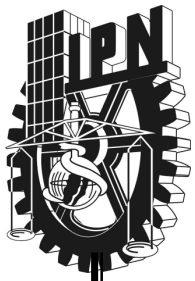


INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE FÍSICA Y MATEMÁTICAS

MEDICIÓN DE RAPIDEZ Y DIRECCIÓN DEL VIENTO UTILIZANDO EL MICROCONTROLADOR MC68HC912

T E S I S

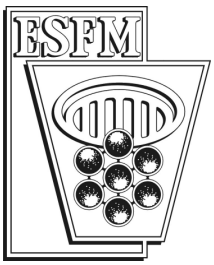
QUE PARA OBTENER EL GRADO DE LICENCIADO

EN FÍSICA Y MATEMÁTICAS

P R E S E N T A:

XOCHITL RAMÍREZ MARQUINA

DIRECTOR DE TESIS: M. EN C. EDGARD MORENO GARCIA



MÉXICO, D. F. FEBRERO 2006

AGRADECIMIENTOS

Señor en especial a ti por darme la oportunidad de realizar día a día mis proyectos y anhelos. Porque siempre estuviste conmigo.

A mis padres por todo su sacrificio y entrega.Este logro también es de ustedes.

A mi esposo por su amor y compresión. Gracias por apoyarme y creer en mi. Te amo.

A mis asesor de tesis Edgard Moreno por la disposición que siempre tuvo al proporcionarme todos los recursos para la realización del trabajo. Gracias por todas sus enseñanzas y por confiar en mi.

A mis profesores Porfirio Reyes y Alfredo Godínez por todo el apoyo que me brindaron, por el interés mostrado durante la realización de este trabago y por ser excelentes personas.Gracias por compartir conmigo sus conocimientos.

ÍNDICE

Pág.

CAPÍTULO 4
UNIDAD CENTRAL DE PROCESAMIENTO

CAPÍTULO 5
PROGRAMAS

CAPÍTULO 6
RESULTADOS Y CONCLUSIONES

APÉNDICES

RESUMEN

Se presenta el desarrollo de un sistema electrónico para medir la rapidez y la dirección del viento en forma continua. Se pretende que el prototipo se utilice en los laboratorios de Servicios a la Navegación en el Espacio Aéreo Mexicano (SENEAM) para fines de mantenimiento, reparación y calibración de sus equipos. Específicamente, se desarrolló el sistema para operar con el sensor de viento Skyvane 2100 fabricado por Qualimetrics Inc. Este sensor se utiliza en las pistas terrestres de los aeropuertos para determinar dos variables meteorológicas importantes: la rapidez del viento y su dirección. El sensor genera una señal cuadrada cuya frecuencia es proporcional a la rapidez del viento y produce un voltaje que es proporcional a la dirección del viento.

El sistema desarrollado utiliza el microcontrolador MC68HC912B32 como su elemento principal, el cual permite realizar por programa gran parte de las funciones de medición. La lectura de las mediciones se presenta en un indicador numérico de tres dígitos con intervalos de 0-99.9 KT (nudos) para rapidez y de 1-360º la dirección. Se utilizan dos recursos del MC68HC912B32 para realizar las mediciones: un sistema temporizador para medir la frecuencia proporcional a la rapidez y otro sistema de conversión Analógica/Digital para medir el voltaje proporcional a la dirección.

Se desarrolló un programa escrito en lenguaje ensamblador propio de la familia MC68HC12, el cuál se encuentra residente en la memoria EEPROM del microcontrolador. Se escribieron subrutinas para medir el periodo en µs y convertir éste valor a nudos (KT); para medir voltaje y convertir éste valor a grados el ángulo de la dirección y para actualizar continuamente los circuitos de lectura.

Como resultado se presenta un prototipo para ser utilizado dentro de los laboratorios de reparación, mantenimiento y calibración de SENEAM. Específicamente, el prototipo opera con el sensor de viento Skyvane 2100, pero puede ser fácilmente adaptado a otro tipo de sensores con salidas similares.

CAPÍTULO 1

GENERALIDADES

En este capítulo se comentan en forma breve algunos conceptos de la meteorología aeronáutica y se plantean los objetivos que se desean alcanzar. Brevemente se justifica la realización del trabajo y se mencionan sus aportaciones. El capítulo termina describiendo el contenido del trabajo.

1.1 CONCEPTOS DE METEOROLOGÍA AERONÁUTICA

La meteorología es la ciencia que se encarga de estudiar los fenómenos atmosféricos, por lo que es esencial el conocimiento de las variables climatológicas para el desarrollo de operaciones aeronáuticas en los aeródromos (área de tierra o de agua destinada a la llegada y salida en movimientos de superficie de las aeronaves), ya que el conocimiento de estas depende la seguridad con la que se lleven a cabo. Entre estas variables son de interés la rapidez y la dirección del viento y para conocer sus valores es necesario contar con los instrumentos y equipos adecuados para cuantificar cada parámetro [1].

RAPIDEZ DEL VIENTO

La rapidez y la velocidad son dos magnitudes cinemáticas que suelen confundirse con frecuencia. La rapidez es una magnitud escalar y se relaciona con la distancia recorrida con el tiempo. La velocidad es una magnitud vectorial que relaciona el cambio de posición con el tiempo.
Los dos principales tipos de instrumentos utilizados para medir la rapidez del viento son el anemómetro rotativo de cubeta y el anemómetro de hélice que constan de dos elementos: el sensor y el transductor. El sensor es un dispositivo que rota por la acción de la fuerza del viento y el transductor es el dispositivo que genera una señal eléctrica [1].

DIRECCIÓN DEL VIENTO

Se define como la orientación del vector del viento en la horizontal. Para los propósitos meteorológicos, la dirección del viento se define como la dirección desde la cual sopla el viento y se mide en grados en la dirección de las agujas del reloj a partir del Norte verdadero. El instrumento más común para medir la dirección del viento es la paleta de viento. Las paletas de viento señalan la dirección desde la cuál este sopla. Un transductor comúnmente utilizado para las aplicaciones de los modelos de calidad del aire es el potenciómetro. El voltaje del potenciómetro varía directamente con la dirección del viento. Un potenciómetro es un resistor variable.

Cuando la dirección del viento cambia, el eje de la paleta se mueve y hace que la resistencia del potenciómetro varíe. Esta modificación está relacionada con la dirección del viento [1].

1.2 OBJETIVO

El principal objetivo del trabajo es desarrollar un sistema electrónico que mida la rapidez y la dirección del viento, utilizando como herramienta principal un microcontrolador de la familia MC68HC12. El prototipo del instrumento será utilizado en los laboratorios de mantenimiento y calibración de SENEAM, cuyas características principales sean:

- Medición de la rapidez del viento en un intervalo de 0 a 99.9 nudos.
- Medición de la dirección del viento en un intervalo de 1-360^{o}.

1.3 JUSTIFICACIÓN

La relevancia del trabajo que se presenta consiste en crear la infraestructura tecnológica dentro de la ESFM-IPN, que permita a baja escala, apoyar y asesorar a la industria y a las empresas gubernamentales como SENEAM.

1.4 APORTACIONES

Como producto del trabajo se presenta el prototipo de un instrumento electrónico que mide la rapidez y la dirección del viento, utilizando como herramienta principal un microcontrolador de la familia MC68HC12.

1.5 CONTENIDO

En el capítulo 2 se describe el sensor de viento utilizado por SENEAM y se propone un sistema electrónico a desarrollar. En el capítulo 3 se presentan las principales características del MCU MC68HC912B32 y en el capítulo 4 se dan los elementos básicos para su programación. El capítulo 5 describe los algoritmos del programa desarrollado en lenguaje ensamblador, las principales funciones y las subrutinas utilizadas. Por último, el capítulo 6 describe las pruebas y los resultados obtenidos en la realización del trabajo.

CAPÍTULO 2

PLANTEAMIENTO DEL PROBLEMA

En este capítulo se da una breve descripción de lo que es SENEAM, el cual requiere de un instrumento que mida la rapidez y la dirección del viento para utilizarlo dentro de sus laboratorios de mantenimiento y calibración. También se describen las características y el funcionamiento del sensor del viento original que utiliza SENEAM el cuál será conectado al instrumento a desarrollar. Finalmente, se propone un sistema electrónico de medición donde se utiliza como elemento principal al sistema de desarrollo SIS68HC912.

2.1 SENEAM (SERVICIOS A LA NAVEGACIÓN EN EL ESPACIO AÉREO MEXICANO)

SENEAM es un órgano desconcentrado dependiente de la Secretaría de Comunicaciones y Transportes que se creó por acuerdo presidencial en 1978 [2]. Uno de sus principales objetivos consiste en proporcionar los servicios de ayuda a la navegación aérea tales como son: meteorología, radio-ayudas, telecomunicaciones aeronáuticas y control de tránsito aéreo. La Organización de la Aviación Civil Internacional (OACI) dependiente de la Organización de Naciones Unidas (ONU) tiene la misión de normar, regular y emitir recomendaciones inherentes al transporte aéreo mundial en la parte civil. México, como Estado signatario del Convenio de Chicago, es responsable del espacio aéreo sobre el territorio nacional, así como aquellas partes sobre el alta mar del Océano Pacífico y Golfo de México en donde el convenio obliga a prestar los servicios a la navegación aérea. Esta responsabilidad dentro del Poder Ejecutivo Federal esta a cargo de la Secretaría de Comunicaciones y Transportes, la cual a encomendado a SENEAM proporcionar los servicios a la navegación aérea en esta porción del espacio.

Entre los servicios que proporciona SENEAM se encuentra el de meteorología aeronáutica. Con este servicio, la empresa de aviación y el piloto reciben información referente al tiempo que hay en las áreas terminales que circundan a los aeropuertos de origen y destino de vuelo, así como las condiciones meteorológicas para seguir la mejor ruta que garantice la seguridad del vuelo o en algunos casos la ruta más corta, para poder conducir con eficiencia el vuelo de la aeronave. En las operaciones de despegue y aterrizaje de las aeronaves es importante monitorear ciertas variables meteorológicas como son, entre otras, la dirección y la rapidez del viento presente en las pistas. Este monitoreo se realiza en las propias pistas de los aeropuertos por medio de estaciones remotas y autónomas que miden diversas variables meteorológicas y que transmiten la información medida hacia diversas dependencias, especialmente a la torre de control.

Como se plantea en el capítulo 1, el objetivo de este trabajo es diseñar y desarrollar un sistema electrónico que permita medir la rapidez y la dirección del viento en condiciones de laboratorio. El prototipo será utilizado dentro de los laboratorios de meteorología de SENEAM para fines de prueba, reparación y calibración de los equipos remotos localizados en las pistas. Según el Manual del Meteorólogo Observador, publicado por SENEAM [3], las variables de rapidez y dirección del viento deben ser registradas con ciertos formatos. En las siguientes líneas se transcriben del manual antes citado, las instrucciones para reportar los valores de las variables de interés.

9.2.4 d<u>ddff(f)G$f_m$$f_m$($f_m$)KT dndndnVdxdxdx Dirección e Intensidad del Viento.</u>

9.2.4.1 <u>La Dirección (ddd).</u> Se reportará con tres dígitos, en decenas de grados (el tercer dígito siempre será cero 0). Direcciones menores a 100 grados deberán ser precedidas de “0”. Por ejemplo un viento de dirección de los 90 grados es codificada como “090”.

9.2.4.2 <u>La Intensidad (ff(f)).</u> Deberá ser codificada en nudos usando dos dígitos en decenas y si se requiere, tres dígitos en centenas. La intensidad menor a 10KT deberá codificarse anteponiendo un cero y deberá terminar con la abreviatura KT que indica que la intensidad esta reportada en nudos. Ejemplo: un viento con una intensidad de 8 nudos deberá codificarse “08KT”; un viento con intensidad de 112 nudos se codificará “112KT”.

9.2.4.6 <u>Viento en Calma.</u> Cuando la intensidad del viento es de 1 nudo o menor se codificará 00000 seguido de la abreviatura (KT).
Ejemplo: 00000KT

El prototipo a desarrollar deberá cumplir con los formatos indicados anteriormente.

2.2 SENSOR DE VIENTO

Para monitorear la rapidez y la dirección del viento que se encuentran presentes en las pistas de los aeropuertos nacionales, SENEAM utiliza un sensor de viento fabricado por Qualimetrics, Inc., llamado “Skyvane” modelo 2100 [4]. Este sensor combina la durabilidad de un instrumento de servicio pesado, con la respuesta característica de peso ligero en la copa y las aspas. La forma aerodinámica del sensor (ver ilustración de la figura 2.1) alinea su cuerpo con la dirección del viento, mientras las 4 aspas de la hélice giran con la rapidez del viento. La dirección del viento es detectada por un potenciómetro localizado en la base del Skyvane. El eje móvil de este potenciómetro se encuentra acoplado mecánicamente al cuerpo del sensor, de tal manera que la resistencia del potenciómetro cambia su valor cuando cambia la dirección del viento. El potenciómetro se alimenta con un voltaje de 5V para producir en su conexión central un voltaje de salida proporcional a la resistencia del potenciómetro y por lo tanto

proporcional a la dirección del viento. Como se ilustra en la figura 2.1, en el eje de la hélice del sensor se encuentra un disco ranurado asociado a un fotointerruptor, el cuál produce una señal cuadrada, cuya frecuencia es proporcional a la rapidez del viento.

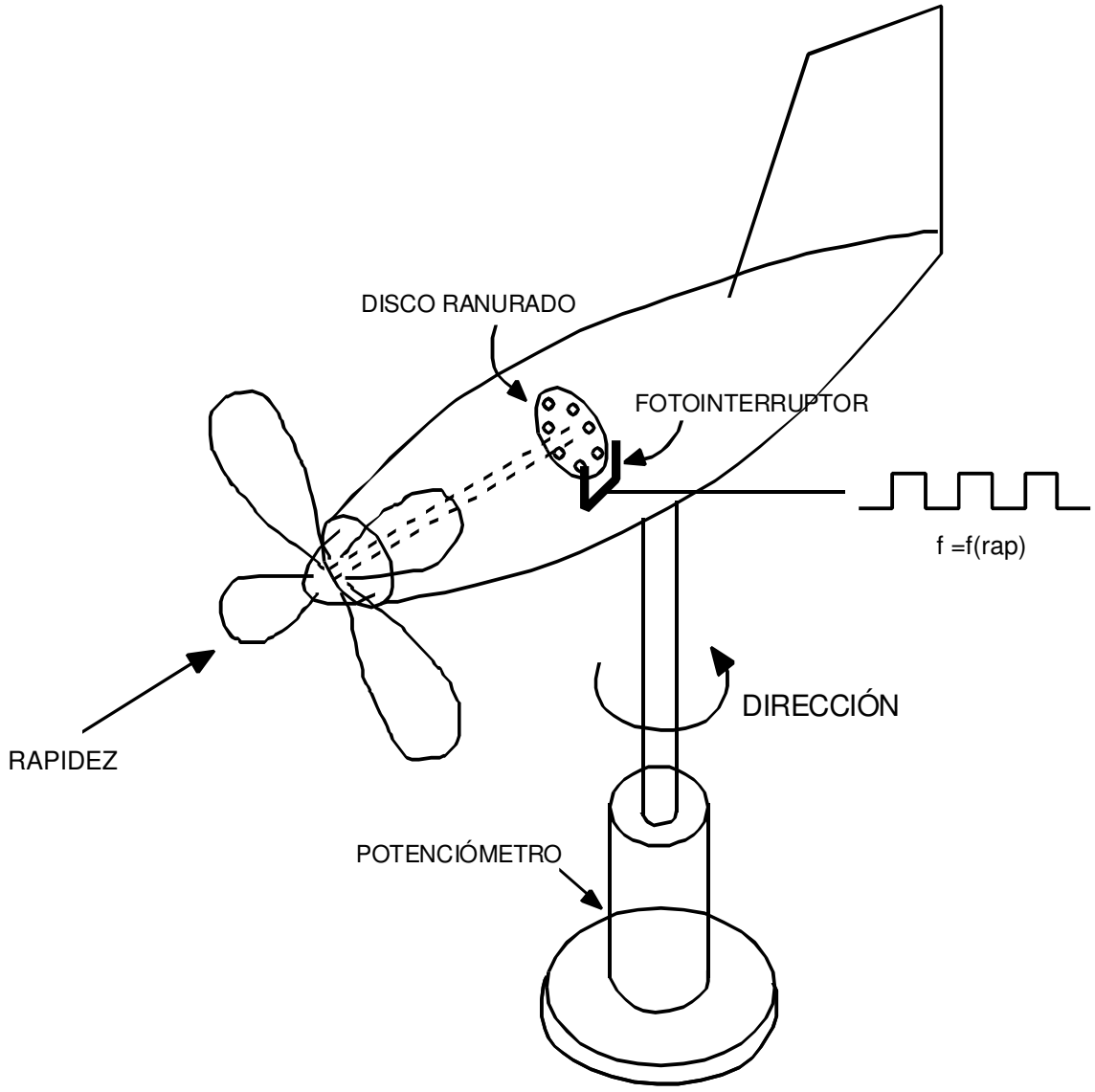


Figura 2.1.- Sensor de viento Skyvane 2100

CARACTERÍSTICAS

- El sensor está construido de fibra de vidrio y aluminio.
- Utiliza acero inoxidable o componentes de latón para el movimiento de todas sus partes.

- Sus componentes pueden durar y resistir vientos de hasta 200 mph.
- Puede operar en lugares rigurosos, incluso a bordo de naves marítimas.
- La salida del sensor suministra salidas compatibles con señales electrónicas con módulos condicionados y equipo con datos lógicos.

ESPECIFICACIONES DEL SKYVANE MODELO 2100

	Generales	Rapidez del viento	Dirección del viento
Tamaño	760x762 mm		
Peso/Embarque	5.4 kg/11.3 kg		
Intervalo		0-200 mph (0-90 m/s)	1-360°
Umbral de inicio		2 mph (0.9 m/s)	
Tracking completo		3 mph (1.3 m/s)	
Distancia constante		1.9 m	
Exactitud		±1 mph < 30 mph ±3% > 30 mph	±2°;±5° al Norte
Resolución		0.3 mph	1°
Salida del sensor		100 mph,HF (461Hz)	POT, 0-1KΩ
Hélice		4 aspas de 350 mm	
Constante de tiempo			< 1 s
Tipo de potenciómetro			modelo de plástico single-wiper

En las siguientes líneas se describen las ecuaciones de transferencia para cada una de las variables que detecta el sensor de viento.

RAPIDEZ DEL VIENTO

El transductor de la rapidez del viento es un tacómetro de alta frecuencia (HF) que produce a la salida una onda cuadrada de +12 VDC con una frecuencia proporcional a la rapidez del viento. La frecuencia de la señal de salida varía de 0 a 925Hz para un intervalo de la rapidez del viento de 0 a 200 mph [4]. La tabla 2.1 muestra una lista de valores proporcionados por el fabricante de la frecuencia de salida para la rapidez del viento. Estos datos se graficaron y se ajustaron a una línea recta (figura 2.2) para obtener la siguiente ecuación de transferencia del sensor:

$$r(mph) = 0.2158 \frac{mph}{Hz} * f + 0.4418\, mph \qquad (2.1)$$

Tabla 2.1.- Valores de entrada (rapidez) y de salida (frecuencia) reportados para el sensor de viento

Entrada, mph	Salida, Hz
10	44.29
18.57	84
20	90.63
30	136.97
30.65	140
40	183.31
45.76	210
50	229.65
60	275.99
70	322.33
80	368.67
90	415
91.08	420
100	461.34
110	507.68
120	554.02
130	600.36
140	646.7
150	693.04
160	739.38
170	785.72
180	832.06
190	878.4
200	924.74

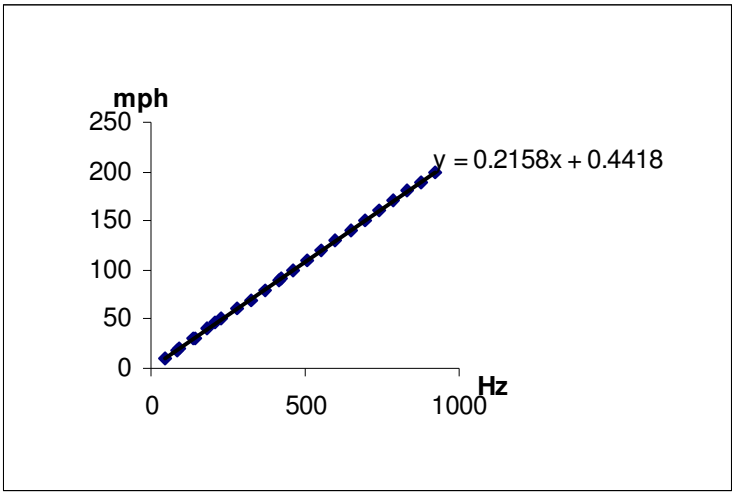


Figura 2.2.- Gráfica y ecuación de transferencia de la tabla 2.1

Meteorológicamente, es común expresar la rapidez del viento en nudos (Knots, KT), la cual se obtiene por la siguiente relación [4]:

$$r(KT) = 0.8689 * r(mph) = 0.187509 \frac{KT}{Hz} * f + 0.384\ KT \qquad (2.2)$$

o bien, en términos del periodo:

$$r(KT) = 0.187509 \frac{KT}{Hz} * \frac{1}{T} + 0.384\ KT \qquad (2.3)$$

En la tabla 2.2 se presentan los factores de conversión x para expresar la rapidez del viento en diferentes unidades a partir de la ecuación 2.1.

Tabla 2.2.- Conversión de unidades de la rapidez del viento

Salida,Hz	KT x =0.86890	Km/h x = 1.60934	m/s x = 0.44704	ft/s x = 1.46660
44.29	8.69	16.09	4.47	14.67
84	16.13	29.88	8.30	27.23
90.63	17.38	32.19	8.94	29.33
136.97	26.07	48.28	13.41	44.00
140	26.64	49.33	13.70	44.96
183.31	34.76	64.37	17.88	58.66
210	39.76	73.64	20.46	67.11
229.65	43.45	80.47	22.35	73.33
275.99	52.13	96.56	26.82	88.00
322.33	60.82	112.65	31.29	102.66
368.67	69.51	128.75	35.76	117.33
415	78.20	144.84	40.23	131.99
420	79.14	146.58	40.72	133.57
461.34	86.89	160.93	44.70	146.66
507.68	95.58	177.03	49.17	161.32
554.02	104.27	193.12	53.64	175.99
600.36	112.96	209.21	58.11	190.66
646.7	121.65	225.31	62.59	205.32
693.04	130.33	241.40	67.06	219.99
739.38	139.02	257.49	71.53	234.66
785.72	147.71	273.59	76.00	249.32
832.06	156.40	289.68	80.47	263.99
878.4	165.09	305.78	84.94	278.65
924.74	173.78	321.87	89.41	293.32

DIRECCIÓN DEL VIENTO

El transductor de la dirección del viento es un potenciómetro sin tope de 1KΩ que se polariza con un voltaje constante. La salida lineal del brazo central es un voltaje proporcional a la dirección del viento en el intervalo de 1-360°. La hendidura del potenciómetro está orientada directamente hacia el Norte. El viento hace girar el cuerpo del sensor y este giro se transmite mecánicamente al eje del potenciómetro. Con un voltaje aplicado al potenciómetro de +5V, el movimiento del sensor provoca un voltaje de salida que varía de 0 a 5V. Este voltaje lineal corresponde a los grados del ángulo de la dirección del viento de 1-360° [4]. Con esta información se determina la ecuación de transferencia que relaciona el voltaje de salida Vd con el ángulo de entrada Az, de la siguiente manera:

$$Vd = 13.9276 \frac{mV}{°} * Az - 13.9276\, mV \qquad (2.4)$$

2.3 PROPUESTA DE SOLUCIÓN

En la figura 2.3 se presenta el procedimiento experimental propuesto para medir la rapidez y la dirección del viento utilizando el sensor descrito en la sección anterior. La propuesta se basa en el uso del microcontrolador MC68HC912B32 fabricado por Motorola, el cual contiene entre otros recursos un Convertidor Analógico/Digital (CAD) que permite medir el voltaje proporcional a la dirección del viento y un sistema temporizador con el cual es posible realizar la medición de la frecuencia ó periodo proporcional a la rapidez del viento.

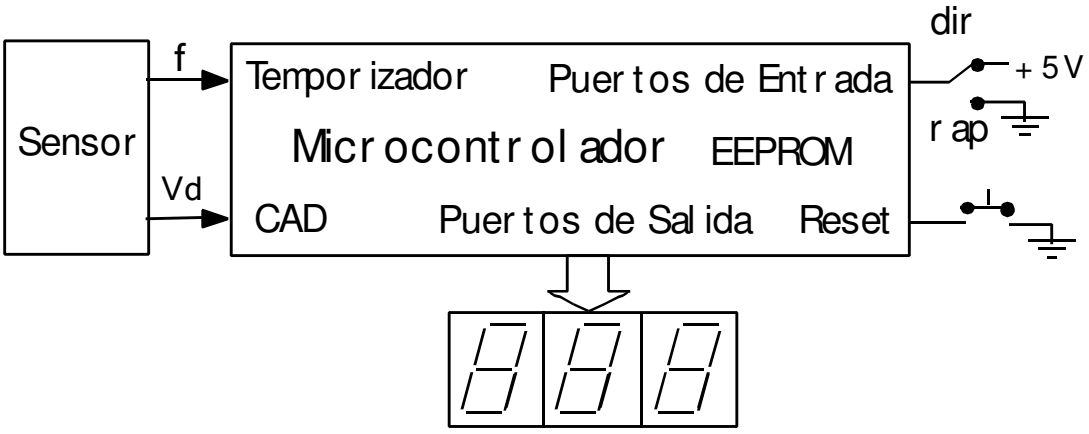


Figura 2.3.- Sistema Propuesto para medir la rapidez y la dirección del viento

Los procesos de medición se controlan por medio de un programa escrito en lenguaje ensamblador, el cuál debe estar almacenado en la memoria EEPROM interna del microcontrolador. La ejecución del programa inicia cuando se enciende el instrumento ó cuando se pulsa el botón de reset. El programa lee el estado del interruptor de función y de acuerdo al valor leído ejecuta la subrutina de medición correspondiente. El valor medido de la frecuencia f o del voltaje Vd es procesado por el programa y enviado a los indicadores digitales de lectura en las unidades adecuadas para cada medición. El programa se ejecuta indefinidamente mostrando en la lectura digital el valor actual de la variable medida. En cualquier momento el interruptor de función puede ser activado para cambiar la variable de medición.

2.4 SISTEMA ELECTRÓNICO

El sistema electrónico se elaboró en base al sistema de desarrollo SIS68HC912 de RACOM Microelectronics [5], el cual incluye las fuentes de alimentación y solo requiere de un transformador de 12 VCA para que pueda operar. El sistema proporciona, por medio de conectores machos en línea doble, todos los recursos del microcontrolador MC68HC912B32 (MCU). En la figura 2.4 se muestra la conexión del sistema de desarrollo con los circuitos externos de lectura, con las salidas del transductor y con el interruptor de función.

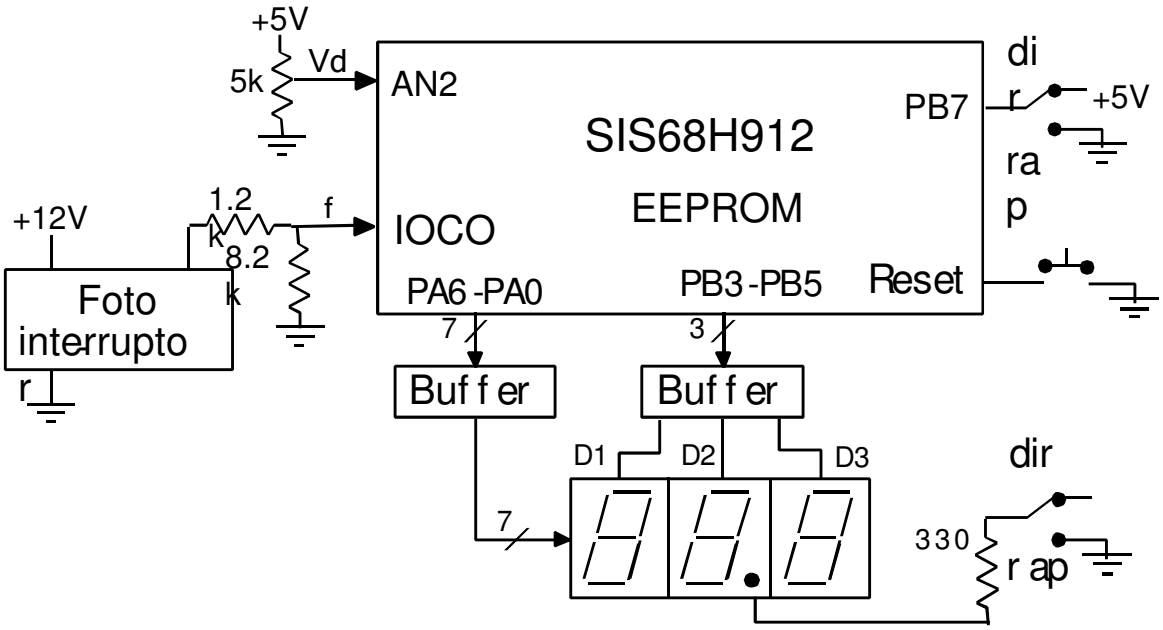


Figura 2.4.- Sistema electrónico con el sistema de desarrollo SIS68H912

La señal analógica Vd proporcional a la dirección del viento se conecta directamente a la entrada analógica AN2 del sistema de conversión A/D del MCU. La señal cuadrada generada por el fotointerruptor tiene niveles altos de 12V, por lo tanto se utiliza un atenuador resistivo para disminuir este valor a niveles aceptados por la entrada IOCO del sistema temporizador del MCU (5V máximo). El estado del interruptor de función puede ser leído por medio del puerto PB7. Se utilizan 7 líneas del puerto A (PA6-PA0) para enviar un código de 7 segmentos hacia los circuitos de lectura. Tres líneas del puerto B (PB3-PB5) se utilizan para encender cada uno de los 3 dígitos en forma multiplexada. Estas 10 líneas de salida del microcontrolador son acopladas a los circuitos de lectura por medio de alimentadores ó "buffers" para manejar las corrientes que demandan los circuitos de lectura. En el Apéndice A se presenta el diagrama esquemático del circuito de lectura, así como el diseño del circuito impreso desarrollado. El Apéndice B muestra como es físicamente el sistema de desarrollo utilizado y el sistema electrónico elaborado. Los procesos de medición y de lectura se controlan por medio de un programa escrito en el lenguaje ensamblador propio del MC68HC912B32, tal y como se describe en el capítulo 5.

MEDICIÓN DE LA DIRECCIÓN DEL VIENTO

El proceso de conversión Analógica/Digital dentro del MCU produce un código binario de 8 bits de salida para un determinado voltaje de entrada. La ecuación de transferencia para la conversión A/D en 8 bits y con un intervalo de entrada (Vd) de 0 a 5V es la siguiente:

$$Ds = \frac{255LSB}{5000mV} * Vd \qquad (2.5)$$

Ds representa el número decimal de salida del código generado en la conversión A/D y Vd es el voltaje proporcional a la dirección del viento. Substituyendo la ecuación (2.4) en (2.5) obtenemos la ecuación de transferencia completa en la medición y despejando el ángulo de la dirección del viento Az, se llega a la siguiente relación:

$$Az = \frac{359^{o}}{255LSB} * Ds + 1^{o} \qquad (2.6)$$

La ecuación (2.6) debe ser resuelta por el programa del microcontrolador. La medición de la dirección se presenta como un entero de 3 dígitos para un intervalo de 1 a 360°.

MEDICIÓN DE LA RAPIDEZ DEL VIENTO

Como se describe en la sección 2.2, el sensor produce una señal cuadrada con una frecuencia y periodos proporcionales a la rapidez del viento. De acuerdo a la tabla 2.1 el intervalo de la frecuencia es de 44.29Hz a 924.74Hz. Para nuestra aplicación el intervalo para la rapidez del viento es de 0 a 99.9 nudos (KT), de tal manera que con los

tres dígitos de lectura se puede mostrar el valor medido con una resolución de 0.1 KT. De la ecuación (2.3) se puede observar que la rapidez a medir debe ser mayor a 0.384 KT para evitar valores negativos de T. Se propone iniciar el proceso de medición a partir de una rapidez 0.5 KT. De acuerdo a la ecuación (2.3), para un intervalo de rapidez del viento de 0.5 a 99.9 (KT), corresponde un intervalo del periodo a medir de 1.616 s hasta 1.884 ms. Para una rapidez inferior a 0.5 KT el programa de control mantiene una lectura de 00.0 en los circuitos de lectura. El programa del microcontrolador debe resolver la ecuación (2.3) para determinar la rapidez en nudos.

El principio de medición de periodo se basa en el conteo de pulsos de reloj (N), de periodo conocido (Tck) y estable, durante el tiempo que se desea medir. De esta manera el tiempo medido esta dado por: T = N*Tck. Para realizar la medición del periodo se utiliza el sistema temporizador del MCU, el cuál contiene un contador libre que permanentemente se encuentra contando la señal del reloj del sistema (Tck). Este contador tiene una capacidad de 16 bits, es decir el máximo tiempo que puede contar es de $T(max) = 2^{16}-1*Tck = FFFFh*Tck$ (h denota un valor hexadecimal). Cuando el contador alcanza su máxima cuenta, continua contando desde un valor 0 y se produce un sobreflujo. La señal del reloj tiene una frecuencia de 8MHz [6] y un periodo de 0.125µs, éste valor representa la resolución en la medición del tiempo. La máxima capacidad de conteo de tiempo es de 8.191ms.

El proceso básico propuesto para medir el periodo consiste en registrar en una variable Ti el contenido del contador libre cuando se detecte una transición de nivel de la señal a medir. Con la siguiente transición, el contenido del contador libre se registra en la variable Tf y el tiempo se determina con la diferencia Tf-Ti. Debido a que no existe sincronía entre la señal interna de reloj de MCU y la señal a medir, es posible que la diferencia Tf-Ti produzca un valor negativo que debe ser tomado en cuenta durante el procesamiento del periodo. Por otro lado también es posible que existan sobreflujos del contador libre en la medición de periodos grandes, dichos sobreflujos deben ser contados para poder procesar la diferencia de tiempo adecuadamente. En la sección 5.3 se describe con más detalle el algoritmo de medición de periodo.

CAPÍTULO 3

DESCRIPCIÓN DEL MICROCONTROLADOR MC68HC912B32

En este capítulo se describe en forma general los conceptos más importantes del microcontrolador MC68HC912B32 (MCU, Unit Microntroller) y sus principales recursos. Se estudia la arquitectura del MCU y las funciones de cada una de sus terminales, las cuales le permiten comunicarse con el exterior. Se describen 7 modos de operación que determinan la forma de ejecución y operación del MCU. Se estudian sus 8 puertos utilizados como entradas y/o salidas para el control y acceso de varios subsistemas. Posteriormente se da una breve descripción de los 4 posibles mapas de memoria y finalmente, también se describe en forma breve, la función de los registros de control más importantes utilizados en la realización de este trabajo.

3.1 RECURSOS

Para poder efectuar la medición de la rapidez y dirección del viento se utiliza un miembro de la familia de microcontroladores MC68HCx12xx de la marca Motorola, donde las x designan a un miembro particular de la familia. Dicho microcontrolador es el MC68HC912B32. Este microcontrolador está fabricado con tecnología HCMOS (semiconductor de óxido metálico complementario de alta densidad) y contiene en un sólo encapsulado los circuitos necesarios para realizar funciones periféricas avanzadas [6], tales como:

- 32 Kbytes de memoria Flash EEPROM.
- 1 Kbyte de memoria RAM.
- 768 bytes de memoria EEPROM.
- Interface de comunicación serial asíncrona (SCI).
- Interface periférica serial (SPI).
- 4 canales de modulación por ancho de pulso (PWM).
- 8 canales temporizados y acumulador de pulsos de16 bits.
- Convertidor Analógico/Digital de 8 bits (CAD).
- Bus externo multiplexado.
- 63 líneas de I/O para propósito general.

Otras propiedades importantes de la familia son:

- Fabricación con tecnología semiconductor de óxido complementario (CMOS) de alta densidad (HCMOS).
- Emplea arquitectura Von Newman, es decir, que las instrucciones y los datos comparten el mismo espacio de memoria.

- Procesador CISC (Complex Instrucción Set Computer) de l6 bits.
- Sistema de interrupción en tiempo real.
- Sin consumo de energía en estado de Paro o Espera.
- Alta inmunidad al ruido.
- Sistema de Vigilancia de Operación Correcta (COP).
- Alta rapidez de operación.

El incluir todas estás funciones en un sólo circuito monolítico hacen de este microcontrolador una herramienta muy poderosa en una gran diversidad de aplicaciones.

3.2 ESTRUCTURA INTERNA Y TERMINALES DE CONEXIÓN

La figura 3.1 muestra el diagrama a bloques del MCU donde se indican sus recursos internos y sus terminales de conexión disponibles hacia el exterior [6]. Muchas de estas terminales tienen funciones alternas que pueden ser programadas por medio de un conjunto de registros de control, los cuales se describen en la sección 3.6. La asignación de las terminales del MCU se muestra en la figura 3.2. A continuación se da una breve descripción de los principales bloques y líneas de conexión, donde el * al final de una designación denotará que esta función se activa con un nivel bajo cero o un cero lógico en su respectiva terminal.

TERMINALES DE ALIMENTACIÓN Y TIERRA

V_{DD}	Alimentación interna.
V_{SS}	Tierra interna.
V_{DDX}	Alimentación externa.
V_{SSX}	Tierra externa.
V_{DDA}	Voltaje para el convertidor Analógico/Digital. Permite el suministro de voltaje para poder realizar la conversión en el CAD.
V_{SSA}	Tierra del convertidor Analógico/Digital.
V_{RH} ,V_{RL}	Voltajes de referencia del convertidor Analógico/ Digital.
V_{FP}	Voltaje de programación para la memoria Flash EEPROM y requerido para una operación normal.
V_{PP}	Alto voltaje requerido para la memoria EEPROM utilizado para realizar pruebas en modos especiales.

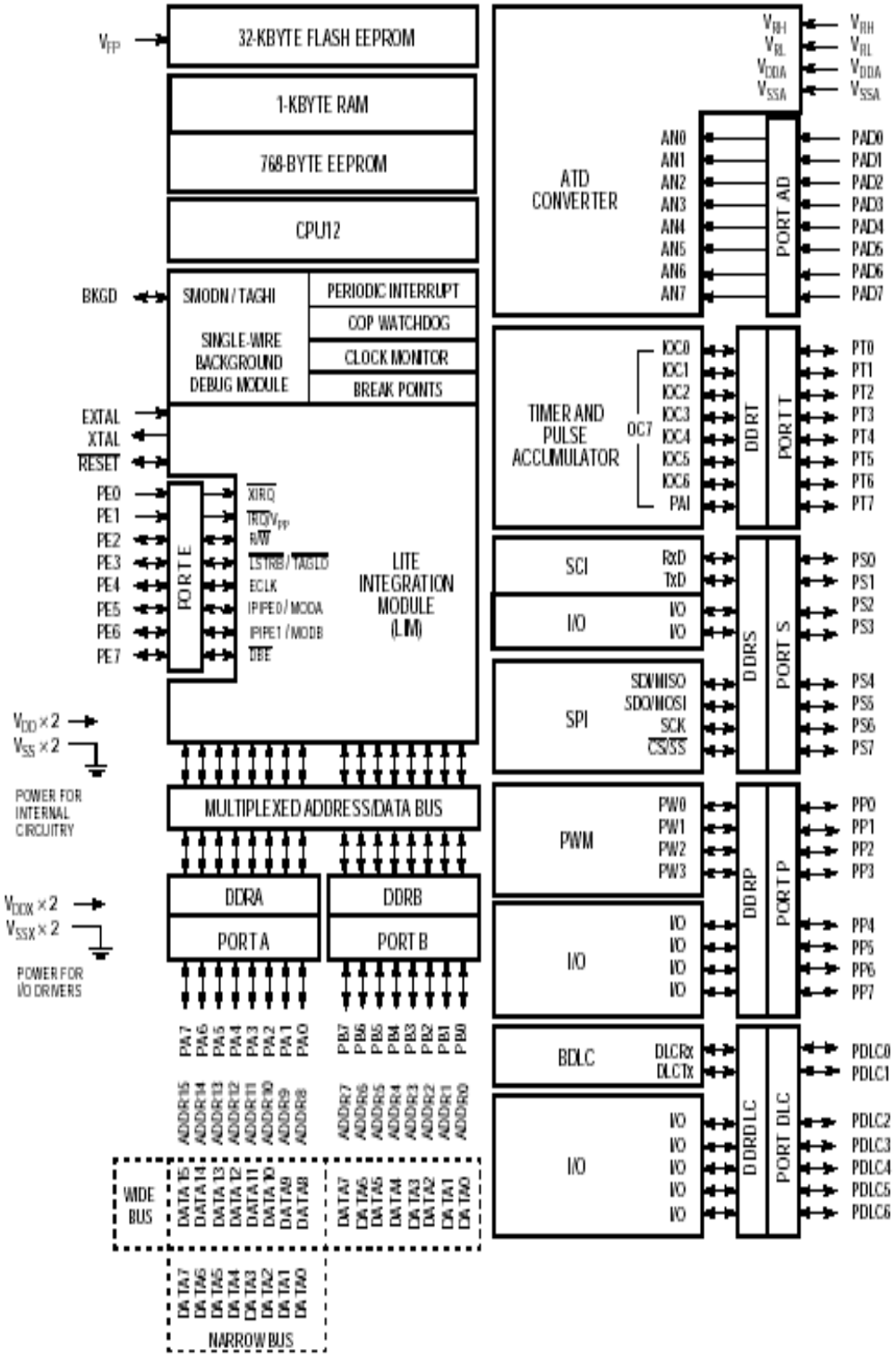


Figura 3.1.- Diagrama a bloques del microcontrolador MC68HC912B32

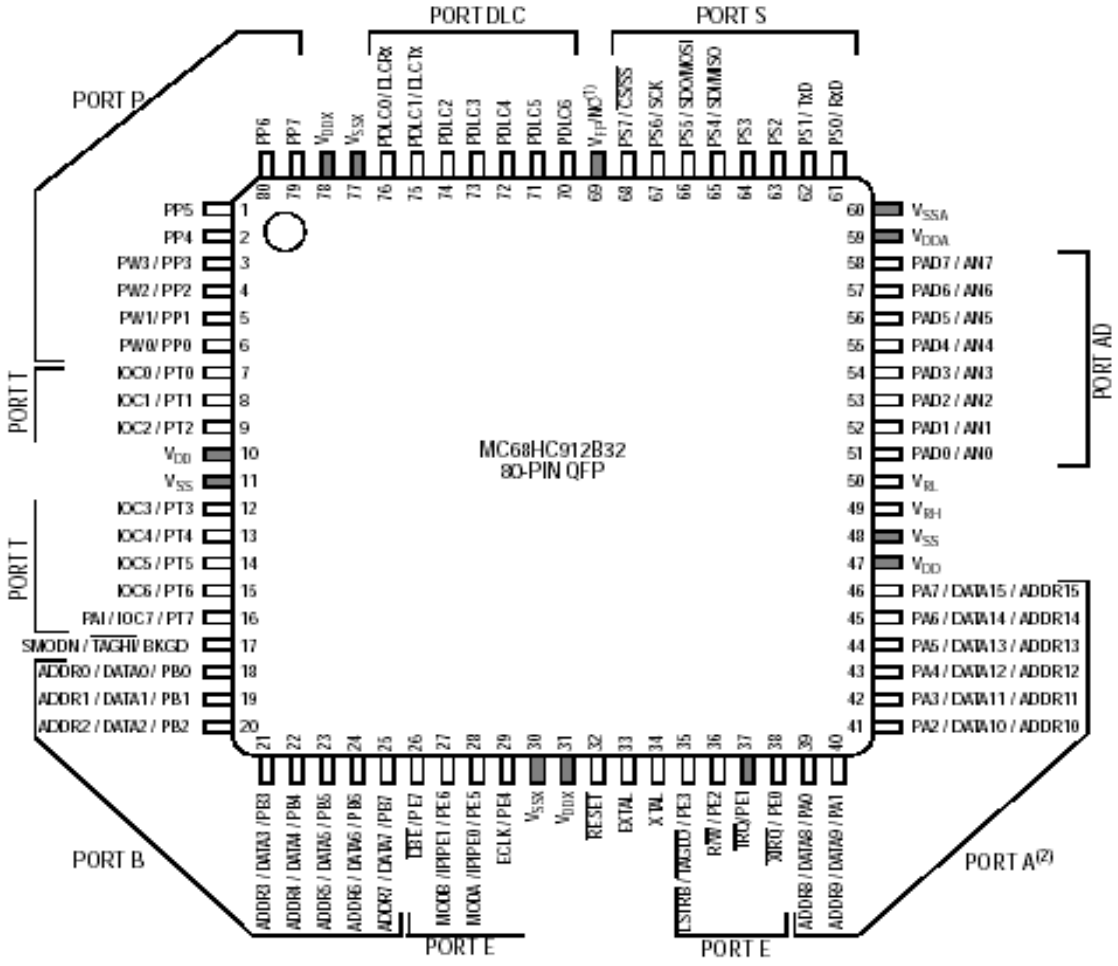


Figura 3.2.- Asignación de pines del MC68HC912B32

OSCILADOR (EXTAL, XTAL, ECLK)

Las terminales EXTAL y XTAL son utilizadas para conectar entre ellas un circuito oscilador externo en base a un cristal o resonador cerámico, o bien para aplicar una señal externa compatible con CMOS a la terminal EXTAL. En ambos casos, se controla al circuito oscilador interno de tal manera que se produce una señal de reloj E con una frecuencia dos veces menor a la frecuencia del oscilador externo. Esta señal de reloj E representa el ciclo de máquina y es la señal básica para las operaciones internas del MCU. La señal de reloj E está disponible hacia el exterior para sincronizar eventos externos por medio de la línea ECLK. La máxima frecuencia de operación del MCU es de 8 MHz, la cual corresponde a una máxima frecuencia del circuito de oscilación externo de 16 MHz. Todas las señales de reloj incluyendo el ECLK, están en paro cuando el MCU está en modo STOP.

TERMINAL RESET*

Es una terminal bidireccional de control y su función principal es de entrada para inicializar al MCU en un estado conocido cuando detecta un pulso de nivel bajo o "0" lógico. En operación normal del MCU esta línea puede actuar como salida para indicar que una falla interna ha sido detectada en el sistema de monitoreo del reloj o en el sistema de vigilancia de operación correcta (COP, Computer Operating Properly).

INTERRUPCIONES (IRQ*,XIRQ*)

El MCU tiene 2 terminales de entrada para solicitar una interrupción: IRQ* y XIRQ*. Cuando se activa la terminal IRQ*, su petición interrupción es o no aceptada, de acuerdo a una habilitación por medio de programa y se dice entonces que es una interrupción enmascarable. Cuando se activa la terminal XIRQ*, su petición de interrupción es atendida en el momento en que sucede el evento externo y entonces se dice que es una interrupción no-enmascarable. La activación de IRQ* se puede programar para que sea con una transición negativa o que sea sensitiva a nivel. La entrada XIRQ* es siempre sensitiva a nivel. El MCU tiene otras terminales y recursos internos que son capaces de generar una solicitud de interrupción del tipo enmascarable.

3.3 MODOS DE OPERACIÓN

El MCU se inicializa aplicando un pulso externo a la línea de reset y opera en uno de siete posibles modos, de acuerdo a los niveles lógicos presentes en las líneas MODA, MODB y BKGD durante éste pulso de reset. En la tabla 3.1 se indica el modo de operación del MCU después del pulso inicial de reset de acuerdo al código formado por las entradas de modo [6]. Cada modo tiene asociado un mapa de memoria.

Tabla 3.1.- Modos de operación de MC68CH912B32

BKGD	MODB	MODA	MODO SELECCIONADO
0	0	0	Especial “single chip “
0	0	1	Especial Expandido Limitado
0	1	0	Especial Periférico
0	1	1	Especial Expandido Extenso
1	0	0	Normal “single chip”
1	0	1	Normal Expandido Limitado”
1	1	0	Reservado
1	1	1	Normal Expandido Extenso

Existen dos modos básicos de operación:

- **Normal.** Varios registros y bits están protegidos contra cambios accidentales. Este modo de operación cuenta con tres tipos de configuración.

- **Especial.** Permite proteger los registros de control y bits de propósitos generales para pruebas y emulaciones. Tres modos de operación especiales corresponden a los modos de operación normal utilizados en la fabricación de pruebas y sistemas de desarrollo.

MODO NORMAL "SINGLE CHIP"

En este modo no hay direcciones externas ni buses de datos. El MCU opera como dispositivo único y todos los programas y fuentes de datos se encuentran en el chip. Las terminales de los puertos A, B y E se pueden configurar como entradas y/o salidas de propósito general.

MODO NORMAL EXPANDIDO LIMITADO

Las terminales de los puertos A, B y E se pueden configurar como entradas y/o salidas de propósito general. Los puertos A y B son utilizados como un bus de direcciones de 16 bits. El puerto A se configura como un bus multiplexado de datos y direcciones (byte alto), mientras que el puerto B es configurado como un bus de direcciones de 8 bits (byte bajo). Este modo es utilizado por sistemas de producción de bajo costo que utilizan 8 bits de memoria (RAM o EEPROM) limitada.

MODO NORMAL EXPANDIDO EXTENSO

En este modo, los puertos A y B son utilizados como un bus multiplexado de direcciones y datos de 16 bits.

MODO ESPECIAL "SINGLE CHIP"

Este modo puede ser usado para forzar al MCU a activar el modo BDM en todos los sistemas de debugeo a través de la terminal BKGD. El MCU no va en busca del vector de reset y ejecuta el código como si estuviera en otros modos; el MCU opera como un dispositivo único en donde todos los programas y espacios para los datos se encuentran dentro del chip, por lo que no hay direcciones ni buses de datos externos. Las terminales externas de los puertos pueden ser utilizadas como entradas y/o salidas de propósito general.

MODO ESPECIAL EXPANDIDO LIMITADO

Los puertos A y B son utilizados por el bus de datos de 16 bits. El puerto A es utilizado como un bus multiplexado de datos y direcciones, el puerto B es utilizado como un bus

de direcciones. En este modo, el dato de 16 bits representa un byte en un tiempo, el bit alto (puerto A) seguido del byte bajo (puerto B).

MODO ESPECIAL EXPANDIDO EXTENSO

Los puertos A y B son utilizados como un bus multiplexado de direcciones y datos. El bus de control esta relacionado con las terminales del puerto E y son configuradas para el servicio de las funciones del bus de control y no como entradas y/o salidas.

MODO ESPECIAL PERIFÉRICO

En este modo, la CPU no está activa. Otro microcontrolador maestro puede controlar el periférico del chip para realizar pruebas. No es posible cambiar de este modo a través del reset. En este modo, el debugeo de Background no debe ser utilizado ya que puede haber conflictos entre el BDM y el maestro externo, causando operaciones incorrectas de ambos modos.

MODO BACKGROUND DEBUG (BDM)

El BDM es un modo de operación auxiliar utilizado por sistemas de desarrollo. BDM esta implementado en el hardware dentro del chip y suministra un conjunto completo de operaciones para el debugeo. Algunos comandos del BDM pueden ser ejecutados mientras la CPU opera en forma normal. El BDM puede ser usado en los modos de operación anteriormente descritos, excepto en el modo especial periférico Una vez habilitado, el modo background puede ser activado a través de un comando serial vía la terminal BKGD o mediante la instrucción BGND. Mientras el BDM esta activado, la CPU puede interpretar comandos especiales de debugeo, leer y escribir los registros, registros periféricos y localidades de memoria.

3.4 PUERTOS

El MCU tiene incorporados ocho puertos los cuales pueden ser utilizados como entradas y/o salidas (E/S) de propósito general, o bien, pueden ser programados para realizar otras funciones alternas [6]. Cada puerto esta asociado a un registro de 8 bits que pueden leerse y escribirse en cualquier momento. La operación de los puertos se programa por medio de los registros de control. Las siguientes líneas describen los puertos que se utilizaron en el desarrollo del prototipo, los registros de control que se mencionan se describen en la sección 3.6.

PUERTO A

Las terminales del puerto A son utilizadas como un bus datos y direcciones en modos expandidos, en modo “single chip” las terminales son utilizadas como E/S de propósito general. En este modo, el registro de control DDRA determina cuando cada terminal

del puerto A es una entrada o una salida. Cuando se activa a “1” un bit en DDRA hace que la terminal correspondiente del puerto A sea configurada como una salida; si se limpia un bit en DDRA, la terminal correspondiente al puerto A es configurada como una entrada. Después del pulso de reset, se limpia el contenido del registro DDRA por lo que las terminales del puerto A son configuradadas como entradas.

Terminal	Función Alterna	
PA0	DATA8	ADDR8
PA1	DATA9	ADDR9
PA2	DATA10	ADDR10
PA3	DATA11	ADDR11
PA4	DATA12	ADDR12
PA5	DATA13	ADDR13
PA6	DATA14	ADDR14
PA7	DATA15	ADDR15

PUERTO B

Las terminales del puerto B son utilizadas como un bus datos y direcciones en modos expandidos, en modo “single chip” las terminales son utilizadas como E/S de propósito general. En este modo, el registro de control DDRB determina cuando cada terminal del puerto B es una entrada o una salida. Cuando se activa a “1” un bit en DDRB hace que la terminal correspondiente del puerto B sea configurada como una salida; si se limpia un bit en DDRB, la terminal correspondiente en el puerto B es configurada como una entrada. Después del pulso de reset, se limpia el contenido del registro DDRB y por lo tanto las terminales del puerto B son configuradas como entradas.

Terminal	Función Alterna	
PB0	DATA0	ADDR0
PB1	DATA1	ADDR1
PB2	DATA2	ADDR2
PB3	DATA3	ADDR3
PB4	DATA4	ADDR4
PB5	DATA5	ADDR5
PB6	DATA6	ADDR6
PB7	DATA7	ADDR7

PUERTO AD

Es la entrada para el subsistema Analógico/Digital o bien es utilizado como E/S de propósito general. Cuando la función de conversión Analógica/Digital no esta habilitada, el puerto tiene 8 terminales de entrada para propósito general. El bit ADPU del registro ADTCTL2 habilita la función A/D. Las terminales del puerto A/D son de entrada analógica y el registro de dirección de datos no está asociado con el puerto.

Terminal	Función Alterna
PAD0	AN0
PAD1	AN1
PAD2	AN2
PAD3	AN3
PAD4	AN4
PAD5	AN5
PAD6	AN6
PAD7	AN7

PUERTO T

Este puerto provee entradas de captura de eventos y salidas de comparación en el subsistema del temporizador y el acumulador de pulsos. Las terminales de este puerto también pueden ser utilizadas como líneas de E/S de propósito general. El bit TEN del registro de control TSCR habilita la función del temporizador y el subsistema del acumulador de pulsos se habilita con el bit PAEN en el registro de control PACTL. Cuando el puerto T es de E/S, el registro DDRT determina la dirección de cada terminal; si un bit del registro DDRT es limpiado, la terminal correspondiente se configura como una entrada.

Terminal	Función Alterna	
PT0	IOC0	-
PT1	IOC1	-
PT2	IOC2	-
PT3	IOC3	-
PT4	IOC4	-
PT5	IOC5	-
PT6	IOC6	-
PT7	IOC7	PA1

3.5 MAPAS DE MEMORIA

El MC68HC912B32 tiene 1 Kbyte de memoria estática RAM utilizada para el almacenamiento de instrucciones, variables y datos temporales durante la ejecución del programa. Después del pulso del reset, la dirección donde inicia la memoria RAM se encuentra en la localidad de memoria $800 pero puede ser asignada en cualquier límite de 2 Kbytes dentro del espacio de direccionamiento estándar de 64 Kbytes. El mapeo interno de la memoria RAM está controlado por cinco bits del registro de control INITRM. El arreglo de la memoria RAM ocupa el primer Kbyte de un bloque de 2 Kbytes.
También cuenta con 768 bytes de memoria EEPROM la cual se puede activar con el bit

EEON del registro de control INITEE. El mapeo interno de la memoria EEPROM es controlado por cuatro bits del registro INITEE. Después del pulso de reset, el espacio direccionado de la memoria EEPROM comienza en la localidad de memoria $0D00 ($ denota un valor hexadecimal) y puede ser remapeado en cualquier bloque de 4 Kbytes dentro del espacio de direccionamiento estándar de 64 Kbytes.

Los 32 Kbytes de memoria Flash EEPROM puede ser remapeada en cualquier parte superior o inferior del espacio de direccionamiento de 64 Kbytes. Cuando ocurren conflictos de mapeo, los registros y las memoria RAM y EEPROM tienen prioridad sobre la memoria Flash EEPROM. Para utilizar la parte de memoria expandida se debe estar operando en alguno de los modos expandidos. El diagrama de la figura 3.3 ilustra el mapa de memoria para cada modo de operación inmediatamente después del pulso de reset.

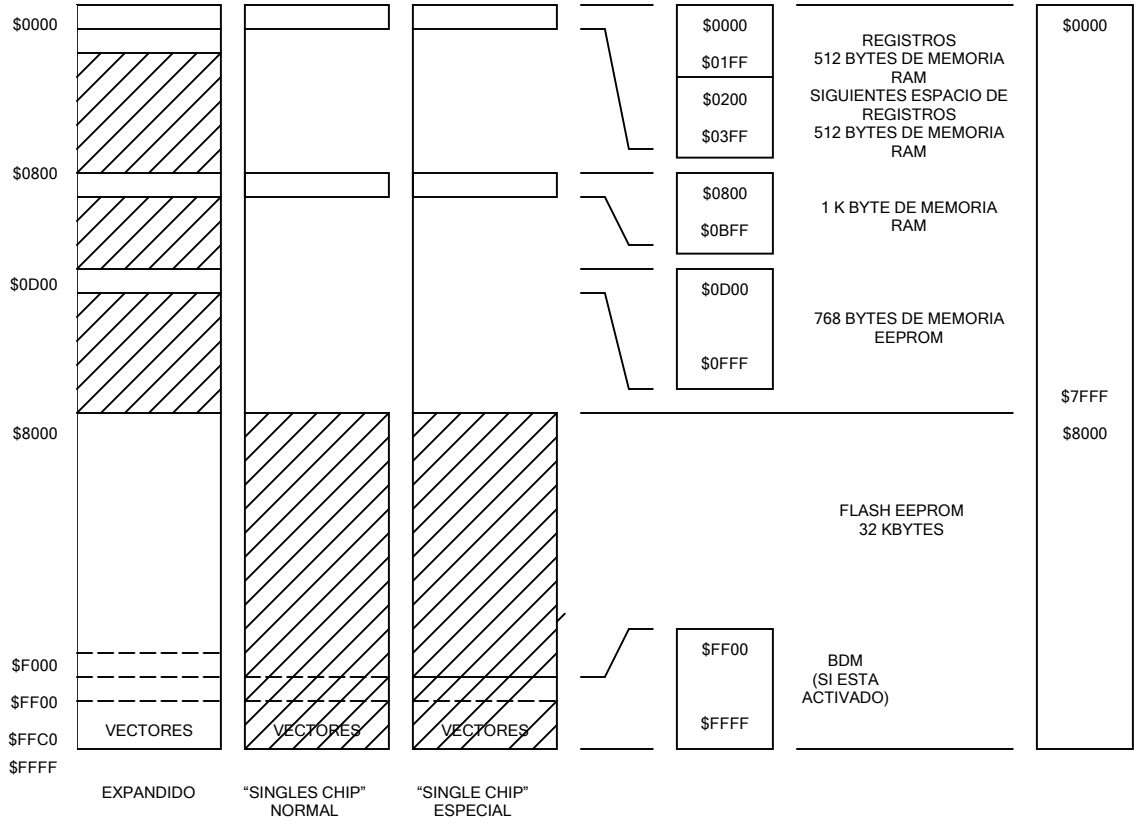


Figura 3.3.- Mapa de memoria del MC68HC912B32

3.6 REGISTROS DE CONTROL

El área de memoria que utilizan los registros de control ocupa los primeros 512 bytes de un bloque de 2 Kbytes. Este bloque puede ser remapeado en cualquier límite de 2 Kbytes dentro de un espacio de direccionamiento estándar de 64 Kbytes, mediante los bits correspondientes en el registro de control INITRG. Después del reset, los 512 bytes del bloque de registros de control se encuentran localizados a partir de la localidad $0000. La Tabla 3.2 muestra los registros de control utilizados en el programa para la medición de la rapidez y dirección del viento [6].

Tabla 3.2.- Registros de control del MC68HC912B32

Registro	Dirección	Descripción
PORTA	$0000	Registro del puerto A, utilizado como E/S.
PORTB	$0001	Registro del puerto B, utilizado como E/S.
DDRA	$0002	Controla la dirección de datos para el puerto A.
DDRB	$0003	Controla la dirección de datos para el puerto B.
ATDCTL2 ATDCTL3	$0062 $0063	Estos registros son utilizados para seleccionar el modo de encendido y control de interrupción.
ATDCTL5	$0065	Este registro de control selecciona el modo de conversión, el canal de conversión y el inicio de la conversión A/D.
ATDSTAT	$0066-$0067	Estos registros de estado del convertidor A/D contienen las banderas que indican que se ha completado la conversión.
PORTAD	$006F	Registro de entrada de datos del puerto A/D.
ADR0H ADR1H ADR2H ADR3H ADR4H ADR5H ADR6H ADR7H	$0070 $0072 $0074 $0076 $0078 $007A $007C $007E	Estos registros contienen el resultado de la conversión Analógico-Digital.
TIOS	$0080	Selecciona al puerto T como E/S.
TCNT	$0084-$0085	Registro de 16 bits de sólo lectura que almacena el contenido del contador libre del sistema temporizador.
TSCR	$0086	Registro del sistema del temporizador.
TCTL1 TCTL2 TCTL3 TCTL4	$0088 $0089 $008A $008B	Registros que se utilizan para programar la transición activa de las entradas de captura del sistema temporizador.
TMSK1	$008C	Los bits de TMSK1 corresponden bit por bit al estado del registro TFLG1. Si una bandera es puesta a “1”, se puede demandar una solicitud de interrupción por

		hardware.
TMSK2	$008D	Selecciona el tiempo de preescala y habilita la interrupción por sobreflujos.
TFLG1	$008E	Cuando existe una transición activa, se captura el tiempo del contador en el registro correspondiente y una bandera es puesta en el bit correspondiente de este registro.
TFLG2	$008F	Cuando existe un sobreflujo es puesta una bandera en el bit correspondiente de este registro.
TCO TC1 TC2 TC3 TC4 TC5 TC6 TC7	$0090-$0091 $0092-$0093 $0094-$0095 $0096-$0097 $0098-$0099 $009A-$009B $009C-$009D $009E-$009F	Estos registros son utilizados para poner el valor del contador cuando es detectada una transición activa en la correspondiente entrada de captura .
PORT	$00AE	Este registro funciona como E/S o bien como captura de entrada y salida comparada en el temporizador y el acumulador de pulsos.
DDRT	$00AF	Programa la dirección de las terminales del puerto T cuando opera como E/S de propósito general.

CAPÍTULO 4

UNIDAD CENTRAL DE PROCESAMIENTO

En este capítulo se describen los registros de la Unidad Central de Procesamiento (CPU, Central Procesing Unit), los modos de direccionamiento y el conjunto de instrucciones que maneja el microcontrolador MC68HC912B32. La CPU es el circuito responsable de interpretar y ejecutar los códigos, escritos en lenguaje de máquina, que representan las instrucciones del microcontrolador. La CPU extrae de la memoria los códigos binarios que representan instrucciones y datos que conforman un programa. Los códigos de instrucción son decodificados para generar en los circuitos una única secuencia de eventos lógicos para cada código de instrucción en particular.

4.1 MODELO DE PROGRAMACIÓN

LA CPU es una unidad de procesamiento de alta velocidad de 16 bits. Contiene amplios registros internos de hasta 20 bits para instrucciones matemáticas extendidas de alta velocidad. La CPU permite ejecutar instrucciones con número impar de bytes, incluyendo muchas instrucciones de un sólo byte, lo cual permite el uso eficiente de espacio en la memoria ROM. Programando la información en los buffers de instrucciones, la CPU siempre tiene un acceso inmediato de un mínimo de tres bytes de código de máquina al inicio de cada instrucción. La CPU también ofrece un conjunto extenso de capacidad de direccionamiento indexado. Los registros son una parte integral de la CPU y no son direccionados como si fueran localidades de memoria [7]. La figura 4.1 muestra el modelo de programación para la CPU.

ACUMULADORES A Y B

Los acumuladores A y B son registros de 8 bits de propósito general que son utilizados para almacenar datos desde la memoria y enviar datos hacia la memoria ó un dispositivo de entrada/salida. También se utilizan para mantener resultados de operaciones aritméticas y lógicas.

ACUMULADOR D

La CPU del MCU puede realizar operaciones de 16 bits para lo cuál utiliza los acumuladores A y B conjuntamente como un sólo acumulador llamado D. El acumulador A es el byte alto del registro D mientras el acumulador B es su byte bajo. El acumulador D es utilizado en las operaciones de suma, resta, multiplicación y división de 16 bits y también para intercambiar datos con los dos registros de índice X e Y.

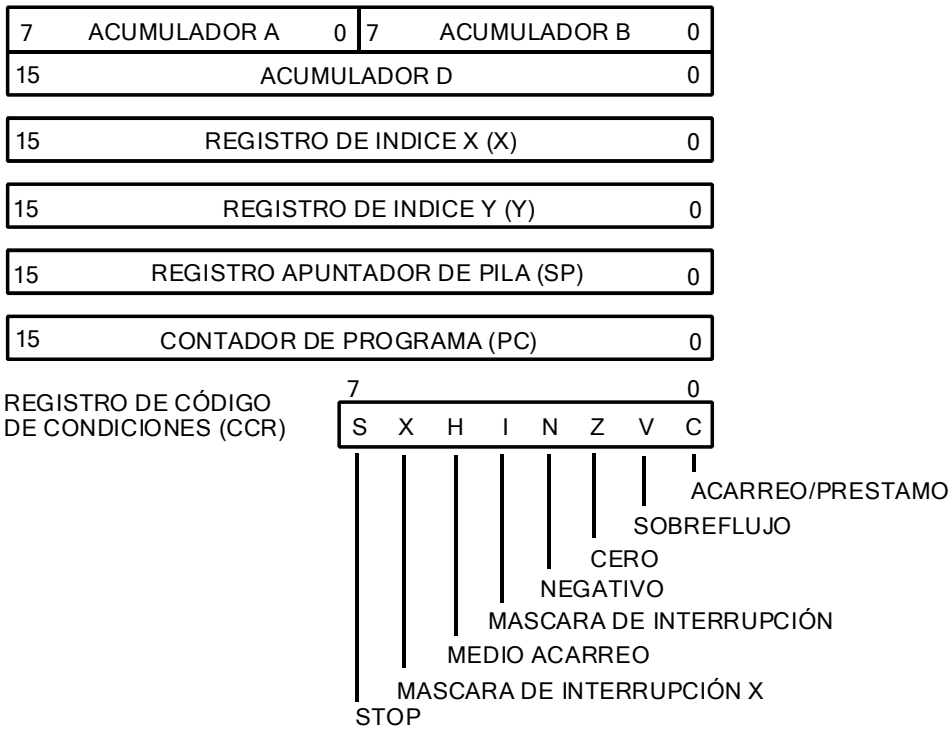


Figura 4.1- Modelo de programación del MC68HC912B32

REGISTROS ÍNDICE X & Y (IX,IY)

Los registros de índice X e Y son de 16 bits y se utilizan en el modo de direccionamiento indexado (de índice). En éste modo de direccionamiento, el contenido del registro de índice de 16 bits se suma a un corrimiento (offset) de 5,8 ó 16 bits (2 bytes) que forma parte de la instrucción para determinar la dirección efectiva del operador. En el siguiente ejemplo se muestra como la CPU calcula una dirección efectiva usando el registro índice X, la instrucción LDAA (Load accumulator A) carga en el acumulador A el contenido de la dirección efectiva.
Ejemplo:

Formato de la Instrucción: LDAA $22,X
Offset : $22
Dirección efectiva: IX+$22

Si suponemos que el contenido de IX es $C412, entonces la instrucción anterior copia en el acumulador A el contenido de la localidad de memoria con dirección $C434.

APUNTADOR DE PILA (STACK POINTER, SP)

La pila ó stack es un área de memoria que puede ubicarse en cualquier espacio de direcciones dentro de los 64 Kbytes y puede extenderse a cualquier tamaño hasta llegar a ser igual al total de memoria disponible en el sistema. La pila se utiliza para almacenar temporalmente datos y direcciones cuando la CPU ejecuta una llamada de subrutina ó una rutina de interrupción, también puede ser utilizada para el amacenamiento temporal de datos. El apuntador de pila es un registro de 16 bits dentro de la CPU cuyo contenido representa la dirección de la siguiente localidad libre de la pila. Cada vez que un byte es almacenado en la pila, su apuntador es automáticamente decrementado, y cada vez que un byte es sacado de la pila el apuntador es automáticamente incrementado. Normalmente, el apuntador de pila es inicializado con una de las primeras instrucciones de un programa. La figura 2.2 muestra el contenido del apuntador de pila antes y después de almacenar 4 bytes:

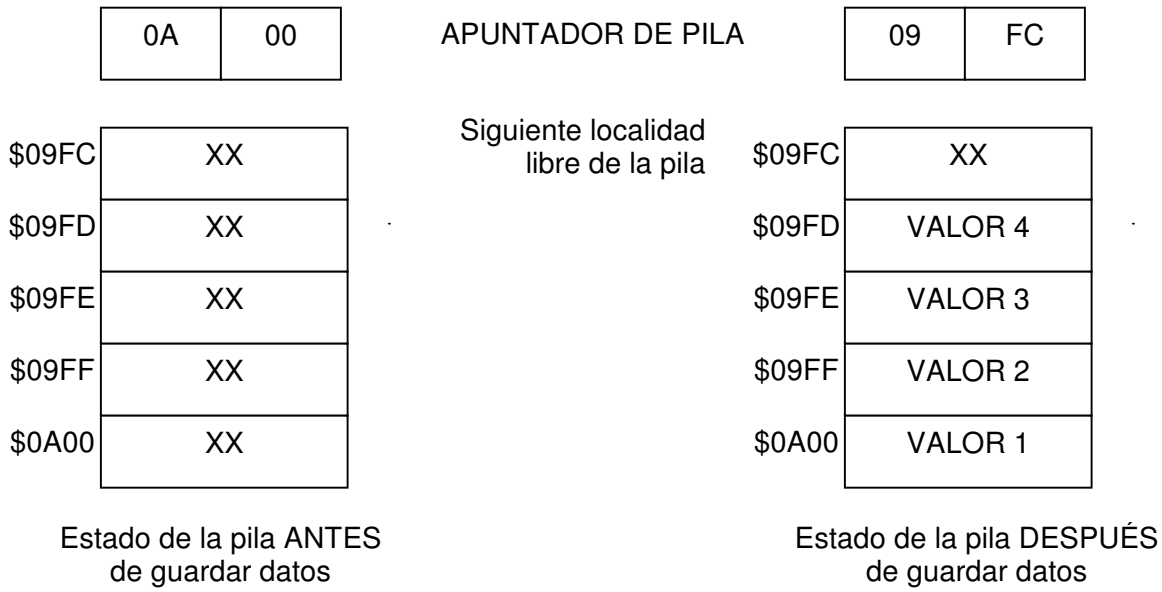


Figura 4.2.- Apuntador de pila antes y después de guardar datos

CONTADOR DE PROGRAMA (PC)

El contador de programa es un registro de 16 bits que contiene la dirección de la siguiente instrucción a ser ejecutada. Cuando la CPU ejecuta una llamada a una subrutina o a una rutina de interrupción, el contenido del contador de programa es automáticamente almacenado en la pila. En la figura 4.3 se muestra el orden en el cual el contador de programa es guardado en la pila cuando se ejecuta un salto a subrutina (JSR, Jump to Subrutine). La última instrucción de una subrutina debe ser un regreso

de subrutina (RTS, Return from Subrutine), y la ultima instrucción de una rutina de interrupción debe ser un regreso de una rutina de interrupción (RTI, Return from Interrupt). Estas dos instrucciones restablecen el contenido del contador de programa almacenado en la pila para continuar la ejecución del programa principal.

Localidad de memoria donde se encuentra almacenado el código.	Instrucción	Operando	Comentarios
$C122	JSR	$C300	Dirección donde inicia la subrutina.
$C125	LDAA	$00	Próxima instrucción a utilizarse.

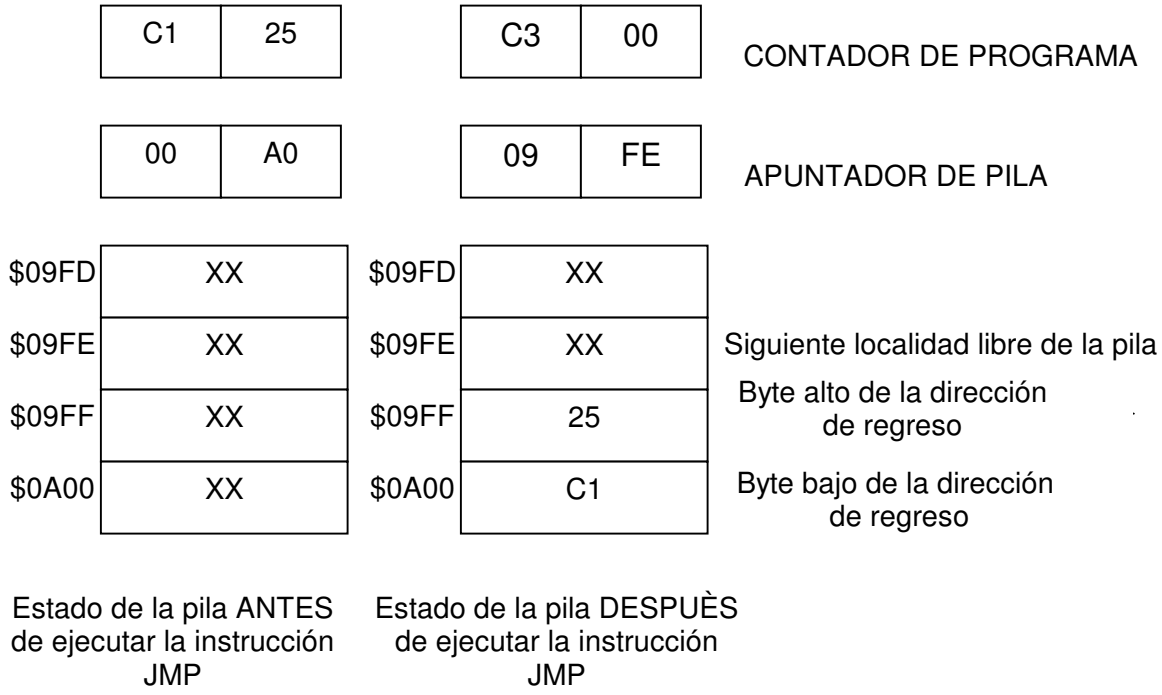


Figura 4.2.- Apuntador de pila antes y después de guardar datos

REGISTRO DE CÓDIGO DE CONDICIÓN (CCR)

El Registro de Código de Condición (CCR) es un registro de 8 bits, 5 de los cuáles contienen cierta condición después de ejecutarse una instrucción (C, V, Z, N y H), 2 bits de máscara para habilitar/deshabilitar interrupciones y un bit para deshabilitar la instrucción STOP. Los 5 bits de condición también son llamados banderas de estado o

bits de banderas. En las siguientes líneas se describen cada uno de los bits que forman el registro de código de condiciones.

Bandera de Acarreo (C)

El bit C es puesto a 1 si la Unidad Lógica Aritmética (ALU) de la CPU realiza un acarreo ó un préstamo durante la última operación aritmética. El bit C también es afectado después de la ejecución de instrucciones de corrimiento.

Bandera de Sobreflujo (V)

El bit V se pone a 1 si hubo un sobreflujo aritmético como resultado de una operación, de otra manera el bit V es puesto a 0. La CPU del MCU es capaz de manejar números en complemento a 2 donde el bit más significativo representa el signo. El intervalo de un número de 8 bits en complemento a 2 es de -128 a +127 en decimal y para un número de 16 bits es de -32,768 a +32,767 en decimal. La bandera V es el resultado de la operación lógica OR exclusiva entre la bandera de acarreo y el acarreo del bit 6 al bit 7 en operaciones de 8 bits o el acarreo del bit 14 al bit 15 en operaciones de 16 bits.

Bandera de Cero (Z)

El bit Z es colocado en un nivel 1 si el resultado de la última operación aritmética, lógica ó de manipulación de datos fue cero. Cando el resultado de la operación es diferente de cero, la bandera Z es colocada en un nivel 0.

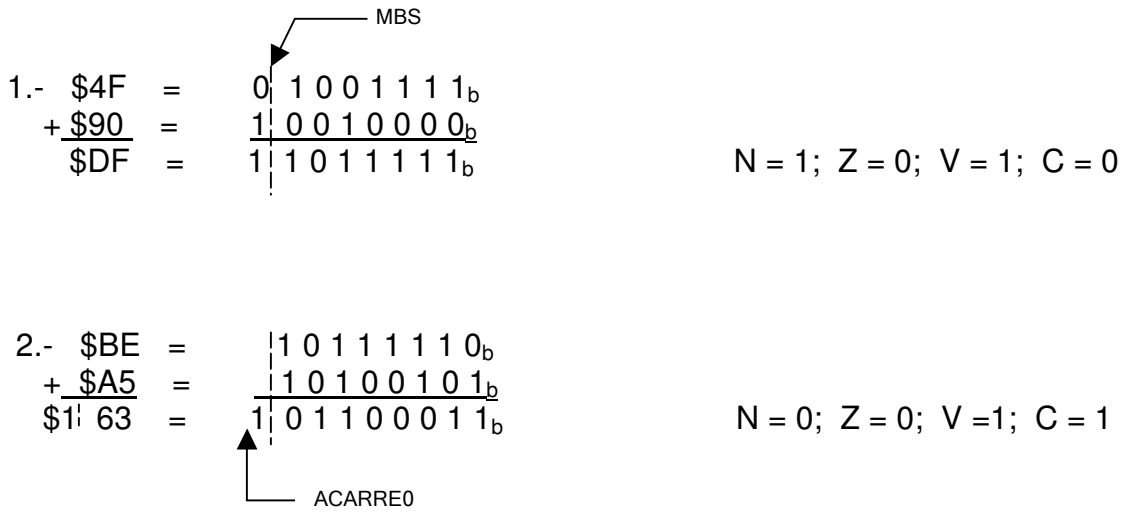
Bandera de Negativo (N)

La bandera N contiene la información del bit más significativo (MBS) del resultado de la última operación. El bit N es colocado en un nivel 1 si el resultado de la última operación aritmética, lógica ó de manipulación de datos fue negativo. Cuando el resultado de la operación es positivo la bandera N es colocada en 0.

Bandera de Medio Acarreo (H)

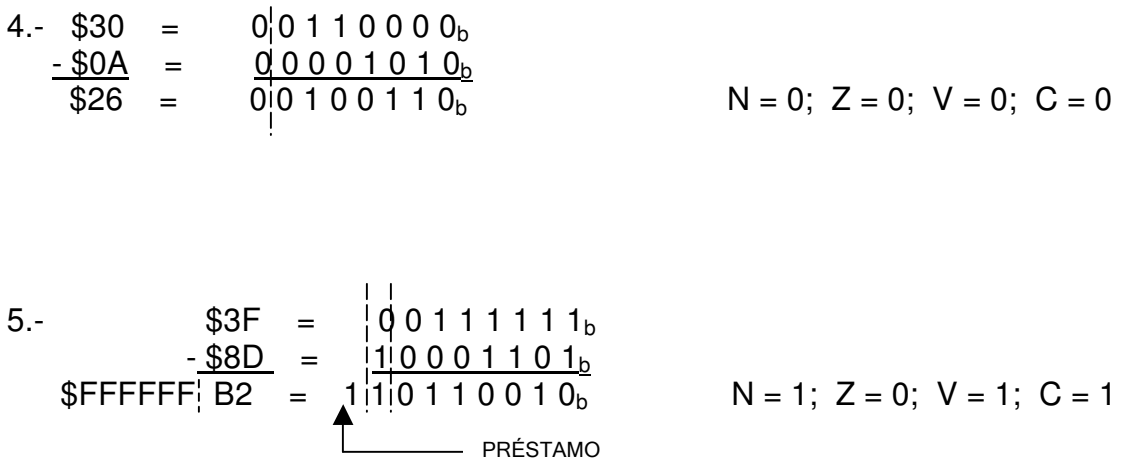
La bandera de medio acarreo es utilizada solamente para operaciones aritméticas en decimal codificado en binario (BCD). El bit H se pone a 1 cuando ocurre un acarreo entre los bit 3 y 4 durante las instrucciones de suma, de otra manera se pone a 0. Para ver de qué manera son afectados éstas banderas a continuación se muestran algunos ejemplos de operaciones básicas con números hexadecimales.

Adición
N: se activa cuando el MBS del resultado es igual a 1.
Z: se activa si el resultado es cero.
V: se activa si existe un sobreflujo complemento a 2 de la operación.
C: se activa si ha ocurrido un acarreo.



Sustracción

N: se activa cuando el MBS del resultado es igual a 1.
Z: se activa si el resultado es cero.
V: se activa se existe un sobreflujo complemento a 2 de la operación.
C: se activa si ha ocurrido un préstamo (cuando la magnitud del sustraendo es mayor que la del minuendo).



Máscara de Interrupción (I)

El bit I habilita/deshabilita todas las interrupciones enmascarables. Mientras el bit I este en un nivel lógico 1, la CPU no reconoce las interrupciones y continua la ejecución del programa normalmente. Cuando se activa la terminal de RESET*, el bit I se pone a un nivel lógico 1 y solamente puede ser limpiado por una instrucción en el programa, es decir, las instrucciones enmascarables solo pueden ser habilitidas por programa.

Máscara de Interrupción (X)

El bit X habilita/deshabilita la señal de interrupción solicitada en la terminal XIRQ*. La interrupción en esta terminal es reconocida por la CPU cuando el bit X esta en un nivel lógico 0 y no es reconocida por la CPU cuando el bit X esta en un nivel lógico 1.

Bit de Paro (S)

Este bit es usado para permitir o no permitir la ejecucción de la instrucción STOP. Si el bit S es colocado en un nivel lógico 1 y la instrucción STOP es ejecutada por la CPU, esta será interpretada como la instrucción de no operación (NOP, No Operation) y la CPU ejecutará la siguiente instrucción. Si el bit S es limpiado a un nivel lógico 0 y la instrucción STOP es ejecutada, todo el sistema de la señal de reloj es detenido y el sistema es colocado en la condición de espera. Para restablecer la operación normal se debe activar la terminal de RESET* o mediante una solicitud de interrupción IRQ* o XIRQ*.

4.2 MODOS DE DIRECCIONAMIENTO

La CPU12 acepta los siguientes tipos de datos [7]:

- Datos de un bit.
- Números enteros con signo y sin signo de 8 y 16 bits.
- Fracciones sin signo de 16 bits.
- Direcciones de 16 bits.

El modo de direccionamiento determina como la CPU tiene acceso a las localidades de memoria mediante el uso de instrucciones y operandos para el desarrollo de su programación. La CPU del MC68HC12B932 cuenta con 6 modos de direccionamiento: Inherente, Inmediato, Directo, Extendido, Relativo e Indexado. Cada modo de direccionamiento excepto en el modo inherente genera una dirección efectiva de 16 bits. La dirección efectiva es la dirección de la localidad donde se localiza el dato a manipular.

MODO INHERENTE (INH)

En este modo de direccionamiento, las instrucciones que se utilizan no contienen operandos ó bien los operandos estan dentro de los registros internos de la CPU. En cualquier caso, la CPU no necesita tener acceso a cualquier localidad de memoria para ejecutar la instrucción.
Ejemplo:

INY El operando es un registro interno de la CPU.

MODO INMEDIATO (IMM)

Los operandos en el modo de direccionamiento inmediato forman parte del conjunto de instrucciones. Cuando un operando es llamado en el modo de direccionamiento inmediato por una instrucción, es puesto inmediatamente en la cola de instrucciones. El modo de direccionamiento inmediato se especifica anteponiendo el carácter ´#' al operando.
Ejemplo:

LDAA #$20 Carga el acumulador A con el número hexadecimal 20.

MODO DIRECTO (DIR)

Este modo de direccionamiento también es llamado direccionamiento página-cero porque es utilizado para accesar operandos en el rango de direcciones que va desde $0000 hasta $00FF. Como las direcciones siempre inician con $00, únicamente los ocho bits de orden bajo del direccionamiento necesitan ser incluidos en la instrucción. El sistema puede ser optimizado si colocamos los datos más comunes en esta área de memoria los datos de acceso en forma frecuente.
Ejemplo:

STX $31 Almacena el valor que hay en IX, en la dirección $0031 los ocho bits de orden alto y en la dirección $0032 los ocho bits de orden bajo.

MODO EXTENDIDO (EXT)

En este modo de direccionamiento, la dirección completa de 16 bits de localidad de memoria al ser operada, es suministrada por la instrucción. El modo de direccionamiento extendido puede ser utilizado para ingresar a cualquier localidad de los 64 Kbytes del mapa de memoria.
Ejemplo:

LDAA $F0A0 Carga el Acumulador A con el valor de la dirección $F0A0.

MODO RELATIVO (REL)

Este modo de direccionamiento únicamente es utilizado por instrucciones de ramificación. Las instrucciones de ramificación condicionales cortas y largas utilizan el

modo de direccionamiento relativo exclusivamente, pero las instrucciones de ramificaciones con manipulación de bits (BRSET, BRCLR) utilizan un modo de direccionamiento múltiple, incluyendo el modo relativo. Las instrucciones de ramificación cortas esta constituido por un código de operación de 8 bits seguido de un offset (compensación) de 8 bits. Las instrucciones de ramificaciones largas constan de un prebyte de 8 bits, un código de 8 bits y un offset signado de 16 bits, contenido en 2 bytes. Los 8 y 16 bits signados del offset son números en complemento a 2, esto permite realizar ramificaciones hacia delante y hacia atrás; el rango del offset para ramificaciones cortas es de $80(-128) a $7F(127) bytes y el rango del offset para ramificaciones largas es de $8000(-32768) a $7FFF(32767). Si el offset es cero, la CPU ejecuta la instrucción inmediata a la instrucción de ramificación, sin tomar en cuenta la prueba involucrada.

MODO INDEXADO (INDX,INDY)

El direccionamiento indexado utiliza un postbyte además de cero, uno o dos bytes de extensión después de la instrucción del código de operación. El postbyte y la extensión realizan las siguientes tareas:

- Especifica cual registro indexado es utilizado.
- Determina si un valor en el acumulador A es utilizado como un offset.
- Habilita automáticamente pre o post-incremento y pre o post-decremento.
- Especifica el tamaño del incremento o decremento.
- Especifica el uso de 5, 9 o 16 bits de offset con signo.

Todos los modos de direccionamiento indexado utilizan un registro de la CPU de 16 bits e información adicional para crear una dirección efectiva. En la mayoría de los casos, la dirección efectiva especifica la localidad de memoria afectada por la operación. En algunas variaciones de direccionamiento indexado, la dirección efectiva especifica la localidad de un valor que apunta a la localidad de memoria afectada por la operación.
Ejemplo: Considere que IX tiene el valor de $1000 antes y después de la ejecución.

```
LDAA 0,X    Carga el acumulador A con el valor que contiene la
                                    dirección $1000.
```

4.3 CONJUNTO DE INSTRUCCIONES

En la arquitectura del MC68HC12 toda la memoria, las entradas y salidas se encuentran mapeadas en el espacio de direcciones común de 64 Kbytes. Esto permite que el conjunto de instrucciones a ser utilizado sea el mismo para tener acceso a la memoria, registros de control y E/S. El conjunto de instrucciones que reconoce el MCU se pueden clasificar en grupos operativos como se describen a continuación.

MOVIMIENTO DE DATOS

Este grupo se divide en 5 instrucciones básicas para movimiento de datos: cargar, almacenar, transferir, intercambiar y mover información entre acumuladores, localidades de memoria y dispositivos de entrada/salida.
Ejemplos:

LDAB Carga el contenido de una localidad de memoria en el acumulador B.

XGDY Intercambia el contenido del acumulador doble D y el registro índice X.

ARITMÉTICAS

Este grupo de instrucciones incluye suma, resta, multiplicación, división, comparación, incrementos, decrementos, complementos a 1 complementos a 2, comprobación de un bit, y ajuste decimal. Para este grupo de instrucciones la CPU es capaz de realizar operaciones signadas en complemento a 2, operaciones sin signo en binario natural y operaciones decimales.
Ejemplos:

MUL Multiplica un valor sin signo de 8 bits (acumulador A) con otro valor sin signo de 8 bits (acumulador B) para obtener un resultado sin signo de 16 bits (acumulador D).

CMPA Compara el contenido del acumulador A con el contenido de una localidad de memoria activando el código de condición el cual puede ser utilizado para condiciones de ramificación lógica y aritmética.

OPERACIONES LÓGICO–BOOLEANAS

Estas instrucciones realizan una operación lógica (AND, OR incluyente, OR excluyente) de un acumulador de 8 bits ó el CCR y el valor contenido en memoria.
Ejemplo:

ORAA Ejecuta la operación lógica OR-inclusiva entre el contenido del acumulador A y el contenido de una localidad de memoria.

PRUEBA Y MANIPULACIÓN DE BITS

Las operaciones de prueba y manipulación de bits utilizan un valor mascarable para probar o cambiar el valor de un bit individual en un acumulador o localidad de memoria.
Ejemplo:

BSET Activa múltiples bits en memoria. Los bits a ser activados son especificados por unos en el byte de máscara. Los otros bits en memoria no son afectados.

DESPLAZAMIENTOS Y ROTACIONES DE BITS

Existen desplazamientos y rotaciones de bytes en los acumuladores y localidades de memoria. El bit que ha sido desplazado se encuentra en la bandera C para facilitar operaciones múltiples con el byte.
Ejemplo:

LSLB Desplaza todos los bits del acumulador B un lugar hacia la izquierda. El bit 0 es cargado con cero. El bit de estado C del CCR es cargado con el bit más significativo del acumulador B.

RAMIFICACIÓN

Una instrucción de ramificación provoca un cambio en la secuencia cuando existen las condiciones especificadas. La CPU utiliza tres clases de ramificaciones: cortas, largas, y condicionadas por bits.
Ejemplo:

BLS Si BLS se ejecuta inmediatamente después de ejecutar una de las siguientes instrucciones: CBA, CMPA, CMPB, CMPD, CPX, CPY, SUBA, SUBB o SUBD, la rama ocurrirá si y sólo si el número binario sin signo en el acumulador es menor o igual que el número binario sin signo contenido en la localidad de memoria.

SALTOS Y SUBRUTINAS

Las instrucciones de salto (JUMP) causan un cambio inmediato en la secuencia. Está instrucción carga el contador de programa con cualquier dirección en los 64 Kbytes del mapa de memoria y el programa sigue su ejecución en esa dirección. La dirección puede proveerse como una dirección de 16 bits o determinarse de varias maneras en el modo de direccionamiento indexado. Las instrucciones de subrutina optimizan el proceso de control de transferencia de un código segmentado que ejecuta una tarea específica, una ramificación corta (BSR), un salto a subrutina (JSR). Las subrutinas que se encuentran dentro de los 64 Kbytes de espacio de memoria se terminan con un retorno a la subrutina (RTS).

INTERRUPCIÓN

Las instrucciones de interrupción dirigen la transferencia de control a una rutina que ejecuta una tarea crítica. En este grupo se encuentran 3 instrucciones para poder atender interrupciones generadas por señales externas al MCU e interrupciones generadas por programa. La CPU también puede ser programada para esperar a que ocurra una interrupción externa. Cuando se reconoce una interrupción todos los registros de la CPU son guardados en la memoria RAM de la pila. La rutina de servicio a la interrupción debe terminar con la instrucción RTI para restablecer los registros de la CPU con los valores que tenían cuando se reconoció la interrupción. Si no hay otra interrupción pendiente se restaura la ejecución normal con la siguiente instrucción

ejecutada antes de la interrupción.

APUNTADOR DE PILA Y REGISTROS ÍNDICE

Las instrucciones de este grupo funcional se utilizan cuando se requiere manejar más de 8 bits en operaciones aritmeticas y para intercambiar datos de 16 bits entre el acumulador D y los registros de índice X e Y. Tambien el contenido del Apuntador de Pila puede ser almacenado o intercambiado con los registros de índice X e Y.
Ejemplo:

PULX El registro índice X es cargado con la dirección indicada por el apuntador de pila. El SP es incrementado en dos.

REGISTRO DE CÓDIGO DE CONDICIÓN

Son formas especiales de instrucciones matemáticas y de transferencia de datos que pueden ser utilizados para cambiar el registro de código de condición.
Ejemplo:

CLC Limpia el bit I mascarable del CCR.

INSTRUCCIONES DIVERSAS

La instrucción de No Operación (NOP) no afecta el contenido de cualquier registro de la CPU ni el estado lógico de cualquier bit del CCR son a menudo utilizadas para reemplazar otras instrucciones durante el debugeo del software. La instrucción de paro (STOP) apila una dirección de retorno y el contenido de los registros y acumuladores de la CPU, entonces se detienen todos los relojes del sistema. La instrucción de espera (WAIT) apila una dirección de retorno y el contenido de los registros y acumuladores de la CPU, entonces espera un servicio de interrupción requerido; las señales del sistema de reloj no se detienen.

4.4 RESETS Y VECTORES DE INTERRUPCIÓN

El HC12 permite realizar otras tareas mientras espera a que suceda un evento. Cuando el evento ocurre, el HC12 deja lo que estaba haciendo para ejecutar una tarea específica y cuando termina regresa al lugar donde estaba antes de que ocurriera el evento. El MC68HC12B32 cuenta con una tabla de vectores de interrupción que pueden ser utilizadas por el programador, cada vector ocupa dos bytes de memoria los cuales deben ser inicializados con la dirección de un servicio de rutina apropiado. Los vectores de reset y de interrupción del CPU están localizados en la memoria Flash EEPROM por lo que están protegidos contra escritura; no obstante, el programa D-BUG12 puede redireccionar estos vectores a un área dentro de la memoria RAM. La Tabla 4.1 muestra las fuentes de interrupción y su vector correspondiente [8].

Tabla 4.1 Mapa de memoria de los Vectores de Interrupción

Interrupción	Vector Normal	Vector en D-BUG12	CCR (BIT)	Habilitación local del registro (bit)
BDLC	$FFD0-$FFD1	$OB10-$OB11	I	BCRI (IE)
ATD	$FFD2-$FFD3	$OB12-$OB13	I	ATDCTL2 (ASCIE)
RESERVADO	$FFD4-$FFD5	$OB14-$OB15	I	
SCI	$FFD6-$FFD7	$OB16-$OB17	I	SC0CR2 (TIE, TCIE,RIE,ILIE)
SPI	$FFD8-$FFD9	$OB18-$OB19	I	SP0CR1 (SPIE)
Flanco del acumulador de pulsos.	$FFDA-$FFDB	$OB1A-$OB1B	I	PACTL (PAI)
Desbordamiento del Acumulador de pulsos	$FFDC-$FFDD	$OB1C-$OB1D	I	PACTL (PAOVI)
Desbordamiento del Temporizador	$FFDE-$FFDF	$OB1E-$OB1F	I	TMSK2 (TOI)
Canal 7del temporizador	$FFE0-$FFE1	$OB20-$OB21	I	TMSK1 (C7I)
Canal 6 del temporizador	$FFE2-$FFE3	$OB22-$OB23	I	TMSK1 (C6I)
Canal 5 del temporizador	$FFE4-$FFE5	$OB24-$OB25	I	TMSK1 (C5I)
Canal 4 del temporizador	$FFE6-$FFE7	$OB26-$OB27	I	TMSK1 (C4I)
Canal 3 del temporizador	$FFE8-$FFE9	$OB28-$OB29	I	TMSK1 (C3I)
Canal 2 del temporizador	$FFEA-$FFEB	$OB2A-$OB2B	I	TMSK1 (C2I)
Canal 1 del temporizador	$FFEC-$FFED	$OB2C-$OB2D	I	TMSK1 (C1I)
Canal 0 del temporizador	$FFEE-$FFEF	$OB2E-$OB2F	I	TMSK1 (C0I)
por tiempo real	$FFF0-$FFF1	$OB30-$OB31	I	RTICTL (RTIE)
IRQ	$FFF2-$FFF3	$OB32-$OB33	I	INTCR (IRQEN)
XIRQ	$FFF4-$FFF5	$OB34-$OB35	X	None
SWI	$FFF6-$FFF7	$OB36-$OB37	None	None
Instrucción ilegal (TRAP)	$FFF8-$FFF9	$OB38-$OB39	None	None
Reset por temporizador COP	$FFFA-$FFFB	$OB3A-$OB3B	None	Velocidad seleccionada en el COP
Reset por fallo de reloj	$FFFC-$FFFD	$OB3C-$OB3D	None	COPCTL (CME, FCME)
Reset externo o de encendido	$FFFE-$FFFF	$OB3E-$OB3F	None	None

CAPÍTULO 5

PROGRAMAS

Este capítulo se inicia describiendo dos herramientas de software que se utilizaron para desarrollar el programa de medición de la rapidez y la dirección del viento. Se da una descripción detallada de las principales funciones del programa con sus correspondientes diagramas de flujo, así como las subrutinas que se desarrollaron. El programa se ejecuta independientemente de la PC y esta escrito en lenguaje ensamblador propio de la familia de microcontroladores (MCU´s) MC68HC12. Este programa se encuentra almacenado permanentemente en los 768 bytes de memoria EEPROM. En el Apéndice C se encuentra el listado completo del programa en lenguaje ensamblador.

5.1 HERRAMIENTAS DE DESARROLLO

Las herramientas de software utilizadas para el desarrollo del programa que mide la rapidez y dirección del viento son dos: un programa de monitor y exploración llamado D-BUG12 que se encuentra residente en la memoria Flash EEPROM del microcontrolador y otro programa llamado IASM12 el cual es ejecutado en la PC. Ambos programas establecen una comunicación tipo serie como se ilustra en la figura 5.1. El programa D-BUG12 utiliza la interfase de comunicación serie (SCI) del sistema de desarrollo SIS68HC912 [5] y el programa IASM12 utiliza un puerto serie de la PC (COM1 o COM2) para cargar programas en la memoria RAM del MCU.

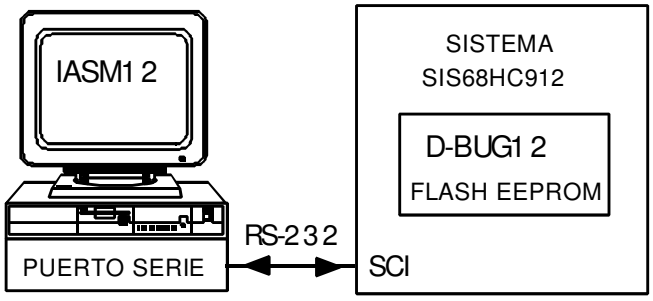


Figura 5.1.- Comunicación serie entre el IASM12 y el D-BUG12

El programa IASM12 presenta un ambiente de trabajo que contiene un editor para poder escribir y modificar programas en lenguaje ensamblador, contiene también un ensamblador para generar el código objeto en lenguaje de máquina. El IASM12 permite generar un código fuente con extensión .ASM en lenguaje ensamblador y puede producir tres archivos de salida: un archivo objeto que es el lenguaje de máquina del procesador con extensión .S19, otro archivo de listado, que es una copia del texto de entrada con varias anotaciones tales como localidades de memoria utilizadas, código de

máquina y tiempo en ciclos de reloj entre otros, con extensión .LST y un tercer archivo utilizado para simuladores e interfases de usuario. En las siguientes líneas se describen en términos generales las principales funciones del ambiente IASM12 [10] y las teclas que invocan su ejecución:

F1 **HELP** (AYUDA) Presenta en pantalla información para el correcto uso del IASM12.

F2 **SAVE** (SALVAR) Almacena el programa que se encuentre en el editor de IASM12 con la extensión .A12 en cualquier directorio, subdirectorio o bien en cualquiera de la unidades del disco duro.

F3 **LOAD** (CARGAR) Recupera un programa en el editor de IASM12.

F4 **ASSEMBLE** (ENSAMBLAR) Ensambla el programa que se encuentre cargado en el editor y crea el programa objeto con extensión .S19 (el formato S es particular de Motorola).

F5 **EXIT** (SALIR) Causa un final en la sesión de edición-ensamblado, sale del editor de IASM12 y regresa al procesador que lo invoco.

F7 **COMM** (COMUNICACIÓN) Permite abrir una ventana de comunicación en la parte inferior de la pantalla para enlazarse con el programa D-BUG12 y ejecutar sus comandos.

F9 **DOS SHELL** (Ir al DOS) Sale del editor y entra al DOS para ejecutar sus comandos. Para retornar, se escribe EXIT.

F10 **MENU** Presenta un menú de opciones donde las más importantes sirven para seleccionar los parámetros de la comunicación serie: puerto, razón de transferencia, número de bits y bits de paro.

El programa D-BUG12 contiene funciones básicas para cargar, probar y depurar programas, revisar y modificar datos en memoria y los contenidos de los registros del MCU. También permite ejecutar programas instrucción por instrucción o a la velocidad del procesador así como grabar o leer información en una unidad de almacenamiento de la PC. Los comandos con los que cuenta este programa utilizados durante la realización del programa fueron los siguientes:

G [<Dirección>]
Inicia la ejecución de un programa. El usuario puede opcionalmente especificar la dirección desde donde debe iniciar la ejecución del programa. La ejecución del programa continúa hasta que se encuentra un punto de paro o se pulsa el interruptor de reset en la tarjeta de evaluación.

HELP
Permite al usuario visualizar en la terminal remota información de los comandos disponibles en el programa de monitor D-BUG12 con la finalidad de tener referencias rápidas de ellos.

LOAD [Dirección Offset]
Carga un programa en memoria RAM o EEPROM.

MD <Dirección inicial>[<Dirección final>]
Muestra el contenido de las localidades de memoria de un determinado bloque.

MM <Dirección>[<Dato>]
Examina y/o modifica el contenido de una localidad de memoria.

RD
Muestra el contenido de los registros de la CPU.

RESET
Inicializa el microcontrolador.

RM
Modifica el contenido de los registros de la CPU.

T [<Número de pasos>]
Permite al usuario monitorear la ejecución de un programa instrucción por instrucción. Opcionalmente el usuario puede ejecutar un conjunto de instrucciones y detener el programa.

5.2 PROGRAMA PRINCIPAL

El programa diseñado para realizar la medición de la rapidez y dirección del viento se inicia cuando se enciende el instrumento y se reinicia cuando se oprime el botón de reset en el instrumento de medición. El programa comienza inicializando las direcciones de los registros, puertos, variables y vectores de interrupción que serán utilizados durante la ejecución del programa. De acuerdo a la posición del interruptor selector de función, el programa ejecuta una de las dos funciones principales: medición de la rapidez o medición de la dirección. La subrutina seleccionada se ejecuta indefinidamente, midiendo en forma continua su correspondiente variable y actualizando el indicador de lectura con el valor medido, hasta que el interruptor cambia de posición seleccionando la ejecución de la otra función. El programa termina cuando se apaga el instrumento. La figura 5.1 muestra un diagrama de flujo del programa principal y el listado del programa completo escrito en lenguaje ensamblador se muestra en el Apéndice C.

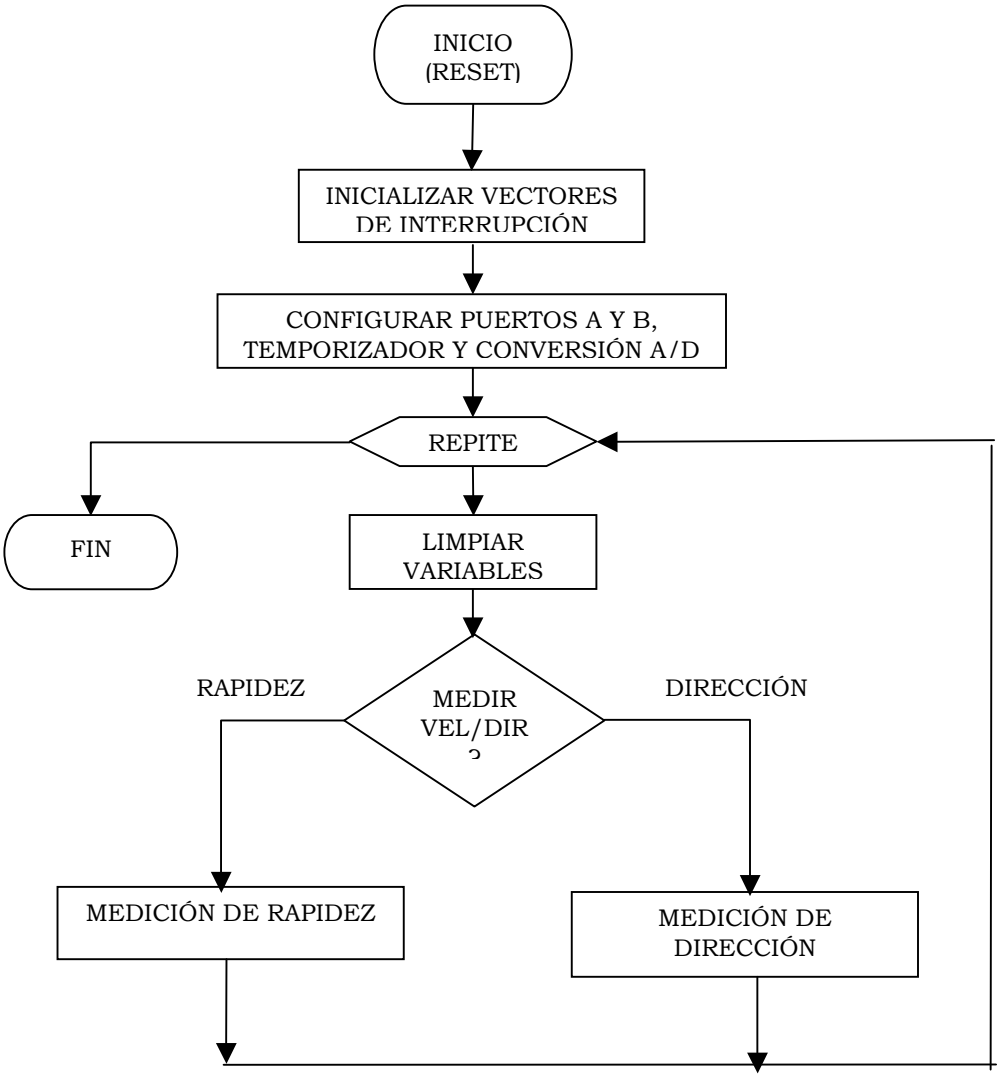


Figura 5.1.- Diagrama de flujo del programa principal para medir la rapidez y la dirección del viento

El origen del programa se encuentra ubicado dentro de la memoria EEPROM en la dirección $0D00. El programa inicia limpiando el registro de control COPCTL para que el programa se ejecute correctamente dentro de la EEPROM. Los vectores de interrupción COPF(Cop Failure Reset), COPCMF(Cop Clock Monitor Fail Reset) y RESET se inicializan con la dirección $0D00, esto permite que el programa se ejecute totalmente independiente de una PC. Como se menciona en la sección 2.3, la medición de rapidez se realiza midiendo la frecuencia ó el periodo de la señal cuadrada generada por el sensor de viento. En este trabajo se realiza la medición de periodo. Las transiciones positivas de esta señal (0->1) producen una interrupción para atender a la subrutina que mide el periodo. El vector de esta interrupción, al que llamamos VI1, es inicializado con la dirección donde se localiza la subrutina PERIODO (sección 5.3.1). Cuando se detecta un sobreflujo, se produce una interrupción para atender a la

subrutina que cuenta los sobreflujos que ocurrieron entre dos transiciones positivas. El vector de sobreflujos (VSF) es inicializado con la dirección donde se localiza la subrutina SOBREFLUJO.

Como se muestra en el diagrama de la figura 5.1, el programa continúa configurando los puertos A y B como entradas y/o salidas de acuerdo a la función que deben realizar, habilita el sistema de temporización 0 en el registro TSCR y configura una terminal del puerto T (sección 3.4) como entrada (IOC0 en la figura 2.4). En esta entrada se detecta la señal producida por el sensor de viento descrito en la sección 2.2. En el registro TCTL4 se programa el tipo de transición (positiva o negativa) que se va a detectar en la entrada IOC0 y se limpia la bandera correspondiente en el registro TFLG1. Esta bandera se pone a 1 cuando se detecta la transición activa y en ese momento el contenido actual del contador libre se transfiere al registro TC0 asociado al temporizador 0.

El programa continua habilitando el sistema de conversión Analógico/Digital por medio de los registros ATDCTL2 y ATDCTL3. Se configura el registro ATDCTL5 para que las conversiones se realicen en forma continua y también en este registro se elige una terminal del puerto A/D (AN2 en la figura 4.2) como la entrada del voltaje analógico suministrada por el sensor de viento. El resultado de la conversión se almacena en el registro ADR2H. Se limpian las variables que se utilizan en el programa y el interruptor de función indica que opción será atendida a través de la terminal PB7 del puerto B (figura 2.4) para realizar la medición correspondiente.

5.3 FUNCIÓN DE MEDICIÓN DE LA RAPIDEZ

Como se menciona en la sección 2.4, la medición de rapidez del viento se realiza midiendo el periodo de la señal cuadrada generada por el sensor y resolviendo la ecuación (2.3) Para realizar la medición de periodo se utiliza el temporizador 0 del MCU con una entrada de captura IOC0 (ver figura 2.4). Después de detectar la primera transición, el temporizador se programa por medio de registros de control para reconocer dos interrupciones: la primera se produce cada vez que se ocurre un sobreflujo en el TCNT y la segunda, cuando son detectadas las transiciones positivas (0->1) en la entrada IOC0.

Cuando se produce un sobreflujo se genera una interrupción la cual es atendida por la subrutina SOBREFLUJO en la cual se incrementa la variable NSF y se limpia la bandera correspondiente en TFLG2 para detectar el siguiente sobreflujo. Si se detecta una transición positiva se produce una interrupción y el contenido del contador libre (TCNT) se transfiere al registro de datos asociado al temporizador 0 (TCO) y se ejecuta una subrutina de atención a la interrupción, llamada PERIODO, para calcular el tiempo entre dos transiciones adyacentes. En la figura 5.2 se presenta un diagrama de flujo de la medición de rapidez del viento.

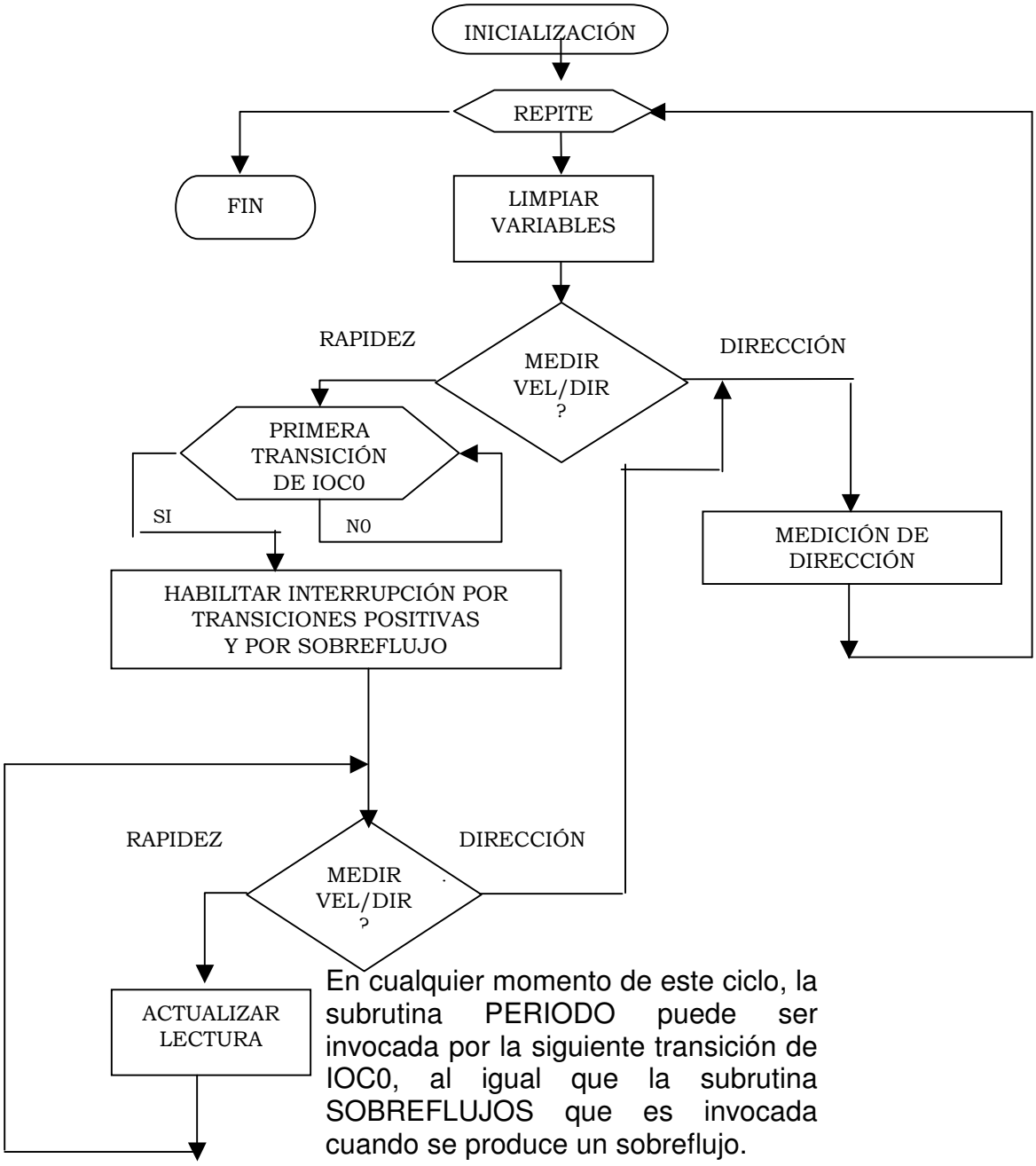


Figura 5.2.- Diagrama de flujo para medir rapidez

El programa se mantiene en un estado de espera hasta que es detectada la primera transición positiva generada por el sensor de viento y entonces el valor del contador libre (TCNT) es transferido al registro correspondiente (TC0) y almacenado en la variable T1. Después de la primera transición detectada, se habilita la interrupción por la entrada IOC0 y las subsecuentes transiciones generan una interrupción. Cuando se reconoce la interrupción por transiciones positivas, el programa realiza un salto a la dirección especificada en el vector de interrupción (VI1), la cual corresponde al inicio de la subrutina PERIODO. También se habilita la interrupción por sobrefujos. Si se

reconoce la interrupción por sobreflujos, el programa ejecutará un salto a la dirección especificada en el vector de interrupción (VSF), que corresponde al inicio de la subrutina SOBREFLUJO.

El programa verifica el estado del interruptor de función y si este se encuentra en la opción medir rapidez, actualiza los indicadores numéricos con los datos en nudos calculados en la subrutina PERIODO. Durante la actualización de los datos se producen sobreflujos y cada vez que ocurre uno, se demanda una interrupción para ser contabilizado y almacenado en la variable NSF. La variable NSF nos indica cuantos sobreflujos ocurrieron entre dos transiciones positivas. El programa regresa nuevamente a examinar el estado del interruptor de función y se mantiene actualizando los circuitos de lectura repetidamente. Cuando se detecta la siguiente transición activa en la entrada IOC0 se produce un salto del programa por interrupción para atender a la subrutina PERIODO. Esta subrutina calcula el periodo, convierte el valor calculado a nudos y actualiza los datos que son enviados al circuito de lectura. Cuando termina la subrutina el control del programa regresa al lugar donde la interrupción fue reconocida y la lectura se mantiene presentando los últimos datos en nudos calculados en PERIODO. La medición de la rapidez termina cuando se elige la opción medir dirección en el interruptor de función, ó bien cuando se apaga el instrumento.

5.3.1 SUBRUTINA PERIODO

En el diagrama de flujo de la figura 5.2, cuando se detecta la primera transición positiva en la entrada IOC0, el valor del contador libre (TCNT) es transferido al registro correspondiente (TC0) y almacenado en la variable T1. Después se habilita la interrupción por transiciones positivas en la entrada IOC0 y el programa se mantiene actualizando los circuitos de lectura y contando sobreflujos (NSF). Cuando se detecta la siguiente transición en IOC0, el contenido del contador libre TCNT nuevamente se transfiere al registro TCO y se produce una interrupción para ejecutar la subrutina PERIODO.

Como se observa en el diagrama de flujo de la figura 5.3, esta subrutina inicializa variables. Se examina primero que la variable NSF sea menor que 197 pues el valor mínimo de medición (0.5KT) corresponde a un periodo de 1,616,400μs aproximadamente. Para valores mayores a 197 se almacena en las variables DIG+1, DIG+2, DIG+3 el código en 7 segmentos correspondiente al cero ($40). Se limpia la variable NSF y se actualiza la variable T1 con el contenido del registro TC0=TCNT, se limpia la variable NSF y la bandera de transiciones positivas para terminar la subrutina. Si la variable NSF es menor que 197 se actualiza la variable T2 con el contenido del registro TC0=TCNT y se procede a calcular el periodo para almacenarlo en una variable llamada MED aplicando la ecuación (5.1), Donde T1* es el complemento de T1, esto es: T1* = FFFFh - T1 y Tsf es el tiempo en que dura un sobreflujo, con un valor Tsf = FFFFh.

$$MED = T2 + T1^* + (NSF-1)Tsf \qquad (5.1)$$

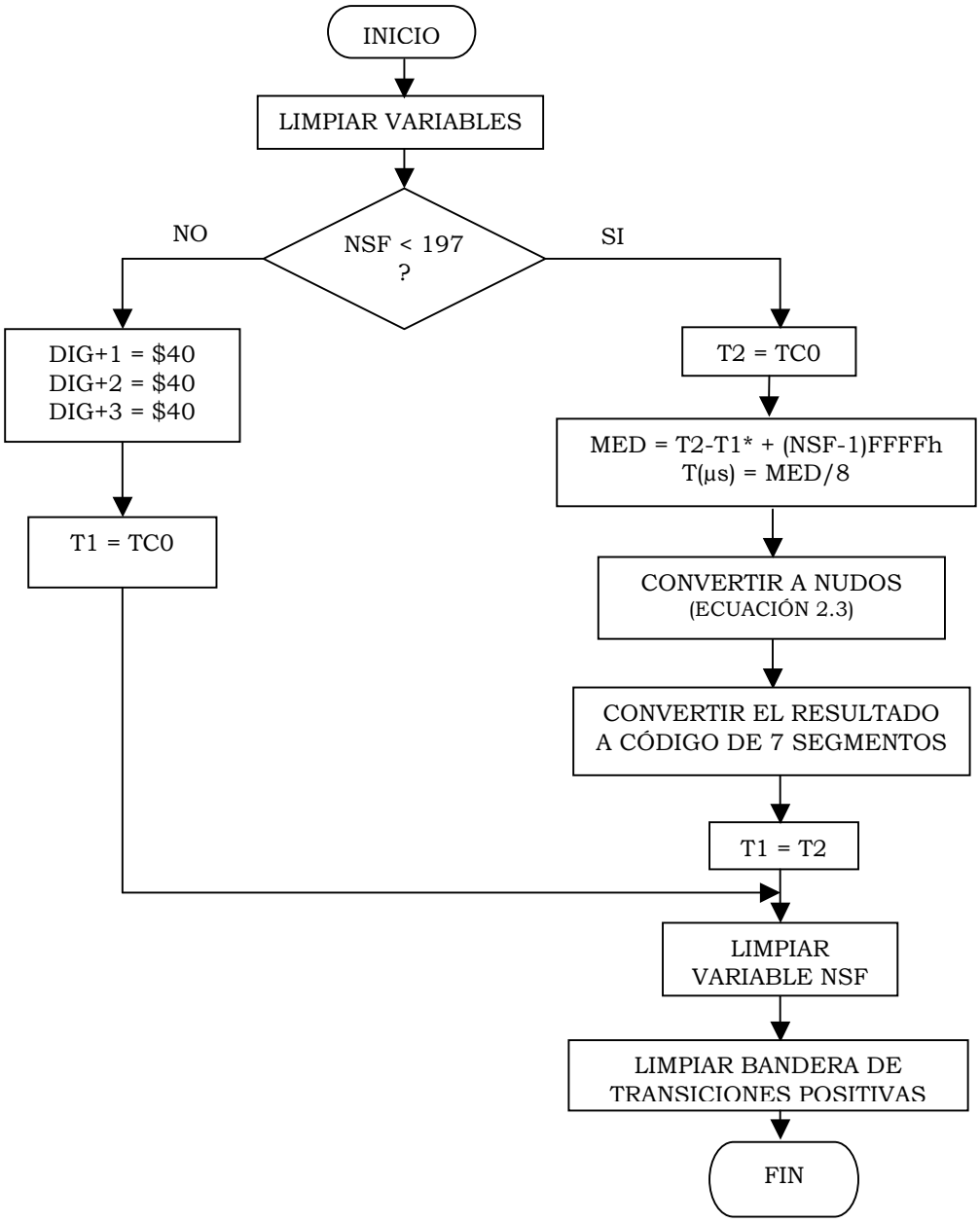


Figura 5.3.- Diagrama de flujo de la subrutina PERIODO

Para obtener el periodo en unidades de microsegundos, el valor de MED debe ser multiplicada por el periodo de la señal de reloj Tck = 0.125µs, o bien, dividirla entre la frecuencia de 8MHz. En nuestro caso resulta más sencillo dividir entre el entero 8 y obtener el resultado en microsegundos. En el diagrama de la figura 5.3 el programa continúa resolviendo la ecuación que convierte el periodo en nudos (ecuación 2.3) y el resultado obtenido es convertido a 6 dígitos, cada uno en código de 7 segmentos para ser guardados en localidades de memoria (DIG, DIG+1, DIG+2, DIG+3, DIG+4, DIG+5).

Por último, se guarda el valor de la variable T2 en la variable T1, se limpia la variable NSF, la bandera de transiciones positivas y el programa regresa al punto donde se reconoció la interrupción.
En la figura 5.4 se muestran diferentes casos de la señal a medir IOC0 con respecto a la ocurrencia de los sobreflujos producidos por el contador libre (TCNT). En el primer periodo de TC0(0) a TC0(1) no hay sobreflujos (NSF=0) y de la ecuación 5.1 el periodo es:

$$MED = T2+T1^*+(NSF-1)Tsf = T2 + (FFFFh - T1) - FFFFh = T2-T1$$

En los dos siguientes periodos, de TC0(1) a TC0(2) y de TC0(2) a TC0(3), T1>T2 y bajo esta condición siempre se tiene por lo menos un sobreflujo. Con NSF=1 de la ecuación 5.1 se tiene:

$$MED = T2+T1^*+(NSF-1)Tsf = T2 + T1^*$$

En el periodo de TC0(3) a TC0(4), T1>T2 y NSF=2, entonces:

$$MED = T2+T1^*+(NSF-1)Tsf = T2 + T1^* + FFFFh$$

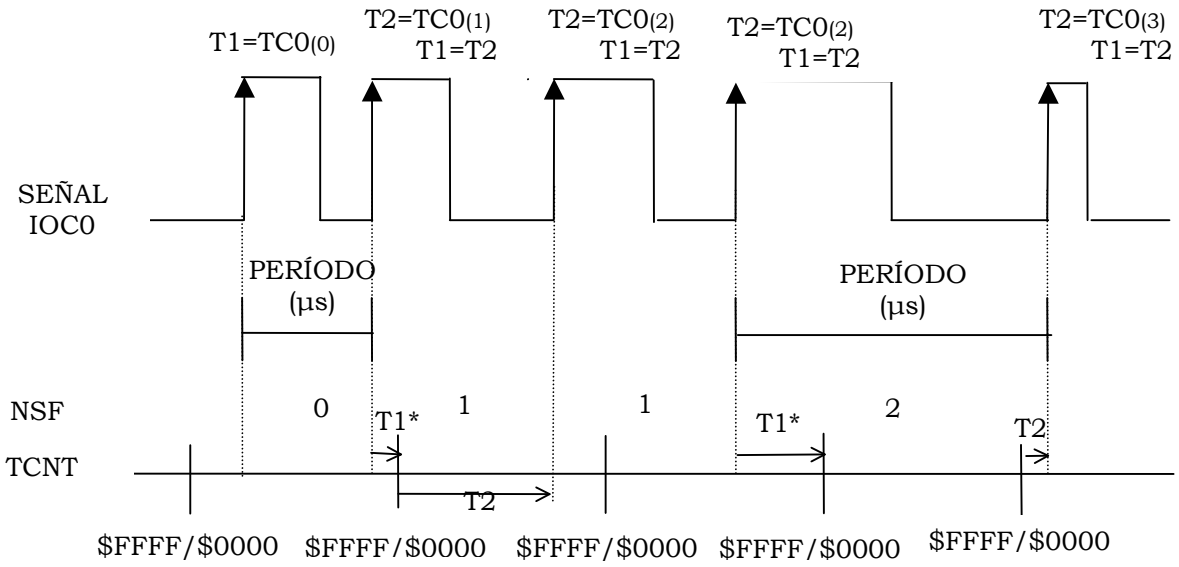


Figura 5.4. Diferentes casos de la señal IOC0 respecto a la ocurrencia de sobreflujos.

En el diagrama de la figura 5.3 el programa continúa resolviendo la ecuación que convierte el período en nudos (ecuación 2.3) y el resultado obtenido es convertido a código de 7 segmentos [12] para ser guardado en localidades de memoria (DIG, DIG+1, DIG+2,DIG+3,DIG+4,DIG+5). Por último, se guarda el valor de la variable T2 en la variable T1, se limpia la variable NSF y la bandera de transiciones positivas. La subrutina regresa al punto donde fue invocada y el programa continua ejecutándose.