UNIVERSIDAD AUTONOMA DE NUEVO LEON
FACULTAD DE INGENIERIA MECANICA Y ELECTRICA
DIVISION DE ESTUDIOS DE POST-GRADO

DESARROLLO DE UN CURSO DE MICROCONTROLADORES QUE INCLUYE: EL PLAN DE ESTUDIOS, APUNTES Y LISTA DE PRACTICAS DE LABORATORIO

TESIS

EN OPCION AL GRADO DE MAESTRO EN CIENCIAS
DE LA INGENIERIA ELECTRICA
CON ESPECIALIDAD EN ELECTRONICA

QUE PRESENTA

RENE BRIONES LARA

SAN NICOLAS DE LOS GARZA, N. L. SEPTIEMBRE DE 1994

DESARROLLO DE UN CURSO DE MICROCONTROLADORES QUE INCLUYE: EL PLAN DE ESTUDIOS, APUNTES Y LISTA DE PRACTICAS DE LABORATORIO

TM
Z5853
.M2
FIME
1994
B7

1020070669

UNIVERSIDAD AUTONOMA DE NUEVO LEON

FACULTAD DE INGENIERIA MECANICA Y ELECTRICA
DIVISION DE ESTUDIOS DE POST-GRADO

DESARROLLO DE UN CURSO DE MICROCONTROLADORES QUE INCLUYE. EL PLAN DE ESTUDIOS, APUNTES Y LISTA DE PRACTICAS DE LABORATORIO

TESIS

EN OPCION AL GRADO DE MAESTRO EN CIENCIAS
DE LA INGENIERIA ELECTRICA
CON ESPECIALIDAD EN ELECTRONICA

QUE PRESENTA

RENE BRIONES LARA

S/N NICOLAS DE LOS GARZA N. L. SEPTIEMBRE DE 1994

TM
Z5853
.M2
FIME
1994
E7

FONDO TESIS

166785

UNIVERSIDAD AUTONOMA DE NUEVO LEON
FACULTAD DE INGENIERIA MECANICA Y ELECTRICA
DIVISION DE ESTUDIOS DE POST-GRADO

Los miembros del comité de tesis recomendamos que la presente tesis realizada por el Ing. René Briones Lara sea aceptada como opción para obtener el Grado de Maestro en Ciencias de la Ingeniería Eléctrica con Especialidad en Electrónica.

El Comité de Tesis

M.C. Ronald López Gómez.
Asesor

M.C.Juan Diego Garza Gonzalez
Coasesor

M.C.Luis Manuel Camacho Velázquez
Coasesor

División de Estudios de Postgrado
M.C.Marco Antonio Méndez Cavazos

San Nicolás de los Garza, N.L. Septiembre de 1994

Agradecimiento

La tarea de escribir este curso resultó algunas veces agotadora, pero siempre fue instructiva y gratificante. Durante el desarrollo de esta actividad hubo que apartarse de amigos; pero el esfuerzo es compensado por los logros obtenidos.

Hablando de esfuerzos, quiero hacer patente mi profundo agradecimiento al **Ing. Ronald López** por su apoyo, sugerencias y tiempo que me brindó durante el desarrollo de esta tesis.

INDICE

I.	Prólogo	1
II.	Síntesis	2
III.	Introducción	3
IV.	Metodología de la Enseñanza	4
V.	Definición del Curso	8
VI.	Conclusiones	9
VII.	Referencias y Bibliografía	10
VIII.	Anexos	11

VIII.1 Programa de Estudios
VIII.2 Lista de Prácticas de Laboratorio
VIII.3 Apuntes
VIII.4 Otros Aspectos

I. PROLOGO

Hoy más que nunca en forma acelerada y sistemática, la educación asume un papel protagónico ante la globalización de la economía de mercado. El proceso de cambio que afecta al mundo, y que alcanza a todas las esferas de la actividad humana, promueve la incentividad del progreso técnico como expresión de una verdadera revolución científico-tecnológica, para adaptarlo a las tendencias que exige el desarrollo.

En el camino hacia la próxima centuria hay una palabra clave: **Globalización.** Si adherimos al principio de que el mundo se globaliza y que no vamos a poder ser una isla, entonces, ya no se trata de determinar como formar profesionistas en un país determinado, sino como formarlos para el mundo.

Apartir de ahora no vamos a poder ser tan originales: **nuestros profesionistas se tienen que parecer cada vez más a los de otros sitios.** Además, así como hay un proceso de movimiento de los productos, también hay un proceso de movimientos de los factores. Por eso, nuestros ingenieros deberán estar preparados para ser transferibles, es decir, que si no trabajan aquí, lo harán en alguna parte de la economía global. Y aunque no fuera así, igual debemos formarlos de ese modo. Porque para poder competir, nuestros profesionistas deberán de estar capacitados para construir productos con los estándares que se están usando en todo el mundo.

Con la finalidad de actualizar los conocimientos, y tomando como base las razones anteriormente expuestas, fundamento la justificación de elaborar mi tesis "Desarrollo de un Curso de Microcontroladores, que incluye: El plan de Estudios, Apuntes y un Listado de Prácticas de Laboratorio", el cuál espero sea utilizado en el Sistema Nacional de Institutos Tecnológicos .

II. SINTESIS

La presente tesis presenta un Curso de Microcontroladores desde un punto de vista totalizador, pues toma en cuenta tanto la preparación teórica como la práctica.

Aplicando una nueva **Metodología de la Enseñanza** que incorpora nuevos modelos en educación, es elaborado **El Programa de Estudios** para orientar las actividades y esfuerzos, para lograr los objetivos generales del curso, es decir:

- Aprender a programar microcontroladores,
- Aprender la forma de conectarlos adecuadamente a dispositivos de entrada o de salida.
- Realizar diseños de equipo electrónico usando como elemento básico un microcontrolador.

Los Apuntes de Microcontroladores toman como base principalmente el Microcontrolador MC68HC11A8. Los apuntes son desarrollados como fundamentación teórica, y en él se presentan:

- Los conceptos generales de microcontroladores estableciendo las diferencias con los microprocesadores.
- La arquitectura del hardware de un microcontrolador MC68HC11A8.
- El Conjunto de Instrucciones y sus Modos de Direccionamiento.
- Las técnicas y herramientas utilizadas como Sistema de Desarrollo de Hardware.
- La utilización de los Puertos para comunicación Paralelo en modo directo y con protocolo de comunicación.
- La descripción y uso del convertidor Análogo-Digital.

El Listado de Prácticas de Laboratorio sirve de guía para la capacitación en el uso de microcontroladores . El desarrollo de las mismas proporciona la experiencia en la manipulación del conjunto de instrucciones y del soporte de hardware, además, mediante el uso de la computadora, se adquiere la habilidad en el manejo del Programa Ensamblador y del Programa Monitor, permitiendo al estudiante adquirir seguridad en la funcionalidad de los microcontroladores para poder emprender proyectos de mayor magnitud.

III. INTRODUCCION

Para preparar profesionistas cuya función social y desempeño, contribuyan al desarrollo y mejoramiento de las condiciones de vida en el país, la Escuela de Graduados tiene como una de sus metas: la formación de personal docente a Nivel Licenciatura. Esto impone la exigencia de revisar y elevar la calidad de la educación.

El mejoramiento de la calidad de la educación atañe entre otros, a las Escuelas de Graduados, sobre todo ante los cambios acelerados de la tecnología contemporánea, que con frecuencia propicia la obsolescencia de los conocimientos o técnicas aprendidas, así como el uso de cursos, apuntes y prácticas de laboratorio también anticuados. Esta tesis, pretende contribuir en ofrecer un Curso de Microcontroladores a nivel licenciatura, para ser utilizado principalmente en el Sistema Nacional de Institutos Tecnológicos, e incluye como factores:

A. Un adecuado enfoque en el Programa de Estudios.
B. La información teórica necesaria en unos Apuntes de Microcontroladores.
C. La capacitación práctica mediante Prácticas de Laboratorio.

Sostengo la idea de que el Programa de Estudios, los Apuntes y el Desarrollo de Prácticas de Laboratorio, son el medio en que convergen y se sintetizan los elementos de un modelo académico y que a partir de él se definen y orientan las interacciones que caracterizan la docencia.

Diseñar o rediseñar un Curso de Microcontroladores se concibe desde este punto de vista, como una actividad totalizadora que exige preparación y capacitación teórica y práctica o sea: aplicación de teorías, conceptos y estrategias de información pertinentes.

IV. METODOLOGIA DE LA ENSEÑANZA.

La Enseñanza Tradicional se ha ido transformando para dar paso a la llamada "Sistematización de la Enseñanza". Esta última tiene como característica, incorporar todas aquellas ideas que se han experimentado en educación al desarrollo de modelos en los que el funcionamiento de la tecnología educativa es óptima, es decir, modelos en los que la interacción entre los recursos humanos, materiales y técnicos, es fundamental para el logro de objetivos instruccionales. Además, el énfasis del proceso instruccional está centrado especialmente en la actividad del estudiante.

La Sistematización de la Enseñanza se manifiesta concretamente, en la elaboración de programas por objetivos, formulados de acuerdo al enfoque conductista, programas en los que el elemento central es el objetivo operacional. Es precisamente en este elemento del modelo académico que doy forma al proyecto que hoy me ocupa. Intento dar forma al Curso de Microcontroladores como resultado de este enfoque de la tarea educativa, del aprendizaje y de la enseñanza.

Se trata de que en la realización del proceso educacional programado, se obtenga un verdadero desarrollo personal de los estudiantes y maestros, a la vez que produzca aprendizajes significativos .

Este Curso de Microcontroladores es una propuesta de aprendizajes que guardan una relación directa y concreta entre: el programa de estudios, los apuntes y el desarrollo de prácticas de laboratorio. La validez de los aprendizajes propuestos por lo tanto, se realizan en función del valor que estos tienen para desarrollar la formación de esta disciplina en particular y en su contribución al cumplimiento del curso bajo estudio.

El Programa de Estudios es una guía fundamental, pues funciona como criterio de orientación para la selección y diseño de las actividades de aprendizaje. Además, es un elemento de comunicación entre docentes y alumnos, porque al ser un instrumento de trabajo, genera compromisos para ambos al orientar sus actividades y esfuerzos, al logro de los aprendizajes que se han fijado como necesarios a partir del objetivo general del curso.

Los Apuntes y las Prácticas de Laboratorio del Curso de Microcontroladores corresponden a la instrumentación didáctica, que junto con el programa de estudios, ocupan momentos diferentes pero complementarios y necesarios de un mismo proceso; es decir, si bien la instrumentación didáctica (apuntes y práctica de laboratorio) deben hacer acopio de la experiencia y posibilidades personales tomando como base fundamental el programa de estudios, pretendiendo, que aún con las diferencias individuales, todos los estudiantes logren los conocimientos propuestos.

De aquí que se planteen como **Elementos del Programa** los *siguientes* puntos:

1. DATOS GENERALES DE IDENTIFICACION DEL CURSO.

1.1. Nombre del Curso. Es la denominación oficial que aparecerá en la carrera.

1.2. Carrera. *Se especifica* la(s) carrera(s) a la(s) que va dirigida el curso.

1.3. Lugar y fecha. Se anotan los datos correspondientes, mes y año en que se elaboró el programa.

2. HISTORIA DEL PROGRAMA

2.1. Lugar y fecha de revisión.

2.2. Participantes.

2.3. Cambios y justificación. Se enuncian los materiales de los que parte para elaborar o revisar el programa. Así como las modificaciones que se realizan y su justificación. Este apartado es acumulativo y deben conservarse en él, todos los cambios y justificaciones que se le hagan al programa, desde su primera elaboración, hasta su última revisión para darse cuenta de su evolución.

3. UBICACION DE LA ASIGNATURA. En este apartado se especifica:

3.1. Las relaciones que a nivel de tema tiene esta asignatura con otras *anteriores y posteriores.*

3.2. Las aportaciones concretas de la asignatura al desempeño profesional.

4. OBJETIVOS GENERALES DEL CURSO. Esto corresponde a los enunciados de los conocimientos, así como a las habilidades y aptitudes globales, que concretizan los productos de aprendizajes que se esperan lograr al final del curso y delimitan su profundidad y alcance; orientando el desglose de las unidades de aprendizaje.

5. **TEMARIO.** Es la relación de temas y subtemas, que los contenidos con los que se trabajará para lograr los objetivos generales .

6. **APRENDIZAJES REQUERIDOS.** En este apartado se enlistan los aprendizajes de las asignaturas ya cursadas y que son necesarios para que el alumno tenga una mayor probabilidad de lograr los aprendizajes planteados en los objetivos del curso.

7. **UNIDAD DE APRENDIZAJE.** En estas se observa lo siguiente:
Su determinación se hace con criterios que toman en cuenta, tanto la estructura lógica de la ciencia (estructura conceptual) así como las capacidades y aprendizajes ya logrados por el estudiante en otros cursos. También se toma encuenta lo que se proyecta lograr en esta asignatura y que está delineado en los objetivos generales.

Las Unidades de Aprendizaje son un elemento resultante de la división del curso tomando en cuenta los criterios anteriores; debido a esto, su contenido temático puede constituirse de uno o más temas, o ser un conjunto de subtemas agrupados, cuando la unidad temática es muy compleja o extensa y que para fines de aprendizajes, se necesita y se posibilita su división.

La estructura lógica de la asignatura y los aprendizajes propuestos en los objetivos generales, dan lugar a que la última unidad haya sido diseñada de manera que permite evaluar en forma integra, los aprendizajes expuestos en el objetivo general. En la redacción de las Unidades de Aprendizaje se especifican los siguientes componentes:

7.1. Número progresivo que le corresponde en el programa
7.2. Contenido temático de la unidad.
7.3. Objetivo Educacional.
7.4. Aprendizajes Intermedios.
7.5. Bibliografía.

A continuación se detalla los tres últimos componentes.

OBJETIVO EDUCACIONAL. El aprendizaje logrado representa una modificación de las estructuras del conocimiento existente, originados por la asimilación y acomodación de nuevos conocimientos y experiencias; tales procesos internos se manifiestan con ejecuciones concretas, que el que aprende es capaz de realizar. A estas manifestaciones se les denomina "Productos de Aprendizajes", de los que son ejemplos los siguientes: resolver

problemas de cierto tipo, diseñar proyectos, elaborar trabajos bibliográficos, expresar argumentos, comparar teorías, etc.

Al formular los objetivos educacionales, se han relacionado estas ejecuciones o capacidades complejas con contenidos temáticos, para así obtener enunciados que señalen a maestros y alumnos el resultado esperado de cada unidad.

La complejidad, significado y relación de estos aprendizajes, es tomada en cuenta para que la redacción sea clara y bien definida, de tal manera, que funcionan como guía para diseñar los aprendizajes intermedios y permiten seleccionar actividades de enseñanza que propician el aprendizaje. Así mismo, funcionan como instrumentos y situaciones de evaluación válidos, para verificar si los objetivos han sido logrados por los estudiantes.

APRENDIZAJES INTERMEDIOS. En los objetivos educacionales se enuncian los aprendizajes complejos en términos de productos de aprendizajes, por lo que se hace necesario un desglose de estos en otros más sencillos, que son llamados Aprendizajes Intermedios. Estos se obtienen para dar respuesta a la siguiente pregunta: Que debe conocer y saber hacer el alumno para lograr el objetivo educacional? Se enlistan entonces estos componentes más sencillos en una secuencia y graduación para que el estudiante adquiera el aprendizaje completo. Así mismo, son enunciados claramente y son suficientes en cantidad para que con su combinación se logre el aprendizaje global.

BIBLIOGRAFIA. Aquí se especifican los materiales bibliográficos relacionados con los contenidos de la unidad y suelen ser libros, partes de libros, artículos de revistas, etc.

V. DEFINICION DEL CURSO.

El presente curso de Microcontroladores incluye: El Programa de Estudios, Apuntes y un Listado de Prácticas de Laboratorio. Está dirigido principalmente a estudiantes de nivel Licenciatura de Ingeniería Electrónica. Sin embargo, también puede ser utilizado por todos aquellos estudiantes, profesionistas y técnicos con conocimiento de electrónica y circuitos digitales, que tengan necesidad y deseos de conocer como funcionan y se aplican los microcontroladores.

En este curso se analizan las diferentes técnicas disponibles para el diseño de sistemas digitales en base a microcontroladores. El curso está basado principalmente en la familia de microcontroladores MC68HC11 de Motorola. Se debe de contar con un sistema de desarrollo de software consistente de un Procesador de Texto, Programas Ensambladores y Monitores residentes en equipo compatible IBM y hardware (micromódulos) para que el alumno realice sus prácticas y proyectos.

Al terminar el curso, el alumno será capaz de:

- Planear la solución a un problema digital en base a microcontroladores.
- Diseñar el hardware necesario tomando como base un modulo comercial y realizar adiciones a este.
- Diseñar el software utilizando ensambladores y sistemas de desarrollo. Integrar todo el conjunto, probarlo y corregirlo.

El curso está estructurado para un semestre realizando las siguientes actividades:

- Asistir 4 hs. de clase teórica por semana y estudiar el material.
- Asistir 2 hs. por semana al laboratorio para hacer una tarea práctica.
- Presentar cuando menos 2 examenes parciales teóricos y examen final de tipo práctico.

Dado lo anterior, el alumno debe dedicar 10 horas en promedio por semana, además de las horas de clases, para realizar actividades directamente relacionadas con el curso.

VI. CONCLUSIONES

El tratar de realizar un Curso de Microcontroladores es desde mi punto de vista, inherentemente mas difícil de desarrollar y planear, sobre todo si lo comparamos con otros cursos como: matemáticas, circuitos eléctricos, etc. Además es más difícil de elegir la secuencia de los temas.

El presente Curso de Microcontroladores representaba un reto que fue resuelto aplicando Tecnologías Educativas que involucran la interacción entre los recursos humanos, materiales y técnicos.

Considero que alcance con éxito el objetivo originalmente planteado de desarrollar un Curso de Microcontroladores, pues para lograr que el estudiante adquiera los conocimientos suficientes sobre microcontroladores, realicé como guía fundamental El Programa de Estudios , y desarrollé como instrumentación didáctica: los Apuntes y el Listado de Prácticas de Laboratorio.

Sin embargo, los apuntes no deben ser tomados como definitivos, pues la tecnología referente a microcontroladores está avanzando a pasos agigantados, lo cual hace necesario su revisión periódica, para así mantenerlos actualizados conforme avance la tecnología, y de esa manera evitar su obsolescencia

Si bien intenté ser lo suficiente claro al exponer cada tema, puede ser necesario para afianzar los conocimientos, el volver a leer algunos temas después de haber avanzado en algunos capítulos, especialmente en lo referente a los capítulos 3 y 4. Sin embargo esto es común en este tipo de material de alta complejidad.

El constante avance en la tecnología y la necesidad de mantener los conocimientos actualizados, fue la principal razón para elaborar un Curso de Microcontroladores, el cual espero sea de gran utilidad para muchos estudiantes.

VII. REFERENCIAS Y BIBLIOGRAFIA

Tokheim,Roger L.
Fundamentos de los Microprocesadores
Ed. McGraw Hill , Segunda Edición 1991

MacKenzie I. Scott
The 8051 Microcontroller
Ed. Merril, 1992

Bishop, Ron
Basic Microprocessors and the 6800
Ed. Hayden Book Company, Inc., 1979

Gilmore, Charles M.
Principios de Microprocesadores
Ed. Limusa, 1989

Gil Padilla, Antonio J.
Electrónica General 1. Dispositivos y Sistemas Digitales
Ed. MacGraw Hill, 1989

Motorola
HCMOS Single-Chip Microcontroller MC68HC11E9
Motorola Inc., 1988

Motorola
M68HC11 Reference Manual
Motorola Inc., 1990

Dirección General De Institutos Tecnologicos
Documento Técnico sobre los Programas de Estudios
Elaborados con Unidades de Aprendizaje, 1985

Nerici, G. Imideo
Metodología de la Enseñanza
Ed. Kapelusz, 1985

Peaget,Jean
Teoría de psicogénesis del conocimiento. Problemas de Psicología Genética., 1980

VIII. ANEXOS

VIII.1. PROGRAMA DE ESTUDIOS

1. DATOS GENERALES DE IDENTIFICACION DEL CURSO.

1.1. Nombre del curso: **MICROCONTROLADORES.**
1.2. Nivel: **LICENCIATURA.**
1.3. Carrera(s):**INGENIERIA ELECTRONICA.**
1.4. Lugar y fecha de elaboración: Mty., N.L. Junio 1994.

2. HISTORIA DEL PROGRAMA.

2.1. Lugar y fecha de revisión:
2.2. Participantes:
2.3. Cambios y justificaciones:

3. UBICACION DE LA ASIGNATURA.

3.1.1. Relación con Asignaturas Anteriores:
- Lógica Digital.
- Programación.
- Teoría de Control.

3.1.2. Relación con Asignaturas Posteriores:
- Microprocesadores II.

3.2. Aportación al Desempeño Profesional:
- Conocer el funcionamiento y uso de microcontroladores y otros circuitos para aplicaciones en aparatos electrónicos.

4. OBJETIVOS GENERALES DEL CURSO.

- Programar microcontroladores usando programas y métodos actuales (Ensambladores, simuladores, Lenguajes de bajo nivel).
- Conectar adecuadamente elementos o dispositivos de entrada, salida e interfaces al exterior.
- Diseñar dispositivos electrónicos usando microcontroladores como elemento básico.

5. TEMARIO.

UNIDAD 1.-CONCEPTOS GENERALES DE MICROCONTROLADORES.

-Introducción a los microcontroladores.
-Tipos de Computadoras y Microprocesadores vs. Microcontroladores.
-El *Microcontrolador como circuito integrado.*
-Conexiones.
-Organización de memoria.

UNIDAD 2.-CONJUNTO DE INSTRUCCIONES

-Modelo del programador.
-Modos de Direccionamiento.
-Instrucciones del Microcontrolador.
-Programa Fuente y Editores.
-Programa Objeto y Ensambladores.
-Programa Monitor e Interconexiones.

UNIDAD 3.-PUERTOS PARALELO Y CONVERTIDOR ANALOGO-DIGITAL

-Puertos
-Comunicación Paralelo.
-Proceso de Conversión.
-Operación en canal único y multicanal.
-Registro de Control.

UNIDAD 4.-PROYECTO.

6. *APRENDIZAJES REQUERIDOS.*

- Diseño de Circuitos Combinacionales.
- Diseño de Circuitos Secuenciales.
- Unidad Lógica-Aritmética, Unidad de Control, Unidades
- Entrada/Salida.
- Algoritmos computacionales.

7. UNIDADES DE APRENDIZAJE

UNIDAD I.- CONCEPTOS GENERALES DE MICROCONTROLADORES.

OBJETIVO EDUCACIONAL.- *Describir el funcionamiento y arquitectura de un* microcontrolador. Desarrollando además un vocabulario de trabajo, en donde sea capaz de utilizar apropiadamente la terminología, siglas, abreviaciones y modismos necesarios.

APRENDIZAJES INTERMEDIOS.

Subtema.-Introducción a las computadoras.

1.1. Describir los componentes básicos y el funcionamiento de una computadora.

Subtema.-Tipos de Computadoras y Microprocesadores vs. Microcontroladores

1.2. Clasificar las computadoras de acuerdo a sus características. Describir las diferencias en la arquitectura, aplicaciones y conjunto de instrucciones entre un microprocesador y un microcontrolador.

Subtema.-El Microcontrolador como circuito integrado.

1.3 Establecer las características generales de un microcontrolador y explicar la función que realiza cada uno de los bloques (subsistemas) que lo conforman.

Subtema.-Conexiones y Terminales..

1.4. Expresar la función que realiza cada terminal de un microcontrolador y reconocer el significado de las siglas que las representan.

Subtema.-Organización de la memoria.

1.5. Describir la organización de la memoria en cada modo de operación.

UNIDAD II.-CONJUNTO DE INSTRUCCIONES.

OBJETIVO EDUCACIONAL.- Aplicar el conjunto de instrucciones de un microcontrolador en la elaboración de programas en lenguaje ensamblador, convertirlos a lenguaje de máquina, transferirlos al microcontrolador y hacerlos funcionar.

APRENDIZAJES INTERMEDIOS.

Subtema.-Modelo del Programador.

2.1. Comprender la función del CPU y los registros internos que son utilizables por el programador.

Subtema.-Modos de Direccionamiento.

2.2. Comprender el significado de los diversos modos de direccionamiento.

Subtema.-Instrucciones del Microcontrolador.

2.3..Entender la acción que realiza cada instrucción y su efecto en los contenidos de los registros y memoria antes y después de ser ejecutadas.

Subtema.-Programa Fuente y Editores.

2.4. Realizar un programas fuente en lenguaje ensamblador y usar un Editor de Texto para convertirlos a lenguaje ASCII.

Subtema.-Programa Objeto y Ensambladores.

2.5. Utilizar un programa ensamblador para obtener el programa objeto en el lenguaje de máquina.

Subtema.-Programa Monitor e Interconexión con el microcontrolador.

2.6. Utilizar un programa monitor para transferir el programa objeto al microcontrolador y realizar las conexiones necesarias para probar su funcionamiento.

UNIDAD III.-PUERTOS PARALELO Y CONVERTIDOR ANALOGO-DIGITAL

OBJETIVO EDUCACIONAL.-Utilizar los puertos paralelo para transmisión y recepción de datos mediante programas en ensamblador, y aplicar el convertidor análogo-digital en la manipulación de señales.

APRENDIZAJES INTERMEDIOS.

Subtema.-Puertos

3.1. Programar las terminales de los puertos como: entradas, salidas, bidireccionales o para aplicaciones especiales.

Subtema.-Comunicación Paralelo.

3.2. Realizar Comunicación Paralelo en: Modo Directo, Strobe Simple y Protocolo Total (Full Handshake).

Subtema.-Operación en canal único y multicanal.

3.3 Realizar programas para la operación de conversión en canal único o en multicanal.

Subtema.-Registros de Control.

3.4 Entender el efecto de los diversos registros de control del convertidor Análogo-Digital.

UNIDAD 4.-PROYECTO.

OBJETIVO EDUCACIONAL .- Aplicar los conceptos desarrollados durante el curso, a la solución de un problema digital. Tomar como base el hardware de soporte del MC68HC11A8 y hacerle las adiciones necesarias. Además, diseñar el software requerido utilizando el Editor de Texto, el Programa Ensamblador y el Programa Monitor PCbug11 para integrar todo el conjunto, probarlo y corregirlo.

VIII.2. LISTA DE PRACTICAS DE LABORATORIO

La elaboración de Prácticas de Laboratorio proporciona la experiencia en la manipulación del conjunto de instrucciones y en el hardware de soporte del MCU. Además, mediante *el uso de la computadora se adquiere* la habilidad en el manejo de los Programas Ensamblador y del Programa Monitor. Lo anterior permitirá al estudiante el poder emprender proyectos de mayor magnitud con la plena seguridad de su funcionalidad.

Es recomendable que durante el desarrollo y como reporte de cada práctica, el alumno realice lo siguiente:

- Descripción de la práctica.
- Elaboración del Diagrama de Flujo.
- *Edición del Programa.*
- Ensamblado del Programa.
- Elaboración del diagrama de conexiones con el hardware de soporte del MCU.
- Depuración y puesta en operación mediante el uso del Programa Monitor.
- Observaciones y Conclusiones del desarrollo de la práctica.

A continuación se presenta una lista de las prácticas sugeridas:

Practica #1. Descripción del Sistema de Desarrollo: Editor de Texto, Programa Ensamblador, Programa Monitor, Hardware de soporte del MCU.

Practica #2. Programa que realice la suma y resta de dos números.

Práctica #3. Programa que lea las señales de entrada proporcionada por interruptores y envíe a la salida para mostrar las condiciones mediante el encendido y apagado de leds

Práctica #4. Elaboración de un programa que encienda leds de izquierda a derecha en forma secuencial, con la posibilidad de variar la rapidez de encendido.

Práctica #5. Diseño de una interfaz para mostrar números hexadecimales en un display.

Practica #6. Diseño de una interfaz para lectura de un teclado hexadecimal y mostrar los valores en un display.

Práctica #7. Conversión de un voltaje analógico a señal digital.

Práctica #8. Programa que realíce operaciones de suma, resta y multiplicación de doble precisión.

Práctica #9. Programa que realíce operaciones con tablas de senos y cosenos.

Práctica #10 Verificación de la operación del proyecto del curso.

VIII.3 APUNTES

CURSO DE MICROCONTROLADORES

ING. RENE BRIONES LARA

San Nicolás de los Garza, N.L. Septiembre de 1994

PREFACIO

Estos apuntes son desarrollados para formar parte de un Curso de Microcontroladores, y examinan las características del hardware (equipo) y del software (programación) de los microcontroladores en general; pero esencialmente de los dispositivos de la familia M68HC11 de Motorola. Estos apuntes están dirigidos a estudiantes de Ingeniería Electrónica y a todos aquellos estudiantes, profesionistas y técnicos con conocimientos de electrónica y circuitos digitales, que tengan deseos de conocer como se aplican los microcontroladores.

En el desarrollo de estos apuntes se ha tomado como prototipo el microcontrolador MC68HC11A8, dada su amplia aplicación en procesos de control industrial. Se pretende que al final del curso, el estudiante realice un proyecto que involucre: diseñar, implementar y documentar un proceso controlado por el microcontrolador MC68HC11A8, además que incorpore hardware y software con su propia originalidad.

Dado que el MC68HC11A8, así como todos los microcontroladores contienen un alto grado de funcionalidad, se hace énfasis en su arquitectura y programación en vez de en los detalles eléctricos. Los tópicos de software son desarrollados en función de aplicar: el Programa Ensamblador AS11 y el Programa Monitor PCbug11.

El capítulo 1 sirve como introducción a los microcontroladores, con énfasis en las diferencias entre los microcontroladores y los microprocesadores. El capítulo 2 describe la arquitectura del hardware del microcontrolador MC68HC11A8 que forma parte de la familia M68HC11. El capítulo 3 introduce el Conjunto de Instrucciones, empezando con definiciones de los Modos de Direccionamiento del MC68HC11A8. El Conjunto de Instrucciones ha sido dividido en grupos de instrucciones (aritméticas, lógicas, de transferencia de datos, etc.), lo cual facilita la presentación paso a paso. Numerosos ejemplos breves se presentan para muchas instrucciones, incluyendo sus modos de direccionamiento.

Los capítulos 4, 5 y 6, progresan a través de las características del MC68HC11A8, no debe precipitarse en estos capítulos, pues es esencial un sólido entendimiento de la arquitectura del hardware y del conjunto de instrucciones del microcontrolador al ir avanzando en estos tópicos.

El capítulo 4 presenta las técnicas y herramientas que son utilizadas en un Sistema de Desarrollo de Software, es importante la elaboración de prácticas en este punto, pues la experiencia en la manipulación del conjunto de

instrucciones nunca podrá ser sobre enfatizada. Los estudiantes deben ser animados en completar las prácticas utilizando la computadora, observando la salida y los mensajes de error provenientes del Programa Ensamblador AS11 y el Programa Monitor PCbug11, y viendo la utilidad de la conversión del programa fuente a hexadecimal.

El capítulo 5 trata sobre Puertos Paralelo para realizar la comunicación o interacción con el mundo real. Y en el capítulo 6 es presentada la conversión análogo-digital.

La culminación de un Curso de Microcontroladores mediante un proyecto es quizá lo mas importante, pues desarrolla en el estudiante el potencial de emprender proyectos de gran escala. Tópicos de programación estructurada, ensamblado, monitoreo , depuración, y eslabonamiento de un programa, producirán un cambio significativo en muchos estudiantes.

En estos apuntes se hace un amplio uso de la literatura de Motorola, especialmente de los dispositivos de la familia M68HC11. En el apéndice se hace un listado completo del Conjunto de Instrucciones del MC68HC11A8.

Existe una dificultad muy especial en la terminología utilizada para describir el funcionamiento de los microcontroladores, sobre todo si se intenta traducirla. Dado que es más común utilizar los términos en inglés, opté por utilizar los términos originales en muchos casos.

La tecnología referente a microcontroladores, y a todos los campos de la electrónica en general, está avanzando a pasos agigantados; por lo que opté por numerar las hojas de cada capítulo en forma independiente, empezando la numeración con el número del capítulo, por ejemplo: 2.1, 2.2, 2.3, etc. Esto permitirá modificar cada capítulo en forma independiente, para así actualizar estos apuntes conforme avance la tecnología y de esa manera evitar su obsolescencia.

Deseo expresar mi profundo agradecimiento al Ing. Ronald López por sus valiosas sugerencias, apoyo y tiempo, que me brindó durante el desarrollo del escrito de este curso.

TABLA DE CONTENIDOS

Número de Párrafo	Título	Número de Página
	Capítulo 1	
	Conceptos Generales de Microcontroladores	
1.1	Introducción	1.1
1.2	Organización de la Computadora	1.1
1.3	Operación de la Computadora	1.3
1.4	Tipos de Computadoras	1.7
1.5	Microprocesadores vs. Microcontroladores	1.8
1.6	CPU Unidad de Procesamiento Central	1.11
1.7	Memorias	1.13
1.7.1	Sistema de Numeración binario y hexadecimal	1.14
1.7.2	Clases de Memorias	1.16
1.8	Buses de Dirección, Datos y Control	1.18
1.9	Puertos I/O	1.20
	Capítulo 2	
	El Microcontrolador Como Circuito Integrado	
2.1	Introducción	2.1
2.2	Características Generales	2.1
2.3	Descripción General del Hardware	2.3
2.4	Conexiones o Terminales	2.7
2.5	Modos de Operación	2.11
2.6	Organización de la Memoria	2.13
2.6.1	RAM	2.13
2.6.2	EXTERNA	2.14
2.6.3	BLOQUE DE REGISTROS	2.14
2.6.4	EEPROM	2.14
2.6.5	ROM	2.14
	Capítulo 3	
	Conjunto de Instrucciones	
3.1	Introducción	3.1
3.2	Modelo del Programador	3.1
3.3	Modos de Direccionamiento	3.5
3.4	Conjunto de Instrucciones	3.9
3.4.1	Nomenclatura	3.9
3.4.2	Clasificación de las Instrucciones	3.12
3.4.2.1	Instrucciones Aritméticas	3.13
3.4.2.2	Instrucciones Lógicas	3.22

TABLA DE CONTENIDOS continuación...

Número de Párrafo	Título	Número de Página
3.4.2.3	Instrucciones de Transferencia de Datos	3.29
3.4.2.4	Instrucciones de Bifurcación o Salto	3.33
3.4.2.5	Instrucciones de Llamada y Retorno de Subrutina	3.35
3.4.2.6	Instrucciones Misceláneas	3.41
	Capítulo 4	
	Sistema de Desarrollo de Software	
4.1	Introducción	4.1
4.2	Técnicas de Programación	4.3
4.3	Lenguajes de Programación	4.4
4.4	Editor de Texto	4.4
4.5	Programa Ensamblador AS11	4.5
4.5.1	Formato o Sintaxis de Líneas del Programa Fuente	4.5
4.5.1.1	El Campo de la Etiqueta	4.5
4.5.1.2	El Campo de Operación	4.6
4.5.1.3	El Campo del Operando	4.7
4.5.1.3.1	Expresiones	4.7
4.5.1.4	El Campo de Comentarios	4.9
4.5.2	Directivas del Ensamblador	4.9
4.5.2.1	BSZ	4.10
4.5.2.2	EQU	4.10
4.5.2.3	FCB	4.10
4.5.2.4	FCC	4.10
4.5.2.5	FDB	4.11
4.5.2.6	FILL	4.11
4.5.2.7	OPT	4.11
4.5.2.8	ORG	4.12
4.5.2.9	PAGE	4.12
4.5.2.10	RMB	4.12
4.5.2.11	ZMB	4.13
4.5.3	Invocación del Ensamblador	4.13
4.5.4	Ejercicio de Escritura y Ensamblado de un Programa	4.15
4.6	Programa Monitor PCbug11	4.18
4.6.1	Introducción	4.18
4.6.2	Como Trabaja el PCbug11	4.18
4.6.2.1	Método Cargador de RAM Interna	4.18
4.6.2.2	Método Cargador de ROM	4.19
4.6.3	Comandos del PCbug11	4.20

TABLA DE CONTENIDOS continuación...

Número de Párrafo	Título	Número de Página
4.6.4	Preparación del MCU y del PCbug11	4.22
4.6.5	Invocación del PCbug11	4.24
4.6.6	Ventanas del PCbug11	4.26
4.6.7	Edición de la línea de Comandos	4.27
4.6.8	Ejemplo de instalar un programa en la EEPROM	4.28
	Capítulo 5 Puertos Paralelo	
5.1	Introducción	5.1
5.2	Puertos	5.1
5.2.1	Puerto A	5.5
5.2.2	Puerto B	5.5
5.2.3	Puerto C	5.6
5.2.4	Puerto D	5.6
5.2.5.	Puerto E	5.6
5.3	Comunicación Paralelo	5.7
5.3.1	Comunicación Paralelo en Modo Directo	5.7
5.3.2	Comunicación Paralelo con Protocolo de Comunicación	5.8
5.3.2.1	Comunicación Paralelo con Control Simple	5.10
5.3.2.2	Comunicación Paralelo con Protocolo Total	5.12
	Capítulo 6 Convertidor Análogo-Digital	
6.1	Introducción	6.1
6.2	Conceptos Generales	6.1
6.3	Terminales V_{RH} y V_{RL}	6.4
6.4	Multiplexador de Entrada	6.4
6.5	Circuito de Control	6.4
6.5.1	Selección de Canales para Conversión A/D	6.5
6.5.2	Modos de Operación de Conversión	6.6
6.5.3	Activación del Sistema A/D	6.7
6.5.4	Selección del Oscilador RC Interno	6.7
	Glosario	G1
	Apéndice	A1

CAPITULO 1

CONCEPTOS GENERALES DE MICROCONTROLADORES.

1.1 INTRODUCCION.

El impacto de las computadoras a sido muy profundo a pesar de haber aparecido hace solo algunas décadas, pues se han venido utilizando en forma general desde 1950. Su presencia es sentida por todos debido a la gran cantidad de publicidad que aparece en periódicos, radio y televisión.

Generalmente las computadoras son consideradas como máquinas "procesadores de datos" o realizando operaciones matemáticas con inagotable competencia. Nosotros confrontamos las computadoras desde un punto de vista bastante distinto, es decir, realizando su trabajo en forma silenciosa, eficiente y cuya presencia generalmente es desapercibida.

Como componente central de una computadora encontramos un circuito integrado llamado microprocesador. Como un concepto nuevo a surgido recientemente un circuito llamado microcontrolador, el cual realiza funciones de control por interactuar con el mundo real, encendiendo o apagando interruptores o realizando actividades de monitoreo. Algunas aplicaciones del microcontrolador son: controladores industriales, hornos de microondas, lavadoras automáticas, automóviles, juguetes, video caseteras, estéreos, etc.

1.2 ORGANIZACION DE LA COMPUTADORA.

Una **computadora** puede ser definida como un sistema de procesamiento de datos, que puede ser programada para operar sin la intervención humana, y que tiene además la habilidad de almacenar y obtener datos. Un sistema básico de computadora se muestra en la figura 1.2.1. La **Arquitectura de la Computadora** es el nombre dado a ésta organización funcional y consta de 5 unidades:

1. **CPU** (Central Procesing Unit).- Unidad de Procesamiento Central.
2. **RAM** (Random Access Memory).- Memoria de Acceso Aleatorio.
3. **ROM** (Read Only Memory).- Memoria de Solo Lectura.

4. **BUSES.-** Conjunto de alambres que transportan información con un propósito común.
5. **PUERTOS I/O** (Input/Output).- Puertos de Entrada/Salida.

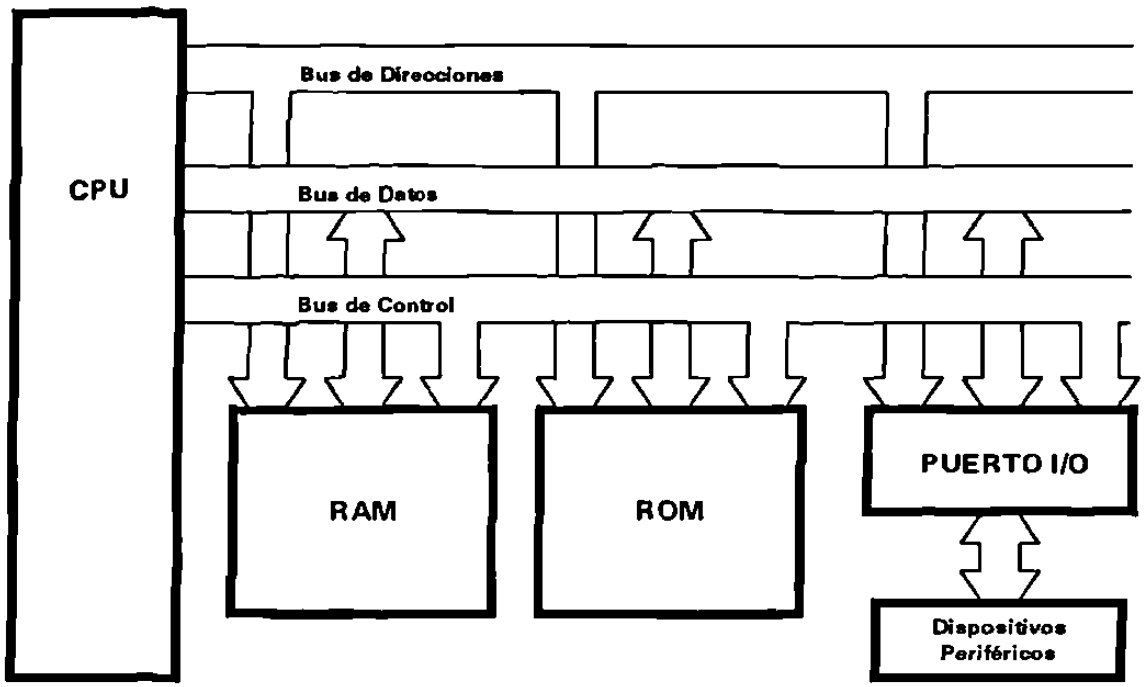


Figura1.2.1 Diagrama de bloques de un Sistema Básico de Computadora.-

El **hardware** es el equipo y constituye las unidades físicas representadas por los bloques de la figura 1.2.1. Para que el hardware sea útil, la memoria del programa debe decir al CPU lo que tiene que hacer. La **Programación** es la preparación de la lista de instrucciones. El **Software** es un término general utilizado para denominar a todos los programas. **Firmware** es el nombre dado al software que es almacenado permanentemente en la memoria de programa.

La **CPU** o Unidad de Procesamiento Central controla todas las unidades del sistema por medio de las líneas o **bus de control.** El **bus de direcciones** es un conjunto de líneas que selecciona una cierta localidad de memoria, puerto de entrada o puerto de salida. El **bus de datos** es el conjunto de líneas de doble sentido o dirección, que se utiliza para introducir o sacar datos de la CPU. Es importante observar que la CPU *puede enviar o recibir datos de la* memoria utilizando el bus de datos.

La **ROM** (Read Only Memory) o Memoria de Solo Lectura habitualmente se utiliza para almacenar el programa en forma permanente. La **RAM** (Random Access **Memory**) o memoria de acceso aleatorio es una memoria de lectura/escritura en donde se almacena en forma temporal los datos o algunos

programas de naturaleza temporal. Por claridad, es costumbre omitir en los diagramas de bloques: las fuentes de alimentación, relojes y algunas líneas de realimentación del CPU.

1.3 OPERACION DE LA COMPUTADORA.

En la figura 1.3.1 se muestra un ejemplo de como opera una computadora y en él se muestra el siguiente procedimiento:

1. Pulsar la tecla A.
2. Almacenar en memoria la letra A.
3. Sacar la letra A en la pantalla del monitor de tubo de rayos catódicos.

El procedimiento de entrada-almacenamiento-salida esbozado en la figura 1.3.1 es una operación común de un sistema de computadora. Para comprender éste procedimiento de transferencia de datos y explicar el uso de las diferentes unidades de la computadora, un diagrama más detallado se muestra en la figura 1.3.2

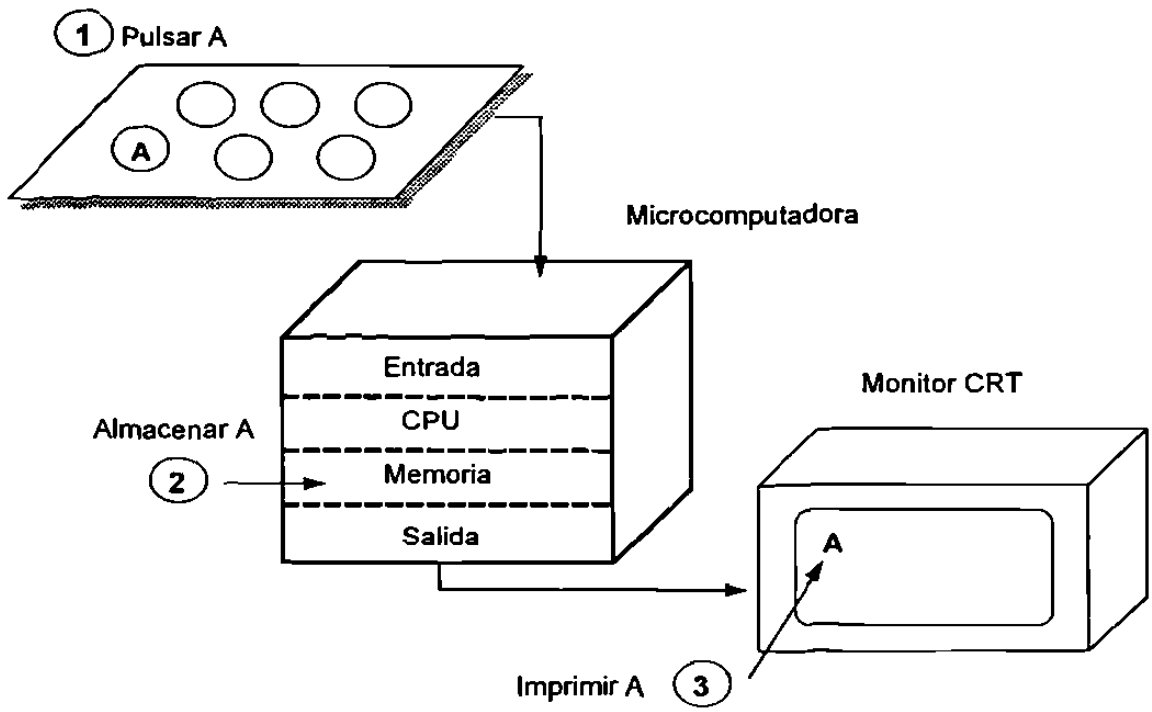


Fig. 1.3.1. Operación típica de la microcomputadora

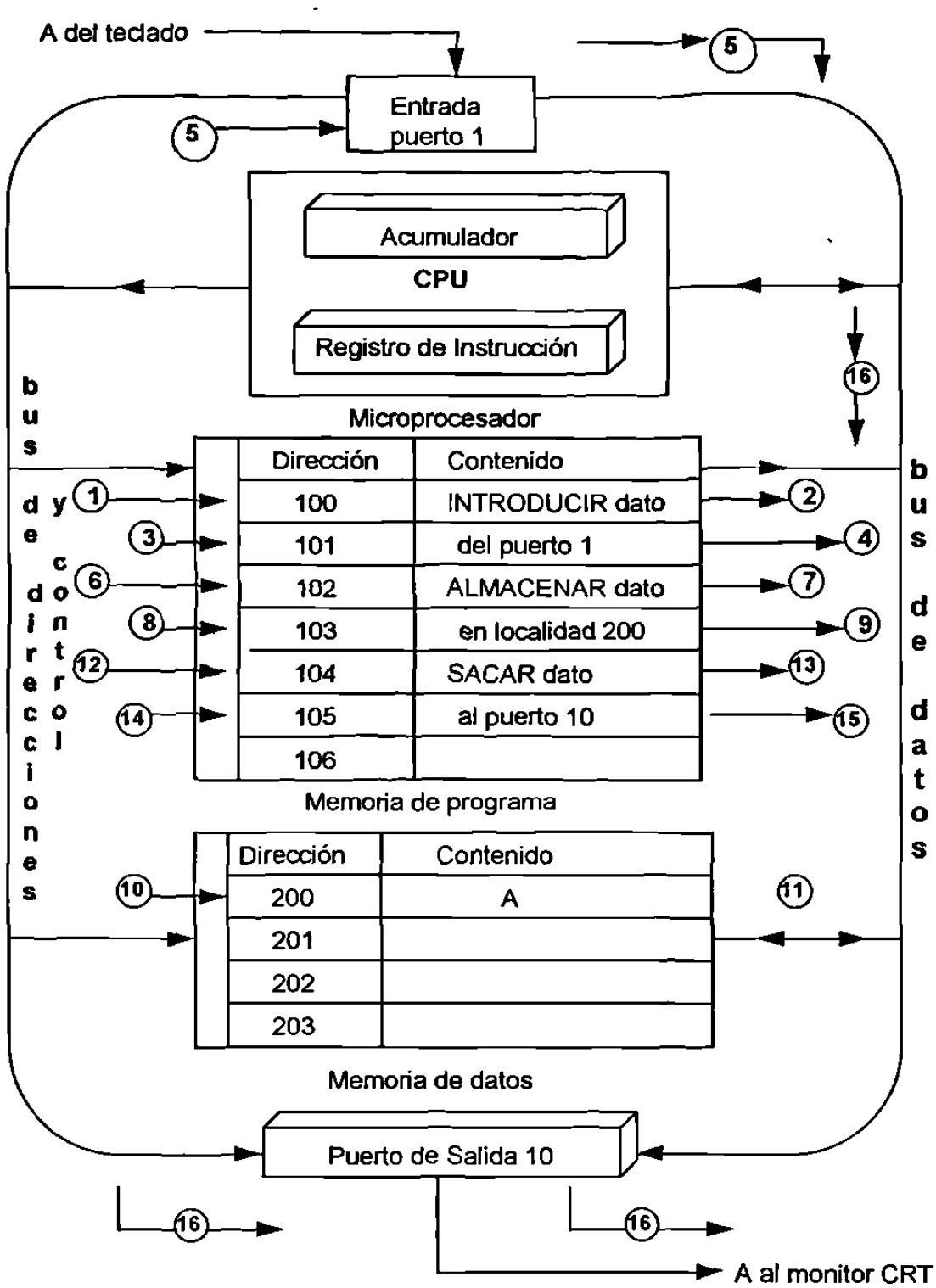


Fig. 1.3.2. Operación paso a paso de una microcomputadora cuando ejecuta las instrucciones de la memoria del programa

Observe cuidadosamente la figura 1.3.2. y vea los contenidos de la memoria de programa. Verá que las instrucciones ya se han cargado en las seis primeras localidades de memoria. Estas instrucciones que están en la memoria de programa son:

1. **INTRODUCIR el dato**, del puerto de entrada 1.
2. **ALMACENAR el dato** del puerto 1, en la localidad 200 de la memoria de datos.
3. **SACAR el dato**, al puerto de salida 10.

Observar que solamente hay tres instrucciones en el programa anterior, aunque puede parecer que hay seis en la memoria de programa de la figura 1.3.2. La razón para pensar esto es que las instrucciones se descomponen habitualmente en varias partes. La primera parte de la instrucción 1 consiste en **INTRODUCIR el dato**. La segunda parte indica de donde proviene el dato **del puerto de entrada 1**. La primera parte de la instrucción se denomina **operación** y la segunda parte **operando**. Operación y operando están ubicados en localidades separadas en la memoria de programa.

Para la primera instrucción de la figura 1.3.2 la localidad 100 de la memoria de programa contiene la operación **INTRODUCIR el dato**, mientras que la localidad 101 contiene el operando **del puerto de entrada 1** que es desde donde se introduce la información.

En el CPU de la figura 1.3.2 se identifican dos nuevas secciones denominadas el registro acumulador y el registro de instrucción.

La secuencia de eventos que ocurren en la computadora en el ejemplo de introducir-almacenar-sacar de la figura 1.3.1, se esboza en la figura 1.3.2. El flujo de instrucciones y datos puede seguirse en el diagrama por los números inscritos en los círculos. Se debe recordar que el CPU es el centro de todas las transferencias de datos y operaciones. En la figura 1.3.2 se pueden seguir los pasos que a continuación se detallan.

Paso 1. La CPU envía la localidad o dirección 100 al bus de direcciones. Una línea de control habilita (conecta) la entrada de lectura en la memoria de programa (leer significa copiar información de una localidad de memoria). Este paso se simboliza en la figura por un 1 inscrito en un circulo.

Paso 2.- La memoria de programa anuncia la primera instrucción **"INTRODUCIR dato"** al bus de datos, y la CPU acepta este mensaje codificado. La instrucción se coloca en una posición de memoria especial de la CPU denominada registro de instrucción. La CPU decodifica o interpreta la instrucción, y determina que necesita el operando de la instrucción (¿De dónde proviene el dato?).

Paso 3.- La CPU envía la localidad 101 al bus de direcciones y la línea de control habilita la entrada de lectura de la memoria de programa.

Paso 4.- La memoria de programa coloca el operando **"del puerto 1"** en el bus de datos. El operando estaba localizado en la localidad 101 de la memoria de programa. Este mensaje codificado **"la dirección del puerto 1"** se acepta en el bus de datos y se coloca en el registro de instrucción. La CPU decodifica ahora la instrucción completa **"INTRODUCIR el dato, del puerto 1"**.

Paso 5.- La CPU hace que se abra el puerto 1 utilizando el bus de direcciones y las líneas de control en la unidad de entrada. La forma está codificada en el acumulador de la CPU.

Es importante observar que la CPU sigue siempre una secuencia de busca-decodifica-ejecuta. Primero *busca* la instrucción en la memoria de programa; segundo, la decodifica y tercero la ejecuta. Trate de observar esta secuencia de busca-decodifica-ejecuta en las dos instrucciones siguientes y continúe con el programa listado en la memoria de programa de la figura.

Paso 6.- La CPU direcciona la localidad 102 en el bus de direcciones; después habilita la entrada de lectura en la memoria de programa utilizando las líneas de control.

Paso 7.- El código de la instrucción **"ALMACENAR el dato del puerto 1"** es leído en el bus de datos y es aceptado por la CPU en el registro de instrucción.

Paso 8.- La CPU decodifica la instrucción **"ALMACENAR el dato del puerto 1"** y determinar que necesita el operando. La CPU direcciona la siguiente localidad de memoria 103 y habilita la entrada de lectura de la memoria de programa.

Paso 9.- El código para **"en la localidad de memoria 200"** es colocado en el bus de datos por la memoria de programa. La CPU acepta este operando y lo almacena en el registro de instrucción. La instrucción completa **"ALMACENAR el dato del puerto 1, en la localidad de memoria 200"** ha sido buscada en memoria y decodificada.

Paso 10.- Ahora comienza el proceso de ejecución. La CPU envía la dirección o localidad 200 al bus de direcciones y habilita la entrada de escritura de la memoria de datos . Escribir significa copiar datos en una localidad de memoria.

Paso 11.- La CPU pone la información almacenada en el acumulador en el bus de datos (la forma codificada de A). La A se escribe en la localidad 200

de la memoria de datos. La segunda instrucción ha sido ejecutada. El proceso de ALMACENAR no destruye el contenido del acumulador, éste contiene todavía la forma codificada de A.

Paso 12.- La CPU debe buscar la información siguiente. Direcciona la localidad 104 y habilita la entrada de lectura de la memoria de programa.

Paso 13.- El código de la instrucción **"SACAR el dato"** se coloca en el bus de datos. La CPU acepta la instrucción en el registro de instrucción, la decodifica y determina que necesita un operando.

Paso 14.- La CPU coloca la localidad 105 en el bus de direcciones y habilita la entrada de lectura de la memoria de programa.

Paso 15.- La memoria de programa pone el código del operando **"al puerto 10"** en el bus de datos. La CPU acepta este código en el registro de instrucción.

Paso 16.- La CPU decodifica la instrucción completa **"SACAR el dato, al puerto de salida 10"** y activa al puerto 10 utilizando el bus de direcciones y las líneas de control en la unidad de salida. La A es transmitida desde el puerto 10 al monitor CRT.

La mayoría de las CPU's transfieren información de forma similar a la detallada en la figura 1.3.2. Las variaciones más grandes están probablemente en las secciones de entrada y salida. A veces se necesitan algunos pasos para lograr que las secciones de entrada y salida operen adecuadamente.

Es importante observar que la CPU es el centro y control de todas las operaciones y sigue la secuencia de busca-decodifica-ejecuta. Siempre las operaciones reales de la CPU son dictadas por las instrucciones listadas en la memoria del programa.

1.4 TIPOS DE COMPUTADORAS.

Las computadoras considerando su tamaño y potencia de cómputo pueden ser clasificadas en:

1. Microcomputadoras.
2. Minicomputadoras.
3. Computadoras de tablero Principal.

Las Microcomputadoras tienen como característica principal su tamaño reducido y el empacado del CPU, el cuál está contenido en un solo circuito integrado llamado microprocesador. Típicas **microcomputadoras** son las computadoras personales, como por ejemplo: IBM PC, Apple Macintosh, y la Commodore Amiga, que incorporan a un microprocesador como su CPU. La RAM, la ROM y los puertos de entrada y salida requieren de muchos circuitos integrados. Conforme se incrementa la cantidad de circuitos, se incrementa la potencia de cómputo. Los circuitos de interfaz varían considerablemente dependiendo de los dispositivos I/O (de entrada/salida). El manejo de la bocina contenida en muchas micro computadoras requiere solo un acoplamiento de compuertas lógicas, pero la interfaz del disco usualmente requiere muchos circuitos integrados. Otras características de las microcomputadoras son que interactuan con un solo usuario y ejecutan un solo programa a la vez.

Las **Minicomputadoras** son más complejas en su arquitectura y tienen un CPU consistente de varios circuitos integrados, lo cuál es necesario para obtener mayor velocidad y potencia de cómputo, además las minicomputadoras son multiusuario y ejecutan simultáneamente varios programas, lo cual es una ilusión resultante de un recurso de la CPU llamado "tiempo compartido".

Las **Computadoras de Tablero Principal** son de gran tamaño y potencia de cómputo, su arquitectura es muy compleja y su CPU consiste de varias tablillas de circuito impreso con múltiples circuitos integrados, además tienen gran velocidad y potencia de cómputo y puede realizar varias tareas al mismo tiempo.

1.5 MICROPROCESADORES VS. MICROCONTROLADORES.

Como fue indicado en el punto anterior, las microcomputadoras tienen un microprocesador como circuito integrado único que realiza las funciones de la CPU. Entonces, ¿Qué es lo que hace al microcontrolador diferente de un microprocesador? Esta pregunta será contestada desde 3 puntos de vista:

1. Arquitectura del hardware.
2. Aplicaciones.
3. Conjunto de instrucciones

Arquitectura del Hardware.

Para resaltar la diferencia entre los microcontroladores y los microprocesadores, la figura 1.2.1. es redibujada mostrando más detalles en la figura 1.5.1.

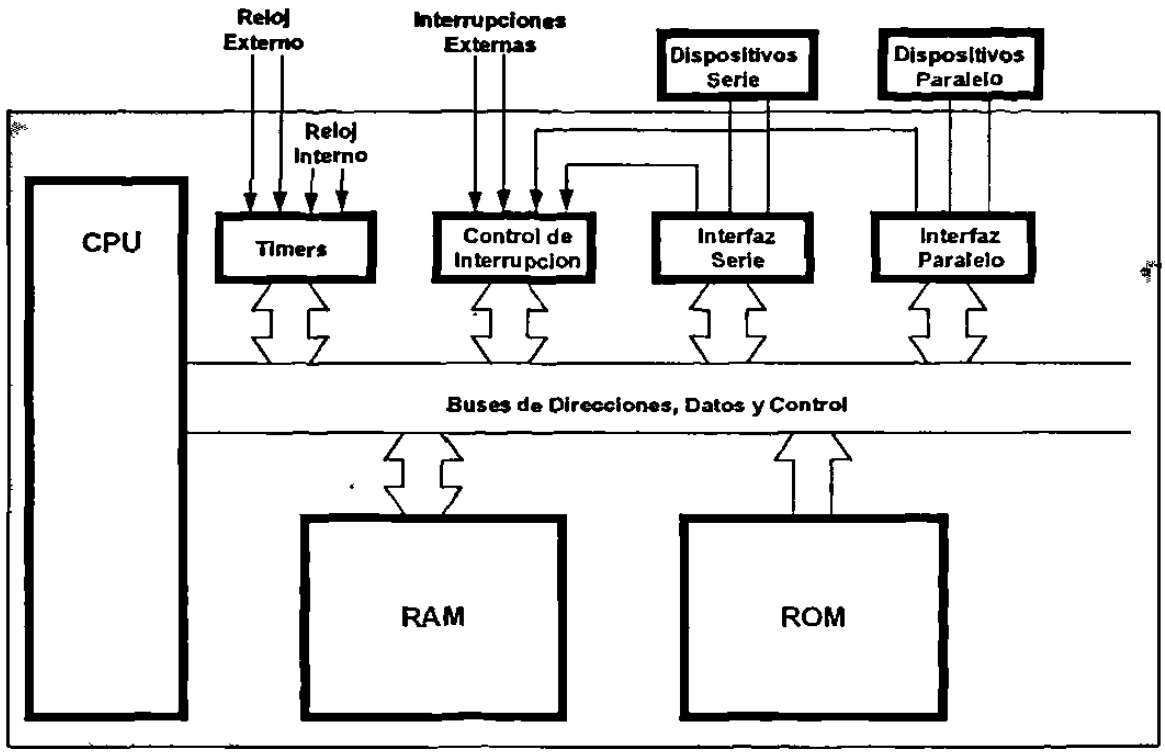


Fig.1.5.1. Diagrama de Bloques de un sistema de Microcomputadora.

Mientras que el microprocesador es un circuito integrado que solo realiza las funciones de CPU, el microcontrolador contiene en un solo circuito integrado al CPU y mucho del circuito restante de un sistema completo de micro- computadora. Los componentes dentro del área sombreada en la figura1.5.1. son una parte integral de muchos circuitos integrados microcontroladores. Además de la CPU, el microcontrolador incluye: RAM, ROM, interfaz serie, interfaz paralelo, timer, y circuitos con un programa de interrupciones, todo en el mismo circuito integrado. Por supuesto, la cantidad de RAM en el chip o pastilla no se aproxima a la que incluye el más modesto sistema de microcomputadora, pero como aprenderemos después, esto no es una limitante, dado que los microcontroladores son intentados para aplicaciones bastante diferentes.

Una importante característica de los microcontroladores es el sistema de interrupciones interconstruido. Como dispositivos orientados al control, los

microcontroladores son generalmente llamados a responder a estímulos externos en tiempo real. Ellos deben realizar rápidas interrupciones, suspendiendo un proceso y ejecutando otro en respuesta a un "evento".

El abrir la puerta de un horno de microondas, es un ejemplo de un evento que debe causar una interrupción en un producto basado en un microcontrolador. Por supuesto, muchos microcontroladores pueden también implementar potentes esquemas de interrupción, pero componentes externos son requeridos. Los circuitos en el microcontrolador incluyen todos las circuitos necesarios para manejar interrupciones.

Aplicaciones.

Los microprocesadores son comúnmente usados como CPU en los sistemas de micro computadoras, para eso fueron diseñados y es donde su mayor potencial descansa. Los microcontroladores por otro lado, son encontrados en pequeños diseños, donde mínima cantidad de componentes realizan actividades orientadas a control. Estos diseños fueron implementados en el pasado con docenas o cientos de circuitos integrados digitales, y ahora, un microcontrolador ayuda en reducir la cantidad total de componentes. Todo lo que es requerido es un microcontrolador, un pequeño número de componentes de soporte y un programa de control en ROM. Los microcontroladores son adecuados para control de dispositivos I/O (entrada/salida) en diseños donde son requeridos una mínima cantidad de componentes, mientras que los microprocesadores son adecuados para procesamiento de información en sistemas de computadoras.

El Conjunto de Instrucciones.

Un Microcontrolador esta previsto para control de entradas y salidas. Las interfaces para muchas entradas y salidas usan un solo bit. Por ejemplo, un motor puede ser encendido o apagado por un solenoide energizado por 1 bit de un puerto de salida. Los microcontroladores tienen instrucciones para activar o desactivar bits individuales y realiza operaciones orientadas a un solo bit como: AND, OR, OR-exclusivo, y realizar saltos cuando un bit es activado o desactivado. Esta potente característica es raramente presente en microprocesadores, los cuales son usualmente diseñados para operar en bytes o con grandes unidades de datos.

En el control y monitoreo de dispositivos quizá con interfaces de 1 bit, los microcontroladores tienen inter-construidos circuitos e instrucciones para operaciones de entrada/salida, sincronización de eventos y habilitación y fijación de niveles de prioridad para interrupciones causadas por estímulos

externos. Los microprocesador generalmente requieren circuitos adicionales (circuitos integrados para interfaz serie, controladores de interrupción, timers, etc.) para realizar operaciones similares. No obstante, la capacidad de procesamiento completo de un microcontrolador nunca se aproxima a la de un microprocesador, dado que la mayor parte del estado real de los circuitos integrados es consumido por las funciones en el chip (pastilla), en detrimento del potencial de procesamiento

Dado que el estado real en el chip es apremiante en los microcontroladores, las instrucciones deben ser extremadamente compactas, con la mayoría de ellas implementadas en un solo byte. Un criterio general de diseño es que el programa de control debe ser fijado en la ROM incluida en el chip, dado que el agregar alguna ROM externa aumentaría mucho el costo del producto final. Un esquema codificado compacto para el conjunto de instrucciones es esencial. Esto es raramente una característica de los microprocesadores; sus potentes modos de direccionamiento traen con ellos instrucciones codificadas menos compactas.

1.6 CPU UNIDAD DE PROCESAMIENTO CENTRAL.

La **CPU** (Central Procesing Unit) o Unidad de Procesamiento Central es el "cerebro" de la computadora, administra todas las actividades en el sistema y realiza todas las operaciones con los datos. La CPU es solamente una colección de circuitos lógicos que realizan en forma continua solo dos operaciones: reconocer instrucciones (fetch instructions) y ejecutar instrucciones (executing).

El **Conjunto de Instrucciones** es un conjunto de códigos binarios que el CPU (Inherentemente en su diseño) es capaz de entender y ejecutar. Estas instrucciones normalmente se agrupan en:

1. Instrucciones Aritméticas (suma, resta, multiplicación, división).
2. Instrucciones Lógicas (AND, OR, NOT, etc.).
3. Instrucciones de Transferencia de datos (Cargar, almacenar, transferir, etc.)
4. Instrucciones de Salto y bifurcación (Saltos Condicionales e Incondicionales.
5. Instrucciones de llamada y retorno de subrutinas.
6. Instrucciones Misceláneas (Interrupciones, no operación, espera, etc.)

La figura 1.6.1 es una vista simplificada del interior de una CPU. En esta figura se pueden observar:

1. **Registros** para almacenar temporalmente la información.
2. **ALU=** Arithmetic and Logic Unit (Unidad Aritmética-Lógica) para realizar operaciones sobre la información.
3. **Decodificador de Instrucciones y Unidad de Control** que determina la operación a realizar y el conjunto de movimientos y acciones necesarias para efectuarlas.

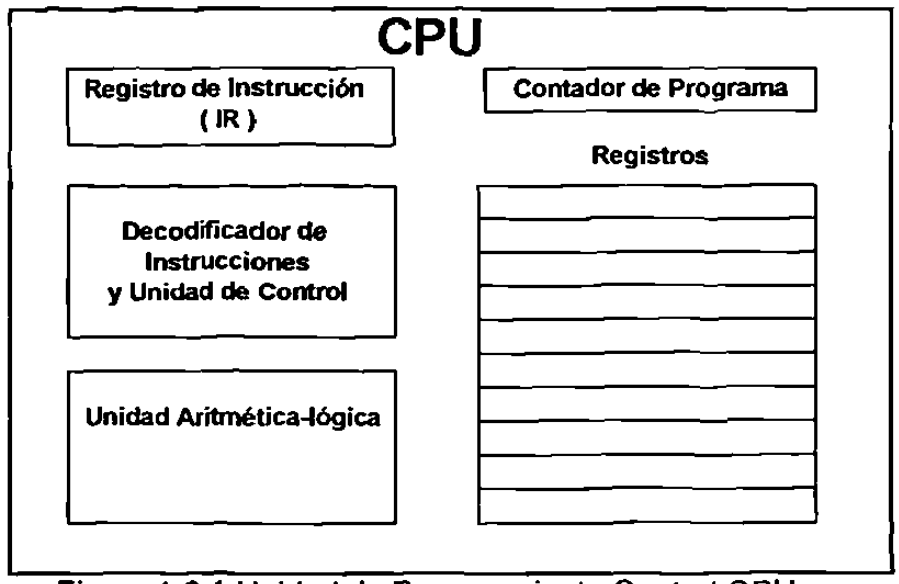


Figura 1.6.1 Unidad de Procesamiento Central CPU

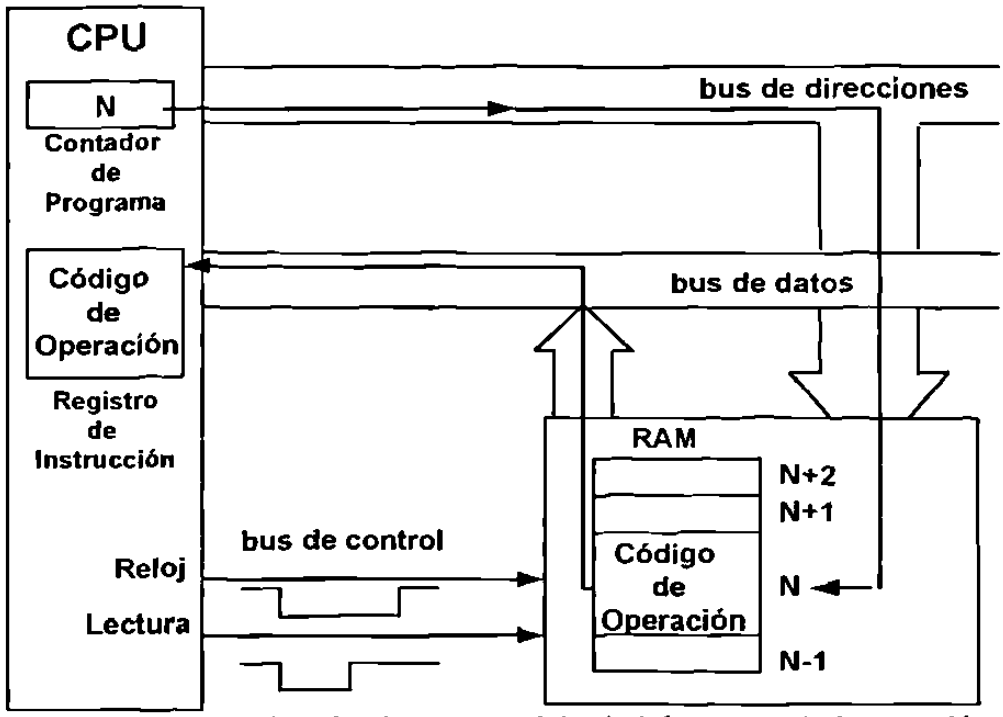


Figura 1.6.2. Actividad de los buses en ciclo de búsqueda de instrucción.

La figura 1.6.2 ilustra el flujo de información para una búsqueda de instrucción. El **buscar una instrucción** (fetch instruction) desde la memoria RAM involucra los siguientes pasos.

a. El contenido del contador de programa es colocado en el bus de direcciones.
b. Una señal de control de lectura es activada.
c. Un dato o código de operación de la instrucción es leído de la RAM y es colocado en el Bus de datos.
d. El código de operación es fijado en el Registro de Instrucciones interno del CPU
e. El Contador de Programa es incrementado para preparar el siguiente reconocimiento de instrucción desde la memoria.

La Ejecución de una instrucción involucra :

a. Decodificar es decir descifrar el código de operación.
b. Generar señales de control.
c. Activar registros internos, entradas y salidas de la ALU (unidad aritmética-lógica).
d. Señalar a la ALU que realice una operación específica.

El **Software** o programa es una serie de instrucciones combinadas para realizar un trabajo importante. El grado en que el trabajo es eficiente y correctamente desarrollado es determinado en su mayor parte por la calidad del software y no tanto en lo elaborado del CPU. Los programas "manejan" al CPU y si hacen algo impropio, entonces imitan la fragilidad de los autores. La frase "la computadora hizo un error" es equivocada. Aunque las fallas del equipo son inevitables, los errores son generalmente un signo de malos programas y errores del operador.

1.7 MEMORIAS.

Los programas y los datos son almacenados en memoria. Las variedades de memorias para computadora son bastas, la terminología que las acompaña es abundante y la tecnología toma rumbos distintos frecuentemente, tal que extensos y continuos estudios son requeridos para mantenerse al tanto de los últimos desarrollos.

La **memorias** son circuitos capaces de almacenar grandes cantidades de información mediante un determinado número de registros que utilizan entradas y salidas comunes para el acceso a todos ellos. Cada uno de estos registros de la memoria recibe el nombre de **localidad** y generalmente están constituidos por los elementos necesarios para almacenar 1, 4 u 8 bits. Un **bit**

es un dígito binario, esta palabra se forma funcionando **binary digit** y es la mínima cantidad de información que se puede almacenar o manejar.

Para cada localidad existe una dirección concreta expresada en forma numérica. Cuando se desea acceder a una determinada localidad es necesario direccionarla, es decir, activar la dirección correspondiente.

La mayoría de los sistemas digitales basados en microprocesadores o microcontroladores actualmente operan con paquetes de 8 bits y las memorias que se utilizan en la actualidad suelen estar constituidas por localidades direccionables que almacenan 8 bits. El conjunto de 8 bits se conoce con el nombre de **byte**. Un **nibble** esta formado por 4 bits, por lo tanto un byte tiene dos nibble.

Write o escritura es la operación de almacenar datos en memoria. **Read** o lectura es la operación de obtener el valor previamente almacenado en una localidad de memoria. Tanto la operación de escritura como de lectura en memoria se realiza seleccionando la localidad deseada a través del bus de direcciones y transportando la información que se desea (escribir o que se ha leído) mediante el bus de datos.

1.7.1 SISTEMAS DE NUMERACION BINARIO Y HEXADECIMAL.

Una determinada localidad de memoria se selecciona generando una combinación binaria, formada por tantos bits como líneas en el bus de dirección existan. Lo más normal es que un sistema digital programable necesite un circuito de memoria de gran capacidad; como consecuencia el número de líneas de direccionamiento será elevado. Utilizar largas expresiones matemáticas formadas por unos y ceros para designar cada localidad, resulta difícil, sobre todo cuando es necesario escribir el programa que hace funcionar al sistema. Los datos almacenados en memoria o aquellos con los que opera el sistema suelen ser de generalmente de 8 o 16 bits, y como consecuencia presentan el mismo problema.

El Sistema de Numeración Hexadecimal se utiliza para facilitar la tarea de programación y representación de localidades y datos. Un mismo número expresado en dicho sistema está formado por la cuarta parte de cifras de las que tendría expresado en binario.

El sistema de numeración hexadecimal (en base 16) utiliza 16 signos diferentes. En la figura 1.7.1 se muestran dichos signos y su equivalencia con el sistema decimal (en base 10) y con el sistema binario (en base 2)

Decimal base 10	Hexadecimal base 16	Binario base 2
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Figura 1.7.1 Equivalencia entre los sistemas de numeración decimal, hexadecimal y binario.

Las transformaciones entre números binarios y hexadecimales son muy sencillas. Cada cuatro bits o cifras binarias pueden ser representadas por un solo signo hexadecimal. Para pasar de binario a hexadecimal, los bits del número binario se agrupan en bloques de cuatro bits comenzando por la derecha. Si el último grupo resulta tener menos de cuatro bits, se podrá completar añadiendo ceros a la izquierda y cada grupo se sustituye por el signo hexadecimal correspondiente.

Los números binarios de cuatro cifras son fácilmente identificables con su equivalente en base 10 y estos últimos se relacionan de manera sencilla con los hexadecimales. Por lo tanto, se trata de realizar un proceso mental que consiste en pasar cada bloque binario a decimal y a continuación éste a base 16.

El siguiente ejemplo disipará cualquier duda: Pasar el número binario 1.1010.1110.0011 a hexadecimal:

0001	1010	1110	0011	binario
1	10	14	3	decimal
1	A	E	3	hexadecimal

Para pasar de hexadecimal a binario el proceso es el inverso al seguido en el ejemplo anterior.

Para obtener el equivalente decimal apartir del número hexadecimal, se sigue el siguiente procedimiento.

1 A E 3

$$3 \times 16^0 = 3 \times 1 = 3$$
$$E \times 16^1 = 14 \times 16 = 224$$
$$A \times 16^2 = 10 \times 16 \times 16 = 2560$$
$$1 \times 16^3 = 1 \times 16 \times 16 \times 16 = 4096$$
$$= 6883$$

Las expresiones que utilizaremos en este y en los siguientes capítulos para denominar direcciones de localidades de memoria estarán comprendidas entre los números hexadecimales 0000 y FFFF. Aunque evidentemente, existen sistemas digitales. que necesitan números de más cifras para direccionar todas las localidades que constituyen su memoria.

Los microprocesadores o los microcontroladores utilizan comúnmente datos de 8 bits y para su representación en hexadecimal se utilizan números comprendidos entre el 00 y el FF. Para indicar que un número esta expresando en el sistema hexadecimal colocaremos $ antes del mismo, por ejemplo: $34.

1.7.2 CLASES DE MEMORIAS.

Son muy abundantes los tipos o clases de memorias disponibles en catálogo. Aquí serán referidos exclusivamente aquellos que están fabricados con materiales semiconductores y más concretamente los circuitos integrados **LSI** (Large Scale Integration) de gran escala de integración. También se hace referencia a dispositivos **MSI** (Medium Scale Intregration) de mediana escala de integración que tienen capacidad de almacenar información, como los registros de desplazamiento o corrimiento.

En la figura 1.7.2.1 se muestra una clasificación de las memorias atendiendo en este caso el siguiente orden: modo de acceso, forma de almacenamiento y tecnología de fabricación.

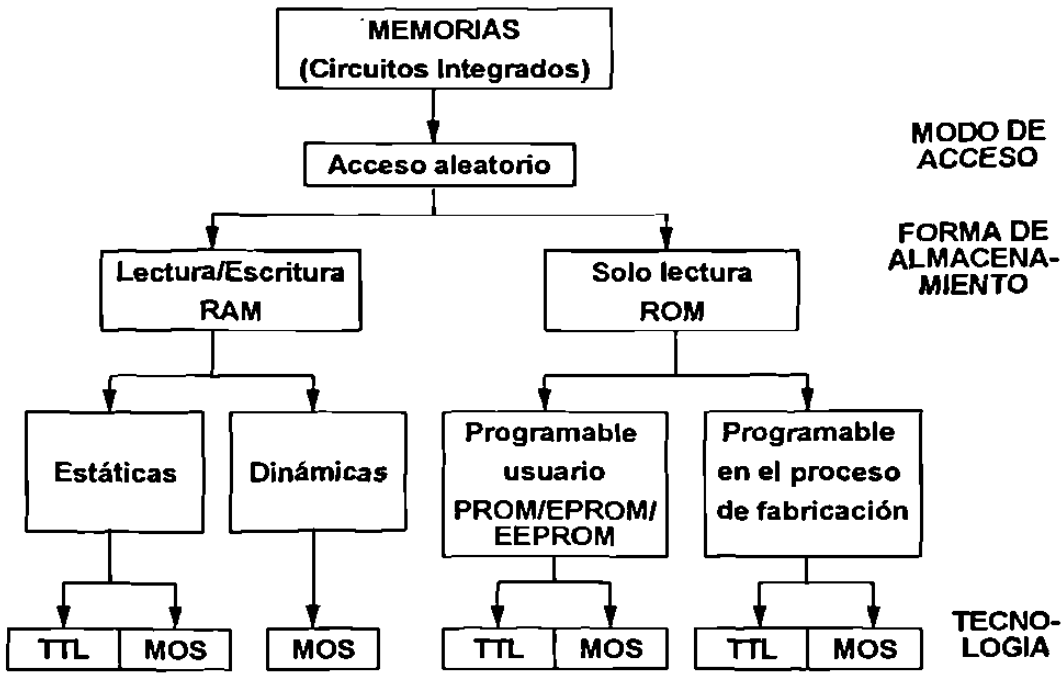


Figura 1.7.2.1 Clases de Memoria de Acceso Aleatorio

Aunque existe una gran variedad, las de mayor uso son las denominadas de acceso aleatorio es decir las **RAM** y **ROM**. En estos dispositivos se emplea el mismo tiempo para acceder a cada una de las localidades, sea cual sea su ubicación.

Las **RAM** (Random Access Memory) son memorias de acceso aleatorio en las que es posible la escritura y lectura de datos. Estas memorias son volátiles, es decir, pierden su información cuando se desconectan de la tensión de alimentación. Se utilizan generalmente para guardar resultados intermedios o finales del proceso y también para almacenar programas procedentes de dispositivos externos tales como discos. Son conocidas también como memorias de trabajo.

Las memorias RAM se dividen en estáticas y dinámicas. Las estáticas mantienen la información siempre y cuando se mantenga la alimentación eléctrica. Las dinámicas, de tecnología MOS exclusivamente, tienen un mayor número de registros por circuitos que las anteriores, y necesitan una señal de refresco controlada por el programa principal para poder mantener la información.

Las **ROM** (Read Only Memory) son memorias (de acceso aleatorio) de solo lectura, estas son no volátiles, es decir, mantienen la información aunque se desconecten de la alimentación eléctrica. Este dispositivo se utiliza para almacenar el programa de control del sistema.

Las memorias ROM de solo lectura se dividen a su vez en:

- **ROM**: Memorias programadas por máscara en el proceso de fabricación.
- **PROM:** Memorias programadas por el usuario en el momento de diseñar el sistema. Solo pueden ser programadas una vez.
- **EPROM**: Memorias de Solo Lectura Reprogramables, esta memoria pueden ser borrada con luz ultravioleta para ser reprogramada con nuevos patrones de bits.
- **EEPROM**: (Electrical Erasable Programmable ROM). Esta memoria puede ser borrada y programada electricamente.

1.8 BUSES DE DIRECCION, DATOS Y CONTROL.

Un **bus** es un conjunto de alambres que transportan información con un propósito común. El acceso a los circuitos de control alrededor del CPU es proporcionado por tres buses: El bus de direcciones, el bus de datos y el bus de control.

El **bus de direcciones** es utilizado para operaciones de lectura y escritura, el CPU especifica la localidad del dato (o instrucción) colocando una dirección en el bus de direcciones y entonces activa una señal en el bus de control indicando si la operación es de lectura o escritura.

La **Operación de Lectura** recupera un byte de dato de una localidad especifica en la memoria y la coloca en el bus de datos. El CPU lee el dato y lo coloca en uno de sus registros internos.

En la **Operación Escritura**, el CPU saca un dato y lo coloca en el bus de datos, debido a la señal de control, la memoria reconoce la operación como un ciclo de escritura y almacena el dato en la localidad especificada.

Dadas **n** líneas de dirección, cada una con la posibilidad de ser uno (1) o cero (0), 2^n localidades pueden ser direccionadas.

Un bus de direcciones de 16 líneas puede direccionar 2^{16}=65,536 localidades. Un bus de direcciones de 20 líneas pueden direccionar 2^{20}=1,048,576 localidades. La abreviación K (por kilo) es estandarizada para 2^{10}=1024, y por lo tanto 16 líneas pueden direccionar $2^6 x 2^{10}$=64k localidades, mientras que 20 líneas pueden direccionar $2^{10} x 2^{10}$=1024K (1Mega) localidades.

El **bus de datos** transporta información entre el CPU y la memoria o entre el CPU y dispositivos I/O (Entrada/Salida). Un intenso esfuerzo de investigación es utilizado en acortar el tiempo de ejecución de las actividades que realiza una computadora. Las computadoras utilizan dos terceras partes de su tiempo simplemente en mover datos. La mayoría de las operaciones de movimiento son entre los registros de CPU y las memorias RAM y ROM, por lo que el número de líneas o ancho del bus de datos es importante para el funcionamiento total. Esta limitación es un cuello de botella. Puede haber una vasta cantidad de memoria en el sistema, y el CPU puede poseer una tremenda capacidad de cómputo, pero el acceso de datos o el movimiento de datos entre la memoria y el CPU vía el bus de datos es el cuello de botella debido al ancho del bus de datos.

Esta característica es tan importante que es común agregar un prefijo que indica el cuello de botella ,es decir, el ancho del bus de datos. La frase "Computadora de 16 bits" se refiere a una computadora con 16 líneas en su bus de datos. Muchas computadoras caen en la clasificación de 4, 8, 16 o 32 bit, la capacidad de cómputo se incrementa conforme el ancho o número de líneas del bus de datos se incrementa.

En la figura 1.2.1 se muestra que el bus de datos es bidireccional, mientras que el bus de direcciones es unidireccional. La información de dirección siempre es proporcionada por el CPU, como es indicado por el sentido de la flecha de la figura 1.2.1., mientras que el dato puede viajar en ambas direcciones dependiendo si la operación es de lectura o escritura. Nota: el término dato es utilizado en sentido general, la información puede ser un dato o una instrucción usada por el programa.

El **bus de control** es una mezcla de señales, cada una con un rol específico en el control de la actividad del sistema. Como regla, las señales de control son señales de sincronización proporcionadas por el CPU hacia los circuitos externos para sincronizar el movimiento de información entre los buses de dirección y de datos.

Para el movimiento de datos entre el CPU y la memoria generalmente se consideran tres tipos de señales tales como: CLOCK (reloj), READ (lectura) y WRITE (escritura). Los nombres y operación de estas señales son altamente dependientes del CPU específico. Las hojas de datos del fabricante deben ser consultados para mayores detalles.

1.9 PUERTOS I/O

Un **PUERTO I/O** es un conjunto de terminales utilizadas para entrada (Input) o salida (Output) de datos en un microcontrolador. Es una práctica generalizada que los puertos estén formados por terminales que realizan funciones semejantes. Entre estas funciones se encuentran:

1. Entradas y/o salidas de propósito general
2. Operación con el sistema del timer (Temporizador)
3. Comunicación paralelo con Strobe (Control) simple
4. Comunicación paralelo con protocolo total (Full handshake)
5. Comunicación serie
6. Conversión análogo digital

Dependiendo de lo elaborado del microcontrolador, es posible que los puertos puedan realizar una o varias de las funciones mencionadas, lo cuál depende de diversos registros de control.

CAPITULO 2

EL MICROCONTROLADOR COMO CIRCUITO INTEGRADO.

2.1 INTRODUCCION.

Los microcontroladores son circuitos integrados LSI (Large Scale Integration) altamente desarrollados, cuyo diseño esta enfocado hacia actividades de control. En este capítulo serán descritos los subsistemas o bloques que comúnmente se encuentran dentro del chip o pastilla de un microcontrolador, además serán descritas las funciones que realiza cada una de sus terminales, y se hará una descripción de la organización de la memoria en el Modo de Operación Cargador.

Se utilizan a partir de este capítulo como prototipo, a los microcontroladores de la familia M68HC11 fabricados por Motorola, debido a que su conjunto de instrucciones es capaz de ejecutar todas las instrucciones de las familias M6800 y M6801 más 91 nuevos códigos de operación. Esto facilita entender el uso de microprocesadores fabricados con anterioridad por Motorola. No se ha utilizado para la descripción microcontroladores fabricados por INTEL, porque generalmente cada uno de sus dispositivos microcontroladores o microprocesadores, utilizan diferentes conjuntos de instrucciones, si bien cabe aclarar que sus caracteristicas y manera de operar son bastante semejantes.

2.2 CARACTERISTICAS GENERALES.

Las siguientes son algunas características relevantes de los microcontroladores de la familia M68HC11:

Características del Hardware:

- 12K bytes de ROM.
- 512 bytes de EEPROM.
- 512 bytes de RAM relocalizables.

- Sistema de Timer de 16 bits con: Preescalador programable de 4 etapas y la posibilidad de: a).- tres funciones de capturas de entrada/cinco de salidas comparadas o, b).- cuatro funciones de captura de entrada/cuatro de salidas comparadas seleccionables.
- Circuito Acumulador de pulsos de 8 bits.
- Interfaz de Comunicación Serie NRZ (No Return to Zero).
- Interfaz Periférico Serie.
- Convertidor Análogo-Digital de 8 bit, 8 canales.
- Circuito de interrupción de Tiempo Real.
- Sistema de Vigilancia de Operación Apropiada de Cómputo.
- Disponible en diferentes tipos de encapsulado.

Características del Software:

- Conjunto de instrucciones mejoradas respecto a M6800/M6801.
- Enteros 16 x 16 y características de División Fraccional.
- Manipulación de bit.
- Modo de Operación de ESPERA (WAIT).
- Modo de Operación de ALTO (STOP).

La familia de microcontroladores M68HC11 esta compuesta de varios miembros, en la figura 2.1 se explica como los números de parte de estos productos es construido.

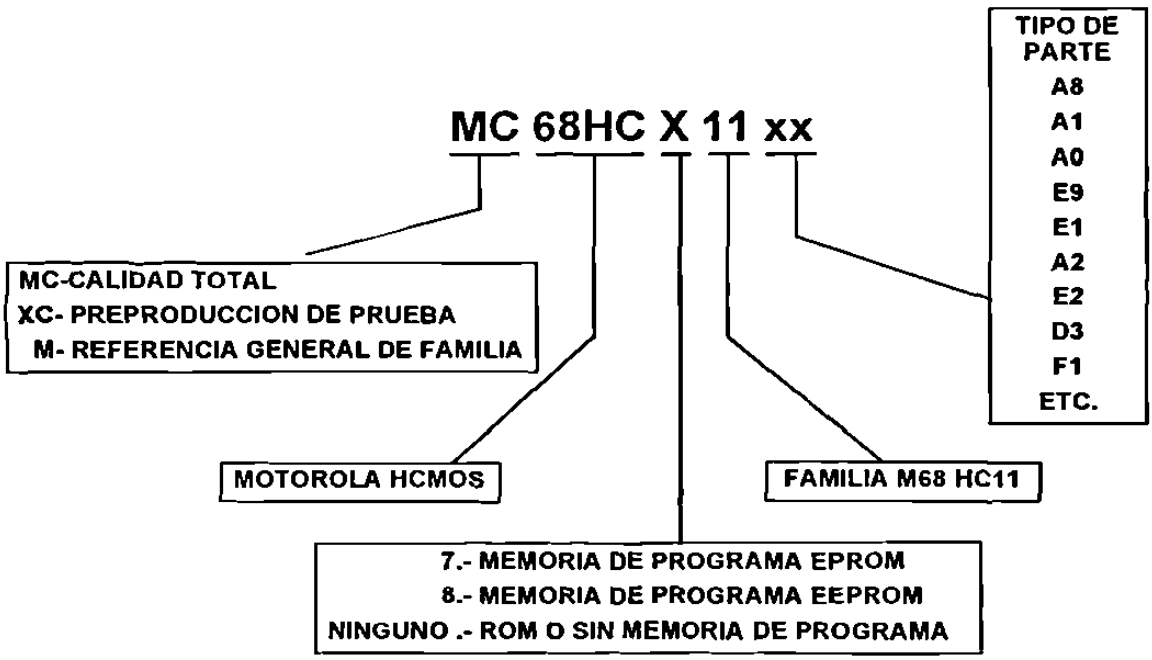


Figura 2.1 Números de parte

Los miembros de la familia M68HC11 difieren principalmente en el tipo y cantidad de memoria. En la tabla 2.1. se resumen las características. Note que los mayores cambios los tienen las variaciones x8, x9, x1, x0. Estas variaciones usan idéntico chip. El registro de configuración (CONFIG) es implementado con celdas EEPROM y es usado para semipermanentemente desactivar la ROM en la variación x1. La ROM y la EEPROM es desactivada en la variación x0.

Tabla 2.1 Microcontroladores de la Familia MC68HC11

No. Parte	EPROM	ROM	EEPROM	RAM	CONFIG 2	COMENTARIOS
MC68HC11A8	---	8K	512	256	$0F	Familia construida alrededor de este dispositivo.
MC68HC11A1	---	---	512	256	$0D	'A8 con ROM deshabilitada
MC68HC11A0	---	---	---	256	$0C	'A8 con ROM y EEPROM deshabilitada
MC68HC811A2	---	---	2K[1]	256	$FF	Sin ROM para sistema expandido
MC68HC811A8	---	---	8K+512	256	$0F	Emulador de EEPROM para A8
MC68HC11E9	---	12K	512	512	$0F	Cuatro entradas de captura
MC68HC11E1	---	---	512	512	$0D	E9 con ROM deshabilitada
MC68HC11E0	---	---	---	512	$0C	E9 con ROM y EEPROM deshabilitado
MC68HC811E2	---	---	2K[1]	256	$FF	Igual que A2 con Timer E9
MC68HC711E9	12K	---	512	512	$0F	
MC68HC11D3	---	4K	---	192	N/A	Versión de 40 terminales de bajo costo
MC68HC711D3	4K	---	---	192	N/A	Versión de D3 con un timer programado
MC68HC11F1	---	---	512[1]	1K	$FF	Alta funcionalidad

Notas::

1. La EEPROM es relocalizable en la parte alta de cualquier página de un bloque de memoria de 4K. La relocalización es hecha con los 4 bits superiores del registro CONFIG.
2. Los valores del registro CONFIG en esta tabla, reflejan el valor programado antes de ser enviado por Motorola

2.3 DESCRIPCION GENERAL DEL HARDWARE.

La tecnología CMOS de alta densidad (HCMOS = High-density Complementary Metal Oxide Semiconductor) es utilizada para producir microcontroladores avanzados altamente sofisticados y con capacidad periférica incluida en el chip, nuevas técnicas combinan pequeño tamaño y alta velocidad con bajo consumo de potencia e inmunidad al ruido. En la figura 2.3.1 se muestra un diagrama de bloques que muestran los subsistemas mayores que a continuación se describen.

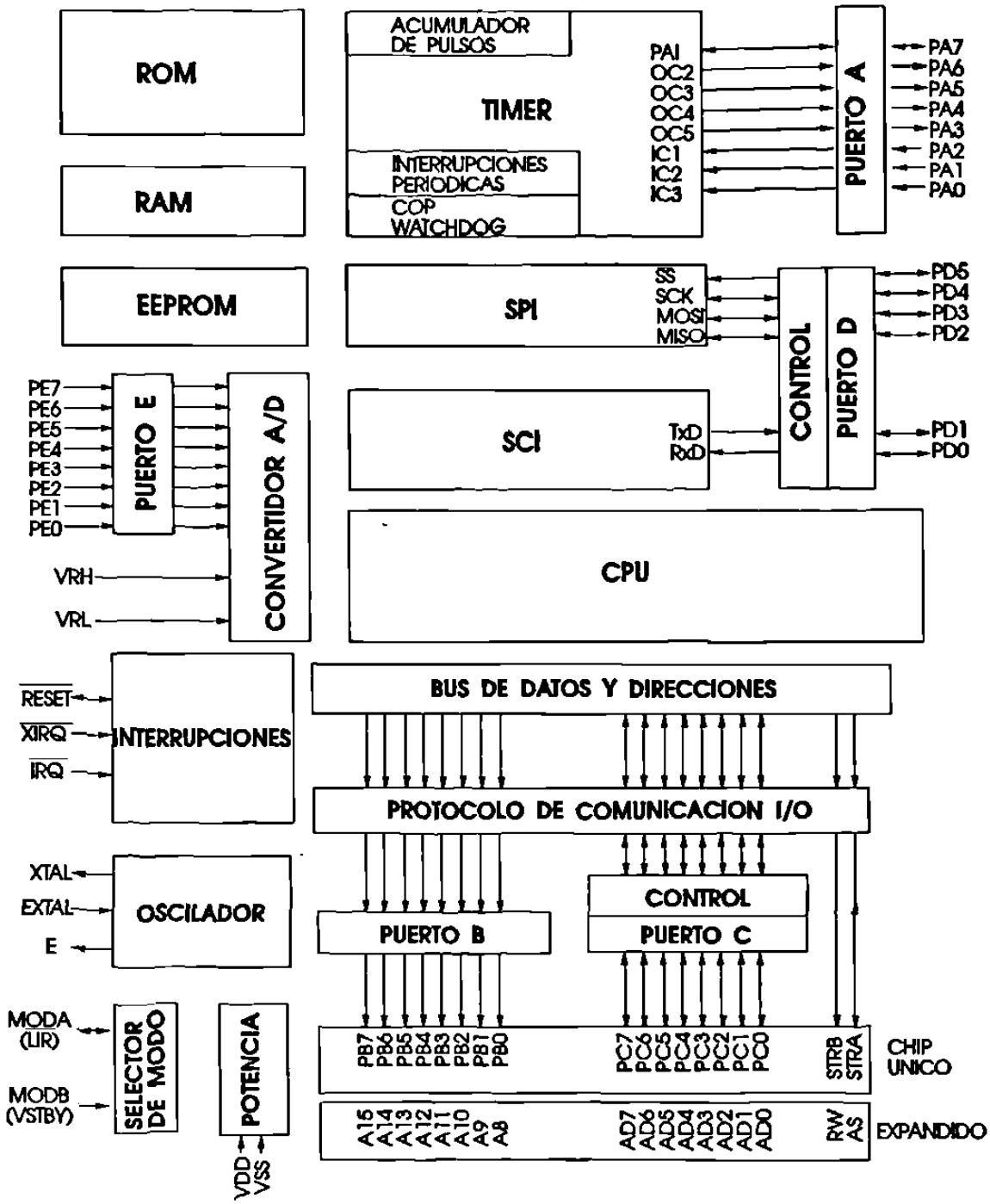


Figura 2.3.1. Diagrama de Bloques.

Memoria ROM.

La memoria ROM es programada por máscara en la fábrica y puede incluir hasta 12k de memoria. Es utilizada para almacenar permanentemente el programa del usuario, y contiene 192 bytes del Programa Cargador (bootstrap) para el Modo de Operación Cargador (Special Bootstrap).

Memoria RAM.

Son disponibles hasta 521 bytes de memoria RAM de celdas estáticas y pueden ser relocalizadas al inicio de cualquier bloque de memoria de 4k. Las aplicaciones de la memoria RAM son muy diversas y generalmente se utiliza para guardar resultados intermedios del proceso, o sea, como memoria de trabajo.

Memoria EEPROM.

Hasta 2k bytes de memoria EEPROM son disponibles en algunos miembros de la familia de microcontroladores M68HC11. La memoria EEPROM puede ser utilizada en la misma forma que la memoria ROM, pero algunas interesantes posibilidades se alcanzan que no son posibles con memorias RAM y ROM. Una vez que la información es programada en la memoria EEPROM incluida en el chip , esta permanece sin cambio aunque la fuente de energía sea removida indefinidamente. A diferencia de la información en ROM, la información en la EEPROM puede ser borrada y reprogramada bajo del control de las instrucciones del software.

Convertidor Análogo-Digital.

Un convertidor Análogo-Digital de ocho canales con una resolución de ocho bits es incluido. El sistema de conversión usa la técnica de redistribución de carga totalmente capacitivo. Dos líneas VRL y VRH son proporcionadas para las entradas de voltaje de referencia, lo cual permite aceptar entradas análogas en el rango de VRL a VRH. Pequeños rangos de entradas análogas pueden ser obtenidos ajustando VRL y VRH a los deseados límites superior e inferior.

Interfaz de Comunicación Serie.

El microcontrolador contiene una Interfaz de Comunicación Serie Asíncrono SCI (Serial Communications Interface) que funciona con un formato standard NRZ (no retorno a cero) con un bit de arranque, ocho o nueve bits de datos y un bit de stop. Una variedad de velocidades de transmisión o baud rate son disponibles, esto permite mediante esta interfaz, que se hagan conexiones a terminales de tubos de rayos catódicos, computadoras personales, etc.

Interfaz de Comunicación Serie Síncrono.

Este subsistema también conocido como Interfaz Periférico Serie ó **SPI** (Serie Peripheral Interface), como su nombre implica, es primordialmente usado para permitir que el microcontrolador se comunique con dispositivos periféricos. El SPI es capaz de realizar la comunicación *con otros procesadores* en un sistema maestro múltiple. Los dispositivos periféricos pueden ser tan simples como los registros de corrimiento **TTL** (Transistor Transitor Logic) o tan complejos como subsistemas completos, tales como driver de display, diodos de cristal líquido o subsistemas convertidores análogo ó digital. El sistema *SPI* es lo suficiente flexible como interfaz directo con numerosos productos standard periféricos de varios fabricantes.

Timer (Temporizador).

El sistema Completo del Timer Principal incluye varias secciones:

a. Un contador de 16 bit continuo (free-running) con un prescalador programable de cuatro etapas que es utilizado para cuenta de eventos o para generar retardos de tiempo por software.
b. Un Circuito de Interrupción Periódico Programable también llamado Interruptor de Tiempo Real, el cual es comúnmente usado para dar la pauta para la ejecución de subrutinas.
c. Sistema de Vigilancia de Apropiada Operación de Cómputo (Computer Operating Properly [COP] Watchdog System) el cual es utilizado para detectar errores de procesamiento de software.
d. Acumulador de Pulsos que se usa para: la cuenta de eventos, medición de frecuencias, interruptor de transición de pulsos, etc.
e. Generador de Baud rate para establecer la rapidez de comunicación serie.

Oscilador.

Este subsistema genera las señales de reloj que controlan la operación del *microcontrolador*. Las *señales* de reloj dependerán de la frecuencia del cristal utilizado. Además contiene un sistema de monitoreo de la señal de reloj para resetear el microcontrolador si ésta es perdida o corre demasiado lenta.

Entradas/Salidas.

Este subsistema de comunicación *I/O* (Input/Output) o de *Entrada/Salida* tiene un total de 40 terminales repartidas en 5 puertos y que junto con las líneas de control llamadas strobe, permiten la comunicación (lectura y escritura) con el exterior.

CPU.

La **CPU** (Central Processing Unit) o Unidad de Procesamiento Central, como fue explicado anteriormente, es el "cerebro" que administra todas las actividades en el sistema y realiza todas las operaciones con los datos.

2.4 CONEXIONES O TERMINALES.

En esta sección ampliaremos el conocimiento de la arquitectura del hardware desde una perspectiva externa, es decir, desde sus terminales. Una breve descripción de la función que realiza cada terminal es presentada, dejando para después en los capítulos correspondientes, el hacer una explicación más detallada. Desde esta sección conviene acostumbrarse a utilizar las siglas en inglés que representan: terminales, registros, secciones, elementos, dispositivos, etc. Por ejemplo: **MCU** (Micro Controller Unit) se utilizará en lugar de microcontrolador.

Como prototipo para la descripción de las terminales se ha elegido al microcontrolador MC68HC11A8 y en la *figura* 2.4.1 se muestra la asignación de las terminales para el encapsulado **PLCC** (Plastic Leaded Chip Carrier) de 52 terminales y el encapsulado **DIP** (Dual In-line Package) de 48 terminales. La diferencia es que en el encapsulado DIP de 48 terminales, 4 terminales del Convertidor Análogo-Digital no salen al exterior. A continuación se describen las terminales.

VDD=Entrada de Voltaje y VSS=Tierra.

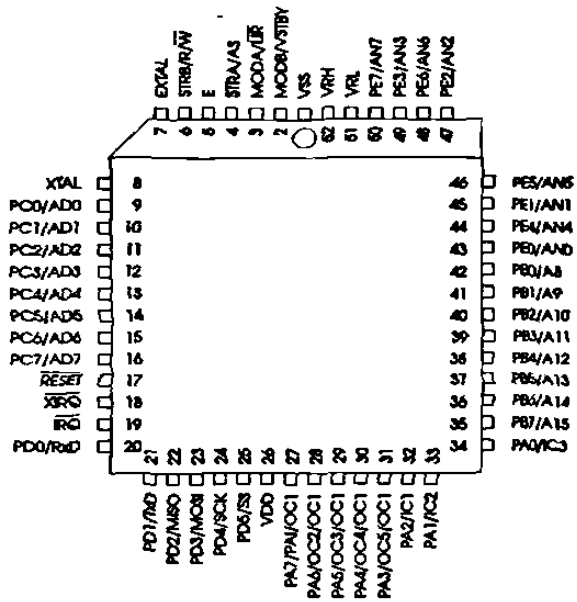
El suministro de Voltaje al MCU es realizado usando estas terminales. VDD es la entrada positiva y VSS es tierra.

RESET.

Esta terminal de control bidireccional activa en baja, es decir, cuando la señal de entrada es cero, es usada como entrada al MCU para inicializarlo en un estado de arranque conocido y actúa como salida de drenador abierto para indicar que una falla interna ha sido detectada, ya sea por el monitor del reloj o por el Sistema de Vigilancia de Operación Apropiada de Cómputo (Computer Operating Properly (**COP**) Watchdog System).

XTAL= Entrada de Cristal y EXTAL= Entrada de Reloj Externo.

Estas dos entradas proporcionan la interfaz para la conexión del cristal o para una señal externa de reloj compatible CMOS y controlan el circuito interno generador de la señal de reloj. La frecuencia aplicada a estas terminales deben ser cuatro veces la **señal de reloj** E deseada.



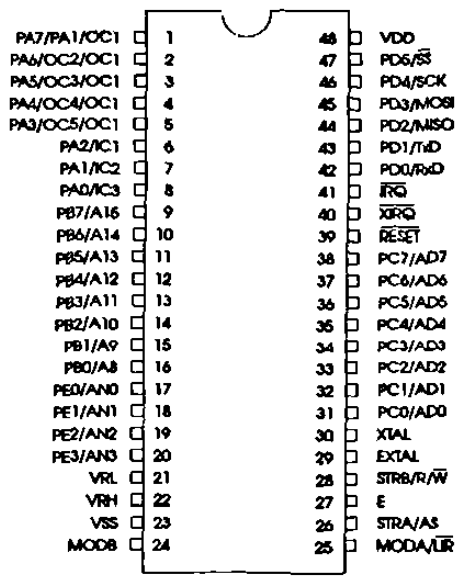


Figura 2.4.1. Asignación de terminales MC68HC11A8

E = Salida de Reloj E.

Esta es la conexión de salida para la señal de reloj E generada internamente, la cual es usada como referencia de tiempo o de sincronización. Cuando la salida de reloj E es baja, un proceso interno se está realizando, y cuando es alta, el dato esta siendo accesado. La señal de reloj E es detenida cuando el MCU esta en modo STOP.

IRQ = Interrup Request

La entrada IRQ o de Requisición de Interrupción proporciona un medio para requerir una interrupción asíncrona para el MCU. La entrada IRQ puede ser programada por el Registro OPTION para que actúe siendo sensible al nivel o siendo sensible a la transición de caída de la señal. Después de reset, IRQ es configurada por default u omisión para operación sensible al nivel.

XIRQ = Non-maskable Interrup Request

La terminal XIRQ o Interrupción No Enmascarable proporciona un medio para requerir interrupciones no enmascarables después de inicialización por reset. Durante reset, el bit X es activado en el Registro del Código de Condición CCR (Condition Code Register), y cualquier interrupción es enmascarada hasta que el software del MCU la habilita. Dado que la entrada XIRQ es sensible al nivel, esta puede ser conectada a una red OR con entradas múltiples con una resistencia de pull-up externa. La terminal XIRQ es generalmente usada como interrupción que detecta pérdida de señal.

MODA/LIR y MODB/VSTBY.

Las terminales MODA/LIR y MODB/VSTBY realizan funciones alternas. Solamente durante el reset, las terminales MODA y MODB son usadas para seleccionar uno de cuatro modos de operación del MCU de acuerdo a la tabla 2.4.1.

Tabla 2.4.1 Selección de Modos de Operación.

MODB	MODA	Modo Seleccionado
0	0	Cargador.
0	1	Prueba.
1	0	Chip Unico.
1	1	Multiplexado Expandido.

En este curso, solamente el Modo de Operación Cargador (Special Bootstrap) es utilizado, pues permite cargar en el MCU los programas del usuario para su verificación, corrección y puesta en operación. En la figura

4.6.4.1. del capítulo 4 se muestra un circuito que puede ser utilizado para operar el MCU en este modo de operación.

Después de que el modo de operación ha sido seleccionado, entonces la terminal MODA asume la función alterna LIR(Load Instruction Register) la cuál proporciona una salida de drenador abierto para indicar que una instrucción esta iniciándose. Todas las instrucciones son realizadas durante una serie de ciclos de reloj E. La señal LIR va hacia abajo durante el primer ciclo de reloj E de cada instrucción (reconocimiento del ciclo de operación). Esta salida se proporciona como una ayuda para la depuración o debug del programa.

Después de que el modo de operación ha sido seleccionado, entonces la terminal MODB asume la función alterna VSTBY que se usa como entrada para potencia de sostenimiento o standby de la RAM, para que su contenido no se pierda cuando VDD no está presente.

VRH-VRL Voltajes de Referencia para el Convertidor Análogo-Digital.

Estas dos terminales proporcional el voltaje de referencia para el circuito Convertidor Análogo-Digital.

STRB - R/W (Strobe B y Read/Write).

En el Modo de Operación Cargador, la terminal STRB (Strobe B) actúa como salida de control programable para Comunicación Paralelo con otros dispositivos externos.

STRA/AS (Strobe A - Address Strobe).

En el Modo de Operación Cargador, esta terminal actúa como entrada de control programable para Comunicación Paralelo con otros dispositivos externos

PUERTO A.

El puerto A puede ser configurado para:

a. Cuatro funciones de captura de entradas (IC1, IC2, IC3, IC4) y tres funciones de salida comparada (OC2, OC3, OC4), y ya sea una para una entrada acumuladora de pulsos o bien otra salida comparada (OC1).

b. Tres funciones de captura de entrada (IC1, IC2, IC3), y cuatro funciones de salida comparada (OC2, OC3, OC4, OC5), y ya sea para una entrada acumuladora de pulsos o bien otra salida comparada (OC1).

Cualquier terminal del Puerto A que no sea usada para su función alterna del TIMER puede ser usada como línea de entrada o salida de propósito general.

PUERTO B.

En el Modo de Operación Cargador (Bootstrap), todas las terminales del Puerto B funcionan como terminales de salida de propósito general. Durante lectura a este puerto, el nivel sensado en el lado de entrada de los drivers de salida del puerto B es leído. El puerto B también puede ser usado en el modo de salida de control (strobe) simple, ocasionando que un pulso de salida aparezca en la terminal STRB cada vez que un dato es escrito en el Puerto B.

PUERTO C.

En el Modo de Operación Cargador, las terminales del Puerto C pueden realizar funciones alternas que incluyen:

a. Entradas/Salidas de Propósito General.
b. Comunicación Paralelo con Control (Strobe) Simple.
c. Comunicación Paralelo con Protocolo Total.

PUERTO D.

Las terminales 0 a 5 del puerto D son usadas como señales I/O (Input/Output = Entrada/Salida) de propósito general. Las terminales del puerto D alternativamente sirven como Interfaz de Comunicación Serie **SCI** (Serial Communications Interface) o como Interfaz Periférica Serie **SPI** (Serial Perifheral Interface) cuando esos subsistemas son habilitados.

La terminal PD0 es la señal receptora de datos **RxD** (Receive Data) para SCI. La terminal PD1 es la señal de salida de transmisión de datos **TxD** (Transmit Data) para SCI. Las terminales PD2 hasta PD5 son dedicadas a SPI. Las terminales PD2 es para la señal **MISO** (Master-In-Slave-Out).

2.5 MODOS DE OPERACION.

Como se mencionó anteriormente, mediante las terminales MODA y MODB se pueden seleccionar cuatro modos de operación para los MCU's de la familia M68HC11, los cuáles son:

a. Cargador (Special Bootstrap)
b. Prueba (Special test)
c. Chip Unico (Single Chip)
d. Multiplexado Expandido (Expanded Multiplexed)

Modo de Operación Cargador (Special Bootstrap)

En este curso solamente este modo de operación es utilizado, pues permite cargar en el MCU los programas del usuario para su verificación, corrección y puesta en operación.

Cuando el MCU es reseteado en el Modo de Operación Cargador, una parte de la ROM de 192 byte que ocupa desde la dirección $BF40 a $BFFF, es habilitada en el chip y el MCU procede a ejecutar el Programa Cargador contenido en esa área. El Programa Cargador que está instalado permanentemente en la ROM inicializa el sistema SCI (Serial Communications Interface), checa opciones de seguridad y acepta que dependiendo del tipo de MCU sean almacenados hasta 512 byte en las localidades $0000 a $01FF. Después de que el byte final es recibido, el control es automáticamente transferido al programa que se inicializa en la localidad $0000.

El Modo de Operación Cargador es muy versátil, dado que esencialmente no hay limitaciones en el tipo de programa del usuario que puede ser cargado en la RAM interna del MCU.

Modo de Operación Prueba (Special Test).

Este modo de operación es primordialmente intentado para pruebas de fábrica . Este modo de prueba no es recomendado para ser utilizado por el usuario debido al reducido sistema de seguridad.

Modo de Operación Chip Unico (Single Chip)

En el Modo de Operación Chip Unico, el Puerto B y el Puerto C funcionan como I/O (Entradas/Salidas) de propósito general o para comunicación paralelo con protocolo de comunicación, donde las terminales strobe A y strobe B proporcionan las señales de control.

Modo de Operación Multiplexado Expandido.

En este modo de operación el MCU tiene capacidad de accesar un espacio de direcciones de 64 k byte. Este espacio total de direcciones incluye la memoria incluida dentro del chip usada en el Modo de Operación Chip Unico más la posibilidad de direccionar localidades para accesar dispositivos periféricos y memoria externa. La expansión del bus de direcciones es realizada utilizando el puerto B y el puerto C y las señales de control AS (Address Strobe) y R/W (Read/Write).

2.6. ORGANIZACION DE LA MEMORIA.

La organización de la memoria para los modos de operación Chip Unico y Cargador para el MC68HC11 es mostrada en la figura 2.6.1. Las localidades de memoria interna son mostradas sombreadas y su contenido es indicado a la derecha. Las localidades marcadas EXT no son utilizadas en estos modos de operación.

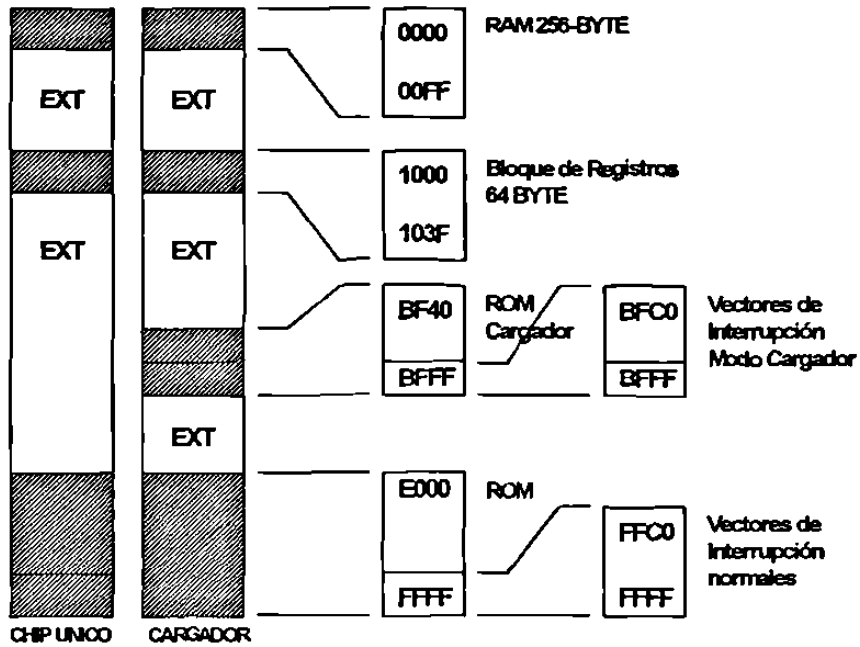


Fig. 2.6.1 MAPA DE MEMORIA EN EL MODO CHIP UNICO Y CARGADOR

En la organización de la memoria de la figura 2.6.1 se observan las siguientes secciones: RAM, EXT, Bloque de Registros, EEPROM, ROM del Programa Cargador y ROM del Programa del Usuario. A continuación se describen estas secciones.

2.6.1. RAM

La RAM interna del microcontrolador MC68HC11A8 consta de 256 byte y después de resetear el MCU empieza en la localidad $0000. Esta memoria es utilizada como memoria de trabajo para almacenar programas y resultados intermedios o finales de un proceso. Está implementada con celdas estáticas que pierden su información cuando se desconecta la tensión VDD. Para mantener la información se requiere una pequeña corriente en la terminal MODB/VSTBY.

2.6.2. EXT

Las localidades marcadas EXT son utilizadas para direccionar memoria externa o dispositivos periféricos externos cuando el MCU está en el Modo de Operación Multiplexado Expandido o de Prueba. Las localidades marcadas EXT **no son utilizadas** en el modo de operación Cargador ni en Chip Unico y direccionar hacia esas localidades puede producir resultados impredecibles

2.6.3. BLOQUE DE REGISTROS.

Los Puertos y Registros de Control en los microcontroladores de la familia M68HC11 están caracterizados por estar localizados en el mapa de memoria y pueden ser direccionados de la misma forma que cualquier localidad de memoria. El Bloque de Registros está constituido con los puertos y los Registros de Control y forma un bloque de 64 localidades de memoria que ocupa desde la dirección $1000 hasta la $103F después de resetear el MCU en el Modo de Operación Cargador. En la tabla 5.2.1 del capítulo 5 se muestra la ubicación de los Puertos y los Registros de Control.

2.6.4. EEPROM.

El MC68HC11A8 incluye en su interior 512 byte de memoria EEPROM del tipo CMOS, y ocupa de la dirección $B600 hasta la $B7FF. Esta memoria es usada en la misma forma que la memoria ROM, pero tiene algunas interesantes posibilidades que no son posibles con memorias ROM y RAM. Una vez que la información es programada en la memoria EEPROM del MCU, esta permanece sin cambio aunque la fuente VDD sea removida indefinidamente. Y a diferencia de la memoria ROM, la información en la EEPROM puede ser borrada y reprogramada mediante el control del software. La EEPROM puede ser borrada y reprogramada mediante un circuito interno alimentado por la fuente VDD, por lo que no es necesaria ninguna fuente de energía especial.

2.6.5. ROM.

La ROM es una memoria de solo lectura y en el chip consta de dos partes: A).- La ROM del Programa Cargador . B).- La ROM del Programa del Usuario.

La **ROM del Programa Cargador** consta de 192 byte que ocupan de la dirección $BF40 hasta la $BFFF. Esta ROM contiene el Programa Cargador que controla el proceso de carga en el Modo de Operación Cargador (Special Bootstrap).

La ROM del Programa del Usuario es utilizada primordialmente para contener las instrucciones del programa de aplicación del usuario y no pueden ser cambiadas porque son *programadas* en el MCU cuando es fabricado. El usuario debe desarrollar y depurar sus programas de aplicación antes de ordenar la producción de MCU's. El MC68HC11A8 dispone de 8 Kbyte de ROM ubicados desde la localidad $E000 hasta la $FFFF.

CAPITULO 3

CONJUNTO DE INSTRUCCIONES

3.1. INTRODUCCION

En este capítulo se hace una descripción del Modelo del Programador, es decir, de los registros internos de CPU *que* intervienen en la ejecución de las instrucciones del MCU. También se incluye una descripción *de los* diferentes Modos de Direccionamiento, o sea, la manera en que son localizados los datos para ser manipulados por el CPU.

El Conjunto de Instrucciones completo que pueden ejecutar los MCU's de la familia M68HC11 es presentado en el Apéndice, y no es necesario analizarlo completo para entenderlo *en su totalidad. Por esa razón solo es* examinado el número suficiente de instrucciones con todos los modos de direccionamiento aplicables expresados brevemente, además son listados los códigos de operación para cada modo de direccionamiento. Muchos ejemplos son presentados para ilustrar el contenido de los registros del CPU y las localidades de memoria aplicables antes y después de la ejecución de una instrucción.

3.2. MODELO DEL PROGRAMADOR.

En la fig. 3.2.1. se observa una vista simplificada del interior del CPU. El CPU es "el cerebro" del MCU , administra todas las actividades en el sistema, y además realiza todas las operaciones con los datos.

La CPU es una colección de circuitos lógicos que realizan en forma contínua solo dos operaciones: buscar instrucciones (fetch instructions) y ejecutar instrucciones (executing instructions).

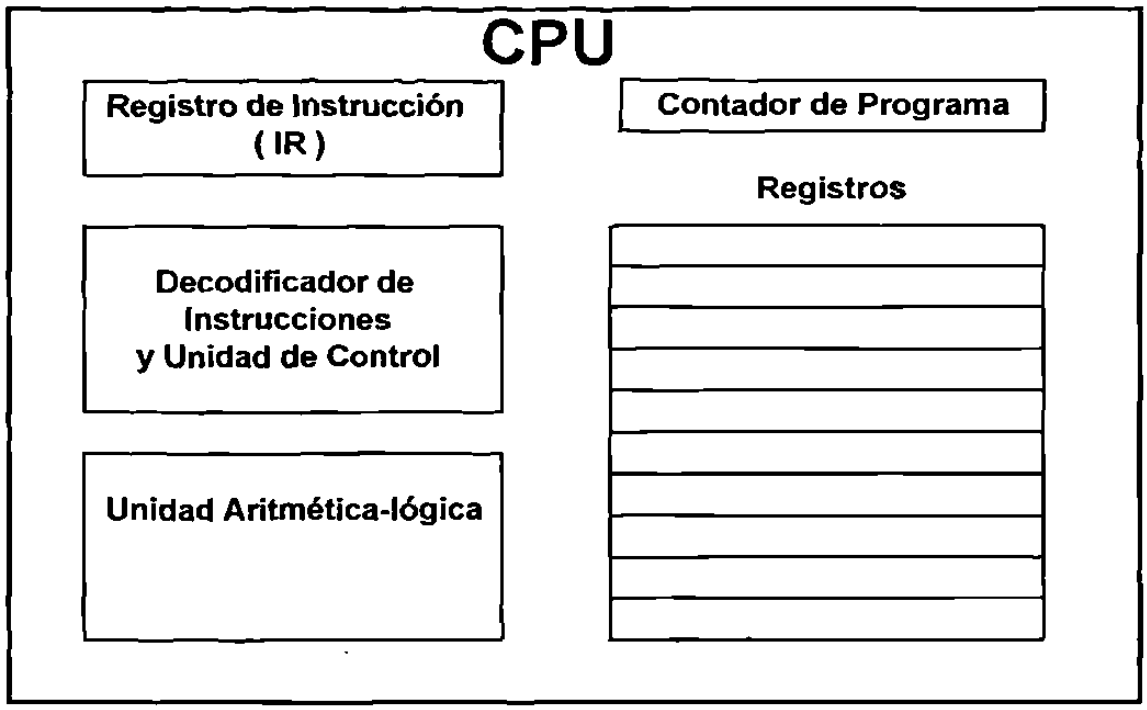


Figura 3.2.1. Modelo del Programador

El **buscar** (fetch) una instrucción desde la RAM involucra los siguientes pasos:

a. El contenido del Contador del Programa **PC** (Program Counter) es colocado en el bus de direcciones.
b. Una señal de control de lectura es activada.
c. Un dato o un código de operación de una instrucción es leído de la RAM y es colocado en el Bus de datos.
d. El Código de Operación **Opcode** (Operation Code) es fijado en el Registro de instrucciones **IR** (Instruction Register).
e. El *Contador de Programa* **PC** (*Program* Counter) es incrementado para preparar la siguiente búsqueda de instrucción desde la memoria.

La **Ejecución** de una instrucción involucra:

a. Decodificar, es decir, descifrar el Código de Operación.
b. Generar señales de Control.
c. Activar registros internos, entradas y salidas de la Unidad Aritmética - Lógica **ALU**.
d. Señalar a la Unidad Aritmética Lógica ALU que realice una operación específica (sumar, restar, etc.)

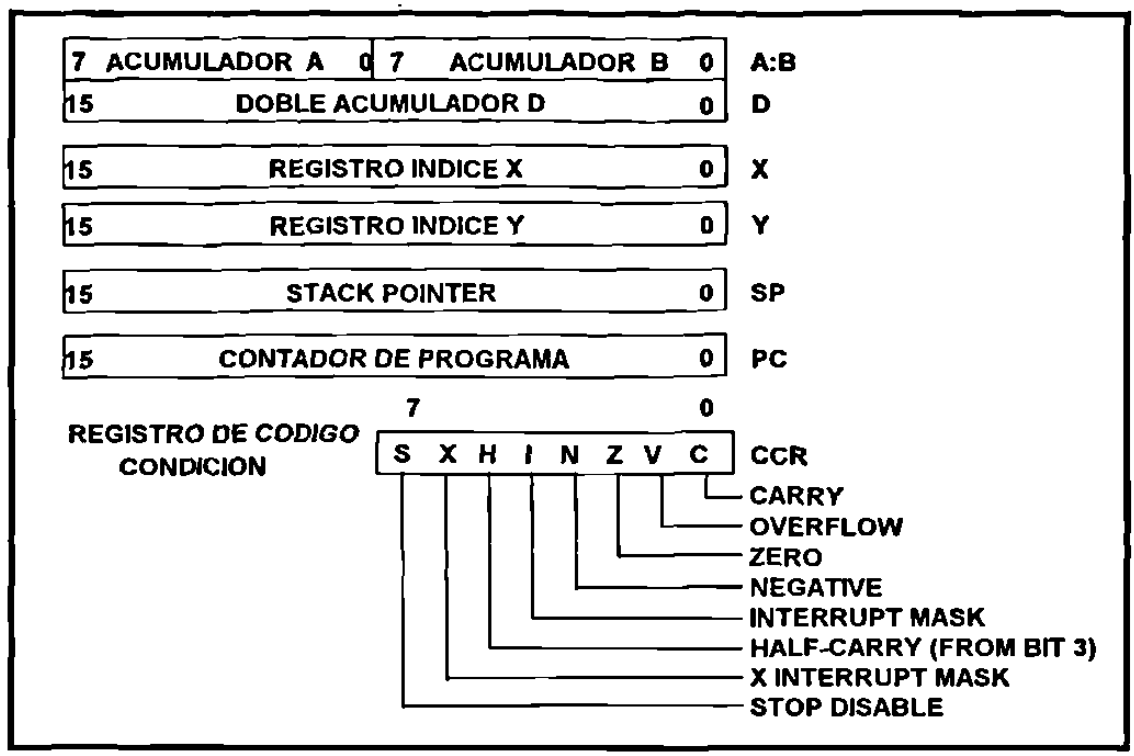


Figura 3.2.2. M68HC11 Modelo del programador

Dada la arquitectura de este CPU, todos los puertos I/O (Entradas/Salidas) y las localidades de memoria son tratados en forma idéntica en el mapa de memoria de 64K bytes. En cambio, los registros del CPU forman parte integral del mismo y no son direccionables, es decir, *no pueden ser* tratados como localidades de memoria. El **Modelo del Programador** es el *nombre dado* al conjunto de registros internos a que tiene acceso el programador. En la figura 3.2.2. se observa el conjunto de registros internos que forman parte del modelo del Programador, los cuales se listan y describen enseguida:

1. Acumulador A, B y D.
2. Registros Indice X e Y.
3. Apuntador de Pila **SP** (Stack Pointer).
4. Contador de Programa **PC** (Program Counter).
5. Registro del Código de Condición **CCR** (Condition Code Register)

Acumuladores A, B y D. Los acumuladores A y B son registros de 8 bits de propósito general usados para mantener los operandos y los resultados de cálculos aritméticos o de manipulación de datos. Los dos acumuladores pueden ser concatenados en un solo acumulador de 16 bit llamado Acumulador D.

Los Registros Indice X e Y son de 16 bits, y son usados para el Modo de Direccionamiento Indexado. Este modo de direccionamiento es usado para el manejo de Tablas.

El Apuntador de Pila SP (Stack Pointer) es un registro de 16 bits que contiene la dirección de la siguiente localidad libre en el stack (pila). El stack es configurado como un registros de lectura/escritura de secuencia **LIFO** (Last-In- First-Out), en este tipo de memoria, el último dato en entrar es el primero en salir. En el Stack Pointer son almacenados datos importantes durante interrupciones o llamadas a subrutina.

Es sumamente importante que el programador en las primeras instrucciones del Programa apunte el Stack Pointer a una área utilizable de la RAM con suficiente espacio.

El Contador de Programa PC (Program Counter) es un registro de 16 bit que apunta siempre a la localidad de la siguiente instrucción que va a ser ejecutada.

El Registro de Código de Condiciones CCR (Condition Code Register) también recibe el nombre de **Registro de Banderas**, pues se les da el nombre de **Banderas** (flags) a los bits contenidos en el registro de condiciones. El registro CCR tiene: 5 indicadores de status (H, N, Z, V, C), 2 bits de interrupciones (I,X) y un bit habilitador de STOP (S). En seguida se describe cada una de las banderas.

S stop disable.- (deshabilitador de STOP).- El bit S es controlado por medio de Software. Cuando es 1 se deshabilita la instrucción STOP, cuando es 0 la instrucción STOP es habilitada.

H half carry.- Este bit de cargo intermedio es activado o puesto en 1, cuando un cargo (carry) ocurre entre el bit 3 y 4 en la Unidad Aritmética Lógica, y es usado para corrección en operaciones aritméticas BCD.

I interrupt mask .- El bit I de esta máscara de interrupción es activado o puesto en 1 por medio de hardware, y por medio de una instrucción de programa se deshabilitan o enmascaran todas las fuentes de interrupción enmascarables.

Z zero.- El bit cero es activado si el resultado de la última operación (aritmética, lógica o al manipular datos) es cero.

C carry/borrow.- El bit C es activado cuando un cargo o préstamo ocurre en una operación aritmética. El bit C también es afectado durante instrucciones de corrimiento o rotación.

3.3. MODOS DE DIRECCIONAMIENTO

Los ***Modos de Direccionamiento*** *son las diferentes* formas en que se especifican el origen o destino de los datos que son manipulados por la CPU. Los MCU's de la familia M68HC11 pueden usar 6 modos de direccionamiento:

1. Inmediato
2. Directo
3. Extendido
4. Indexado
5. Inherente o Implícito
6. Relativo

En los ejemplos de los modos de direccionamiento se aplica lo siguiente:

a. Cuando está presente el símbolo #, indica direccionamiento inmediato.

b. Cuando no está presente el símbolo #, el número siguiente al código de operación es una dirección, excepto en el modo de direccionamiento Inherente.

c. El símbolo **$** indica que el número que le sigue es hexadecimal.

En el **Modo de Direccionamiento inmediato**, la instrucción contiene los datos en el byte o bytes que están inmediatamente enseguida del código de operación.

Ejemplo del Modo de Direccionamiento Inmediato:

LDAA #$25 cargar el acumulador A, con el dato 25 en hexadecimal que se ubica inmediatamente en la siguiente localidad de memoria

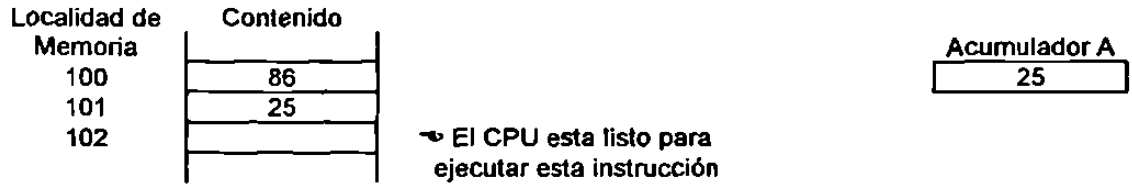
Antes de ejecutar la instrucción: LDAA #$25

Localidad de Memoria	Contenido
100	86
101	25
102	

El CPU esta listo para ejecutar esta instrucción

Acumulador A
00

Después de ejecutar la instrucción: LDAA #$25

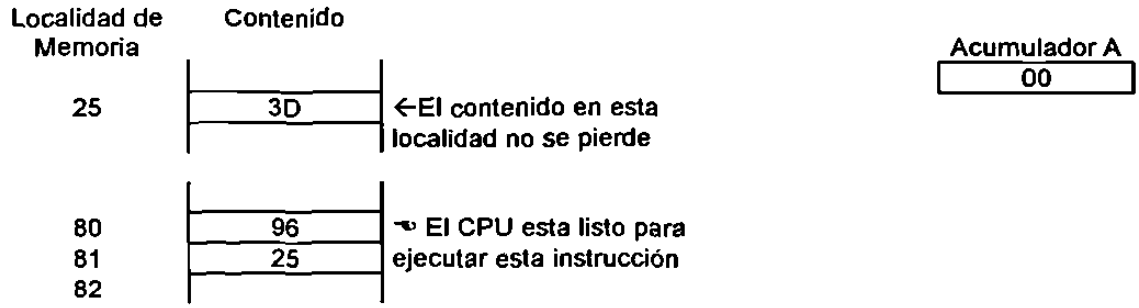


En el **Modo de Direccionamiento Directo** también llamado Direccionamiento de Página Cero, el byte que sigue al código de operación contiene la dirección de la localidad de memoria donde se encuentra el dato; esto es, en los primeros 256 bytes, desde $00 a $FF. En la dirección antes mencionada solo se especifica el byte menos significativo, pues el byte más significativo se asume igual a $00.

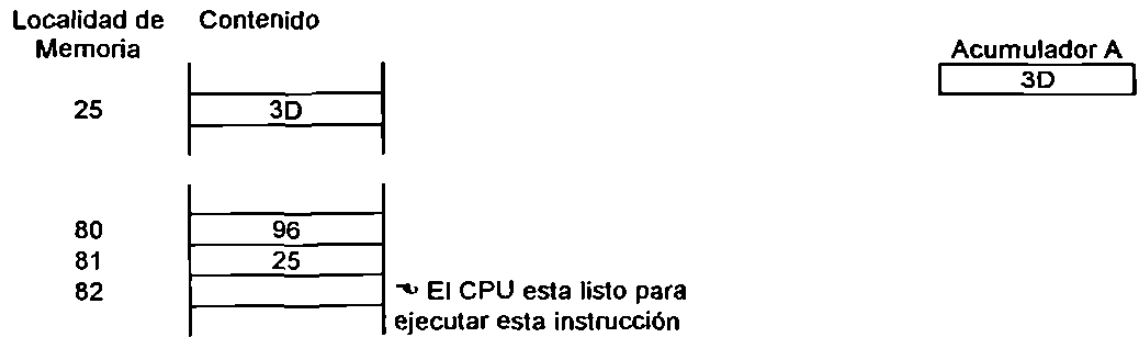
Ejemplo del Modo de Direccionamiento Directo:

LDAA $25 Cargar el acumulador A con el contenido de la localidad de memoria 25 en hexadecimal.

Antes de ejecutar la instrucción: LDAA $25



Después de ejecutar la instrucción: LDAA $25

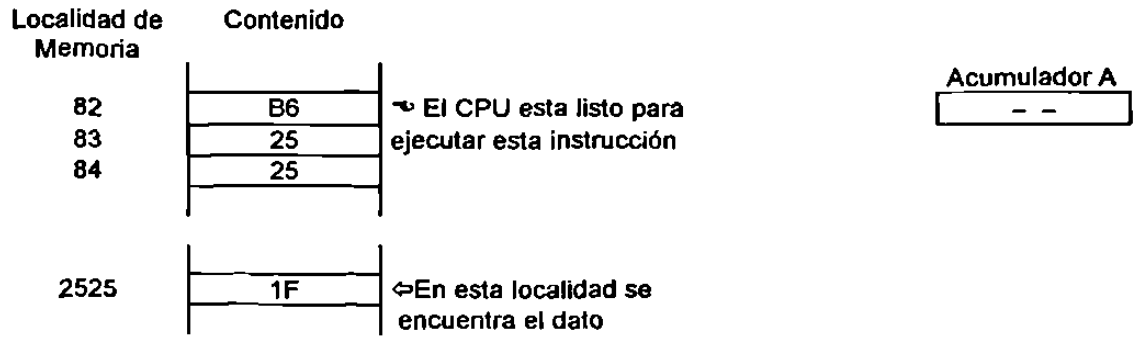


En el **Modo de Direccionamiento Extendido** el segundo y tercer byte enseguida del código de operación contienen la dirección del operando (dato). En este modo de operación se requieren dos bytes para establecer la dirección que puede ser cualquiera de los 64k bytes de memoria.

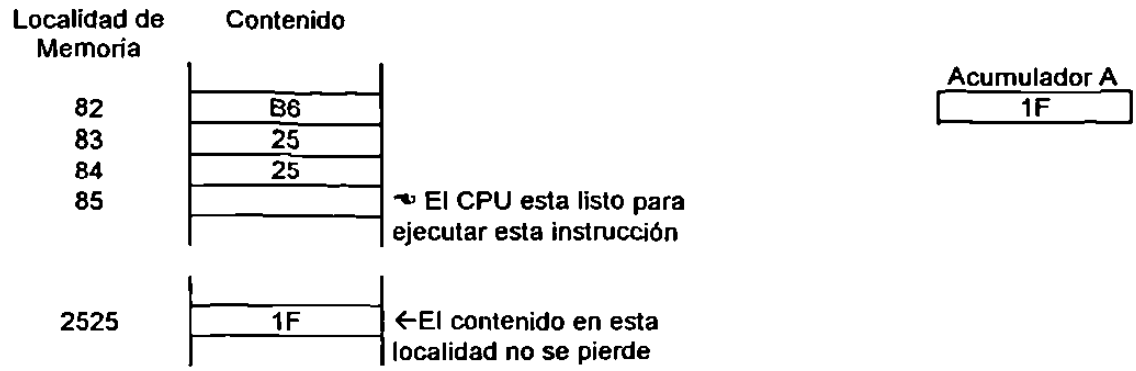
Ejemplo del Modo de Direccionamiento Extendido:

LDAA $2525 cargar el acumulador A con el contenido de la localidad de memoria 2525 hexadecimal.

Antes de ejecutar la instrucción: LDAA $2525



Después de ejecutar la instrucción: LDAA $2525



En el **Modo de Direccionamiento Indexado** uno de los registros índice X o Y es usado para calcular la dirección efectiva. La **dirección efectiva** es un valor de dos bytes, que es generado internamente en cada modo de direccionamiento (excepto en el inherente) para establecer la localidad origen o destino de los datos. En este modo de direccionamiento la dirección efectiva es variable y depende de 2 factores:

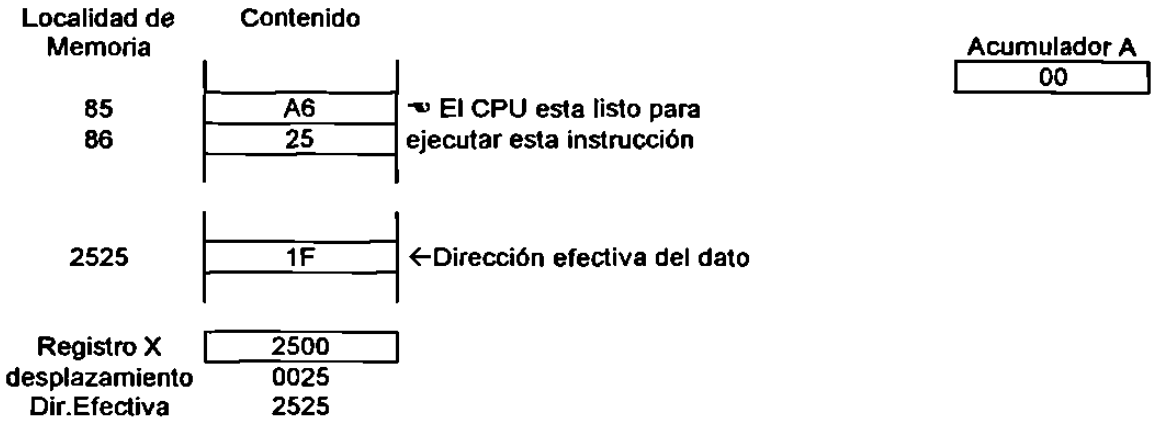
1. El contenido actual del registro índice (X o Y) que está siendo usado.

2. Los 8 bits de desplazamiento (offset) contenidos en la instrucción.

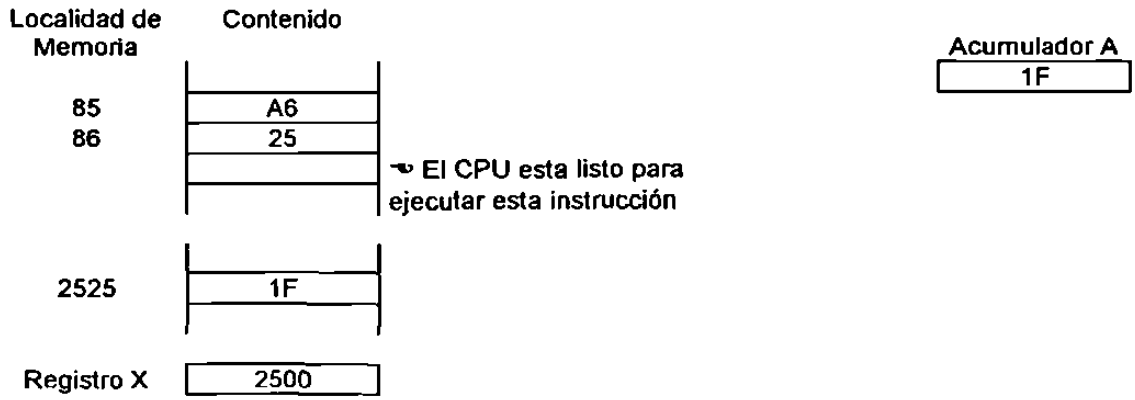
Ejemplo del Modo de Direccionamiento Indexado.

LDAA $25,X cargar el acumulador A, con el contenido de la localidad de la dirección efectiva obtenida al sumar 25 hex al contenido del Registro Indice X.

Antes de ejecutar la instrucción: LDAA $25,X



Después de ejecutar la instrucción: LDAA $25,X



El **Modo de Direccionamiento Inherente o Implícito** se utiliza con instrucciones que involucran exclusivamente los registros internos de MCU y está caracterizado por no requerir operandos, pues la instrucción contiene toda la información necesaria.

El **Modo de Direccionamiento Relativo** es utilizado en instrucciones de salto o bifurcación condicional. El salto solo puede ejecutarse desde -127 a +128 bytes relativos (con respecto) a la instrucción de bifurcación. En el siguiente capítulo veremos como el programa Ensamblador, realiza en forma totalmente transparente para el programador, el cálculo de localidades necesarias de salto.

3.4. CONJUNTO DE INSTRUCCIONES.

El **Conjunto de Instrucciones** es el grupo de instrucciones que un microprocesador o microcontrolador específico puede ejecutar. El Conjunto de *instrucciones* no están estandarizados debido al individualismo de cada fabricante, y a la diferencia de arquitectura y uso de cada dispositivo. En ésta sección será revisado el Conjunto de instrucciones para los MCU's de la familia M68HC11.

Un número suficiente de instrucciones es analizado, lo cual permitirá entender la totalidad del Conjunto de Instrucciones. Todos *los modos de* direccionamiento para esas instrucciones es presentado brevemente y el código de operación **hex** (abreviación de hexadecimal) para cada modo de direccionamiento es listado.

Todas las direcciones y ejemplos utilizados incluyen registros y contenidos de memoria que han sido seleccionados al azar y son mostrados para propósitos ilustrativos solamente.

3.4.1. NOMENCLATURA.

En la explicación de la función que realizan las instrucciones se hace uso de la siguiente nomenclatura:

1 *Los Registros se representan con las siguientes letras:*

A = Acumulador A
B = Acumulador B.
D = Doble Acumulador: Acumulador A concatenado con Acumulador B
CCR = Registro del Código de Condición.
X = Registro Indice X
Y = Registro Indice Y
PC = Contador de Programa (Program Counter).
SP = Apuntador de Pila (Stack Pointer)
M = Localidad de Memoria.

2 En el Registro del Código de Condición **CCR** (Condition Code Register) se usa la siguiente nomenclatura para indicar la actividad de los bits: S, X, H, I, N, Z, V, C.

- = bit no afectado o no importa su valor si proviene de una instrucción anterior.
0 = bit forzado a cero.
1 = bit forzado a uno.

⇕ = bit es puesto en uno o en cero de acuerdo al resultado
⇓ = el bit puede cambiar de uno a cero, permanece en uno, pero no puede cambiar de cero a uno.

3 Todos los números utilizados son hexadecimales.

4 Un símbolo como éste ☜ indica la siguiente instrucción que será ejecutada por la CPU.

5 Una flecha como ésta ⇦ indica que "es colocado en" o "transferido a".

6 Los paréntesis son usados para indicar "el contenido de". Por ejemplo: (B) significa el contenido del acumulador B; (2720 hex) significa el contenido de la localidad de memoria 2720 hex.

7 Las letras mayúsculas entre comillas que aparecen al inicio de la explicación de una instrucción corresponden a su mnemónico. Los ***mnemónicos*** son abreviaturas o siglas de palabras (en inglés) que facilitan el recordar la función que realiza una instrucción y se utilizan para aproximar la lectura y escritura de programas al lenguaje habitual (en inglés).

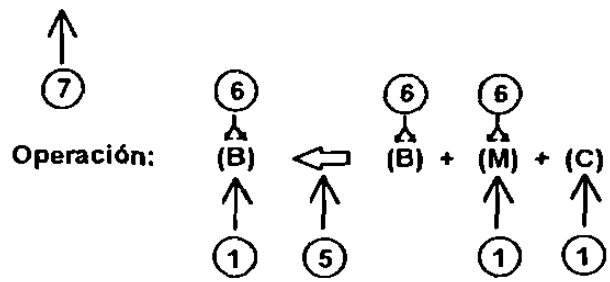
8 Cuando está presente el símbolo # indica direccionamiento inmediato. Cuando no está presente el símbolo #, el número siguiente al código de operación es una dirección.

9 El símbolo $ indica que el número siguiente es hexadecimal.

En el ejemplo siguiente relacionando los números inscritos en los círculos, con los números de los incisos correspondientes de la nomenclatura, se puede observar la aplicación de la misma.

EJEMPLO DESCRIPTIVO DE LA APLICACION DE LA NOMENCLATURA

"ADCB" Add with Carry to B



Descripción: Suma el contenido del Registro B al contenido de la localidad de memoria M y le agrega el contenido del bit C, el resultado es colocado en el Registro B, el contenido de la localidad de memoria M no cambia.

```
     S X H I N Z V C
CCR - - ⇕ - ⇕ ⇕ ⇕ ⇕
        ↓
        (2)
```

Mnemónico: ADCB

Modo de Direccionamiento	Código de Operación	Listado Fuente	Nota de Explicación
Inmediato	(3) ⇩ C9	(8) ⇩ ADCB #$36 ⇧ (9) ⇙	1
Directo	D9	ADCB $91	2
Extendido	F9	ADCB $8671	3 Ver ejem.
Indexado X	E9	ADCB $31, X	4
Indexado Y	18E9	ADCB $D2, Y	5

NOTAS DE EXPLICACION:

1. (B) ⇦ (B) + 36 hex + (C)

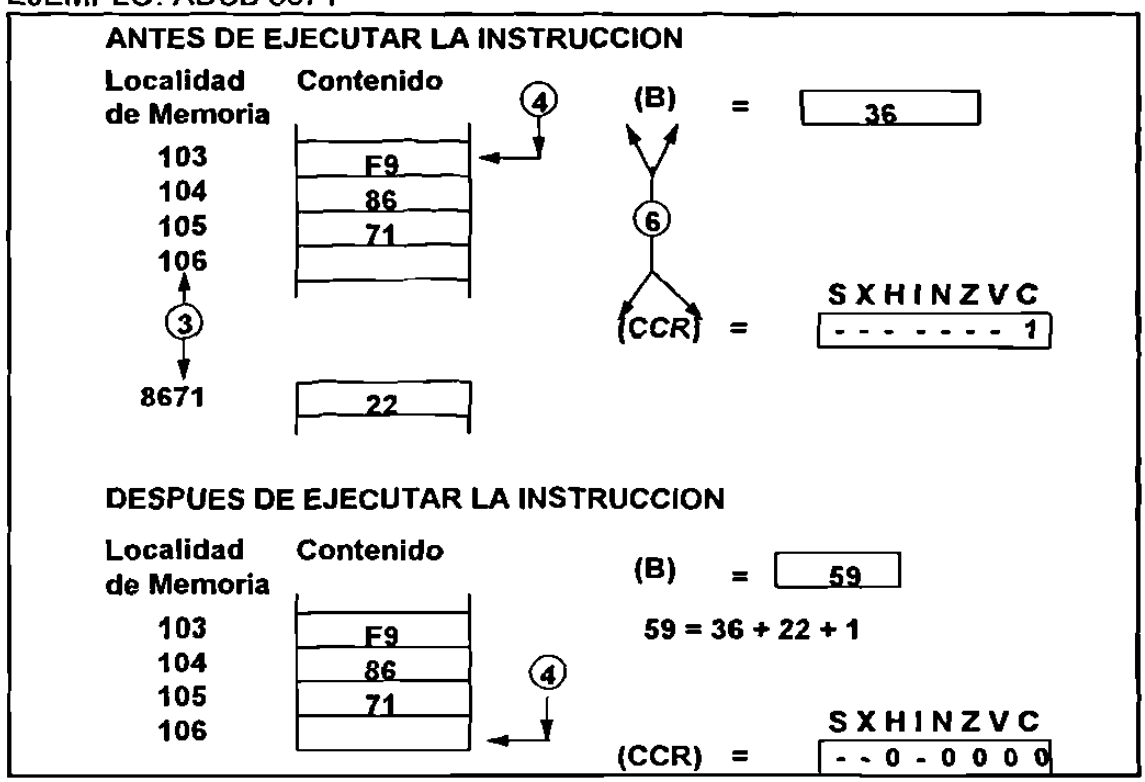
2. (B) ⇦ (B) + (91 hex) + (C)

3. (B) ⇦ (B) + 8671 hex) + (C)

4. (B) ⇦ (B) + ((X) + 31 hex) + (C)
   (6) ↙↘ ↑ (5)

5. (B) ⇦ (B) + ((Y) + D2 hex) + (C)

EJEMPLO: ADCB 8671



3.4.2. CLASIFICACION DE LAS INSTRUCCIONES.

Las instrucciones del conjunto pueden clasificarse de diversas formas, en este capítulo se organizan en las siguientes categorías:

1. Instrucciones Aritméticas: sumar, restar, incrementar, decrementar, comparar, complemento a 2, multiplicar, dividir, etc.

2. Instrucciones Lógicas: AND, OR, OR-exclusivo, etc.

3. Instrucciones de Transferencia de datos: cargar, almacenar, transferir, intercambiar, etc.

4. Instrucciones de bifurcación o salto: bifurcaciones condicionales e incondicionales.

5. Instrucciones de llamada y retorno de subrutinas.

6. Instrucciones Misceláneas: manipulación de interrupciones, No operación, Espera, STOP, Prueba, etc.