comentario que describa brevemente la acción de esa instrucción en el registro que se esté usando. Esto nos servirá para tener un control del programa y que cuando lo estemos depurando al momento en el que se nos señale un error sea más práctico encontrarlo ya que muchas veces los errores de programación pueden tener su inicio en líneas anteriores y no en las líneas que especifica la herramienta de desarrollo. Este es el bit más significativo.

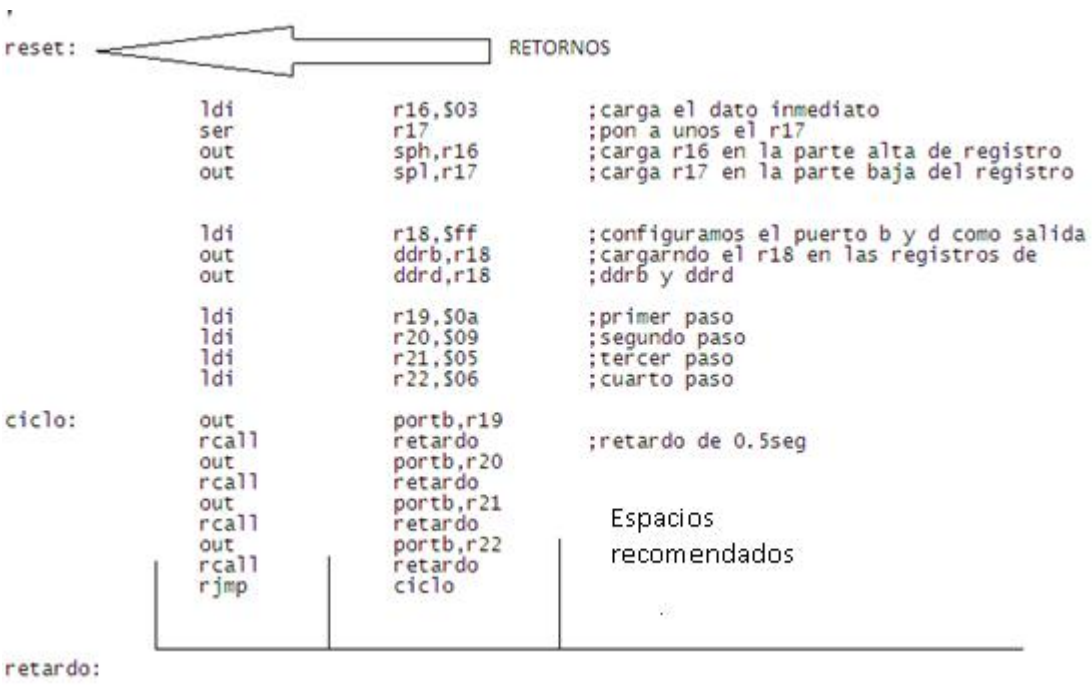


Fig.4.3 Estructura de un programa en ensamblador.

Realizar un programa, que lea datos por los pines 0, 1, 2 y 3 de un puerto. Estas entradas estarán dadas por un DIP switch y las salidas serán los pines 4, 5, 6 y 7 del mismo puerto, en los pines de salida estarán conectados LEDS, en los que se verán las combinaciones que se den con el DIP switch, como se muestra.

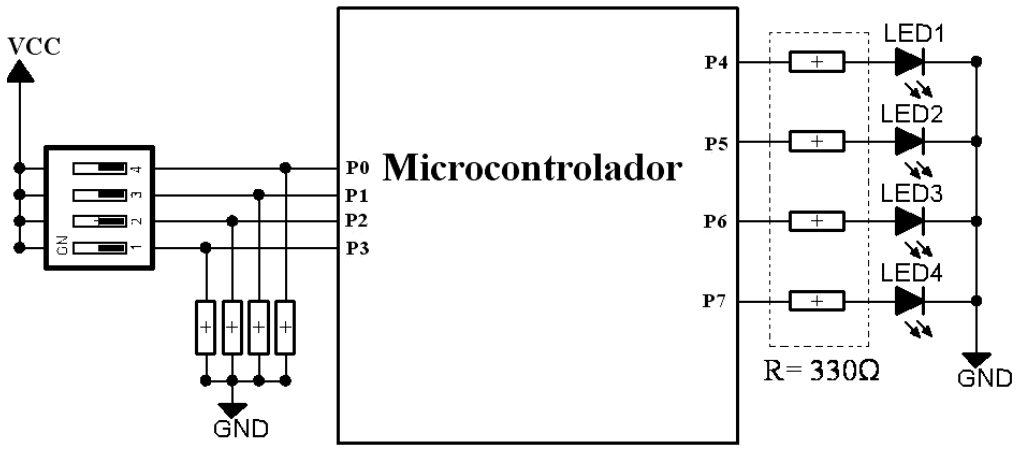


Fig.4.4 Hardware

Se recomienda que las entradas y salidas sean por el mismo puerto, para reafirmar que un puerto puede ser configurado pin por pin.

Cambiar el esquema anterior por el siguiente, para esto se tienen que habilitar las resistencias de carga activa (pull-up) del microcontrolador, con esto nos ahorramos cuatro resistores, como se muestra.

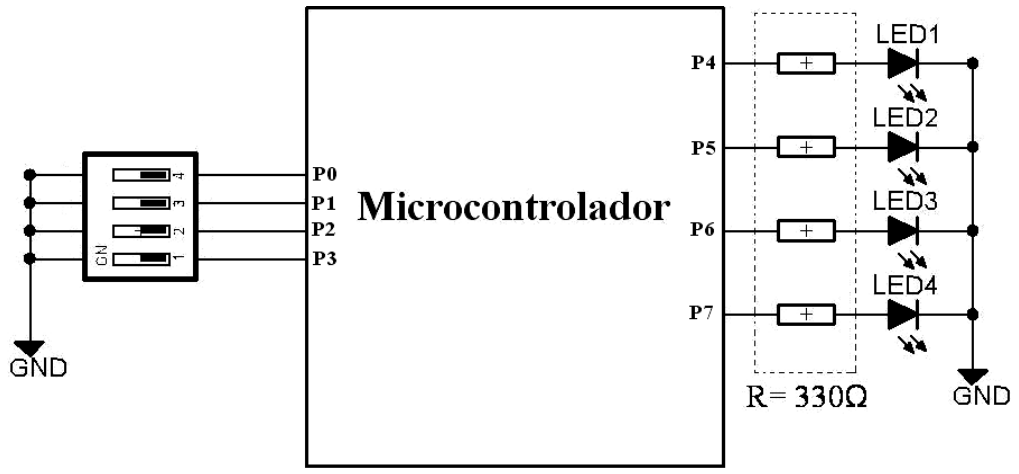


Fig. 4.5 Hardware

Para los dos circuitos anteriores se tiene que medir el voltaje en las entradas, para ver la diferencia, entre entrada de alta impedancia y de carga activa (pull-up). (Cuando el DIP switch esté abierto).

Desarrollar un programa, en el cual se incrementen o decrementen dos registros o localidades de memoria, el primer registro o localidad debe decrementarse de (0xFF a 0x00) o incrementarse de (0x00 a 0xFF), y cuando esto haya sucedido el segundo registro o localidad debe decrementarse o incrementarse en uno (1), y regresar a que el primer registro o localidad se decremente o incremente. El programa debe terminar cuando el segundo registro llega a ser cero, como se muestra en los diagramas.

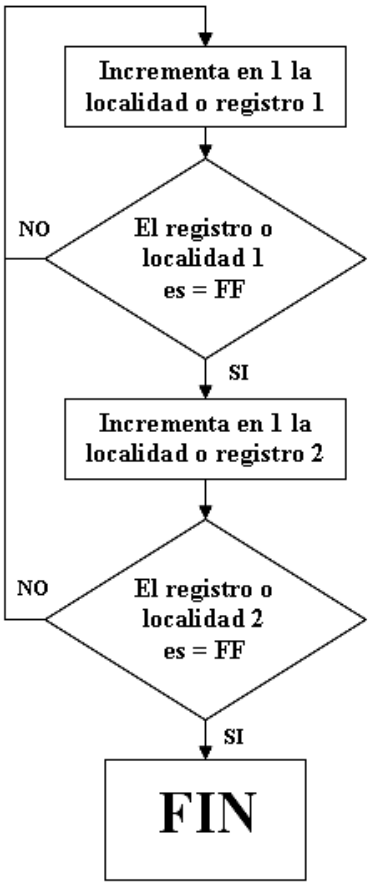


Fig.4.6 Incrementos

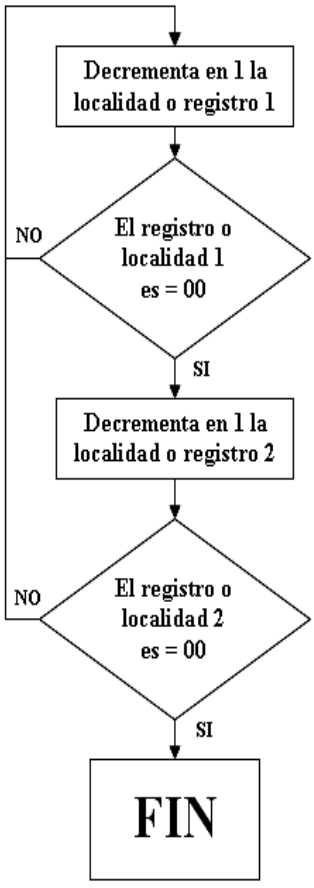


Fig.4.7 Decrementos

NOTA:

Los circuitos armados se entregaran después de la práctica 3 (“Herramientas de desarrollo (software)”.

Sugerencias para el profesor.

1. Explicar qué es un puerto, cómo se configura, y los registros asociados a este.
2. Formar equipos de 2 ó 3 alumnos para que desarrollen el programa, (desarrollo dentro del laboratorio), que consiste en configurar la mitad de un puesto como entrada (pin 0, 1, 2 y 3) y la otra mitad como salida (pin 4, 5, 6 y 7). Las entradas están dadas por el DIP switch.
3. Que los alumnos desarrollen el programa, con entradas de carga activa,(pull-up).
4. Explicar cómo debe de funcionar el programa de incrementos o decrementos, y dejar que lo desarrollen en equipo.

Bibliografía.

La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

PRÁCTICA 4

"PROGRAMACIÓN EN ENSAMBLADOR 2"

(Direccionamiento y pila)

Objetivo General.

Que el alumno conozca que es direccionamiento y pila.

Objetivo Particular.

1. Que el alumno conozca los diferentes tipos de direccionamiento.
2. Que el alumno entienda el concepto de pila.
3. Que el alumno realice su primer programa en ensamblador.

Guía de estudio para el alumno.

1. Busca los tipos de direccionamiento del microcontrolador que utilizarás en las prácticas. En los manuales se encuentra en la parte llamada modos de direccionamiento **(Addressing Modes)** o en el set de instrucciones.
2. Buscar el tipo de pila que tiene el microcontrolador. FIFO o LIFO.

Modos de direccionamiento.

Los modos de direccionamientos son distintas formas que tiene la CPU de acceder a los datos que necesitan una instrucción para su ejecución. En los siguientes ejemplos, se enumeran los más importantes.

1. **Direccionamiento inmediato**: El dato al que se hace referencia se encuentra "dentro" de la instrucción, no es necesario acceder a memoria. Este modo de direccionamiento se indica, (usualmente) mediante el signo **#**, la letra K (literal), etc.

Ejemplo:

Tabla.5.1. El dato, en este caso 15, es cargado directamente al acumulador A.

MNEMÓNICO	DESCRIPCIÓN	DESCRIPCIÓN OPERACIONAL
LD A, #15	Carga A con el dato inmediato	A←15

	OPERANDO	ANTES	DESPUES
LD A, #15	A	¿?	15

2. **Direccionamiento directo**: El dato se encuentra en la dirección de memoria especificada.

Ejemplo:

Tabla.5.2 Lo que tenemos en la localidad de memoria $05 (en algunos entornos el signo de pesos indica que el dato está en notación hexadecimal) pasa al acumulador B y la localidad $05 conserva su valor. .* El contenido de la dirección efectiva es el operando que se utiliza en la instrucción.

MNEMÓNICO	DESCRIPCIÓN	DESCRIPCIÓN OPERACIONAL
LDAB,$05	Carga B con el contenido de la dirección de memoria $05	B←M

	OPERANDO	ANTES	DESPUÉS
LDAB,$05*	B	¿?	26
	M ($05)=26	26	26

3. **Direccionamiento indirecto**: La dirección de memoria es especificada por el contenido de un registro llamado puntero. En lenguaje ensamblador, la notación que se utiliza es, el nombre del registro entre corchetes, ejemplo [X].

Ejemplo:

Tabla.5.3. El registró puntero X está apuntando hacia la localidad 63 (dirección efectiva), dentro de esta localidad está el dato 10, el cual es cargado en el acumulador.

MNEMÓNICO	DESCRIPCIÓN	DESCRIPCIÓN OPERACIONAL
LDA, [X]	Carga A con el contenido de la dirección que se indica dentro del registro [X].	A←[X]

	OPERANDO	ANTES	DESPUES
LD A,[X]	A	¿?	10
	X	63	63
	X=63	10	10

4. **Direccionamiento indexado**: Este modo de direccionamiento se utiliza para acceder a tablas en la memoria. Se toma la dirección del registro índice, se le suma un desplazamiento (offset) y el contenido de esa dirección es el dato buscado.

Ejemplo:

Tabla.5.4. El registro índice en este caso el X está apuntando hacia la localidad 42, se le suma el offset de 5 que está indicado en la instrucción y con esto la localidad efectiva es 47, dentro de 47 está el dato 12, que es el que se carga en el acumulador A.

MNEMÓNICO	DESCRIPCIÓN	DESCRIPCIÓN OPERACIONAL
LDAA, 5X	Carga A con el contenido de la dirección especificada por X, más un offset de 5.	A←[X+5]

	OPERANDO	ANTES	DESPUES
LDAA,5X	A	¿?	12
	X	42	42
	X+5	47	47
	X+5=47	12	12

Nota: Los ejemplos de direccionamiento 1, 2 y 4 son del 68HC11 y el 3 del COP8CBR9.

Pila (Stack).

La pila es memoria dedicada a almacenar datos y direcciones en forma de pila (consecutivamente). En la pila se guardan las direcciones de retorno de subrutina e interrupciones, también se utiliza para guardar el contenido de algunos registros, al inicio de una subrutina de interrupción (se verá en la práctica 5). La instrucción para introducir un dato es PUSH y para sacarlo es POP.

Existen dos tipos LIFO y FIFO:

1. **LIFO (Last In, First Out)**: Estructura donde el último dato en entrar es el primero en salir.

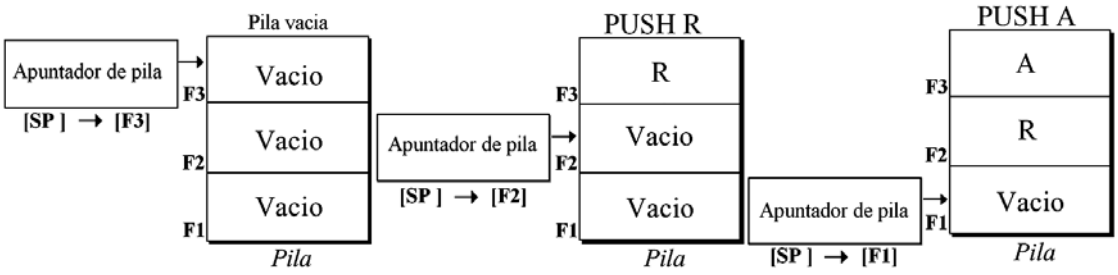


Fig.5.1. Almacenamiento de los datos R y A en la pila.

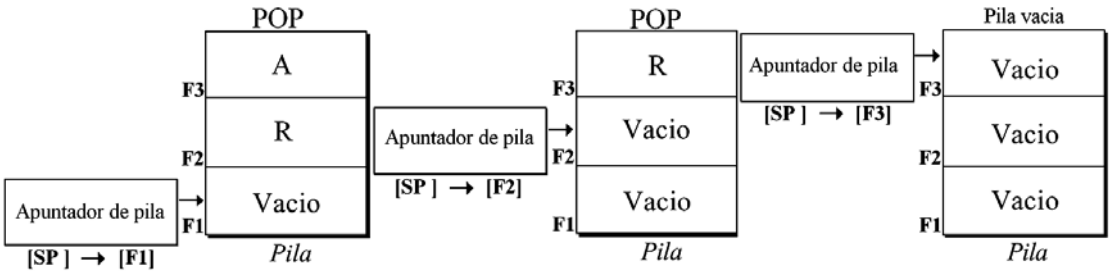


Fig5.2. Recuperación de los datos R y A

2. **FIFO (First In, First Out)**: Estructura donde el primer dato en entrar es el primero en salir.

Puntero de pila (SP, Stack Pointer): Puntero o apuntador de pila, es el registro que apunta hacia la dirección en que se encuentra la pila. **(Puede ser necesario inicializar el puntero cuando se utilizan llamadas a subrutinas e interrupciones).**

Subrutina: Es un segmento de código que se repite varias veces dentro de un programa, pero que sólo se escribe una vez, cuando se quiere acceder a este se realiza un llamado a subrutina, esto se hace a través de una instrucción. Simplifica el programa principal.

Dentro del laboratorio.

1. Realizar un programa, en el cual se introduzcan datos en la pila y después se recuperen, es importante tomar en cuenta el tipo de pila para la recuperación de datos.
2. Realizar un programa que muestre la secuencia en LEDS de la Figura 5.4.

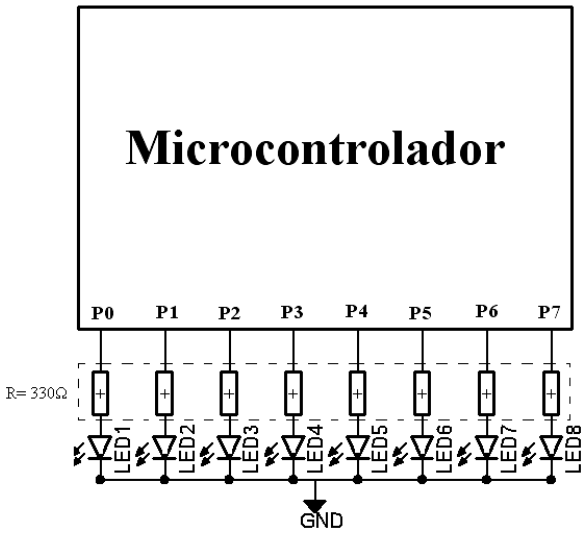


Fig.5.3. Alambrado

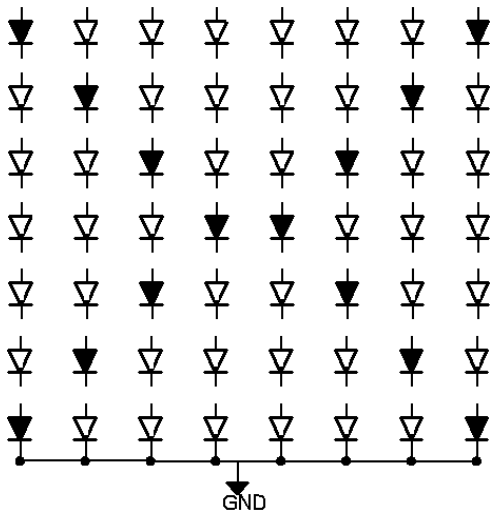


Fig.5.4. Secuencia

Los LEDS que aparecen en la figura 5.4 rellenos de color negro, representan a los LEDS encendidos y los de color blanco a los apagados.

Nota: para poder visualizar los LEDS encendidos y apagados es necesario tener en el programa una subrutina que provoque un retardo, esta puede ser el programa de la práctica anterior donde se tuvo que hacer una rutina de retardo para poder visualizar el conteo.

Sugerencias para el profesor.

1. Explicar los modos de direccionamiento que maneja el microcontrolador elegido para estas prácticas.
2. Explicar cómo funciona la pila (FIFO y LIFO) y dejar un programa que la utilice. (punto 1 del desarrollo dentro del laboratorio)
3. Dejar a los alumnos que realicen el punto 2 del desarrollo dentro del laboratorio y si no hubiera tiempo dejarlo de tarea.

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

PRÁCTICA 5

“INTERRUPCIONES”

Objetivo General.

Que el alumno aprenda el manejo de interrupciones.

Objetivo Particular.

1. Que el alumno sepa que registros están asociados para realizar un o varias interrupciones.
2. Que el alumno conozca los tipos de interrupciones.
3. Que el alumno sepa los pasos que se producen al presentarse una interrupción.

Guía de estudio para el alumno

1. Leé lo correspondiente a las interrupciones (en el manual).
2. Buscar los registros y banderas que hay que configurar para habilitar la interrupción externa o temporizador (externo), según sea el caso.
3. Buscar en qué dirección se encuentra el vector de interrupciones.
4. Buscar las instrucciones asociadas a las interrupciones en el set de instrucciones.
5. Buscar el pin por el que entra la señal de interrupción, ya sea externa o del temporizador (externo).

Introducción.

Las interrupciones son señales internas o externas, que provocan el desvío de la ejecución del programa principal, para atender una serie de instrucciones llamada Rutina de Servicio de Interrupción (RSI) y, al terminar esta rutina, el programa principal continúa donde fue interrumpido.

Como en la RSI pueden ser modificados algunos registros, concretamente aquellos que la RSI va a emplear y alterar su contenido, principalmente el acumulador; es necesario guardarlos al inicio de la rutina de interrupción, para esto se puede usar la pila, localidades de memoria o registros de propósito general. Hay que recordar, restaurar los registros antes de regresar al programa principal, para que este continúe con los valores que tenía antes de ser interrumpido.

Las direcciones de las RSI se encuentran en una tabla en memoria, denominada tabla de vectores de interrupción; en ésta, generalmente cada interrupción tiene una localidad específica a la cual brincará si sucede el evento de esa interrupción. También dentro de esta tabla se encuentra la prioridad de cada una de las interrupciones, si se producen dos interrupciones, se realizará primero la rutina de la de mayor prioridad.

Algunas interrupciones son provocadas por temporizadores, interrupción externa que es a través de un pin, Reset, recepción y transmisión serie, la de software. Existen más y cambian (no mucho) dependiendo del microcontrolador.

Tipos de interrupciones.

Existen 3 tipos de interrupciones, las enmascarables, las no enmascarables y las de software.

1. **Las interrupciones enmascarables**, como su nombre indica, se pueden enmascarar, es decir, habilitar o deshabilitar a través de un bit.
2. **Las interrupciones no enmascarables** no se pueden deshabilitar. Siempre se deben de atender.
3. **Las interrupciones software** son las producidas por el propio programador en momentos conocidos. Como su nombre lo indica se llevan a cabo a través de una instrucción. Esta interrupción puede entrar en la clasificación de las no enmascarables.

También es posible clasificar las interrupciones en internas y externas.

1. **Las interrupciones internas,** son las producidas por circuitos o periféricos integrados dentro del propio microcontrolador.
2. **Las interrupciones externas,** son las producidas por circuitos o periféricos externos.

Pasos que se realizan al producirse una interrupción.

1. Se encuentra en ejecución el programa principal.*
2. Al ocurrir una interrupción, se continua la ejecución de la instrucción en curso hasta finalizarla.
3. Guarda el contenido del contador de programa en la stack (pila).*
4. Guarda el contenido de los registros, esto puede ser automático de lo contrario se tiene que realizar una rutina para hacerlo.
5. Habilitar o deshabilitar interrupciones del mismo nivel, para permitir o evitar interrupciones anidadas. Esto puede ser automático o manual. Se recomienda que no se aniden interrupciones y si fuera el caso de que se aniden, se tiene que evitar el desbordamiento de la stack (pila).
6. Identificar por software el dispositivo que interrumpe, en caso de que no haya identificación hardware.
7. Realiza la RSI.*
8. Recuperar el contenido de los registros, Esto puede ser automático o manual.
9. Regresa el contador de programa al valor que tenía antes del producirse la interrupción, o sea, lo recupera de la stack.*
10. Volver a habilitar interrupciones (automático o manual).

*Lo realiza el microcontrolador de manera automática.

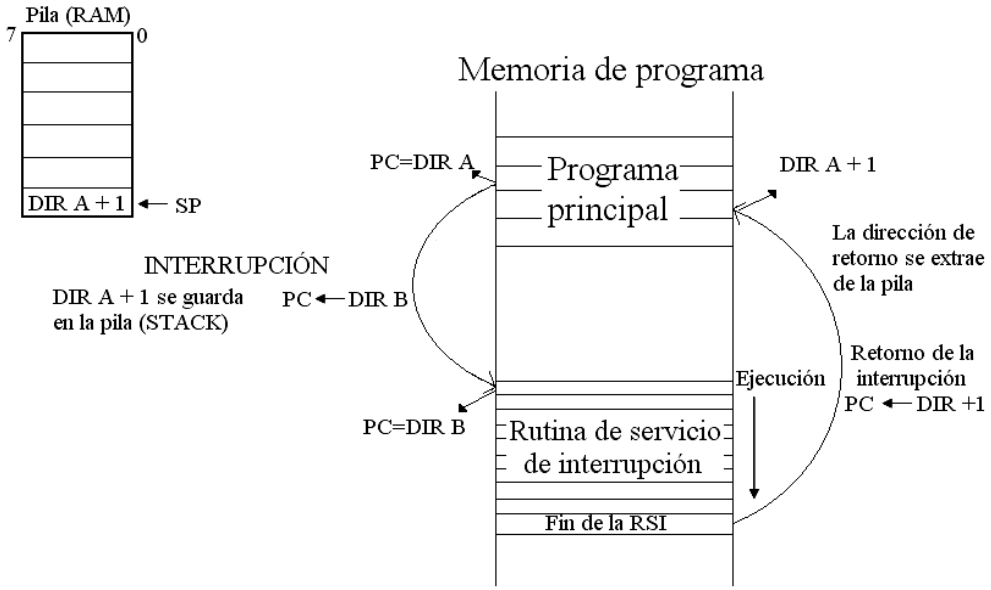


Fig.6.1. Diagrama esquemático de función de las interrupciones.

Para habilitar las interrupciones normalmente se tienen que configurar dos banderas, una que habilita todas las interrupciones enmascarables y otra que habilita cada interrupción en particular.

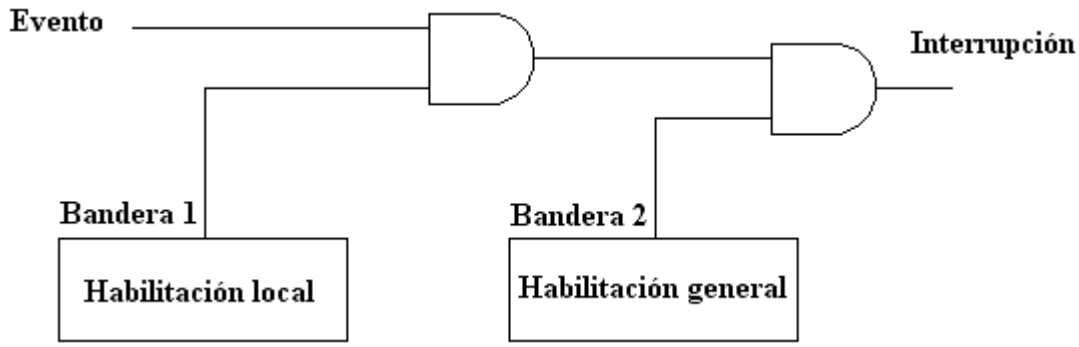


Figura Fig.6.2, muestra un esquema general de una interrupción y sus banderas asociadas.

Como se ve en la figura anterior, se tienen que habilitar las 2 banderas, si sólo se habilita una, cuando suceda el evento no generará una interrupción.

La principal función de las interrupciones es simplificar el control, ya que cuando están habilitadas se atiende a los periféricos sólo cuando lo requieren, y no hay que realizar rutinas para monitorearlos.

Dentro del laboratorio.

1. Realizar un programa cuya rutina principal sea mostrar la secuencia de la figura 5.4, del desarrollo dentro del laboratorio de la practica 4, y al ser interrumpido realice la RSI que muestre un conteo en binario de 0 a 255 y al terminar regrese a la rutina principal.

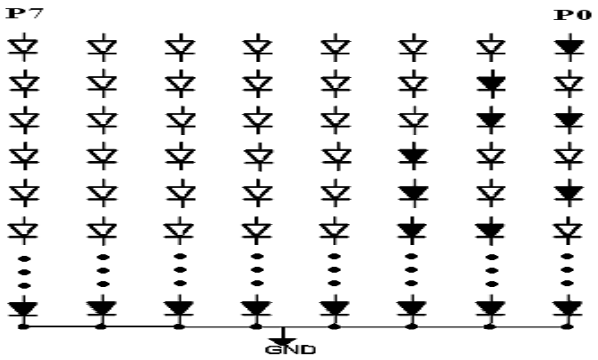


Fig.6.3. Conteo en binario

El hardware para la interrupción externa quedara de la siguiente manera:

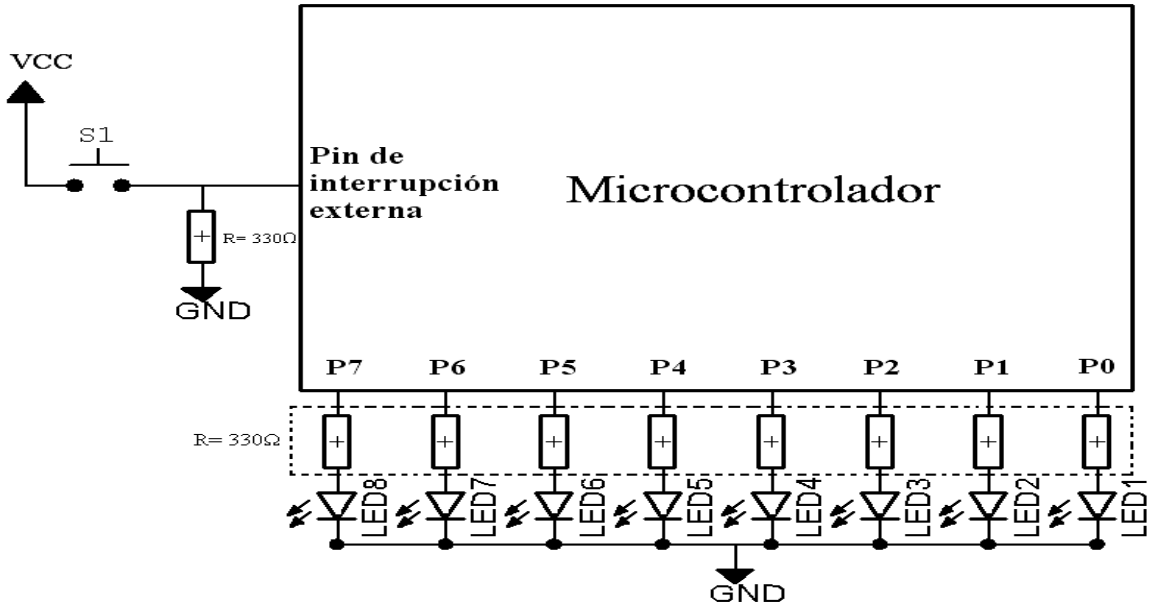


Fig.6.4. El hardware para la interrupción externa

Y para el temporizador (externo) quedara:

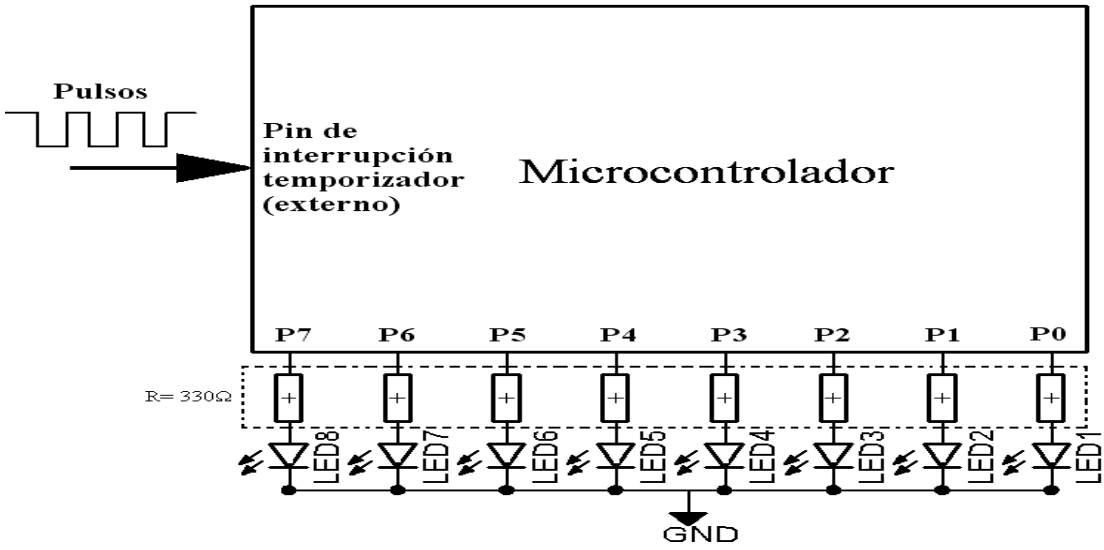


Fig6.5. El hardware para la interrupción por temporizador.

Sugerencias para el profesor.

1. Explicar que es una interrupción y para qué sirve (poner ejemplos).
2. Explicar el concepto de vector de interrupción.
3. Explicar cómo funcionan las interrupciones para el microcontrolador utilizado.
4. Cuestionar al alumno sobre los registros, banderas e instrucciones asociados a las interrupciones.
5. Dejar a los alumnos que desarrollen el programa del punto 1 (del desarrollo dentro del laboratorio), que consiste en utilizar la interrupción externa o la del temporizador (externo).

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

PRÁCTICA 6

"CONVERTIDOR A/D"

Objetivo General.

Que el alumno aprenda a utilizar el convertidor analógico digital que contiene el microcontrolador.

Objetivo Particular.

1. Que el alumno conozca los registros que están asociados al convertidor A/D.
2. Que el alumno realice un programa donde se adquiera una señal analógica y se represente el resultado de esa señal en forma digital en un puerto de E/S.
3. Que el alumno los criterios que se deben de tomar en cuenta para la digitalización de una señal analógica.
4. Que el alumno sepa cómo se realiza el cálculo de la señal analógica convertida en digital y compare el resultado.

Los microcontroladores cuentan, en algunas versiones, con canales de conversión A/D (Analógica-Digital). El convertidor recibe una señal de voltaje analógica (en un pin o par de pines de entrada) y la convierte a un valor digital (bits).

Guía de estudio para el alumno.

1. Leer lo correspondiente al convertidor analógico-digital (en el manual).
2. Buscar los registros y banderas que hay que configurar para habilitar la conversión A/D.

3. Buscar en qué dirección se encuentran los registros de resultado de las conversiones.
4. Buscar el puerto o puertos por los que puede entrar la señal analógica.

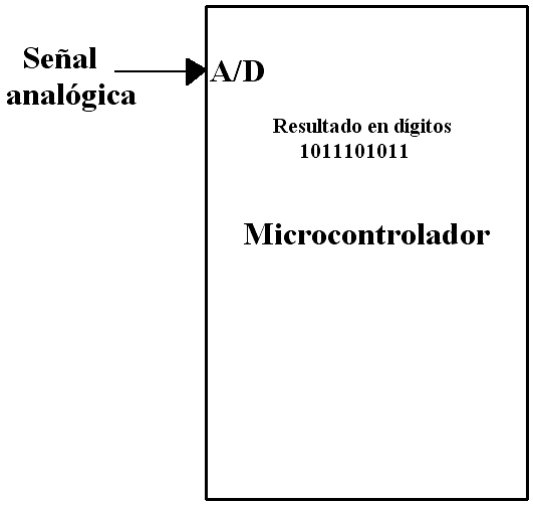


Fig.7.1. Entrada analógica de un microcontrolador.

El valor digital se guarda en registros mapeados en memoria. El rango del voltaje analógico a convertir es definido por los voltajes de referencia, los cuales pueden ser tomados internamente del microcontrolador o del exterior.

El convertidor A/D es de utilidad cuando se requiere leer un valor de un periférico (sensor de temperatura, luz, Convertidor frec-v, etc.).

Existen dos modos de conversión llamados Single (sencillo), con el cual se convierte la señal analógica que se encuentra en un solo pin y el modo Differential (diferencial), recibe de entrada dos señales en un par de pines.

La unidad de conversión A/D tiene registros asociados a esta, algunos de estos pueden ser, los de interrupciones, los de configuración (en los cuales se ven las

velocidades a las que puede trabajar, el inicio y el fin de la conversión, el canal de selección) y los de resultado en los cuales se guarda el resultado de la conversión.

Dentro del laboratorio.

1. Realizar un programa que convierta una señal analógica (variable de 0 a 5v) a digital y el resultado se vea en LEDS, cuando la fuente analógica se encuentre en 0v todos los LEDS deben estar apagados y cuando este en 5v todos deben estar prendidos, para saber si la conversión es correcta, suponiendo que tenemos una resolución de 8 bits, se usa la siguiente regla:

$$\mathrm{ADC} = \frac{\mathrm{V_{in}} * 255}{\mathrm{V_{ref}}}$$

$\mathrm{V_{in}} =$ voltaje censado en la entrada analógica.

$255 =$ la máxima resolución 8 bit´s $=$ FF

$\mathrm{V_{ref}} =$ voltaje de referencia, para este caso es de 5v

$\mathrm{ADC} =$ resultado de la conversión

El diagrama queda de la siguiente manera:

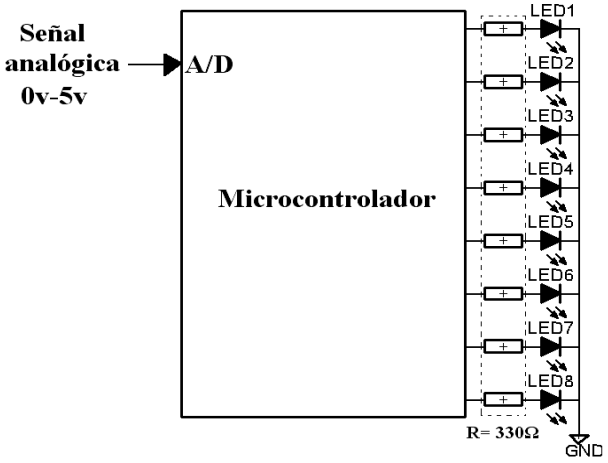


Fig.7.2. Hardware

Sugerencias para el profesor.

1. Explicar cómo funciona la conversión A/D y para qué sirve (poner ejemplos).
2. Explicar cómo funciona la conversión A/D para el microcontrolador utilizado.
3. Dejar a los alumnos que desarrollen el programa del punto 1 (del desarrollo dentro del laboratorio).

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

PRÁCTICA 7

“TRANSMISIÓN SERIE ASÍNCRONA”

Objetivo General.

Que el alumno aprenda a utilizar la transmisión serie asíncrona del microcontrolador.

Objetivo Particular.

1. Que el alumno conozca los registros asociados a la unidad de transmisión serial.
2. Que el alumno tenga la capacidad de configurar las características de los tramas a enviar o recibir.
3. Que el alumno realice un programa que transmita una señal previamente convertida a digital de un micro controlador a otro que reciba la señal y que la mostrara por un puerto de E/S.

Guía de estudio para el alumno.

1. Leé lo correspondiente a la transmisión serie.
2. Buscar los registros y banderas que hay que configurar para habilitar la transmisión serie, control, estado, transmisión, recepción, baud (velocidad) y los de interrupciones.
3. Buscar los pines asociados a la transmisión y recepción serie.

Introducción.

Los microcontroladores cuentan con unidades especiales de transmisión serie que permite comunicarse con otros dispositivos. Algunas de estas son:

1. Adaptador de comunicación serie asíncrona **(UART, Universal Asynchronous Receiver Transmitter).**
2. adaptador de comunicación serie síncrona y asíncrona. También conocida como SCI. (**USART**, **Universal Synchromous Asynchronous Receiver Transmitter)**
3. Es un moderno bus serie para las PC. **(USB, Universal Serial Bus**).
4. **Bus I^2C**, es un interfaz serie de dos hilos desarrollado por Phillips.
5. **CAN** (Controller Area Network), para permitir la adaptación con redes de conexionado multiplexado desarrollado conjuntamente por Bosch e Intel para el cableado de dispositivos en automóviles.

En esta práctica nos enfocaremos en la UART y USART (SCI y SPI) que permiten realizar comunicaciones asíncronas y síncronas (sólo la USART) a distintas velocidades.

Los paquetes usualmente contienen un bit de comienzo (Bit de start); 8 ó 9 bits de datos, un bit de parada (Bit de stop) al final y bit de paridad (par/impar). Todos estos configurables.

Para configurar los parámetros de la comunicación serie, tenemos básicamente, los registro de control, estado, transmisión, recepción, baud (tasa de transmisión dada en bits por segundo) y los de interrupciones. Dentro de los cuales se tienen que realizar las siguientes acciones.

1. Habilitar la interrupción asociada a la transmisión o recepción serie. También pueden ser ambas.
2. Determinar la tasa de transmisión (baud), para esto se tiene una formula. (que viene en el manual del dispositivo a utilizar).
3. Determinar si existirá paridad, en caso de haber, también se tiene que determinar si será par o impar.
4. Se tiene que configurar el bit de inicio (start), en caso que se requiera.
5. Seleccionar si se transmitirán o recibirán datos. También pueden ser ambos.
6. Seleccionar si la transmisión o recepción será síncrona o asíncrona.
7. Seleccionar el tamaño del paquete de datos, 8 ó 9 bits.
8. Seleccionar el modo maestro o esclavo.

Los pasos anteriores no tienen que ser en el orden que se muestran, pero se tienen que definir para que la comunicación serie sea correcta.

Dentro del laboratorio.

1. Realizar un programa en el que entre una señal analógica al microcontrolador 1, la convierta en digital y la transmita de manera serial a microcontrolador 2, el cual la mostrará de manera digital en los LEDS.

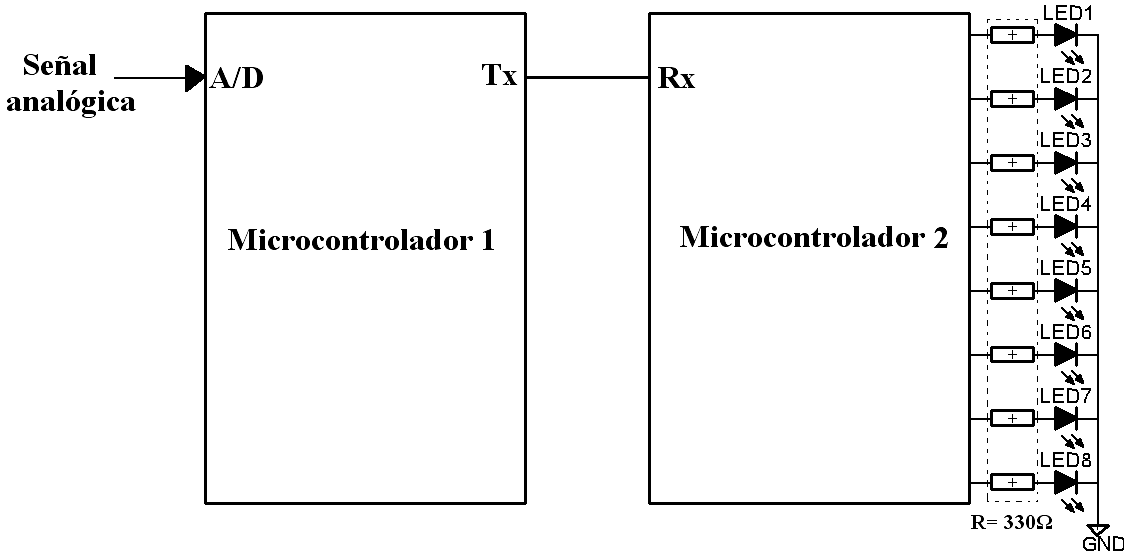


Fig.8.1 Hardware de transmisión serial, de una señal analógica adquirida por medio de la entrada A/D del microcontrolador 1, enviada por el puerto serial al microcontrolador 2 y mostrada en 8 bits por un puerto de salida del microcontrolador 2.

Sugerencias para el profesor.

1. Explicar en qué consiste la transmisión serie.
2. Explicar conceptos importantes para la transmisión serie como son, bit de paridad, baud, bit de inicio, modo maestro o esclavo, etc.
3. Explicar cómo funcionan la transmisión serie para el microcontrolador utilizado.
4. Cuestionar al alumno sobre los registros, banderas e instrucciones asociados a la transmisión serie.
5. Formar equipos de 2 ó 3 alumnos para que desarrollen el programa del punto 1 (del desarrollo dentro del laboratorio), en el que se utiliza el convertidor analógico/digital (que se vio en la práctica anterior) y la transmisión serie.

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).

Práctica 8

"ADQUISICIÓN DE DATOS"

Objetivo General.

Que el alumno visualice en una computadora los datos adquiridos del micro controlador.

Objetivo Particular.

1. Que el alumno realice un interfaz entre el micro controlador y la computadora.
2. Que el alumno tenga la capacidad de leer un puerto de la computadora y graficar los datos con una Interfaz Gráfica de Usuario, **(GUI, Graphical User Interface)**
3. Que el alumno reafirme cada una de las prácticas realizadas.
4. Que el alumno ponga en práctica sus conocimientos de electrónica y los combine para el diseño de un sistema de adquisición de datos.

Guía de estudio para el alumno.

1. Leer lo correspondiente a la transmisión serie.
2. Leer lo correspondiente a conversión A/D**.**
3. Buscar los registros y banderas que hay que configurar para habilitar la transmisión serie, la conversión A/D, ADMUX, velocidad de conversión, resolución, control, estado, transmisión, recepción, baud (velocidad) y los de interrupciones.
4. Buscar los pines asociados a la transmisión, recepción serie y conversión A/D.

Introducción.

La finalidad de la adquisición de datos es medir un fenómeno físico, como la temperatura, presión, sonido, etc. Mientras cada sistema de adquisición de datos se define por sus requerimientos de aplicación, cada sistema comparte un fin en común. El cual es adquirir, analizar y presentar información. Los sistemas de adquisición de datos incorporan señales, sensores, actuadores, acondicionamiento de señales, dispositivos de adquisición de datos y software de aplicación.

La adquisición de datos consiste en tomar una muestra de la variable física que en su caso estemos censando, esta muestra es transformada a niveles de voltaje o corriente, la cual puede ser interpretada por una computadora u otro dispositivo electrónico encargado del almacenamiento, procesamiento e interpretación de la señal.

Sensores o transductores: Los sensores tienen un rol vital en la adquisición de datos, ellos tienen la función de convertir la variable física que se desea registrar en una magnitud eléctrica (voltaje, corriente, resistencia, capacidad, inductancia, etc.).

Tipos de sensores: Existen varios tipos de sensores, de voltaje, de corriente, resistivos, capacitivos, inductivos, etc.

Dato: Representación simbólica (numérica, alfabética...), atributo o característica de un valor.

Adquisición: Toma de un conjunto de variables físicas, conversión en voltaje y digitalización de manera que se puedan procesar en un ordenador.

Sistema: Conjunto organizado de dispositivos que interactúan entre sí ofreciendo prestaciones más completas y de más alto nivel.

Bit de resolución: Número de bits que el convertidor analógico digital (ADC) utiliza para representar una señal.

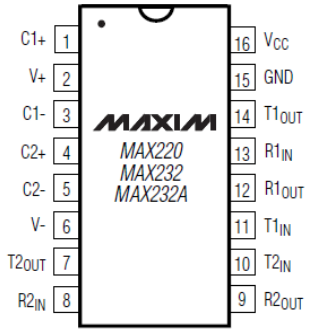
Rango: Valores máximo y mínimo entre los que el sensor, instrumento o dispositivo funcionan bajo unas especificaciones.

Teorema de muestreo de Nyquist-Shannon: Las muestras discretas de una señal son valores exactos que aún no han sufrido redondeo o truncamiento alguno sobre una precisión determinada, esto es, aún no han sido cuantificadas. El teorema demuestra que la reconstrucción exacta de una señal periódica continua en banda base a partir de sus muestras, es matemáticamente posible si la señal está limitada en banda y la tasa de muestreo es superior al doble de su ancho de banda.

Dentro del laboratorio.

1. Realizar un programa en el que entre una señal analógica al microcontrolador 1, la convierta en digital y la transmita de manera serial a puerto serial de tu PC, el cual por medio del circuito MAX232 se acoplará para poder ser mostrada en el ambiente de LabVIEW. (el circuito Max 232 y el programa con el cual mostraremos la señal serán detallados en el mismo proceso de la práctica).
2. Verificar las conexiones del CI MAX232 para acoplar la señal que sale de microcontrolador al puerto serial de la PC.

TOP VIEW



DIP/SO

CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

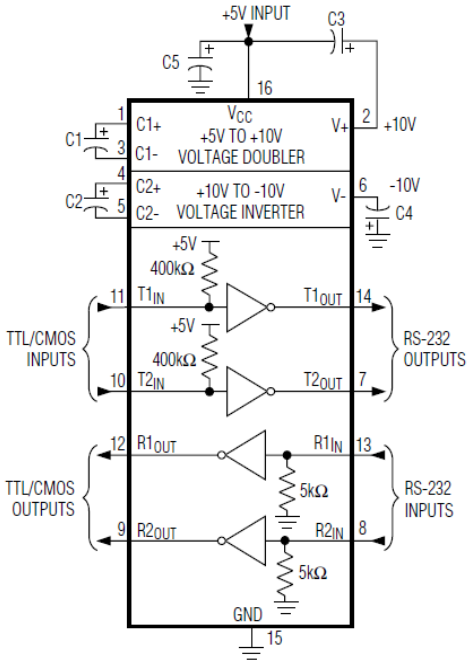


Fig.9.1 Salidas y configuración del Max 232

3. Como se observa en la figura anterior, tenemos la forma de conectar el C,I el valor de los capacitores utilizados se encuentra en la tabla inferior izquierda los pines de entrada y salida, usamos este circuito ya que la entrada serial de la PC es RS232. Otra situación muy importante es que debemos utilizar un conector DB9 hembra y hacer una conexión de la siguiente forma.

Tabla.9.1. Nombre y descripción de los pines del CI Max 232.

Pin	Señal	Nombre	Señal
3	TxD	Transmit	Pc Out
2	RxD	Receive	PC In
7	RTS	Req to Send	RTS Flow
8	CTS	Clear to send	CTS Flow
6	DSR	Data set Ready	Ready
5	SG	Signal Ground	Common
1	CD	Carrier Detect	Modem Conected
4	DTR	Data Terminal Ready	Ready
9	RI	Ring Indicator	Line Ringing

4. Como podemos observar tendremos que soldar un cable plano de 3 hilos al conector DB9 hembra en el pin 3 para transmisión de datos, en el pin 2 para recibir los datos (en este caso sólo mandaremos datos a la PC, así es que podría no ser necesario), y por último el pin 5, que será nuestro GND.
5. El diagrama a bloques del sistema de adquisición de datos quedaría de la siguiente forma.

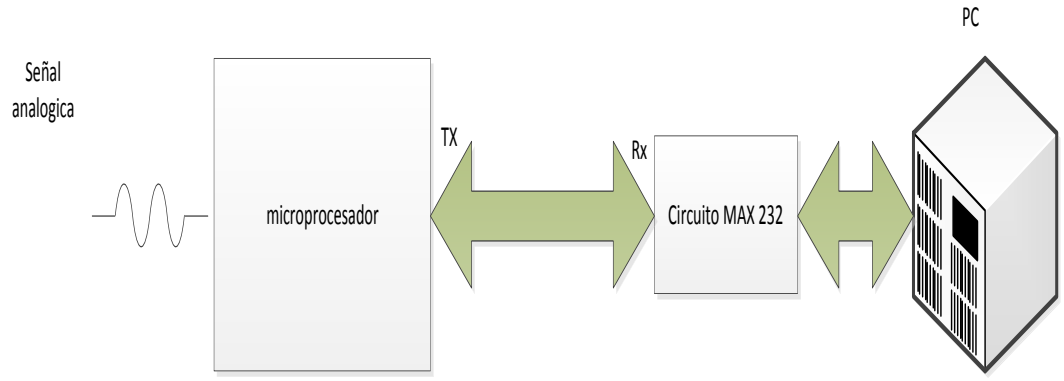


Fig.9.2. Diagrama a Bloques del sistema.

El circuito quedaría de la siguiente forma.

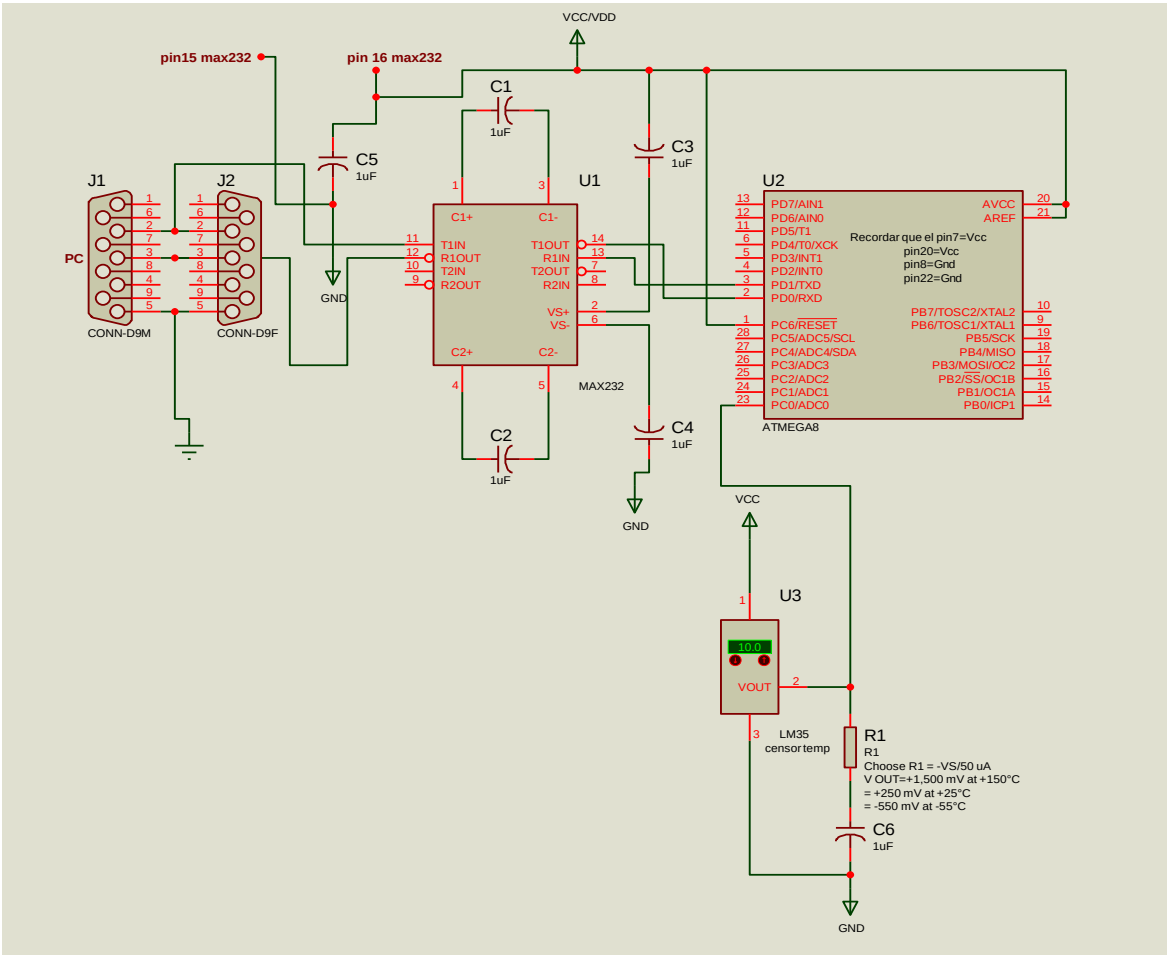


Fig.9.3. Diagrama eléctrico del sistema usando un ATmega8.

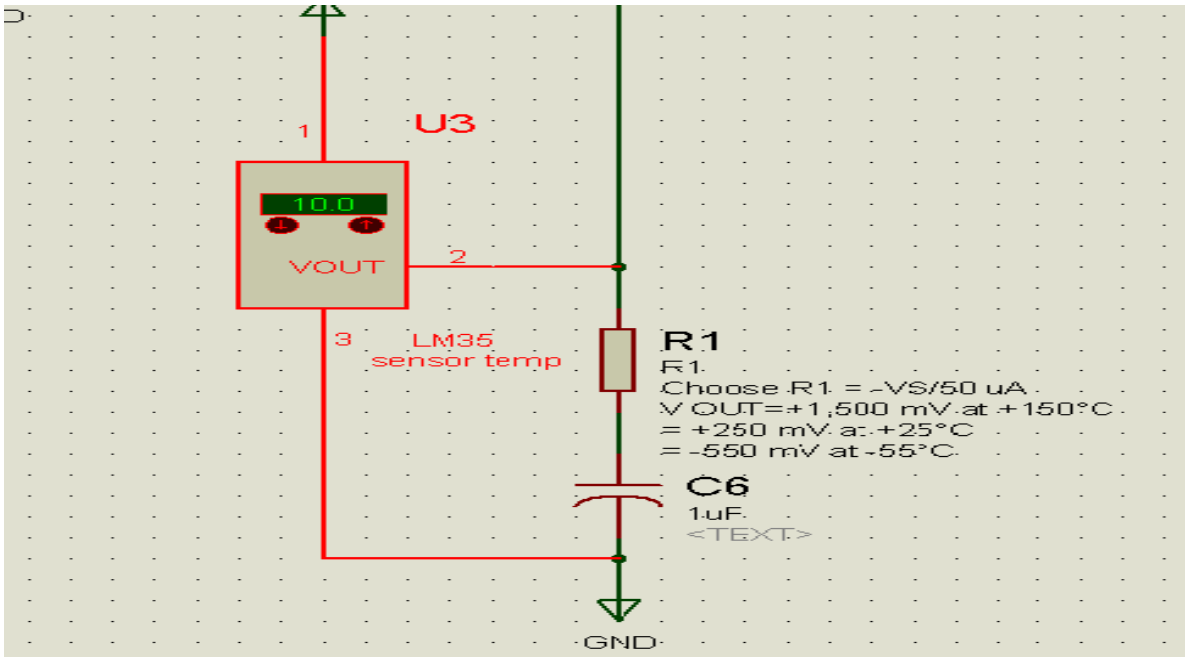


Fig.9.4. Diagrama eléctrico del sistema usando un ATmega8,conexión y cálculo del censor de temperatura LM35.

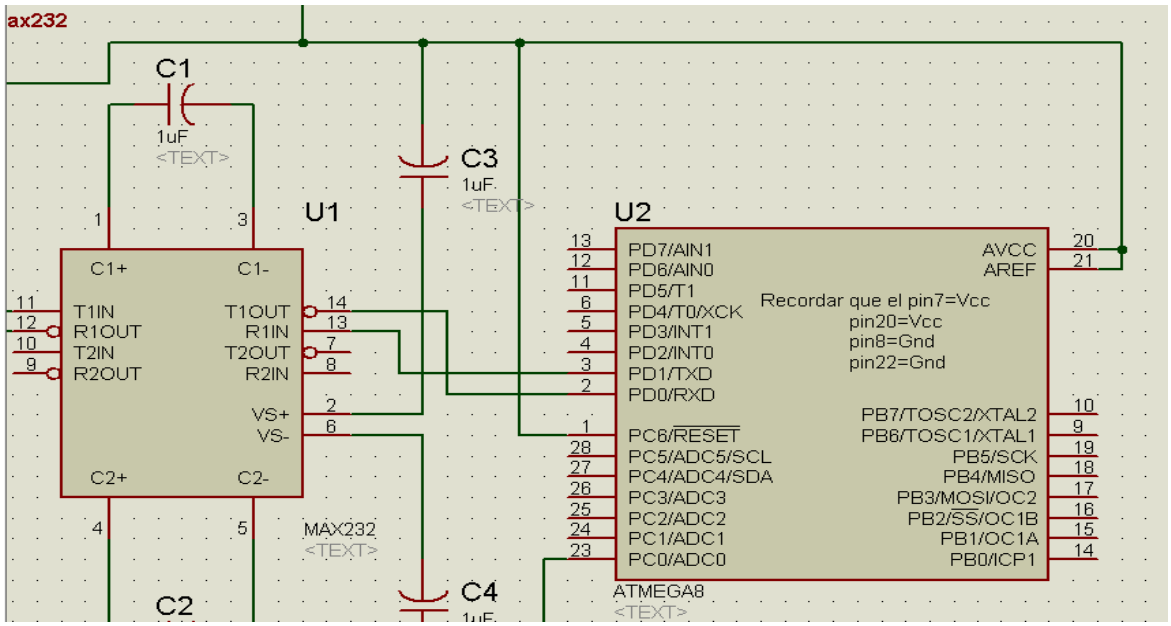


Fig.9.5. Diagrama eléctrico del sistema usando un ATmega8, principales conexiones del ATmega8.

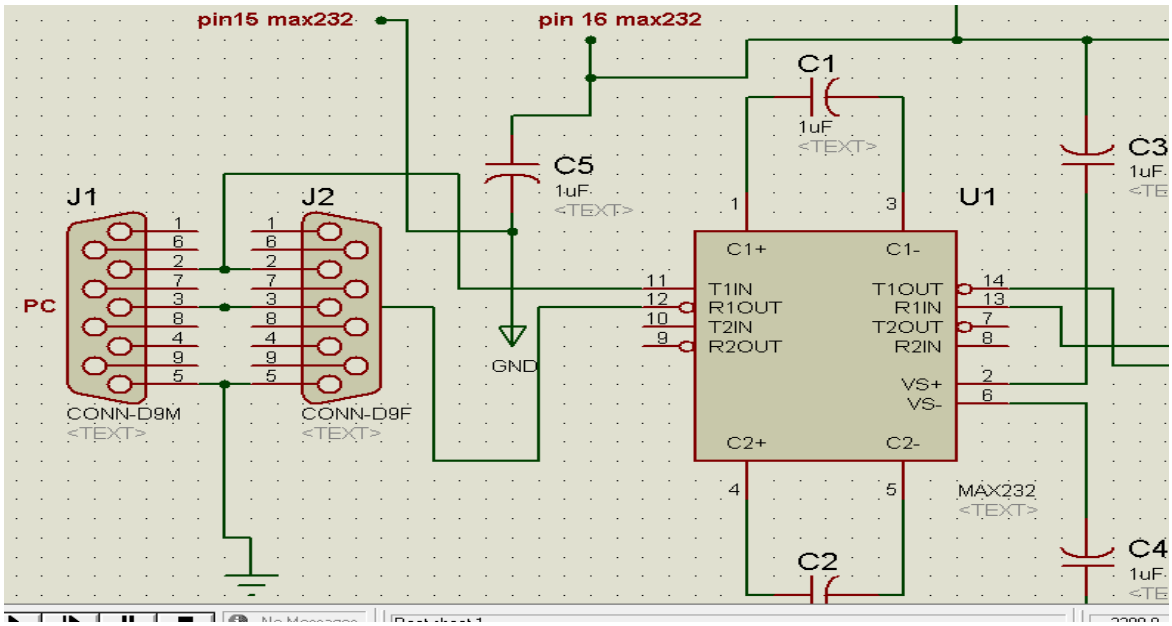


Fig.9.6. Diagrama eléctrico del sistema usando un ATmega8, conexión del puerto serial de la PC con el CI MAX 232.

6. Posteriormente necesitamos un programa que nos ayude a procesar la señal, visualizarla e interpretar dicha señal para, ya sea tomar una decisión, tener un censo para fines estadísticos o mejor aún utilizar esa información para accionar un actuador que nos permita ya sea el caso, aumentar o disminuir la temperatura (esto último se logra programando el puerto paralelo de la computadora para visualizar un bit del puerto paralelo, acondicionarlo por medio del HD74LS245P o PC847X y accionar un relevador a 5Vcd, dependiendo de lo que se quiera accionar, se debe tomar en cuenta la potencia que consume el actuador para elegir bien el relevador).
7. Ahora necesitamos abrir una hoja en blanco de LabVIEW y abrir su diagrama de bloques, posteriormente abrimos la paleta de “Instrument” y escogemos VISA, este es un protocolo de comunicación serial, debemos configurarlo. Si ya se tiene una experiencia con este programa lo más recomendable es bajar el programa ya capturado de la página de National Instruments y adecuarlo

para la visualización de la señal en una gráfica en el panel frontal.

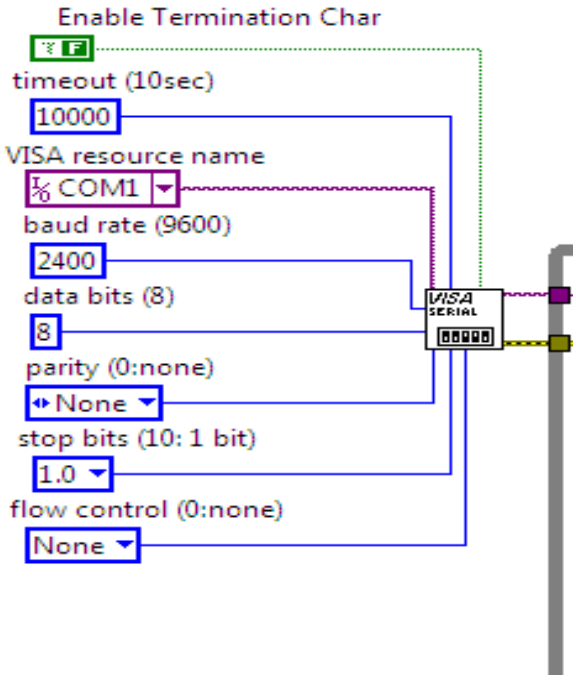


Fig.9.7. Diagrama a Bloques de LabVIEW, configuración del protocolo VISA.

8. Después en una estructura **While Loop** se encierra los siguientes bloques para poder visualizar la señal en el panel frontal.

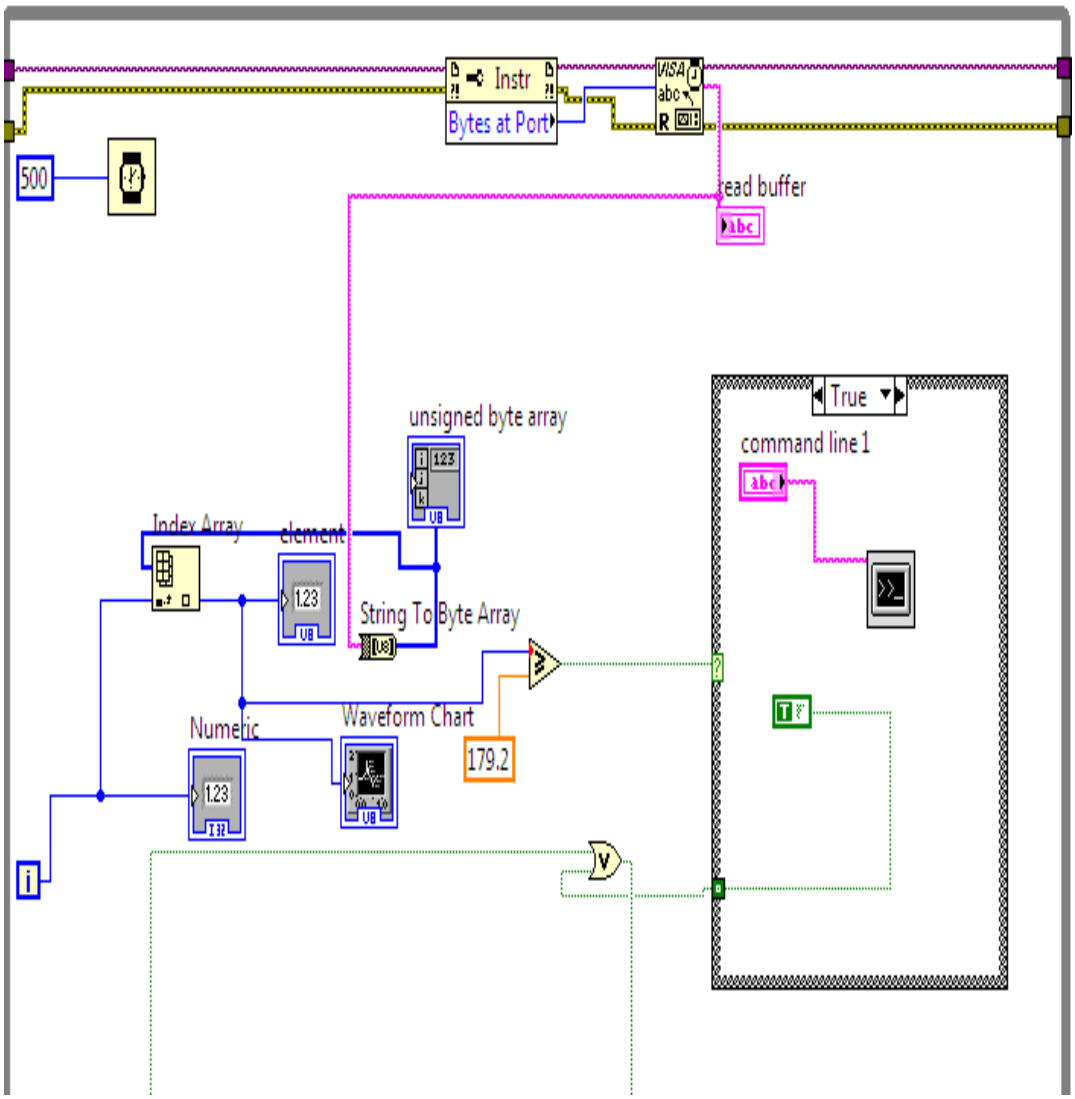


Fig.9.8. Estructura While Loop en el diagrama a Bloques de LabVIEW.

9. La estructura que se encuentra dentro del **While Loop** es un **Case,** esta nos puede ayudar tomar una decisión si es que la información recibida nos ayudará para accionar algún actuador, iniciar un programa etc.

10. Lo que vemos en la figura 9.9. sigue formando parte de la configuración del protocola serial VISA. Esta parte su única función es hacer notar al programa que existe un error de comunicación del puerto.

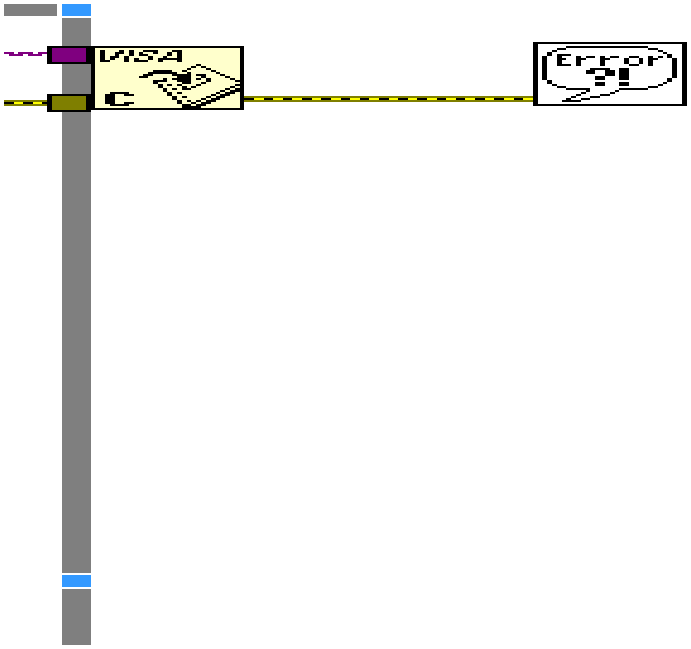


Fig.9.9. Salida de la estructura **While Loop.**

11. El programa visto desde el panel frontal quedaría de la siguiente forma.

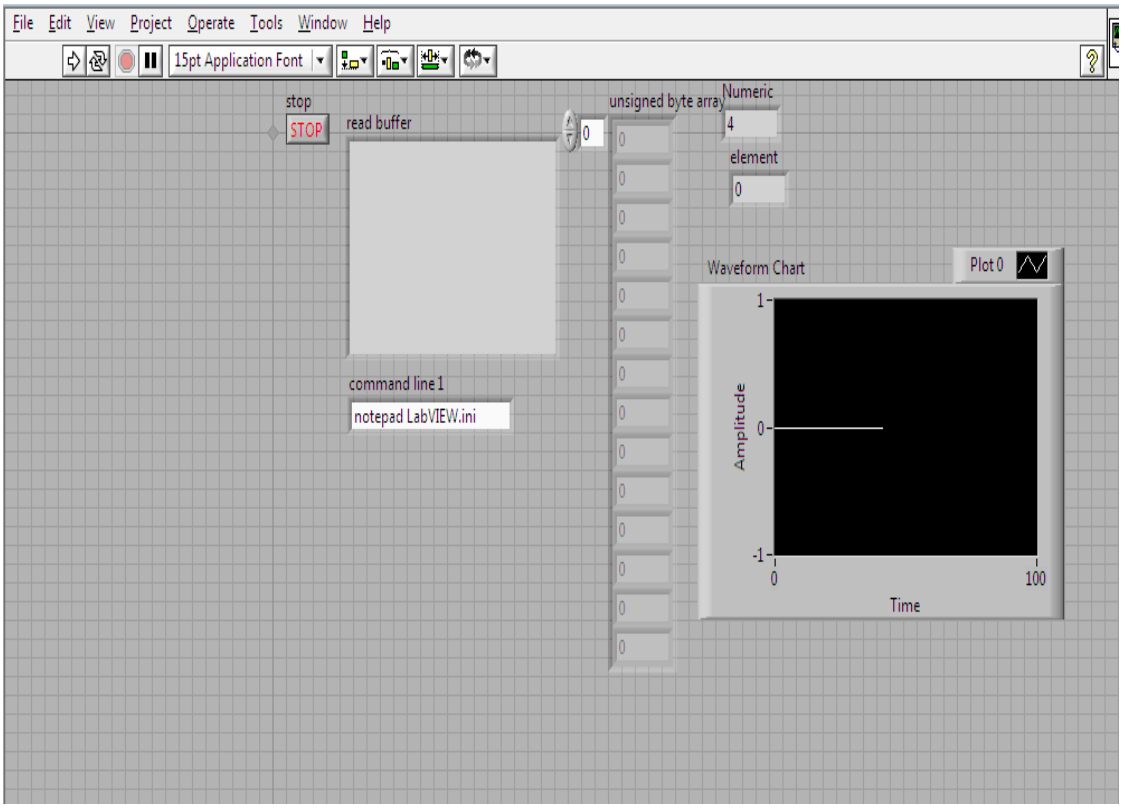


Fig.9.10. Panel Frontal del Programa en LabVIEW de una comunicación serial.

Sugerencia para el profesor

1. Tener cuidado con el manejo de las corrientes del puerto serial ya que podemos dañarlo PC.
2. Que el alumno investigue las características del puerto seria de una PC.
3. Si no está familiarizado con un software de diseño de circuitos se le oriente que software le ayudará con esa tarea.
4. Auxiliar al alumno con el programa propuesto para el procesamiento de la señal o en otro caso puede implementarse cualquier otro software de su agrado. Este es sólo un programa y una pequeña guía, para más información guíese con la ayuda del software.
5. Que el alumno investigue los diferentes protocolos de transmisión serial y en especial el protocolo VISA.

Bibliografía.

1. La que el instructor indique, los manuales del fabricante y los manuales de usuario del dispositivo que se manejara en estas prácticas (recordando que el microcontrolador a manejar depende del profesor).
2. LabVIEW for Everyone Graphical Programming Made Easy and Fun. Jeffry Travis, Jim KringThrid Edition.

Conclusiones.

Como podemos darnos cuenta, no sólo es interesante el estudio de estos componentes de muy alta escala de integración, sino que es casi fundamental saber su funcionamiento y saber cómo implementar un dispositivo de estas características, pero sin salir de nuestros objetivos que son estrictamente académicos, ya que como lo vemos hoy en día, la mayor parte de las cosas que utilizamos van desde lo más cotidiano como son, televisores, reproductores de música, teléfonos fijos, celulares, computadoras, hornos de microondas; hasta armas sofisticadas, están diseñadas con un microcontrolador, que nos ayudan hacer las cosas de forma rápida, sencilla, y lo importante, lo que hace que existan, más económico.

Estos dispositivos también son importantes el diseño de sistemas digitales, como en sistemas de control, sistemas de iluminación, sistemas de comunicación, etc. La mayor parte, hoy en día, de los dispositivo electrónico y sistema digital cuentan con la colaboración de estos dispositivos, en la telefonía, en los instrumentos de medicina, de hecho en cualquier ámbito podemos encontrarlos sin saber, con un microcontrolador.

De tal suerte que espero que sea de gran ayuda para todo aquel que esté interesado en aprender, también que sea un apoyo didáctico para el instructor de este laboratorio y una herramienta más de las muchas que existen hoy en día.

Bibliografía.

AVR. An Introductory Course

John Morton. 2002.

Newnes

What's a Microcontroller.

Andy Lidsay.

Parallax. ver 2.0

Computer Organization and Architecture.

Stallings, William.

Prentice-Hall, Madrid 2007.

AVR 8-bit

Instruction Set

Atmel Corp. (2002).

|

COP8™ Microcontroller

COP8 Flash Family User´s Manual

National Semiconductor Corp. (2000).

MICROCONTROLADOR COP8TM -Manual de Teoría y Práctica Básica-

Jesús Flores V.

Nacional Semiconductor Corp. (2001)

PICmicro™ Mid-Range MCU Family

Reference Manual

Microchip Technology Inc. (1997)

Soluciones a Interfaz por Medio del Puerto Paralelo de la PC

Susana Palacios, Andrés García

ITESM Campus Estado de México (1999)

LabVIEW for Everyone Graphical Programming Made Easy and Fun.

Jeffry Travis, Jim Kring

Thrid Edition

Índice alfabético