ESCUELA POLITÉCNICA DEL EJÉRCITO
SEDE LATACUNGA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA ESPECIALIDAD INSTRUMENTACIÓN

"Diseño e Implementación de una Red I2C para Instrumentación Industrial"

Luis Alberto Barrionuevo Bassante
Edgar Patricio Barrionuevo Bassante

Latacunga – Ecuador
2005

CERTIFICACIÓN

Certificamos que el presente trabajo fue desarrollado por Luis Alberto Barrionuevo Bassante y Edgar Patricio Barrionuevo Bassante, bajo nuestra supervisión.

……

Ing. Eddie Galarza

DIRECTOR DE TESIS

……

Ing. Nancy Guerrón.

CODIRECTOR DE TESIS

AGRADECIMIENTO

Nuestros sinceros agradecimientos a toda nuestra familia y a las personas que han estado junto a nosotros, por su apoyo incondicional demostrado durante este tiempo de esfuerzo y sacrificio.

Patricio Barrionuevo B.

Luis Barrionuevo B.

DEDICATORIA

Dedicamos este trabajo fruto del esfuerzo y dedicación:

Con mucho cariño a la memoria de nuestra madre que pese a su ausencia, dejó sembrado en nosotros el espíritu de lucha y honradez en los actos que enrumban nuestras vidas.

En especial a nuestro padre, muestra de lucha constante y sacrificio por sus Hijos.

A mis Hermanos, creadores del pilar en el que uno se arrima en los momentos más difíciles.

A toda nuestra familia que nos extendió la mano cuando de verdad lo necesitamos.

A nuestros amigos y conocidos que supieron ayudarnos.

Patricio Barrionuevo B.

Luis Barrionuevo B.

ÍNDICE

PAG.

CAPÍTULO II

CAPÍTULO III

CAPÍTULO IV

CAPÍTULO V

CAPÍTULO VI

ANEXOS

ANEXO A: Tarjetas Diseñadas

ANEXO B: Programa de los Microcontroladores

ANEXO C: Manual del Usuario

ANEXO D: Acrónimos principales

INTRODUCCIÓN

El presente proyecto de titulación está enfocado a investigar la tecnología de la Red I2C, cual es el protocolo de comunicación utilizado, la manera de implementar aplicaciones utilizando esta tecnología, además de cómo la aplicación de la red I2C puede ayudar al desarrollo de la Instrumentación Industrial.

Se inicio pensando en el diseño de un sistema que permita la transmisión y recepción de señales estandarizadas en el área de instrumentación. Por tal motivo al final de este trabajo se realiza un prototipo demostrativo de la recepción y transmisión de señales analógicas y digitales, utilizando para el efecto una red I2C.

Ponemos a disposición de todos ustedes. “Diseño e Implementación de una Red I2C para Instrumentación Industrial y la realización de un prototipo para la transmisión y recepción de señales analógicas y digitales empleando la red I2C.

CAPÍTULO I

1. BUS I2C

Para simplificar la interconexión de dispositivos al microprocesador, Philips desarrolló un sencillo bus bidireccional basado en dos hilos por el que se trasmiten los datos vía serie.

Las líneas SDA y SCL son bidireccionales y están polarizadas a positivo mediante resistencia de "pull-up" de forma que en reposo están a nivel alto.
En el bus exciten un maestro (que generan la señal de SCL y controlan la comunicación) y esclavos que responden a peticiones del maestro a velocidades relativamente lentas. El bus I2C emplea comunicación serie, utilizando un conductor para manejar el pulso de reloj y otro para intercambiar datos.

1.1. GENERALIDADES

1.1.1. VERSIÓN 1.0.

Esta versión del bus I2C fue desarrollada en 1992, presenta las siguientes características:

- La programación por software de la dirección de un esclavo es omitido.
- Se añade el modo rápido, esto permite un incremento de hasta 400 kbits/s
- Se añade el direccionamiento a 10 bits. Este permite 1024 direcciones adicionales del esclavo.

1.1.2. VERSIÓN 2.0.

A partir de 1998 el bus I2C se ha convertido en un estándar en el mundo, existen más de 50 compañías con licencia de uso. Muchas de las aplicaciones de hoy requieren altas velocidades y bajos suministros de voltajes. En esta versión actualizada del bus I2C, incluye los siguientes requerimientos y modificaciones.

- El nivel bajo de salida y la histéresis de los dispositivos con una fuente de voltaje de 2V e inferior, ha sido adaptada para encontrar el margen de ruido requerido y mantener compatibilidad con dispositivos que emplean fuentes de voltaje superior.
- El requerimiento de 0.6V a 6mA para los estados de salida de los dispositivos del modo rápido han sido omitidos.

1.1.3. VERSIÓN 2.1.

A partir del 2000, la versión 2.1 del bus I2C incluye las siguientes modificaciones menores.

- Después de una condición de arranque repetida en el modo Hs (módulo de alta velocidad), es posible extender la señal del reloj
- Algunos parámetros de tiempo en el modo Hs han sido disminuidos.

1.1.4. VENTAJAS PARA LOS DISEÑADORES.

Los ICs (circuitos integrados) compatibles con el bus I2C permiten el diseño de un sistema de rápido progreso, que va desde el diseño de diagramas de bloque funcional, hasta su prototipo.

Algunas de las características del bus I2C con los ICs, que son particularmente atractivos para los diseñadores, son las siguientes:

- El protocolo de direccionamiento integrado y la transferencia de datos, permite a los sistemas, tener un software completamente definido.
- Los mismos ICs pueden ser usados a menudo en muchas aplicaciones diferentes.

- Los ICs pueden ser añadidos o quitados de la red sin que esto afecte al resto del sistema.
- Las etapas de diagnóstico y depuración son simples; los errores de funcionamiento pueden ser determinados inmediatamente.
- El tiempo de desarrollo del software puede ser reducido al ensamblar una biblioteca de módulos de software reutilizables.
- Muy bajo consumo de corriente.
- Alta inmunidad al ruido.

- Amplio rango del valor de la fuente de voltaje.
- Amplio rango de operación de temperatura

1.1.5. VENTAJAS PARA LOS FABRICANTES.

Los ICs compatibles con el bus I2C no sólo ayudan a los diseñadores, también dan muchas ventajas al equipo de fabricantes, debido a que:

- El empleo de dos alambres para la comunicación, minimiza la interconexión de los elementos máster y esclavo, del bus I2C; de esta manera los ICs tienen menos pines y no hay tantas pistas.
- La capacidad del máster del bus I2C permite una verificación y alineamiento rápido en el equipo del usuario final mediante conexiones externas a una línea de ensamblaje.
- El protocolo del bus I2C integrado, elimina totalmente la necesidad de decodificadores de dirección.
- La compatibilidad de los ICs con el bus I2C incrementa la flexibilidad del sistema de diseño.
- Permite la construcción sencilla de variaciones del sistema y es de fácil perfeccionamiento para mantener los diseños al día, de esta forma, una familia entera de equipamiento puede ser desarrollado alrededor de un modelo básico.
- Los perfeccionamientos para el equipo nuevo o los modelos con características reforzadas (memoria extendida, control remoto, etc.)

pueden ser creados simplemente en base a las limitaciones de los ICs en el bus.

1.1.6. ESPECIFICACIONES.

Para las aplicaciones de control digital orientadas a 8 bits como aquellas que requieren microcontroladores, ciertos criterios de diseño pueden ser establecidos:

- Un sistema completo, generalmente consiste de por lo menos un microcontrolador y otros periféricos como memorias y / o expansores de I/O.
- El costo de conexión de los dispositivos dentro del sistema deben ser minimizados.
- Un sistema que realiza una función de control no requiere de transferencia de datos a altas velocidad.
- La eficiencia total depende de los dispositivos seleccionados y de la naturaleza de la estructura de interconexión del bus.

Para obtener un sistema que satisfaga estos criterios, se necesita una estructura del bus en serie. Los buses en serie no tiene las mismas capacidades de los buses en paralelo, pero requieren de menos cableado y pocos pines de conexión para los ICs. Sin embargo el bus no es simplemente un medio de interconexión, incorpora todos los formatos y procedimientos para la comunicación dentro del sistema.

Para la comunicación de los dispositivos del bus es necesario tener algún tipo de protocolo de comunicación, el cual evita confusión de comunicación, pérdidas de datos y el bloqueo de la información. Además, los dispositivos rápidos deben ser capaces de comunicarse con los dispositivos lentos sin inconvenientes. El sistema no debe ser dependiente de los dispositivos a él conectados.

Debe existir un procedimiento para determinar que dispositivo controlará el bus y cuando será necesario establecer control.

1.1.6.1. CONCEPTO DEL BUS I2C

El bus I2C apoya cualquier proceso de fabricación con ICs (NMOS, CMOS, bipolar). Los datos en serie (SDA) y el reloj serial (SCL) llevan la información entre los dispositivos conectados al bus. Cada dispositivo es reconocido por una única dirección (ya sea un microcontrolador o una memoria), y puede operar como un transmisor o receptor, dependiendo de la función que debe cumplir. Obviamente, un driver LCD es sólo un receptor, mientras que una memoria puede recibir y transmitir datos.

Un máster, es el dispositivo que inicia una transferencia de datos en el bus y genera las señales del reloj que permiten dicha transferencia. En ese momento, cualquier dispositivo direccionado es considerado como un esclavo.

Tabla 1.1. Definición de la terminología del bus I2C

TÉRMINO	DESCRIPCION
Transmisor	Dispositivo que envía datos hacia el bus.
Receptor	Dispositivo que recibe la información desde el bus
Máster	El dispositivo que inicia una transferencia, genera las señales del reloj y finaliza dicha transferencia.
Esclavo	Un dispositivo direccionado por un máster.
Multi-máster	Más de un dispositivo máster puede intentar controlar el bus al mismo tiempo, sin afectar el mensaje.
Arbitraje	Procedimiento para asegurar que, si más de un máster trata de controlar simultáneamente el bus, solamente a

	uno se le permita hacerlo, así el mensaje no será dañado.
Sincronización	Procedimiento para sincronizar las señales del reloj de dos o más dispositivos.

El bus I2C es un bus multi-máster, esto significa que más de un dispositivo capaz de controlar el bus puede ser conectado a la red. Los másters generalmente son microcontroladores. Si se considera el caso de una transferencia de datos entre dos microcontroladores conectados al bus I2C, las características notables de la relación del máster-esclavo y transmisor-receptor encontradas no son permanentes, y depende solo de la dirección que la transferencia de datos tenga en ese momento. En la Figura 1.1 se ilustra la transferencia de datos entre dos microcontroladores:

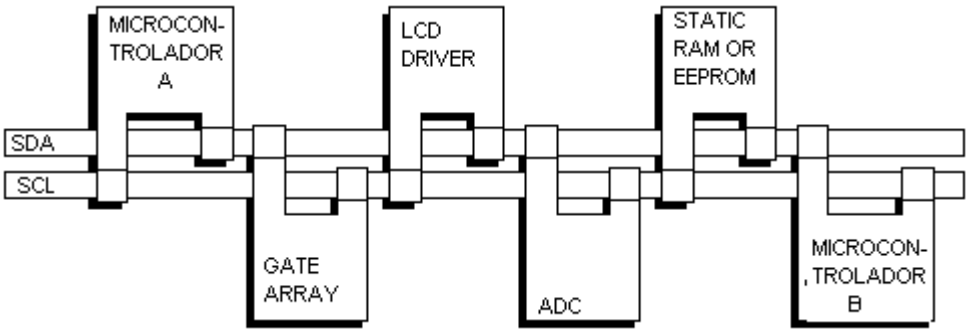


Figura 1.1. Ejemplo de una configuración del bus I2C que usa dos microcontroladores.

El microcontrolador A quiere enviar información al micro controlador B:

- El microcontrolador A (máster), direcciona a el microcontrolador B (esclavo)
- El microcontrolador A (transmisor – máster), envía los datos al microcontrolador B (esclavo-receptor)
- El microcontrolador A finaliza la transferencia.

Si el microcontrolador A quiere recibir información desde el microcontrolador B:

- El microcontrolador A(máster) direcciona a el microcontrolador B (esclavo).
- El microcontrolador A (máster-receptor) recibe los datos desde el microcontrolador B(esclavo-transmisor).
- El micro controlador A termina la transferencia.

Aún en este caso, el máster (microcontrolador A) genera la señal de reloj y termina la transferencia de datos.

Al conectar varios microcontrolador al bus I2C existe la posibilidad de que más de un máster pueda tratar de iniciar la transferencia de datos al mismo tiempo.

Para evitar este caos que podría resultar de tal evento, un procedimiento de arbitraje ha sido desarrollado.

Si dos o más másters tratan de poner información en el bus, el primero en producir un ‘uno ' cuando el otro produce o pone un ‘cero' perderá el arbitraje. La señal del reloj durante el arbitraje es una combinación sincronizada de los señales de reloj generados por los másters que usan el wired-AND conectado a la línea de SCL.

1.1.6.2. CARACTERÍSTICAS GENERALES

Tanto el SDA como el SCL son líneas bi-direccionales conectadas a una fuente positiva de voltaje. Cuando el bus está libre, las dos líneas están en estado alto. La etapa de salida de los dispositivos conectados al bus debe tener un drenaje abierto o colector abierto para ejecutar la función del wired-AND. Los datos en el bus I2C pueden ser transferidos a 100kbit/s o más en el modo estándar, en el modo rápido superior a 400 kbit/s y superior a 3.4 Mbit/s en el modo de alta velocidad.

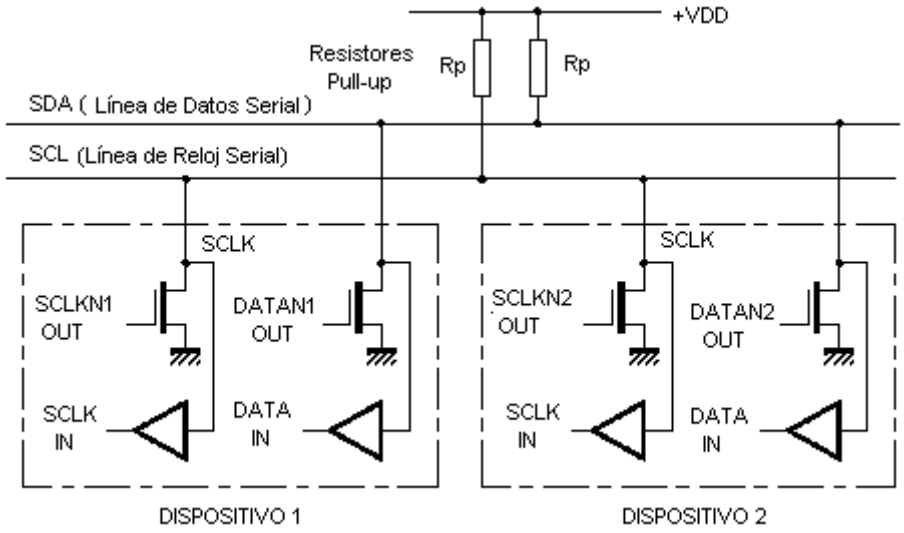


Figura 1.2. Configuración del bus I2C que usa dos microcontroladores.

1.2. TRANSFERENCIA DE INFORMACIÓN.

Debido a la variedad de dispositivos de diferente tecnología (CMOS, NMOS, bipolares) que pueden ser conectados al bus I2C los niveles lógicos “Ø” (bajo) y “1” (alto) no son fijos y dependen del nivel asociado de VDD (fuente de voltaje directo). Un pulso del reloj se genera para cada bit transferido.

1.2.1. VALIDEZ DE LOS DATOS

Los datos en la línea SDA deben ser estables durante el período ALTO del reloj. El estado ALTO o BAJO de la línea de datos puede cambiar sólo cuando la señal del reloj en la línea SCL esté BAJA.

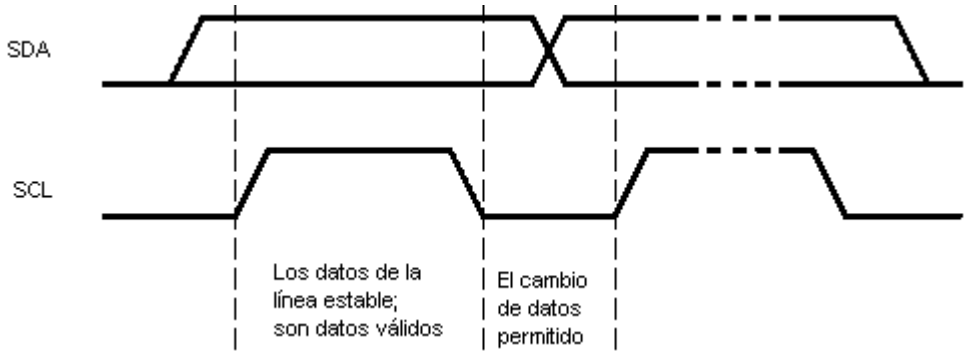


Figura 1.3. Transferencia del bit en el bus I2C.

1.2.2. CONDICIÓN DE INICIO Y PARADA

La condición de INICIO (S) de transferencia de datos, se produce cuando aparece una transición de “1” a “0”en la línea SDA, manteniendo la línea de reloj SCL a “1”.

El FIN de transferencia de datos o condición de PARADA (P) es da por la transición de “0” a “1” en la línea SDA cuando la señal de reloj SCL está a “1”. (Ver Fig.1.4)

Las condiciones de inicio o parada son siempre generadas por el dispositivo máster. Se considera que el bus está ocupado después de la condición de arranque, para que el bus se libere nuevamente debe pasar un cierto tiempo luego de la condición de parada.

El bus permanece ocupado si se genera el arranque repetido en vez de una condición de parada. Al respecto, el arranque (S) y las condiciones de arranque repetido (Sr) son funcionalmente iguales, por lo tanto, el símbolo S será usado como un término genérico para representar a las dos condiciones de arranque y parada repetido.

1.2.3. FORMATO DE BYTE

Cada byte transmitido por la línea SDA debe tener 8 bits de longitud. El número de bytes que puede ser transmitido para cada transferencia de información no es restringido. Cada byte debe estar seguido por un bit de “reconocimiento”. Los datos son transferidos con el bit más significativo (MSB) al inicio del byte (Ver Fig.1.4). Si un esclavo no puede recibir o transmitir otro byte completo de datos hasta que se haya ejecutado alguna otra acción, por ejemplo el servicio de una interrupción interna, puede mantener la línea del reloj SCL en bajo para forzar al máster a un estado de espera. La transferencia de datos continua cuando el esclavo está listo para recibir o transmitir otro byte y la línea del reloj SCL queda libre. Un mensaje que empieza con tal dirección puede ser finalizada por la generación de una condición de parada, aun durante la transmisión de un byte.

1.2.4. RECONOCIMIENTO

Para los que datos que se transmiten en paquetes de 8 bits, no hay límite en cuanto a número de bytes, pero después de cada byte se debe intercalar un bit de reconocimiento (ACK) por parte del dispositivo receptor.

Si el Esclavo es el receptor y no genera el bit de reconocimiento después de cada byte, el Máster aborta la transferencia de información generando un bit de parada (STOP).

Dado que las etapas de salida son de drenaje abierto, los niveles lógicos “0”son dominantes en las líneas, por tanto el Esclavo debe dejar su salida SDA a “1”para que el Máster pueda generar la condición de parada.

Si el dispositivo Máster es el receptor, genera un ACK tras cada byte recibido, permitiendo al Esclavo que continúe enviando bytes. Si el Máster decide finalizar la transferencia, genera un bit de STOP en lugar de colocar un bit de reconocimiento.

Si el Esclavo necesita retardar el envío del siguiente byte (porque no lo tiene todavía disponible), puede situar la línea SCL en estado bajo y forzar al Máster a situarse en un estado de “Espera” puesto que no se podrían generar flancos en SCL

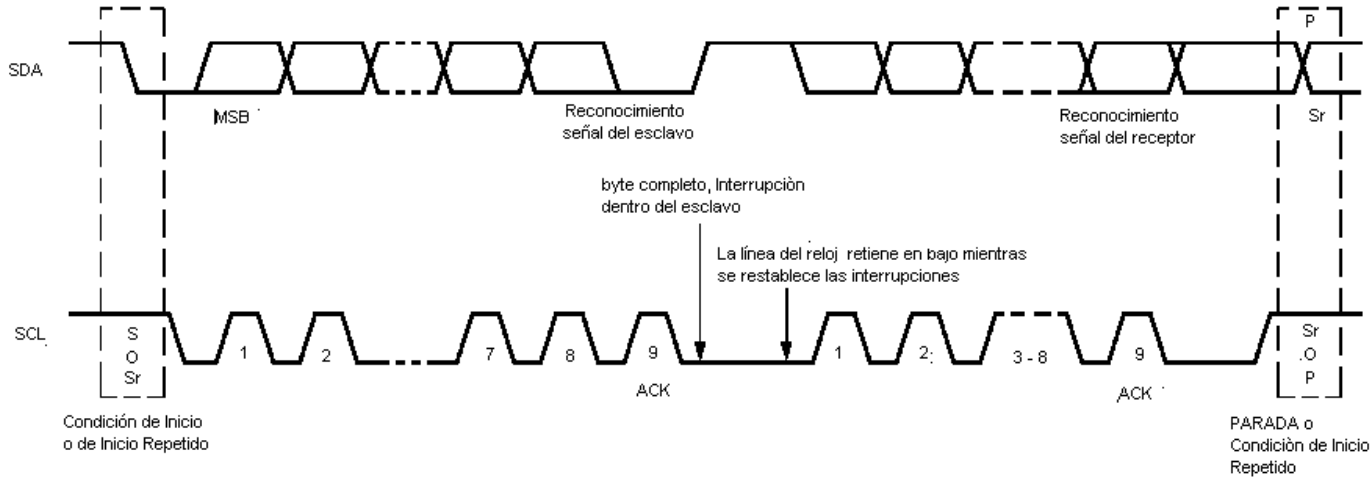


Figura 1.4. Transferencia de datos en el bus I2C

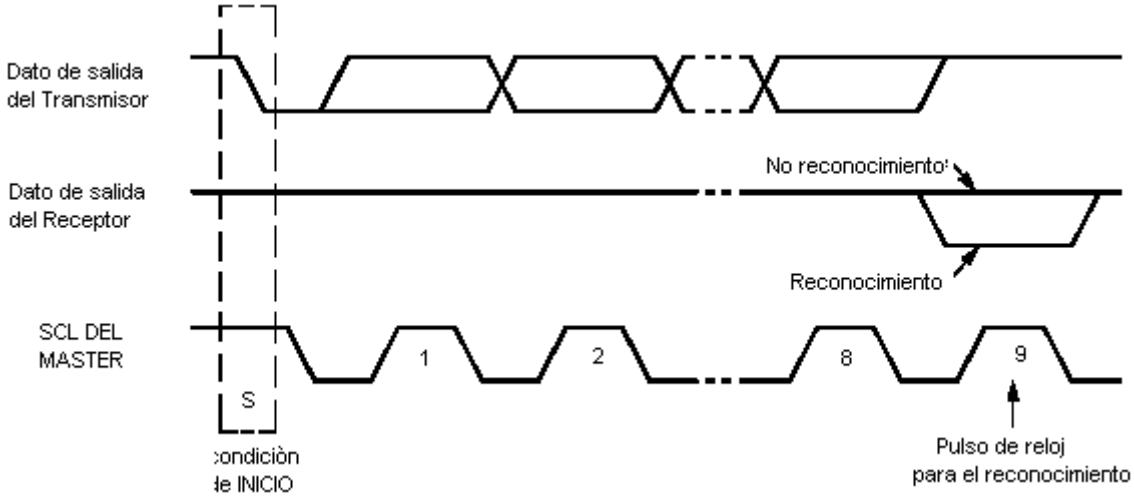


Figura 1.5. Reconocimiento en el bus I2C

1.3. ARBITRAJE Y GENERACION DEL RELOJ

1.3.1. SINCRONIZACIÓN.

La sincronización de los relojes se produce después de que los dispositivos hayan iniciado el arbitraje del bus. Las transiciones en la línea de reloj (SCL) inician el proceso de conteo de los tiempos en los estados alto y bajo para cada reloj. Si un dispositivo pone su etapa de salida de reloj en estado bajo, resultaría dominante por el wire-and (and cableada) existente en la línea SCL

Por tanto la transición de 0 a 1 en la etapa de salida de un reloj puede que no cambie el estado de la línea SCL si hay otro dispositivo que la mantiene a Ø y deberá introducir esperas internas hasta que SCL pase a 1. Cuando esto se produce, todos los dispositivos empiezan a contar el tiempo que tienen que estar en estado alto. El primer dispositivo que cambia a 0 marca el tiempo en estado alto de la línea SCL

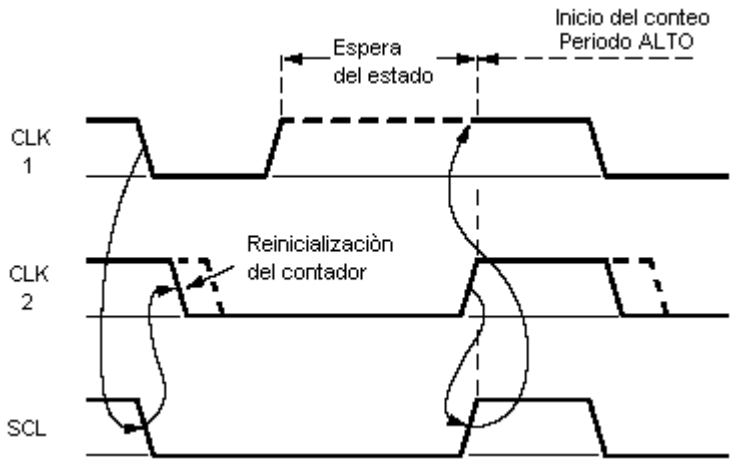


Figura 1.6. Reloj de sincronización durante el procedimiento de arbitraje.

Cuando todos los dispositivos involucrados han registrado su período de BAJA, la línea del reloj será liberada e irá a ALTA. Entonces no habrá diferencia entre el reloj de los dispositivos con el estado de la línea SCL y todos los dispositivos empezarán el conteo de sus períodos ALTA. El primer dispositivo en completar su período de ALTA pondrá de nuevo la línea SCL en estado bajo, de esta forma un reloj sincronizado SCL genera su período bajo y alto determinado el período en alto más corto.

1.3.2. ARBITRAJE

Un máster puede empezar una transferencia sólo si el bus está libre. Dos o más másters pueden generar una condición de inicio dentro del tiempo de mantenimiento mínimo (tHD.STA), lo cual produce una condición de arranque del bus. El arbitraje se lleva a cabo en la línea SDA, mientras que la línea SCL está en un nivel alto, de tal forma el máster que transmita un nivel alto, mientras que el otro máster esté transmitiendo un nivel bajo, dejará de transmitir datos a la etapa de salida porque el nivel en el bus no corresponde a su propio nivel. El arbitraje puede continuar a través de muchos bits, su primer paso es la comparación de la dirección de los bits. Si los másters están tratando de dirigir cada uno al mismo dispositivo, el arbitraje continuará con la

comparación de los datos de los bits si es que son másters transmisores, o reconocimientos de los bits, si son máster receptores. Debido a que la dirección y la información de los datos en el bus están determinadas por el máster ganador, la información no se pierde durante el proceso de arbitraje. Un máster que pierde el arbitraje puede generar pulsaciones del reloj hasta el final del byte en el cual perdió el arbitraje.

Cuando un máster de modo Hs tiene un código único de 8 bits, siempre terminará el arbitraje durante el primer byte, La figura 1.7 muestra el procedimiento de arbitraje de dos másters.

Al momento hay una diferencia entre el nivel de datos internos del máster que genera los datos 1 y el nivel actual de la línea SDA, sus datos de salida están desconectados, lo cual significa que el nivel de salida alto está conectado al bus. Esto no afectará la transferencia de datos iniciada por el máster ganador.

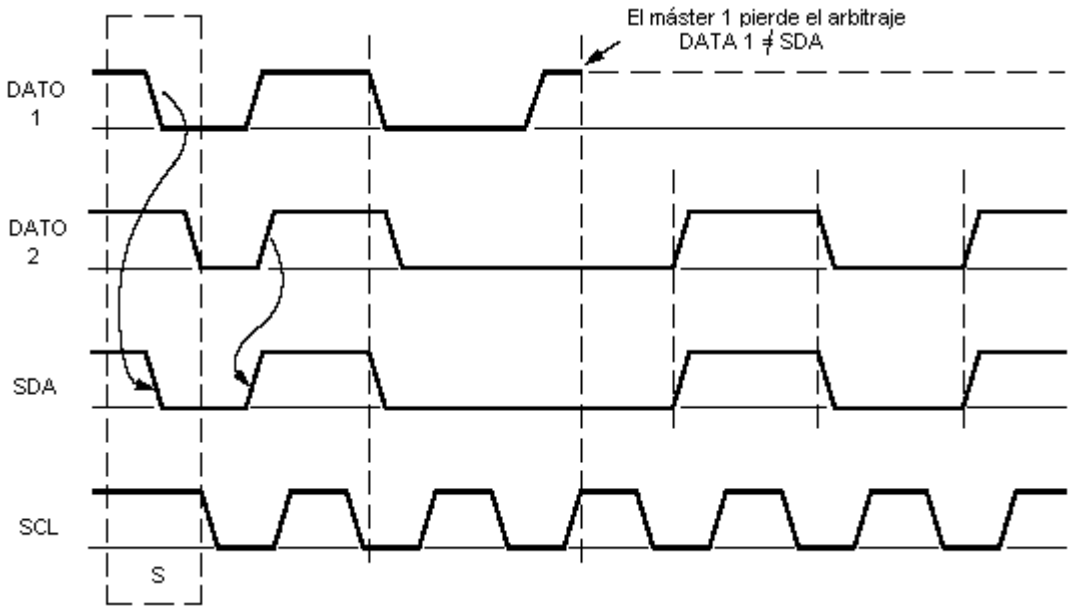


Figura 1.7. El procedimiento del arbitraje de dos másteres.

En este ejemplo, el máster 1 pierde el arbitraje.

Se debe tener en cuenta que el arbitraje no es permitido entre:

- Una condición de arranque repetido y un bit de datos.
- Una condición de parada y un bit de datos.

- Una condición de arranque repetido y una condición de parada.

1.3.3. USO DEL RELOJ Y MECANISMO DE SINCRONIZACIÓN PARA EL HANDSHAKE.

Además de ser usado durante el procedimiento de arbitraje, el mecanismo de sincronización puede ser usado para facilitar a los receptores la transferencia rápida de datos o con el nivel de byte. En el nivel de byte, un dispositivo es capaz de recibir los bytes de datos a un porcentaje rápido, pero necesita más tiempo para guardar un byte recibido o preparar otro byte para ser transmitido. Los esclavos pueden mantener la línea SCL baja después de la recepción y el reconocimiento de un byte para forzar al máster a un estado de espera hasta que el esclavo esté listo para la transferencia del siguiente byte en un procedimiento tipo handshake.

En el nivel del bit, un dispositivo como un microcontrolador, puede disminuir el tiempo de la señal de reloj en la red extendiéndolo lentamente cada período BAJO. Por lo tanto la velocidad de cualquier máster se adapta al porcentaje de la operación interna de este dispositivo.

En el modo Hs, esta característica puede ser usada sólo en el nivel Byte

1.4. DIRECCIONAMIENTO

El procedimiento de direccionamiento para el bus se realiza de tal manera que el primer byte después de la condición de arranque generalmente determina cual esclavo será seleccionado por el máster. La excepción es la dirección llamada “general” la cual se dirige a todos los dispositivos. Cuando se usa esta dirección, todos los dispositivos deberían responder en teoría con un reconocimiento, sinembargo, los dispositivos pueden estar configurados para ignorar esta dirección. El segundo byte de la dirección de la llamada general, define la acción a ser tomada.

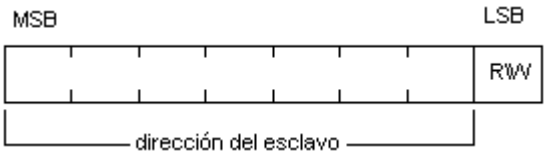


Figura 1.8. El primer byte después de la condición de arranque.

1.4.1. DEFINICIÓN DE BITS EN EL PRIMER BYTE

Los primeros 7 bits del primer Byte forman parte de la dirección del repetidor. El octavo bit es el LSB. (bit menos significativo) que determina la dirección del mensaje.

Un “cero” en la posición menos significativo del primer byte significa que el máster escribirá la información a un esclavo seleccionado, un “uno” en esta posición significa que el máster leerá la información desde el esclavo.

Cuando se envía una dirección luego de la condición de arranque, cada dispositivo en el sistema compara los primeros 7 bits con su dirección. Si ellos coinciden, el dispositivo se considera a sí mismo como direccionado por el máster como un esclavo-receptor o como un esclavo-transmisor dependiendo del bit R/W

Una dirección del esclavo puede ser formada por una parte fija y por una parte programable. Puesto que probablemente habrá varios dispositivos idénticos en el sistema, la parte programable de la dirección del esclavo da el número máximo posible de los dispositivos que pueden ser conectados al bus. El número de bits de dirección programable de un dispositivo depende del número de pines disponibles.

Dos grupos de 8 direcciones (0000XXX y 1111XXX) son reservados para los propósitos indicados en la tabla 1.2. La combinación de bit 11110XX de la dirección del esclavo está reservada para el direccionamiento a 10 bits.

Tabla 1.2. La definición de los bits en el primer byte

LA DIRECCIÓN DEL ESCLAVO	R/W BIT	DESCRIPCION
0000 000	0	Dirección de llamada general
0000 000	1	byte de arranque
0000 001	X	Dirección del BUS
0000 010	X	Formato reservado para buses diferentes
0000 011	X	Reservado para propósitos futuros
0000 1XX	X	El código de máster del modo Hs
1111 1XX	X	Reservado para propósitos futuros
1111 0XX	X	Direccionamiento a 10bits

1.4.2. DIRECCIONES DE LLAMADA GENERAL

La dirección de llamada general se utiliza para el direccionamiento de cada dispositivo conectado al bus. Sin embargo, si un dispositivo no necesita cualquiera de los datos proporcionado por la estructura de la llamada general, puede ignorar esta dirección no emitiendo un reconocimiento.

Si un dispositivo requiere los datos de una dirección de la llamada general, reconocerá esta dirección y se comportará como un esclavo-receptor, el segundo y los siguientes bytes serán reconocidos por cada

esclavo-receptor capaz de manejar estos datos. Un esclavo que no puede procesar uno de estos bytes debe ignorarlo con el no reconocimiento. El significado de la dirección de la llamada general siempre se especifica en el segundo byte.

Se considera los siguientes casos:

- Cuando el bit menos significativo B es un “cero”
- Cuando el bit menos significativo B es un “uno”

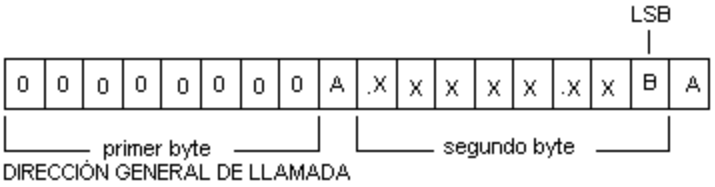


Figura 1.9. El formato de dirección de llamada general.

Cuando el bit es un “cero”, el segundo byte tiene la siguiente definición:

- 00000110 (H‘06’). Restablece y escribe la parte programable de la dirección del repetidor mediante el hardware, al recibir esta secuencia en el 2° byte, todos los dispositivos diseñados para responder a la dirección de la llamada general se restablecerán y guardarán la parte programable de su dirección. Debe asegurarse que ningún dispositivo mantendrá en un nivel bajo la línea SDA o SCL después de aplicar la fuente de voltaje, debido a que estos niveles bajos bloquearían el bus.
- 00000100 (H‘04’).Escribe la parte programable de la dirección del esclavo mediante el hardware. Todos los dispositivos que definen la parte programable de su dirección mediante el hardware (los cuales responden a la dirección general de llamada) mantendrán esta parte programable en la recepción de la secuencia del 2 byte.
- 00000000 (H‘00’).Este código no puede ser usado como segundo byte.
- El resto de códigos no han sido fijados y los dispositivos deben ignorarlos.

Cuando el bit es un “uno”, la secuencia del 2° byte es una llamada general. Por hardware entonces, un máster no conoce con anticipación

que dispositivo tiene que realizar la transmisión de datos, por lo cual, este solamente generará una dirección general de hardware y su propia dirección de identificación para el sistema. Esto se ilustra en la figura 1.10.

El bit #7 permanece en el segundo byte que contiene la dirección del hardware máster, esta dirección es reconocida por un dispositivo inteligente (Ejemplo: un microcontrolador) conectado al bus el cual dirigirá la información desde el hardware máster. Si el hardware máster puede también actuar como un repetidor, la dirección del repetidor es idéntica a la dirección del máster.

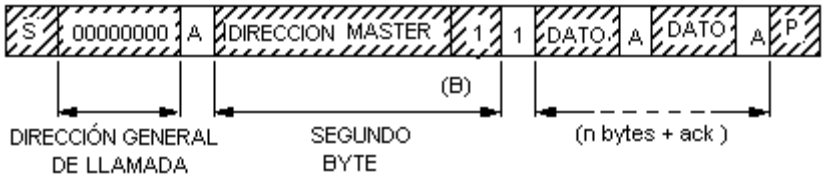


Figura 1.10. Los datos se transfieren desde el hardware del transmisor_ máster

1.4.3. BIT DE INICIO O ARRANQUE

Un microcontrolador que contiene un chip con interfaz del bus I2C puede ser programado solo para ser interrumpida por los pedidos del bus. Cuando el dispositivo no tiene tal interfase, puede monitorear constantemente el bus vía software. Mientras más veces el microcontrolador monitoree el bus, menos tiempo puede llevar a cabo sus funciones normales. Por lo tanto, hay una diferencia de velocidad entre los dispositivos rápidos de hardware y un microcontrolador relativamente lento, el cual se reinstala en el software. En este caso, la transferencia de datos puede estar precedida por un procedimiento de arranque, la cual es mucho más larga que lo normal.

El procedimiento de arranque consta de:

- Una condición de arranque (s)
- Un byte de arranque (00000001)
- Un reloj de pulsación de reconocimiento (ACK)
- Una condición de arranque repetido (Sr)

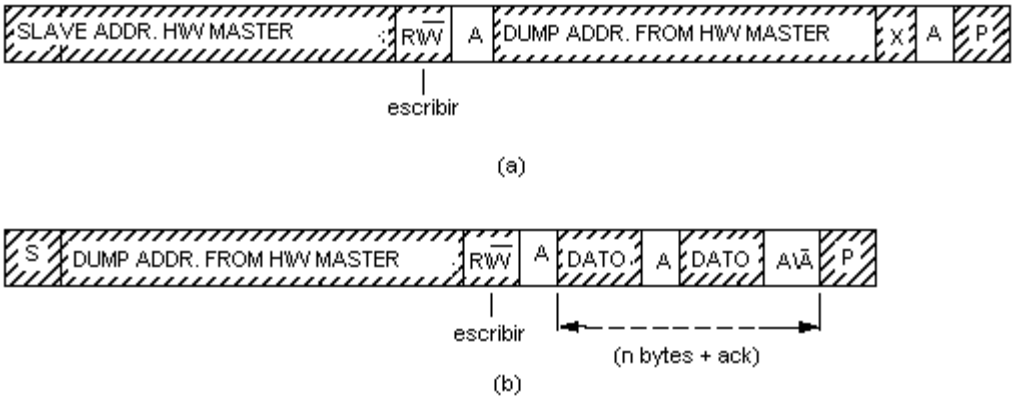


Figura 1.11. La transferencia de datos por un dispositivo hardware transmisor capaz de depositar los datos directamente a los dispositivos del esclavo.

En la figura 1.11. (a) la configuración del máster envía la dirección de depósito al hardware máster.

En la figura 1.11. (b) el hardware máster deposita los datos al repetidor seleccionado.

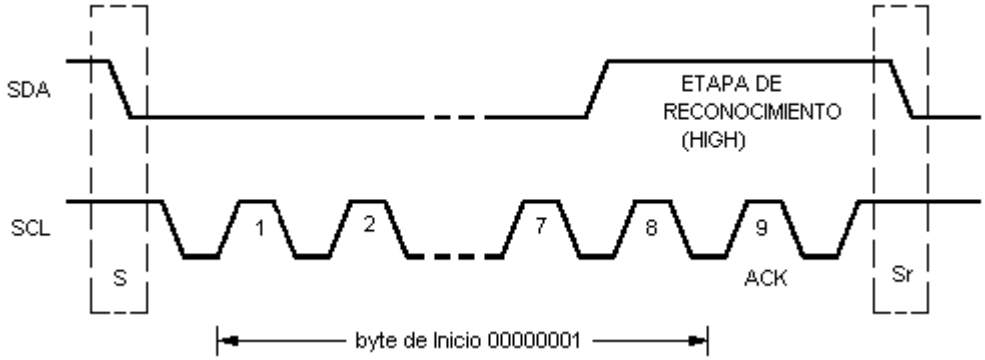


Figura 1.12. Procedimiento de arranque del byte

Después que la condición de arranque (S) ha sido transmitida por un máster el cual requiere acceso al bus, el byte de arranque (00000001) es transmitido. Otro microcontrolador puede utilizar la línea SDA para encontrar la condición de arranque repetido (Sr), la cual luego es usada para la sincronización.

Otro hardware receptor esperará receptar la condición de arranque repetida (Sr), y por lo tanto ignorará el byte de arranque. Se genera una pulsación de reloj de reconocimiento después del byte de arranque. Este está presente sólo para cumplir con el formato de

manejo del byte en el bus. No se permite ningún dispositivo para reconocer el byte de arranque.

1.4.3.1. ESPECIFICACIÓN DE LAS EXTENSIONES DEL BUS I2C EN EL MODO ESTÁNDAR.

La especificación del modo estándar del bus I2C con la transferencia de datos superior a los 100kbit/s y el direccionamiento de 7-bit, ha sido utilizada desde inicios de los años 80. Este concepto creció rápidamente en popularidad y hoy es aceptado en todo el mundo.

La especificación del modo estándar del bus I2C fue mejorada al pasar los años y ahora está disponible con las siguientes extensiones:

- Modo rápido, con una transmisión de bits superior a los 400kbit/s.
- Modo de alta velocidad (modo HS), con una transmisión de bits superior a los 3.4 Mbit/s.
- Direccionamiento a 10 bits, Este permite el uso de más de 1024 direcciones adicionales para el esclavo.

Hay dos razones principales para extender la especificación regular del bus I2C, entre las cuales se tiene:

- Muchas de las aplicaciones de hoy necesitan transferir grandes cantidades de datos en serie y requiere una transmisión de bits sobre los 100kbit/s (modo estándar) o aún 400 kbit/s (modo rápido), como resultado de los mejoramientos en las tecnologías del semiconductor, los dispositivos del bus I2C ahora están disponibles con niveles de transmisión de bits sobre los 3.4Mbit/s (modo HS) sin ningún incremento notable en el costo de la construcción del circuito de interfase.
- Con el direccionamiento de 7- bits, se teniá disponible 112 direcciones, que pronto fueron saturadas, requiriendo así

más combinaciones de direcciones, que permitieran el uso de más esclavo para nuevos dispositivos. Este problema fue resuelto con el nuevo esquema de direccionamiento a 10 bits, el cual permitía un incremento de 10 veces en las direcciones disponibles.

Ahora la interfase del bus I2C en el (modo HS y FS), pueden tener un direccionamiento de 7 a 10 bits para los esclavos. En el caso de requerir un hadware barato es recomendable utilizar un direccionamiento de 7 bits. Los dispositivos con direcciones de 7 a 10 bits pueden ser combinados en el mismo sistema del bus I2C sin tener en cuenta si es que es un sistema de modo F/S o Hs. Los dos existentes y los futuros másters pueden generar ya sea 7 o 10 direcciones de bits.

1.4.3.2. MODO RÁPIDO

Con la especificación de modo rápido del bus I2C, el formato del protocolo, los niveles lógicos y la máxima capacidad de carga para las líneas SDA y SLC citadas en el modo estándar del bus I2C son incambiables. Los nuevos dispositivos con una interfase del bus deben encontrar por lo menos un mínimo de requerimientos de la especificación del modo rápido o modo Hs. Los dispositivos de modo rápido pueden recibir y transmitir en velocidades mayores a los 400 kbit/s. El requerimiento mínimo es que pueden sincronizarse con una transferencia de 400 kbit/s y pueden prolongarse a un período bajo de la señal SCL para disminuir la transferencia. Los dispositivos de modo rápido son compatibles en forma descendente y pueden comunicarse con los dispositivos de modo estándar en un sistema de 0 a 100 Kbit/s del bus I2C, como los dispositivos de modo estándar, no son compatibles en forma ascendente, no deberían ser incorporados en un sistema de modo rápido del bus I2C.

La especificación de modo rápido del bus tiene las siguientes características adicionales comparadas con el modo estándar:

- El nivel máximo de transmisión de bits se incrementa a 400kbit/s.
- El tiempo de los datos en serie (SDA) y las señales del reloj en serie (SCL) han sido adaptadas.
- Los reguladores de salida de los dispositivos de modo rápido incorporan el control de inclinación de los bordes de caída de las señales SDA y SCL.
- Si se apaga la fuente de poder conectada a un dispositivo de modo rápido, los pines de las líneas SDA y SCL deben estar flotando de manera que no obstruyan las líneas del bus.
- Los dispositivos externos conectados a las líneas del bus deben ser adaptados para acoplarse en el tiempo máximo permitido por el modo rápido del bus I2C.

1.4.3.3. MODO Hs.

Los dispositivos del modo Hs pueden transferir la información a un nivel de velocidad mayor a 3.4 Mbit/s, ellos todavía permanecen compatibles con el modo rápido y el modo standard (F/S modo) de los dispositivos de comunicaciones bi-direccionales en un sistema de velocidad variada del bus. La excepción de que el arbitraje y la sincronización del reloj no se ejecuta durante la transmisión en el modo Hs. El mismo protocolo serial del bus y el formato de datos se mantiene con el sistema de modo F/S. Dependiendo de la aplicación, nuevos dispositivos pueden tener una interfase rápida o un modo Hs del bus, aunque los dispositivos de modo Hs son preferidos ya que pueden ser diseñados en un gran número de aplicaciones.

1.4.4. FORMATO DE DIRECCIONES CON 10 BITS

El direccionamiento del 10_bits es compatible y puede ser combinado con el direccionamiento del 7_bits. Si de la combinación reservada para el direccionamiento de 10 bits se utilizan solo los 7 primeros bits

(1111XXX) del primer byte seguido de una condición de arranque (S) o una condición de arranque repetido (Sr), el direccionamiento de 10_bits será utilizado como un direccionamiento de 7_bit. Los dispositivos con las direcciones de 7_bits y con el de 10_bits pueden ser conectadas al mismo bus I2C y ser usados en los sistemas del modo F/S y el modo Hs. Aunque hay 8 posibles combinaciones de los bits de dirección reservada (1111XXX), sólo cuatro combinaciones (11110XX) son usadas para el direccionamiento de 10_bits, las cuatro combinaciones restantes 11111XX son reservadas para futuros incrementos del bus I2C.

La dirección del esclavo de 10_bits, está formada de los dos primeros bytes seguidos de una condición de arranque (S) o una condición de arranque repetido (Sr), los primeros 7_bits del primer byte son la combinación (11110XX), de los cuales los dos últimos bits (XX) son los más significativos (MSBs) de la dirección del 10_bits, el octavo bit del primer byte es el R/W que determina la dirección del mensaje. Un "cero" en la última posición significativa del primer byte, significa que el máster escribirá la información en el esclavo seleccionado. Un "uno" en esta posición significa que el máster leerá la información desde el esclavo seleccionado.

Si el bit R/W es "cero" entonces el segundo byte contiene los restantes 8 bits (XXXXXXXX) de la dirección del 10_bits. Si el bit R/W es "uno" entonces el siguiente byte contiene datos transmitidos desde un esclavo a un máster.

Varias combinaciones de formatos de R/W son posibles dentro de una transferencia que incluye el direccionamiento de 10_bits. Los formatos de transferencia de datos son:

- El máster-transmisor, transmite a un esclavo-receptor con una dirección de 10_bits esto se ilustra en la figura 1.13.

Cuando una dirección del 10_bits seguida de una condición de arranque es transmitida al bus, cada esclavo compara los primeros 7 bits del primer byte de la dirección (11110XX) con su propia dirección y comprueba si el octavo bit de la dirección es 0. Es posible que más de

un esclavo se identifique con la dirección enviada y genere un reconocimiento (A1). Todos estos esclavos compararán los 8 bits del segundo byte de la dirección del esclavo (XXXXXXXX) con sus propias direcciones, pero sólo un esclavo se identificará con la dirección y generará un reconocimiento (A2). Los otros esclavos permanecerán direccionados por el máster hasta que reciba una condición de parada (P) o una condición de arranque repetido (Sr) seguida por una dirección de esclavo diferente.

- El máster-receptor lee al esclavo-transmisor con un direccionamiento de 10_bits.

La dirección de transferencia cambia después del segundo bit de R/W, (véase en la figura 1.14), realizando un reconocimiento del bit 2, el procedimiento es el mismo que el descrito para el máster-transmisor a un esclavo-receptor. El esclavo chequea si los primeros siete bits del primer byte de la dirección del esclavo (Sr) son los mismos que cuando estuvieron después de la condición de arranque (S) y comprueba si el octavo bit (R/W) es 1. Si el esclavo se identifica con la dirección transmitida, considera que ha sido direccionado como un transmisor y genera el reconocimiento A3. El esclavor-transmisor permanece direccionado hasta que recibe una condición de parada (P) o hasta que recibe otra condición de parada repetida (Sr) seguida por una dirección de esclavo diferente. Después de una condición repetida, todos los otros esclavos, también compararán los 7 primeros bits del primer byte de la dirección del esclavo (11110XX) con sus propias direcciones y comprobarán el octavo bit (R/W).

- Formato combinado. Un máster transmite datos a un esclavo y luego lee los datos desde el mismo esclavo (figura 1.15) este máster ocupa el bus todo el tiempo. La dirección de transferencia cambia después del segundo bit (R/W).
- Formato combinado. Un máster transmite los datos a un esclavo y luego los transmite a otros esclavos (figura 1.16). El mismo máster ocupa el bus todo el tiempo.
- Formato combinado. En la figura 1.17 se indica como transmite un máster los datos a un esclavo utilizando un direccionamiento de

7_bits y luego transmite los datos a un segundo esclavo con una direccionamiento de 10_bits. El mismo máster ocupa el bus todo el tiempo.

Figura 1.13. Un máster- transmisor direcciona un esclavo-receptor con la dirección de 10_bits.

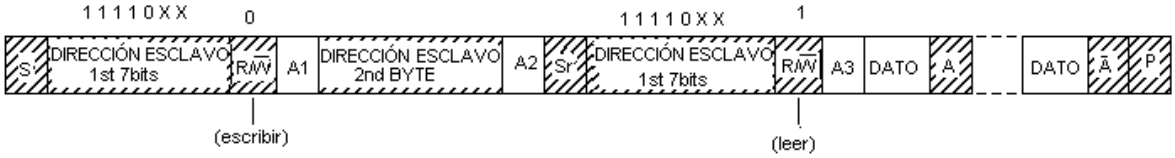


Figura 1.14. Un máster-receptor direcciona un esclavo- transmisor con una dirección del bit_ 10.

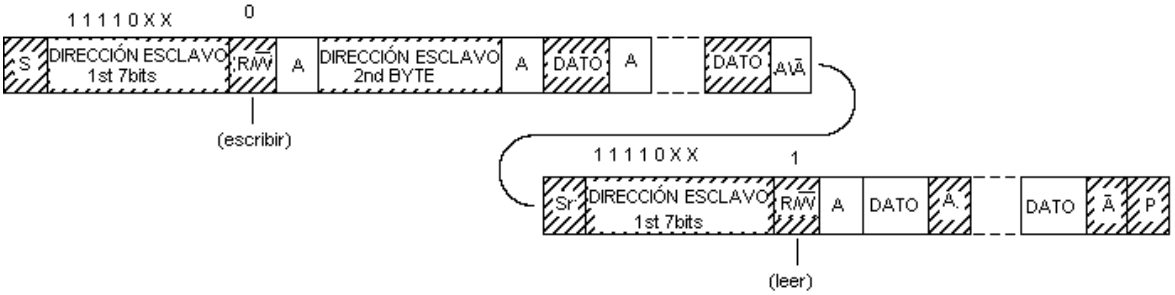


Figura 1.15. Formato combinado. Un máster direcciona a un esclavo con la dirección de 10_bits luego transmite los datos a este esclavo y lee los datos.

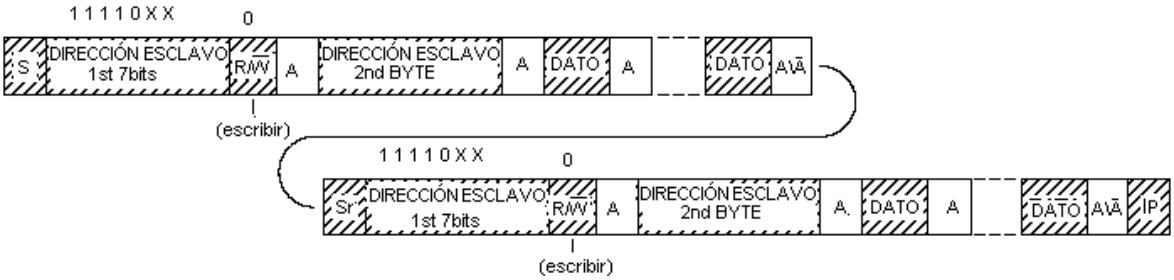


Figura 1.16. Formato combinando. Un máster transmite el datos a dos esclavos, ambos con las direcciones del 10-bit.

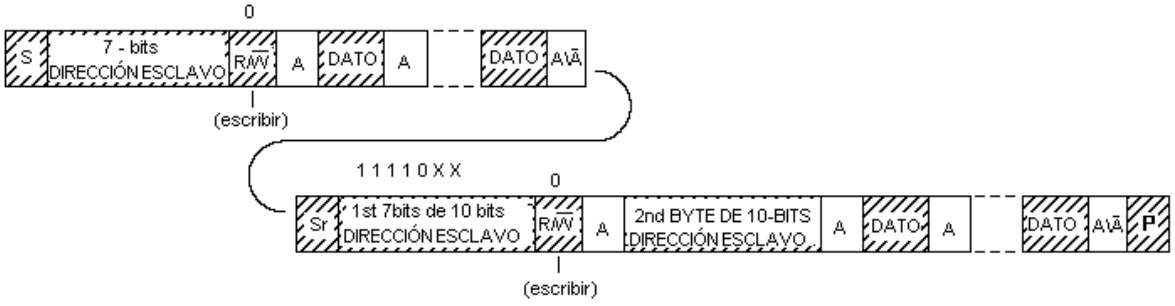


Figura 1.17. Formato combinando un máster transmite los datos a dos esclavos, uno con una dirección del 7-bit, y uno con una dirección del 10-bit.

1.4.5. DIRECCIÓN DE LLAMADA GENERAL Y BYTE DE INICIO CON DIRECCIONAMIENTO DE 10 BITS

El proceso de direccionamiento de 10_bits en el bus I2C, se da de tal manera que los dos primeros bytes después de la condición de arranque (S) generalmente determinan cual esclavo será seleccionado por el máster. Se debe tomar en cuenta que esto no sucede con la dirección de "llamada general" 00000000 (H'00'). Los dispositivos esclavos con el direccionamiento de 10_bits reaccionarán a una llamada general en la misma forma como los dispositivos con el direccionamiento de 7_bit .

El hardware del máster puede transmitir su dirección de 10_bit después de una "llamada general". En este caso, la dirección de llamada general del byte es seguida por dos bytes sucesivos que contienen la dirección del máster-transmisor.

1.5. MODOS DE OPERACIÓN

1.5.1. TRANSFERENCIA DE ALTA VELOCIDAD

Para llevar a cabo una transferencia a una velocidad mayor a los 3.4 Mbit/s. se han hecho las siguientes mejoras para la especificación regular del bus I2C:

- Durante la transferencia del modo Hs no se realiza el arbitraje o sincronización del reloj en los sistemas multimáster, ésto eleva la velocidad de transmisión de bits. El procedimiento del arbitraje siempre finaliza después de una transmisión de código del máster precedente en el modo F/S.
- Los dispositivos del máster del modo Hs generan una señal del reloj con una relación de 1 a 2 en un nivel alto y bajo. Esto

descongestiona los requerimientos de tiempo para la transmisión de bits.

- Como una opción, los dispositivos máster del modo Hs incorporan un puente. Durante la transferencia del modo Hs, los datos de alta velocidad (SDAH) y el reloj serial de las líneas de altas velocidades (SCLH) de los dispositivos del modo Hs son separados por este puente desde las líneas SDA y SCL de los dispositivos del modo F/S. Esto reduce la capacidad de carga de las líneas SDAH y SCLH produciendo una elevación más rápida y la caída de los tiempos, en estas señales.
- Los dispositivos de entrada de modo Hs incorporan un control de salto de los bordes de caída de las señales SDAH y SCAH.

La figura 1.18 indica la configuración física del bus I2C en un sistema sólo con dispositivos de modo Hs. Los pines SDA y SCL en los dispositivos del máster son usados únicamente en el bus de velocidad combinada y no son conectados en un sistema de modo Hs, en tales casos, se pueden usar para otras funciones. Los resistores serie opcionales Rs, protegen las lineas I/O de los dispositivos del bus además contra los picos de alto voltaje en las líneas del bus y minimiza el ruido y la interferencia. Los resistores de empuje a alto Rp mantienen las líneas SDAH y SCLH en un nivel alto y asegura que las señales sean empujadas desde un nivel bajo a un nivel alto dentro del tiempo requerido.

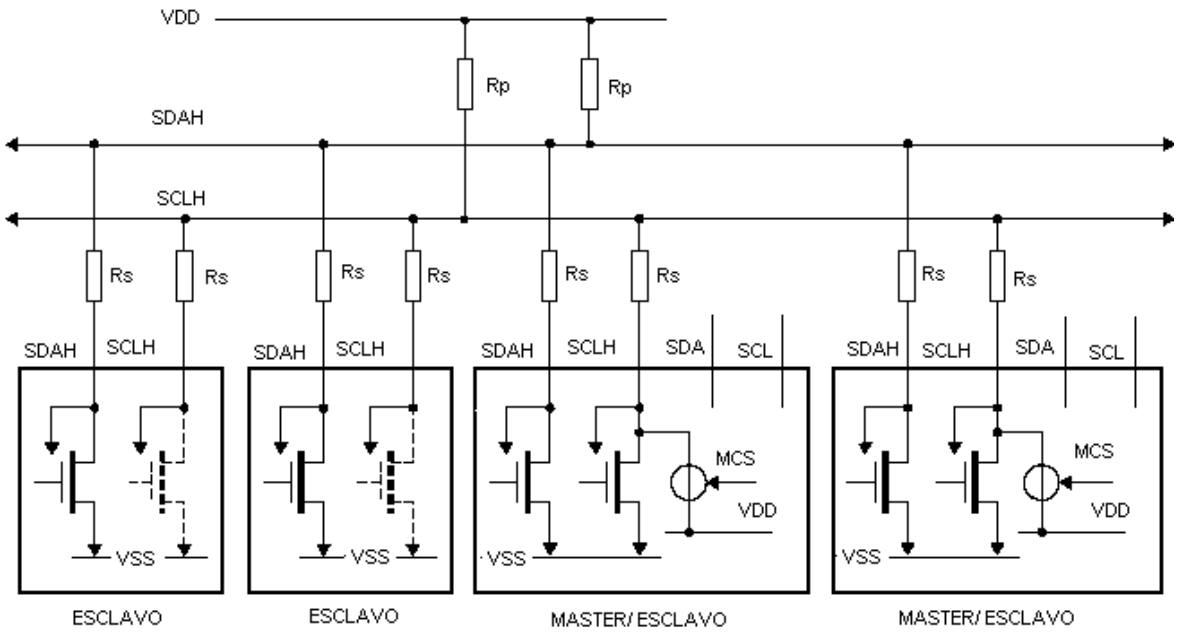


Figura 1.18. La configuración del bus I2C con los dispositivos del modo HS

1.5.2. FORMATO DE TRANSFERENCIA DE DATOS SERIALES EN UN MODE HS

El modo Hs sólo puede comenzar después de las condiciones siguientes:

1. Condición de START (S)
2. Código máster de 8 bits (00001XXX)
3. No reconocimiento del bit (Ā)

Estas condiciones se muestran en la figura 1.19 y tiene dos funciones principales:

- Permite el arbitraje y la sincronización entre los másters en competencia en la velocidades del modo F/S, resultando un máster ganador.
- Este indica el comienzo de la transferencia en el modo Hs.

Los códigos del máster en el modo Hs se guardan en un código de 8 bits, los cuales no son usados para un direccionamiento del esclavo u otros propósitos. Además, como cada máster tiene su propio código único, más de ocho másters en el modo Hs pueden estar presentes en el sistema del bus I2C. El código para un dispositivo máster de modo

Hs es el software programable, y es seleccionado por el diseñador del sistema.

El arbitraje y sincronización del reloj se lleva a cabo durante la transmisión del código máster y el no reconocimiento del bit (Ā), después de lo cual un máster ganador permanece activo. Como no se permite ningún dispositivo para reconocer el código máster, éste está seguido por un no reconocimiento del bit (Ā). Después el bit de no reconocimiento (Ā) la línea SCLH se coloca en un nivel alto, el máster activo conectado al modo Hs y facilita que la fuente de corriente empuje hacia un nivel alto el circuito por la señal SCLH. Entonces el máster activo envía una condición de arranque repetida (Sr) seguida por una dirección del esclavo de 7_bits o de 10_bits, y recibe un bit de reconocimiento (A) desde el esclavo seleccionado.

Después de una condición de arranque repetido y de cada bit de reconocimiento (A) o no reconocido (Ā) el máster activo anula nuevamente su fuente de corriente del circuito que empuja hacia un nivel alto.

La transferencia de datos continua en el modo Hs luego del arranque repetido (Sr) y solamente se desconecta al modo F/S después de una condición de parada (P).

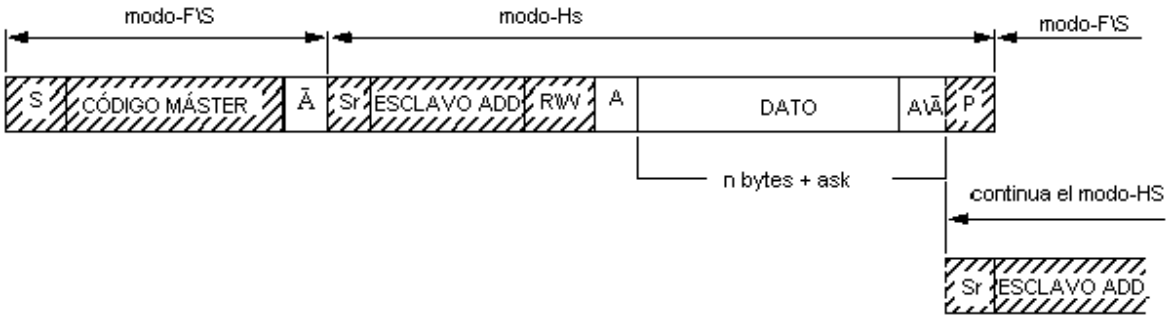


Figura 1.19. Formato de transferencia de datos en el modo HS.

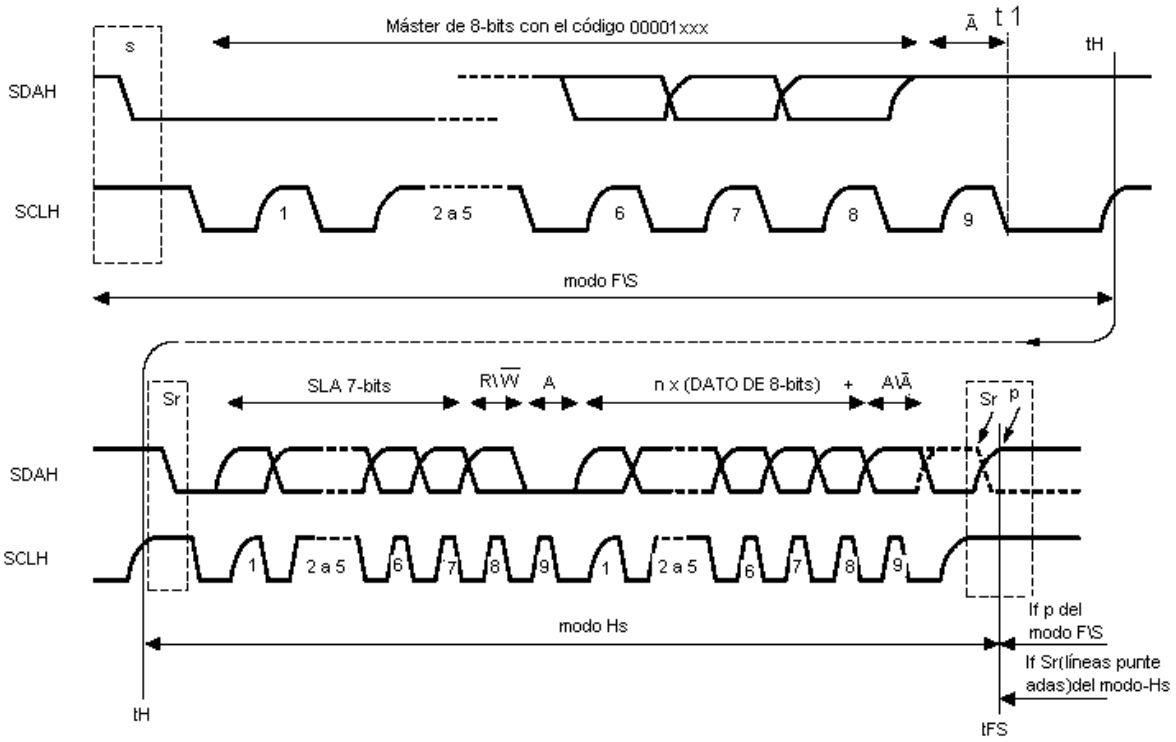


Figura 1.20. Un traslado completo del modo HS

1.5.3. CAMBIO DEL MODO F/S AL MODO HS Y VICEVERSA

Cada dispositivo del modo Hs puede conmutar desde modo rápido hasta el modo Hs y viceversa, esto es controlado por la transferencia serial en el bus I2C. Antes del tiempo t1 en la figura 1.20 cada dispositivo conectado opera en el modo rápido. Entre los tiempos t1 y tH (este intervalo de tiempo puede ser alargado por cualquier dispositivo) cada dispositivo conectado debe reconocer la secuencia “S 00001XXX A” y tiene que cambiar su circuito interno desde la ubicación del modo rápido al modo Hs. Entre los tiempos t1 y tH el máster conectado y los dispositivos del esclavo realizan esta conmutación con las siguientes acciones:

El máster activo:

1. Adapta sus filtros de entrada SDHA y SCLH de acuerdo al requerimiento de supresión de picos en el modo Hs.
2. Adapta la estructuración y tiempos del sostenimiento según los requisitos del modo Hs.

3. Adapta el control de inclinación de las señales de salida SDHA y SCLH de acuerdo al requerimiento del modo Hs.
4. Cambia el número de bits del modo Hs, el cual es requerido después del tiempo tH.

Los másters no-activos, o perdedores:

1. Adaptan sus filtros de entrada SDHA y SCLH de acuerdo al requerimiento de la supresión de picos en el modo Hs.
2. Espera por una condición de parada, para detectar cuando el bus está libre de nuevo.

Todos los esclavos:

1. Adaptan sus filtros de entrada SDHA y SCLH de acuerdo al requerimiento de la eliminación de picos en el modo Hs.
2. Adaptan la estructuración y tiempos del sostenimiento según los requisitos del modo Hs. Este requisito puede cumplirse por la adaptación de los filtros en la entrada.
3. Adapta el control de inclinación de la señal de salida SDHA, si es necesario, para los dispositivos del repetidor. El control de inclinación es sólo aplicable para las señales de salida y dependiendo de las tolerancias del circuito, los requerimientos de los dos modos, rápido y Hs, pueden ser completados sin conmutar su circuito interno.

En el tiempo tFS de la figura 1.20, cada dispositivo de la figura conectada, debe reconocer la condición de parada (P) y cambiar su circuito interno desde el modo Hs hacia atrás hasta la ubicación del modo rápido como se indica antes del tiempo t1.

1.5.4. DISPOSITIVOS DE MODE HS A BAJA VELOCIDAD

Los dispositivos del modo Hs son totalmente compatibles y pueden ser conectados a un sistema del modo F/S del bus I2C. Mientras el código del máster no sea transmitido en tal configuración, todos los dispositivos del máster Hs permanecerán en el modo F/S y se comunicarán en velocidades del modo F/S con su fuente de corriente inutilizada. Los pines SDHA y SCLH son usados para conectar el

sistema del bus al modo F/S, permitiendo que los pines en el dispositivo de modo Hs sean usados para otras funciones.

1.5.5. TRANSFERENCIA DE UN MODO HS EN UN SISTEMA DE BUS DE VELOCIDAD VARIADA

En la figura 1.21 se indica el diagrama de tiempo de una transferencia de datos completa del modo Hs, la cual es llamada por una condición de arranque, un código máster y un no reconocimiento (Ā) (en la velocidad del modo F/S).

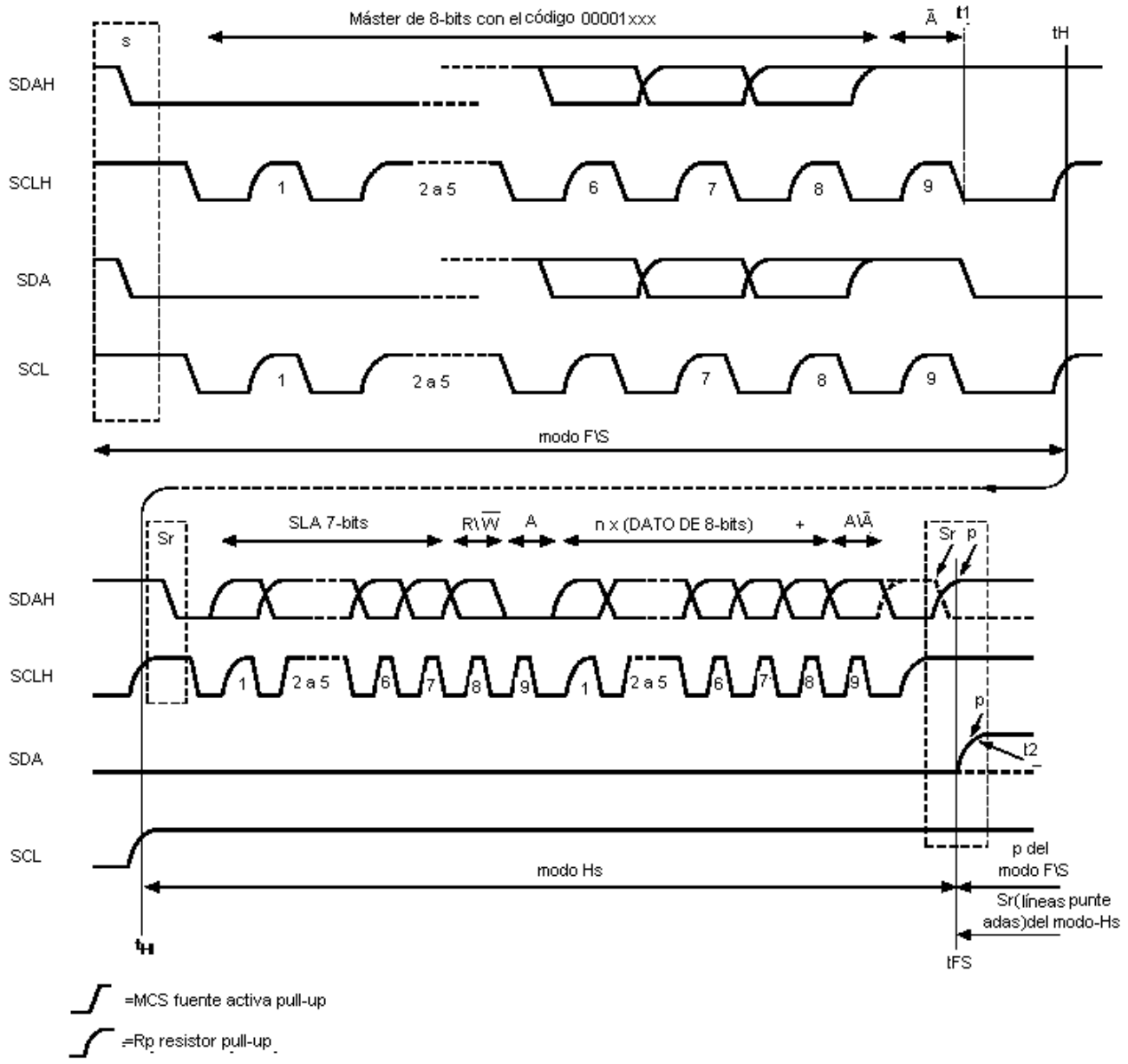


Figura 1.21. Una transferencia completa del bus de velocidad variada.

La transferencia de modo Hs empieza después de th con una condición de arranque repetido (Sr). Durante la transferencia de modo Hs, la línea SCL permanece en un nivel alto y la línea SDA en un nivel de estado fijo y así se realiza la transferencia de una condición de parada (P). Después de cada reconocimiento (A) o no reconocimiento del bit (Ā) el máster activo deshabilita su circuito de corriente que empuja hacia un nivel alto. Esto facilita a los otros dispositivos retrazar la transferencia serial al alargar el período de BAJO nivel de la señal SCLH. El máster activo restablece su circuito de corriente cuando todos los dispositivos están libres y la señal SCLH alcanza un nivel alto y así las velocidades son superiores.

1.6. ESPECIFICACIONES ELÉCTRICAS

1.6.1. DISPOSITIVOS STÁNDARD Y FAST-MODE

Los niveles y corriente I/O, la supresión de picos, el control de inclinación de salida y la capacitancia del pin para el modo F/S de los dispositivos del bus I2C están dados en la tabla 1.3. Las características de toma de tiempo del bus. Los períodos mínimos del nivel alto y bajo del reloj SCL especificados en la tabla 1.4 determinan el máximo porcentaje de la transferencia de bits de 100Kbit/s para los dispositivos de modo estándar, y de 400Kbit/s para los dispositivos de modo rápido. Los dispositivos de modo rápido y modo estándar del bus I2C deben ser capaces de seguir las transferencias a su propio nivel de transmisión de bits, ya sea al ser capaz de transmitir o recibir a esas velocidad o para aplicar el procedimiento de sincronización del reloj, el cual forzará al máster a un estado de espera y alargamiento del período de BAJO nivel de la señal SCL.

Tabla 1.3. Características de las fases SDA y SCL de I/O para los dispositivos del modo F/S del bus I2C.

PARÁMETRO	**SÍMBOLO**	**MODO ESTÁNDAR**		**MODO RÁPIDO**		**UNI-DAD**
		MÍNIMO	MÁXIMO	MÍNIMO	MÁXIMO	
voltaje de entrada de nivel bajo:	VIL	- 0.5	0.3VDD	-0.5	0.3VDD	V
voltaje de entrada de nivel alto:	VIH	0.7VDD	_	0.7VDD	n/a	V
Nivel de voltaje bajo de salida con un descenso de corriente en 3 mA:	VOL	0	0.4	0	0.4	V
Corriente de entrada fija I/O en el pin con un voltaje de la entrada entre 0.1VDD y 0.9VDDmax	Ii	10	10	10	10	μA
Capacitancia para cada pin de I/O	Ci	-	10	-	10	pF

Tabla 1.4. Las características de las líneas del bus SDA y SCL para los dispositivos de F/S-modo.

PARÁMETRO	**SÍMBOLO**	**MODO ESTÁNDAR**	**MODO RÁPIDO**	**UNID.**

		MÍN	MAX	MÍN	MAX	
Frecuencia del reloj SCL:	fSCL	0	100	0	400	kHz
Condición de arranque de tiempo repetido.	tHD;STA	4.0	–	0.6		μs
El período bajo del reloj SCL	tLOW	4.7	–	1.3	–	μs
El período alto del reloj SCL	tHIGH	4.0	–	0.6	–	μs
Tiempo de la condición de arranque repetitivo	tSU;STA	4.7	–	0.6	–	μs
Establecimiento del tiempo en alto de los datos set-up	tSU;DAT	250		100	–	ns
Tiempo de levantamiento de las dos señales SDA y SCL	tr	–	1000	20 + 0.1Cb	300	ns
Tiempo de caída de las dos señales SDA y SCL	tf	–	300	20 + 0.1Cb	300	ns
Set-up tiempo para la condición de PARADA	tSU;STO	4.0	–	0.6	–	μs
Tiempo libre del bus entre una condición de STOP y START	tBUF	4.7	–	1.3	–	μs
Carga capa_ citiva para cada línea de bus	Cb	–	400	–	400	pF
Margen de ruido en el nivel bajo para cada dispositivo conectado	VnL	0.1VDD	–	0.1VDD	–	V

Margen de ruido en el nivel alto para cada dispositivo conectado	VnH	0.2VDD	–	0.2VDD	–	V

1.6.2. DISPOSITIVO HS-MODE

Los niveles y corriente I/O, la supresión de picos, el control de inclinación de la señal de salida y la capacitancia del pin para los dispositivos de modo Hs del bus I2C están dados en la tabla 1.5. El margen de ruido para los niveles altos y bajos en las líneas del bus son las mismas especificadas por los dispositivos del modo F/S del bus.

La condición de arranque “normal” S no existe en el modo Hs. Los parámetros de tiempo para la dirección de bits, y el bit R/W, el reconocimiento del bit y los datos de los bits, son todos los mismos, solamente la elevación del borde de la primera señal del reloj SCLH después de un reconocimiento de bit tiene un valor más grande porque el RP externo tiene que empujar el SCLH sin la ayuda de la fuente de corriente interna.

Los períodos mínimos de alto, bajo, la elevación máxima, y la caída de los tiempos de la señal del reloj SCLH determina el porcentaje más alto de bits. Con una señal SCLH internamente generada con períodos de niveles alto y bajo de 200ns y 100ns respectivamente; un máster de modo Hs puede completar los requerimientos de tiempo por las pulsaciones externas del reloj SCLH (teniendo en cuenta la elevación y caída de los tiempos) por el máximo porcentaje de bits de 3.4 Mbit/s, de esta manera una frecuencia básica de 10MHz, puede ser usada por

un máster de modo Hs para generar la señal SCLH. No hay límites para los períodos de alto y bajo del reloj SCLH y no hay límite para un porcentaje más bajo de bits.

Los parámetros de tiempo son independientes para la carga capacitiva superior a 100 pF para cada línea del bus permitiendo el porcentaje máximo posible de bits de 3.4 Mbit/s. En una carga capacitiva superior en las líneas del bus, el porcentaje del bit decrece gradualmente. Los parámetros de tiempo para una carga capacitiva del bus de 400 pF, permitiendo un porcentaje máximo de bits de 1.7 Mbit/s. Para las cargas capacitivas del bus entre 100 pF y 400 pF, los parámetros de tiempo deben ser interpolados linealmente. La elevación y caída de los tiempos está de acuerdo con la propagación máxima del tiempo de las líneas de transmisión SDAH y SCLH para prevenir los reflejos de los terminales abiertos.

Tabla 1.5. Características de las fases SDAH, ACLH, SDA y SCL, I/O para los dispositivos del bus I2C.

PARÁMETRO	**SÍMBOLO**	**MODO HS**		**UNIDAD**
		MÍNIMO	MÁXIMO	
Nivel bajo del voltaje de entrada	VIL	0.5	0.3VDD	V
Nivel alto de voltaje de entrada	VIH	0.7VDD	VDD + 0.5	V
Entradas de histéresis del disparador	Vhys	0.1VDD	–	V
Nivel bajo del voltaje de salida (drenaje abierto) en SDHA, SDA y SCLH.	VOL	0	0.2VDD	V
En la resistencia del puerto de transferencia, para las dos direcciones	RonL		50	Ω

de corriente en el nivel VOL entre SDA y SDHA o SCL y SCLH en 3 mA.				
En la resistencia del puerto de transferencia entre SDA y SDAH o SCL y SCLH si los dos están en el nivel VDD	RonH	50	–	kΩ
La elevación del tiempo de salida y el tiempo de caída en SCLH con una carga capacitiva de 10 a 100 pF.	trCL, tfCL	10	40	ns
La caída del tiempo de salida en SDAH con una carga capacitiva de 400pF	tfDA	20	160	ns
La amplitud de pulsación de picos en SDAH y SCLH que deben ser suprimidos por los filtros de entrada.	tSP	0	10	ns
La corriente de entrada de cada pin I/O con un voltaje de entrada entre 0.1VDD y 0.9 VDD.	Ii	–	10	μA

1.6.3. VALORES MÁXIMO Y MÍNIMOS DE LAS RESISTENCIAS RP Y RS PARA DISPOSITIVOS DEL BUS I2C STÁNDARD-MODE

El voltaje de la fuente limita el valor mínimo del resistor Rp debido a la mínima corriente especificada de 3 mA, para un valor de V máximo = 0.4 V en la etapa de salida. VDD como una función de Rp mínima se muestra en la figura 1.22. El margen de ruido requerido de 0.1 VDD

para el nivel bajo limita el valor máximo de Rs. El valor máximo de Rs como una función de Rp se muestra en la figura 1.23.

La capacitancia total del alambre conductor, de las conexiones y los pines, limita el valor máximo de Rp debido al incremento de tiempo especificado.

La figura 1.24 muestra a Rp máximo como una función de capacitancia del bus.

El máximo nivel alto de la corriente de entrada de cada conexión de entrada/salida tiene un valor máximo especificado de 10uA. Debido al margen de ruido requerido de 0.2 VDD por el nivel alto, esta corriente de entrada limita el valor máximo de Rp. Este límite depende de VDD. La corriente total de entrada del nivel alto es indicada como una función de Rp máximo en la figura 1.25

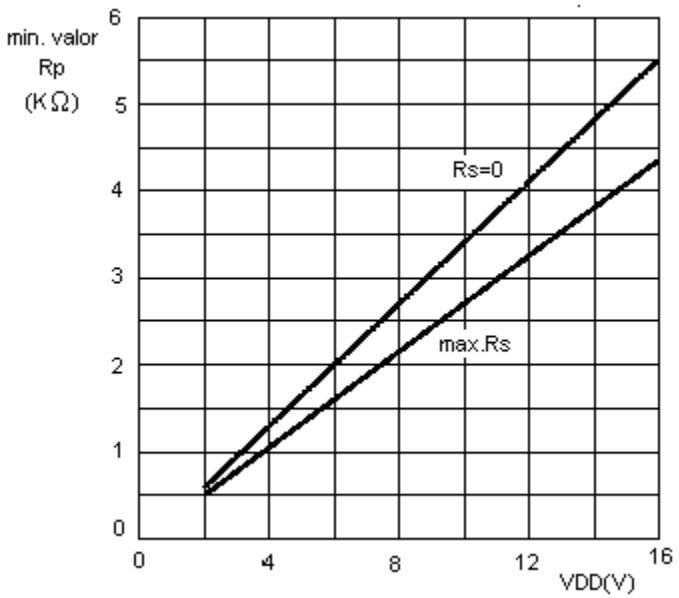


Figura 1.22. Valor mínimo de Rp como función de fuente de voltaje con el valor Rs como parámetro.

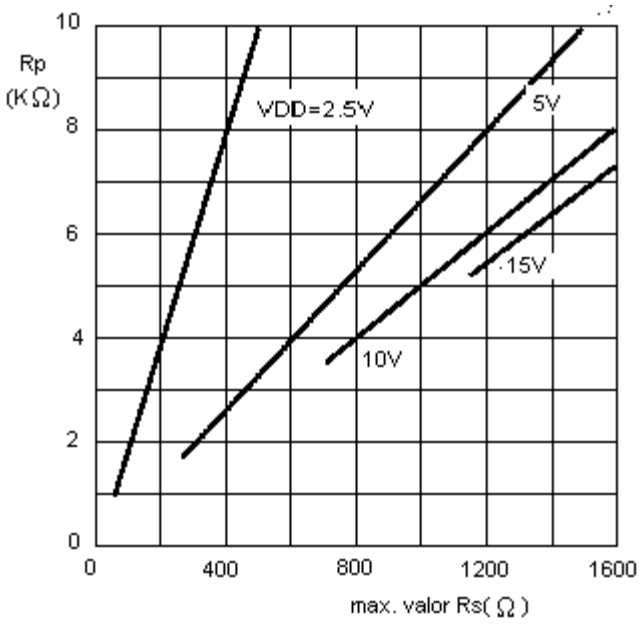


Figura 1.23. Valor máximo de Rs como función del valor de Rp con la fuente de voltaje como parámetro.

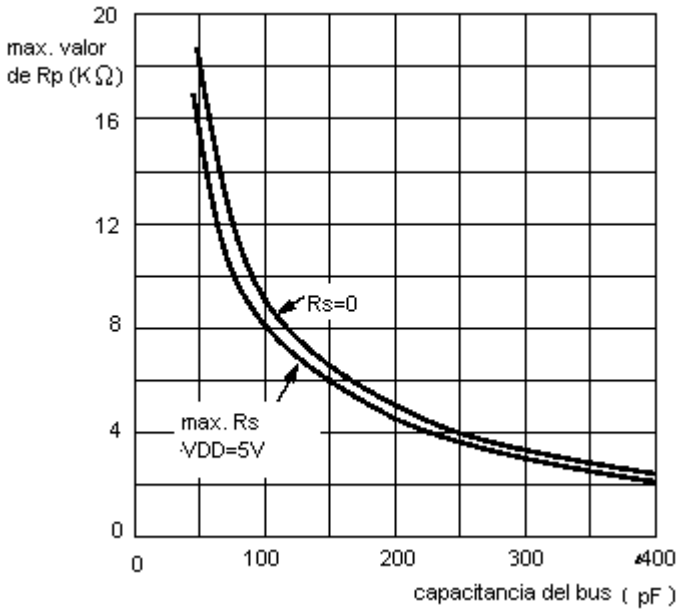


Figura 1.24. Valor máximo de Rp como función de la capacitancia del bus para el modo estándar del bus I2C.

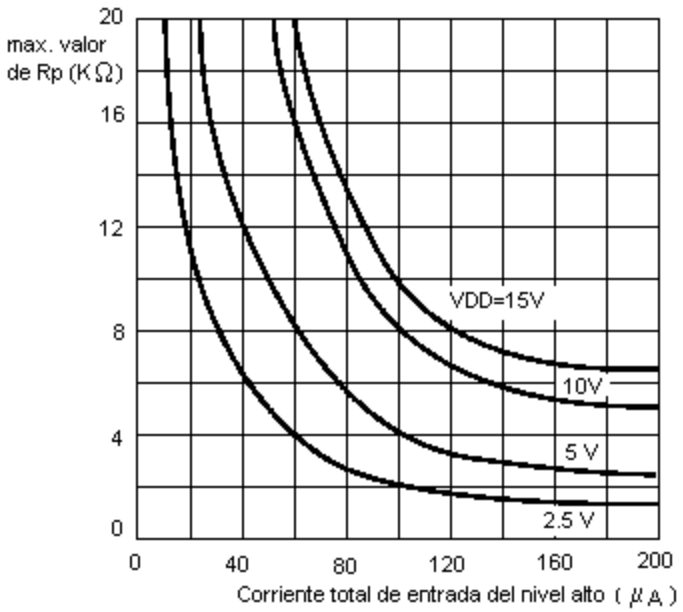


Figura 1.25. Corriente total de entrada del nivel alto como función del valor máximo de Rp con la fuente de voltaje como un parámetro.

1.7. APLICACIÓN

1.7.1. FASES DE SALIDA DEL DISPOSITIVO BUS-I2C DEL FAST-MODE

Las figuras 1.26 y 1.27 demuestran los ejemplos de fases de salida con el control de inclinación en la tecnología CMOS y BIPOLAR. La inclinación del borde de caída está definida por un condensador de tipo “Mylar” (c1) y un resistor (R1). Los valores típicos para C1 y R1 están indicados en los diagramas anteriores.

La tolerancia amplia para el tiempo de salida dada en la tabla 1.3, significa que el diseño no es crítico. El tiempo de caída sólo está ligeramente influenciado por la carga externa del bus (Cb) y el resistor externo (Rp). Sin embargo, la elevación del tiempo tr especificado en la tabla 1.4 está determinada por la capacitancia de carga del bus y el valor del resistor.

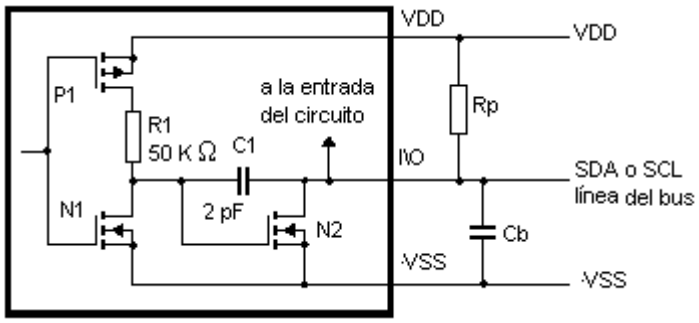


Figura 1.26. Fase de salida de inclinación controlada en la tecnología CMOS.

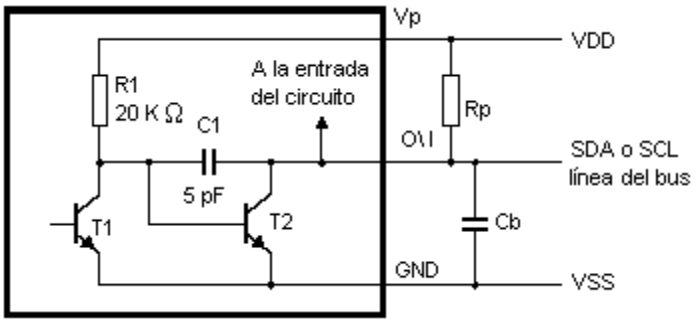


Figura 1.27. Fase de salida de inclinación controlada en la tecnología bipolar

1.7.2. CAMBIO DE CIRCUITOS PULL-UP PARA DISPOSITIVO DEL BUS-I2C DE FAST-MODE

El voltaje de la fuente VDD y el máximo nivel bajo de salida determina el valor mínimo del resistor Rp. Por ejemplo con un voltaje de la fuente de VDD = 5V+- 10% y VOL máximo = 0.4 V con 3 mA, Rp mínimo es de (5.5 – 0.4)/0.003 = 1.7 kΩ, este valor de Rp limita la máxima capacitancia del bus a más o menos 200 pF para encontrar el requerimiento máximo tr de 300 ns. Si el bus tiene una capacitancia superior a ésta, se utiliza el circuito conmutado de la figura 1.28 que es para una fuente de voltaje de VDD = 5 V +- 10% y una carga capacitiva máxima de 400 pF,

Los resistores combinados Rp1 y Rp2 pueden empujar hacia un nivel alto la línea del bus dentro del tiempo de elevación máximo (tr) especificado de 300 ns.

Los resistores Rs son opcionales, protegen a las fases I/O de los picos altos de voltaje y minimizan la diafonía de las señales en las líneas del bus, el valor máximo de Rs está determinado por el voltaje máximo de caída permitido a través de este resistor cuando la línea del bus está conmutando a un nivel bajo con el objeto de apagar Rp2.

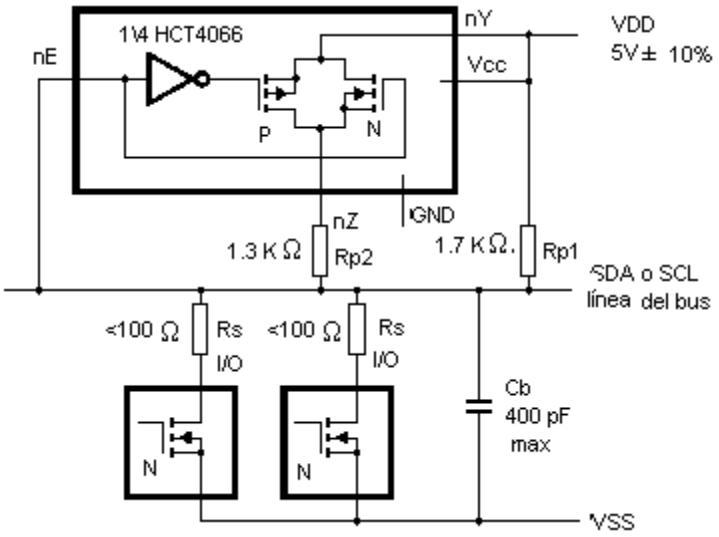


Figura 1.28. Circuito conmutado que empuja hacia un nivel alto

1.7.3. CONFIGURACIÓN DE LAS LÍNEAS DE BUS

El cableado debe ser escogido de tal modo que la diafonía y la interferencia de las líneas del bus sean minimizadas. Las líneas del bus son las más susceptibles a la diafonía y a la interferencia en los niveles altos a consecuencia de la impedancia alta de los dispositivos que empujan hacia un nivel alto al sistema.

Si la longitud de las líneas del bus excede 10 cm. e incluye las líneas VDD y VSS, el modelo de cableado se muestra en la figura 1.29.

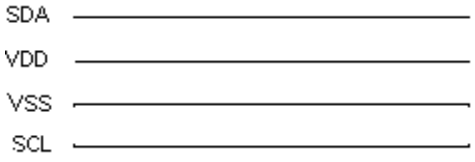


Figura 1.29. Primera configuración de la línea bus

Si sólo la línea VSS está incluida el modelo de cableado se muestra en la figura 1.30.

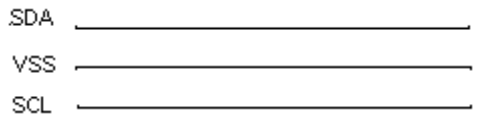


Figura 1.30. Segunda configuración de la línea bus

Estos modelos de cableado producen cargas capacitivas idénticas para las líneas SDA y SCL.

Si las líneas del bus son trenzadas en pares cada línea del bus debe estar trenzada con un retorno a VSS, alternativamente, la línea SCL puede estar trenzada con un retorno VSS y la línea SDA trenzada con un retorno a VDD. En el último caso, los condensadores deben ser usados para desacoplar la línea VDD y VSS, colocado en los dos extremos del par trenzado.

Si las líneas del bus son blindadas (blindaje conectado a VSS), la interferencia se minimiza. Sin embargo, el cable blindado debe tener un acoplamiento de baja capacitancia entre las líneas SDA y SCL para minimizar la diafonía.

1.7.4. VALORES MÁXIMO Y MÍNIMOS DE LAS RESISTENCIAS RP Y RS PARA DISPOSITIVOS DEL BUS I2C DEL FAST-MODE

Los valores máximos y mínimos de los resistores Rp Y Rs para los dispositivos conectados al bus en el modo rápido pueden ser determinados con la ayuda de las figuras 1.22, 1.23 y 1.25, el valor máximo de Rp como una función de la capacitancia del bus es menos de la que se indica en la figura 1.24. El gráfico de reemplazo para la figura 1.24 demostrando el valor máximo de Rp como una función de capacitancia del bus (Cb) para un modo rápido está dado en la figura 1.31.

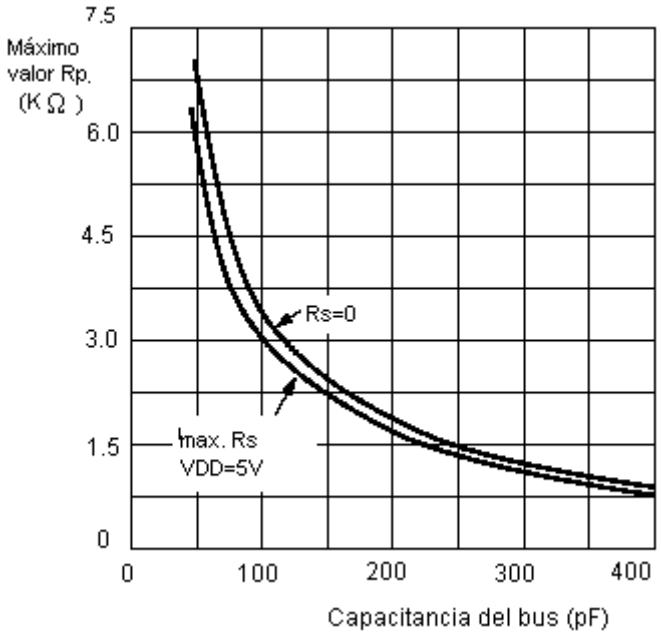


Figura 1.31. Máximo valor de Rp como función de capacitancia par reunir el requisito de máximo tr

1.7.5. VALORES MÁXIMOS Y MÍNIMOS DE LAS RESISTENCIAS RP Y RS PARA DISPOSITIVOS DEL BUS I2C DEL HS-MODE

Los valores máximos y mínimos de los resistores Rp Y Rs conectados al bus I2C en el modo Hs, pueden ser calculados a partir de los datos en tabla 1.5. Muchas combinaciones de estos valores son posibles debido a las diferentes elevaciones y caídas de tiempo, cargas en las líneas del bus, fuente de voltaje, los sistemas de velocidad combinados.

1.7.6. CONEXIÓN DE DISPOSITIVOS CON DIFERENTES NIVELES LÓGICOS

En el bus I2C pueden conectarse dispositivos con diferentes niveles de voltaje, que unen resistores que empujan hacia un nivel alto las líneas de suministro de energía. Aunque ésta es la solución más simple, los dispositivos con niveles de voltaje más bajo que puede se conectar al

sistema deben ser de 5V, lo cuales puede generar mayor costo en la fabricación de estos dispositivos. Al usar un inversor de nivel bi-direccional, es posible interconectar dos secciones del bus, con cada sección conectada a una fuente de voltaje diferente y con niveles lógicos de tensión diferentes. Tal configuración se muestra en la figura 1.32. La sección izquierda “bajo voltaje” tiene resistores que empujan hacia un nivel bajo y dispositivos conectados a una fuente de voltaje de 3.3V, la sección de la derecha “alto voltaje” tiene resistores y dispositivos conectados a una fuente de voltaje de 5V. Los dispositivos de cada sección tienen I/Os con una fuente de voltaje relacionada a los niveles lógicos de entrada y a una configuración de drenaje abierto de salida.

El inversor de nivel de cada línea del bus es idéntico y consiste de un MOSFET, que debe de estar conectado de la siguiente forma; TR1 para la línea serial de datos SDA y TR2 para la línea serial del reloj SCL. Las compuertas “g” (gates) tienen que estar conectadas a la fuente de voltaje más bajo (VDD1), las fuentes (s) a las líneas del bus de la sección “bajo voltaje” y los drenajes (d) a las líneas del bus de la sección “alto voltaje”. Muchos MOS-FETs tienen el substrato internamente conectado con su fuente, si este no es el caso, se debería hacer una conexión externa. Cada MOS-FET tiene un diodo integrado (unión n-p) entre el drenaje y el substrato.

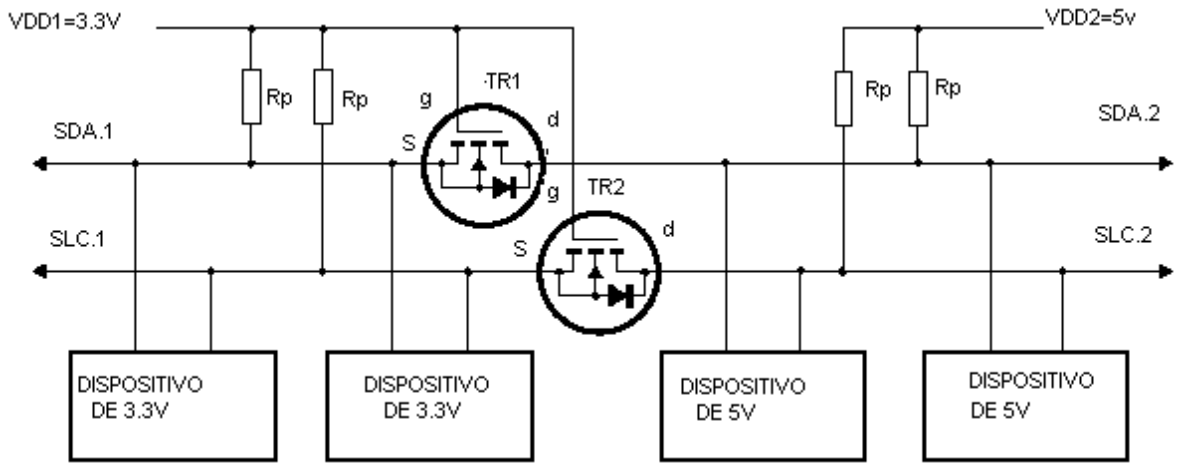


Figura 1.32. Circuito inversor de nivel bidireccional que conecta dos secciones de diferente voltaje en un sistema del bus I2C.

1.8. PROTOCOLO DE COMUNICACIÓN DE LA RED

Como es lógico, para iniciar una comunicación entre dispositivos conectados al bus I^2C se debe respetar un protocolo.

Tan pronto como el bus esté libre, un dispositivo maestro puede ocuparlo generando una condición de inicio. El primer byte transmitido después de la condición de inicio contiene los siete bits que componen la dirección del dispositivo de destino seleccionado y un octavo bit correspondiente a la operación deseada (lectura o escritura). Si el dispositivo cuya dirección se apuntó en los siete bits está presente en el bus éste responde enviando el pulso de reconocimiento ó ACK. Seguidamente puede comenzar el intercambio de información entre los dispositivos.

Cuando la señal R/W está en un nivel lógico bajo, el dispositivo maestro envía datos al dispositivo esclavo hasta que deja de recibir los pulsos de reconocimiento, o hasta que se hayan transmitido todos los datos, en caso contrario, es decir cuando la señal R/W está en un nivel lógico alto, el dispositivo maestro genera pulsos de reloj durante los cuales el dispositivo esclavo puede enviar datos. Luego de cada byte recibido el dispositivo maestro (que en este momento está recibiendo datos) genera un pulso de reconocimiento.

El dispositivo maestro puede dejar libre el bus generando una condición de parada (Stop). Si se desea seguir transmitiendo, el dispositivo maestro puede generar otra condición de inicio en lugar de una condición de parada. Esta nueva condición de inicio se denomina "inicio repetitivo" y se puede emplear para direccionar un dispositivo esclavo diferente ó para alterar el estado del bit de lectura/escritura (R/W).

CAPÍTULO II

2. EL PIC16F877A

2.1. DESCRIPCIÓN DEL PIC 16F877A

El PIC16F877 se pueden considerar como una combinación de las virtudes del PIC 16F84 con la inclusión de los recursos de los PIC 16F73 y 74. Incorporan la memoria FLASH, con una capacidad de 4 K y 8 K palabras de 14 bits, sin cambiar la estructura interna del procesador y conservando el mismo repertorio de instrucciones.

La memoria RAM posee una capacidad de 368 byte, mantienen la misma estructura basada en 4 bancos de 128 bytes cada uno, seleccionables con los bits RP0 y RP1 del Registro de Estado.

La memoria de datos no volátil es de 256 bytes, se manejan hasta 14 posibles fuentes de interrupción y 3 timers.

Además incorporan los siguientes recursos: Dos módulos CCP, comunicación serie, comunicación paralelo, conversor A/D, que serán descritos en esta sección.

2.1.1. PRIMERA SUBFAMILIA PIC 16F87X

Se trata de un conjunto de cuatro modelos de PIC con memoria FLASH que viene a reforzar la posición y la aplicación del anterior PIC 16F84. Los modelos son: PIC 16F873, PIC 16F874, PIC 16F876, PIC 16F877, que se encuentran dentro de la gama media de los PIC 16Fxxx de microcontroladores de 8 bits.

En la tabla 2.1 se ofrece los principales características de los cuatro modelos y en ella se ha incluido al PIC 16F84A para su análisis comparativo.

Tabla 2.1. Características relevantes de los cuatro modelos de PIC 16F87x, junto a las del PIC 16F84A

MODELO		**PIC 16F84A**	**PIC 16F873**	**PIC 16F874**	**PIC 16F876**	**PIC 16F877**
MEM PROG (FLASH)	**Bytes**	1792	7168	7168	14336	14336
	Palabras	1024*14	4096*14	4096*14	8192*14	8192*14
MEM DATOS	**Bytes EEPROM**	64	128	128	256	256
	Bytes RAM	68	192	192	368	368
CONVERSOR A/D		no	5(10 bits)	8(10bits)	5(10bits)	8(10bits)
BOR (detección de perdida d energ.)		no	si	si	si	si
LÍNEAS E/S		13	22	33	22	33
COMUNICACIÓN		no	USART/	USART/	USART/	USART/

SERIE		MSSP	MSSP	MSSP	MSSP
MÓDULOS DE CAPTURA CCP	no	2	2	2	2
TEMPORIZADORES	1-8bits 1-WDT	1-16bit, 2-8bit 1-WDT	1-16bit, 2-8bit 1-WDT	1-16bit, 2-8bit 1-WDT	1-16bit, 2-8bit 1-WDT
FREC. MAX. MHz	20	20	20	20	20
ICSP (programación serie en Circuito)	si	si	si	si	si
ENCAPSULADOS	18P,18SO, 20SS	28SP,28SO	40P,44L, 44PQ,44PT	28SP,28SO	40P,44L, 44PQ,44PT
FUENTES DE INTERRUPCIÓN	4	13	14	13	14
COMUNICACIÓN PARALELO	no	no	si	no	si

2.1.2. SEGUNDA FAMILIA PIC 18Cxx

Esta nueva familia de microcontroladores de 8 bits reúne mejores características que la familia 16F87x; mantienen su compatibilidad con las familias PIC 17Cxxx, 16Cxxx, 16C5x y 12Cxxx, su diseño le permite ser programado de forma muy eficiente con el lenguaje C, alcanzan rendimientos de 10MIPS (millones de instrucciones/seg), disponen de un repertorio de 77 instrucciones de 16 Bits, un rango de direccionamiento de 2 Mbytes de memoria de programa, funcionan a una frecuencia de 40 Mhz y pueden ser programados en serie sobre el circuito de aplicación.

2.1.3. CLASIFICACIÓN DE LOS MICROCONTROLADORES

El conjunto de microcontroladores PIC de 8 bits se distribuye en cuatro gamas:

- **GAMA BÁSICA:** familias PIC 12C5xx y PIC 16C5x, con un repertorio de 33 instrucciones de 12 bits y dos niveles de Pila.
- **GAMA MEDIA:** familias PIC 12C6xx y PIC 16Cxx, con un repertorio de 35 instrucciones de 14 bits, ocho niveles de Pila y un vector de interrupción.
- **GAMA ALTA:** familia PIC 17Cxxx, con un repertorio de 58 instrucciones de 16 bits, con 16 niveles de Pila y 4 vectores de interrupción.
- **GAMA MEJORADA:** familia PIC 18Cxxx con 77 instrucciones de 16 bits, 32 niveles de Pila y 4 vectores de interrupción.

2.2. ARQUITECTURA DE LOS MICROCONTROLADORES

2.2.1. ARQUITECTURA PIC 16F87X

Bajo la denominación PIC 16F87x se hace referencia a una subfamilia de micro controladores PIC de la gama media, que se identifica por tener como memoria de programa una de tipo FLASH y una serie de recursos semejantes a los modelos más potentes.

Dos de los cuatro modelos que componen esta subfamilia están encapsulados con 28 pines (PIC 16F873/6), mientras que los otros dos tienen 40 pines (PIC 16F874/7). Las principales diferencias entre los PIC 16F87x con 28 pines y los de 40 se dan por el número de líneas de E/S disponibles, a continuación se citan las tres diferencias más relevantes:

- Los modelos encapsulados con 40 pines disponen de 5 puertas (PA, PB, PC, PD y PE) de E/S con un total de 33 líneas para conectar a los periféricos exteriores. Los de 28 patitas solo tienen 3 puertas (PA, PB y PC) con 22 líneas de E/S.
- El conversor A/D en los PIC con 28 patitas tienen 5 canales de entrada, pero los de 40 patitas tienen 8.

- Solo los encapsulados con 40 patitas integran una puerta paralela esclava.

2.2.2. PRINCIPALES RECURSOS DEL MICROCONTROLADOR PIC 16F877

Las características fundamentales son las siguientes

- Procesador de arquitectura RISC avanzada.
- Juego de 35 instrucciones de 14 bits de longitud. Todas ellas se ejecutan en un ciclo de instrucciones, menos las de salto que tardan dos ciclos.
- Frecuencia máxima de 20 MHz.
- Hasta 8K palabras de 14 bits para la memoria de código, tipo FLASH.
- Hasta 368 bytes de memoria de datos RAM.
- Hasta 256 bytes de memoria de datos EEPROM.
- Hasta 14 fuentes de interrupción interna y externa.
- Pila con 8 niveles.
- Modos de direccionamiento directo, indirecto y relativo.
- Perro guardián (WDT).
- Código de protección programable.
- Modo SLEEP de bajo consumo.
- Programación serie en circuito con dos patitas.
- Voltaje de alimentación comprendido entre 2 y 5.5V.
- Bajo consumo (menos de 2mA a 5V y 5 MHz)

Adicionalmente se especifica los dispositivos periféricos con que cuenta.

- Timer0: temporizador contador de 8 bits con predivisor de 8 bits.
- Timer1: temporizador contador de ‘16 bits con predivisor.
- Timer2: temporizador contador de 8 bits con predivisor y postdivisor.

- Dos módulos de Captura (CCP) – Modulación por ancho de pulso (PWM).
- Conversor A/D de 10 bits.
- Puerto Serie Síncrono.
- Puerta Paralela Esclava.

En la figura 2.1 se presenta el diagrama de asignación y conexión del encapsulado

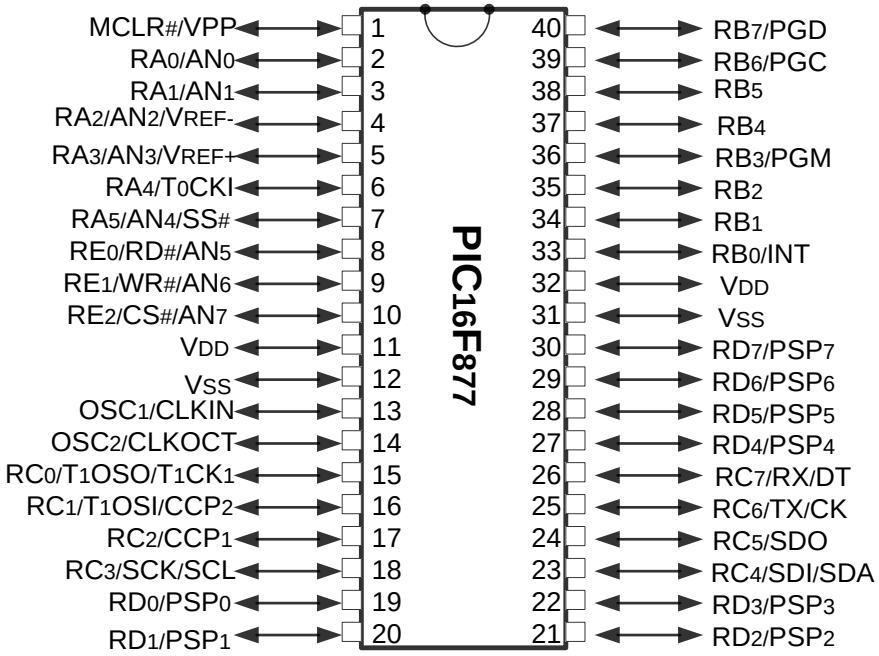


Figura 2.3. Diagrama de asignación y conexiones del encapsulado

2.2.3. DESCRIPCIÓN DE PINES DEL PIC 16F877

Pines de propósito general:

- OSC 1/CLKIN (13): entrada del cristal de cuarzo o del oscilador externo.
- OSC2/CLKOUT (14): salida del cristal de cuarzo. En modo RC, el pin OSC2 saca la cuarta parte de la frecuencia que se introduce por OSC1, que determina el ciclo de instrucción.
- VSS (12-31): conexión a Tierra.

- VDD (11-32): entrada de la alimentación positiva.
- MCLR#VPP/THV(1): entrada de RESET o entrada del voltaje de programación o voltaje alto en el modo test.

Puerta A

- RA0/AN0(2): puede actuar como línea digital de E/S o como entrada analógica al conversor AD (canal 0)
- RA1/AN1(3): igual que la RA0/AN0.
- RA2/AN2/VREF- (4): puede ser línea digital de E/S, entrada analógica o entrada del voltaje negativo de referencia.
- RA3/AN3/VREF+ (5): línea digital de E/S, entrada analógica o entrada del voltaje de referencia positivo.
- RA4/T0CKI (6): línea digital de E/S o entrada del reloj del Timer0. Salida con colector abierto.
- RA5/SS#/AN4(7): línea digital de E/S, entrada analógica o selección como esclavo de la puerta serie síncrona.

Puerta B

- RB0/INT(33): línea digital de E/S o entrada de petición de interrupción externa
- RB1(34): línea de E/S digital.
- RB2(35): línea de E/S digital.
- RB3/PGM(36): línea digital de E/S o entrada del voltaje bajo para programación.
- RB4(37): línea de E/S digital.
- RB5(38): línea de E/S digital.
- RB6/PGC(39): línea digital de E/S. En la programación serie recibe las señales de reloj.
- RB7/PGD(40): línea digital de E/S. En la programación serie actúa como entrada de datos.

Puerta C

- RC0/T1OSO/T1CKI(15): línea digital de E/S o salida del oscilador del Timer1 o como entrada de reloj del Timer1.

- RC1/T1OSI/CCP2(16): línea digital de E/S o entrada al oscilador del Timer1 o entrada al módulo Captura2/salida Comparación2/salida de PWM2.
- RC2/CCP1(17): E/S digital. También puede actuar como entrada Captura 1/Salida comparación1/Salida de PWM1.
- RC3/SCK/SCL(18): E/S digital o entrada de reloj serie síncrona/salida de los modos SPI e I2C.
- RC4/SDI/SDA(23): E/S digital o entrada de datos en modo SPI o I/O datos en modo I2C.
- RC5/SDO(24): E/S digital o salida de datos en modo SPI.
- RC6/TX/CK (25): E/S digital o pin del transmisor del USART asíncrono o como reloj del síncrono.
- RC7/RX/DT(26): E/S digital receptor del USART asíncrono o como datos en el síncrono.

Puerta D

- RD0/PSP0-RD7/PSP7(19): los 8 pines de esta puerta pueden actuar como líneas de E/S digitales o como líneas para la transferencia de información en la comunicación de la puerta paralela esclava.
- Puerta E sólo tiene 3 pines.
- RE0/RD#/AN5(8): E/S digital o señal de lectura para la puerta paralela esclava o entrada analógica

- RE1/WR#/AN6(9): E/S digital o señal de escritura en la puerta paralela esclava o entrada analógica al conversor A/D.
- RE2/CS#/AN7(10): E/S digital o activación/desactivación de la puerta paralela esclava o entrada analógica.

En la figura 2.4 se presenta la arquitectura correspondiente al modelo PIC 16F877 de 40 pines.

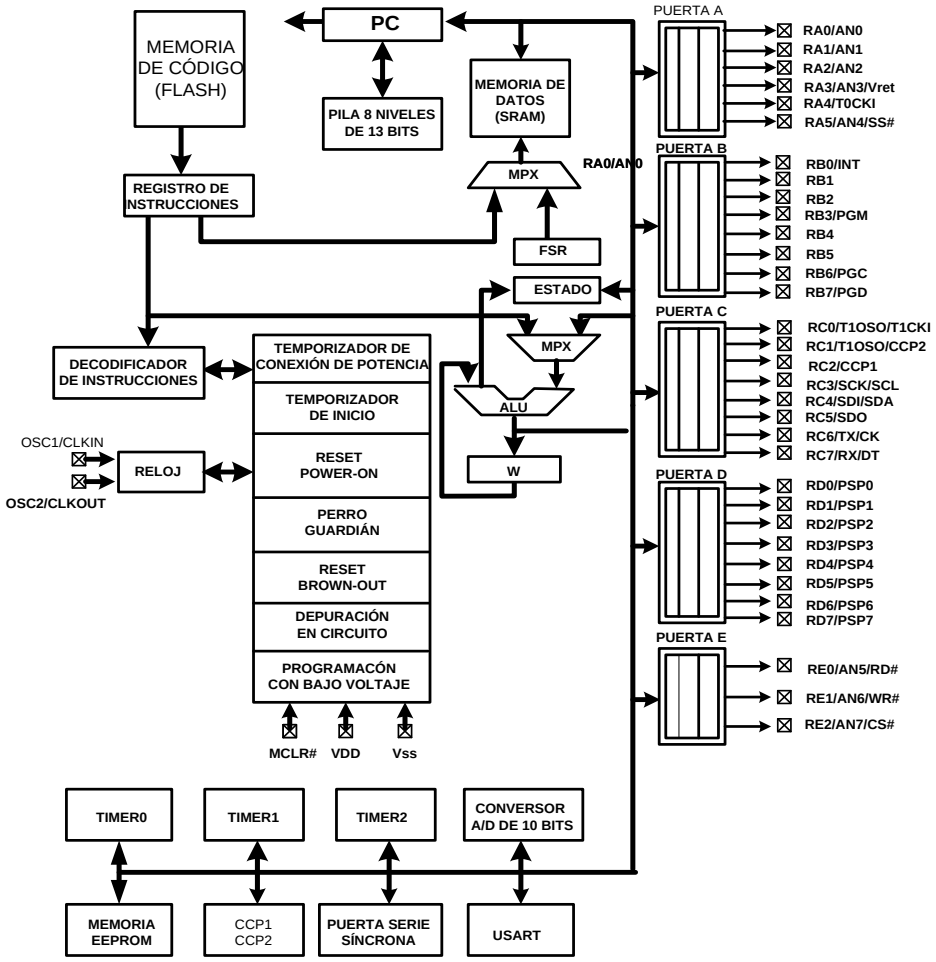


Figura 2.4. Arquitectura del PIC 16F877

2.3. DISTRIBUCIÓN DE MEMORIA

2.3.1. MEMORIA DE PROGRAMA.

La memoria FLASH, en la que se graba el programa de aplicación en el PIC16F877, puede tener una capacidad de 4K u 8K palabras de 14 bits cada una. Dicha memoria está dividida en páginas de 2 K palabras y está direccionada con el PC, que tiene un tamaño de 13 bits. La pila, que tiene 8 niveles de profundidad, es transparente para el usuario, es decir, funciona automáticamente y no dispone de instrucciones para guardar o sacar información. Con la instrucción CALL y con las interrupciones, el valor del PC se graba en el nivel superior. Con

las instrucciones RETURN, RETFIE y RETLW el valor contenido en el nivel superior de la pila se carga en el PC. Al poseer la pila sólo 8 niveles le corresponde al programador preocuparse por los anidamientos en las subrutinas para no sobrepasar dicho valor. El vector de Reset ocupa la dirección 0000h y el vector de Interrupción la 0004h, igual que en el P1C16F84.

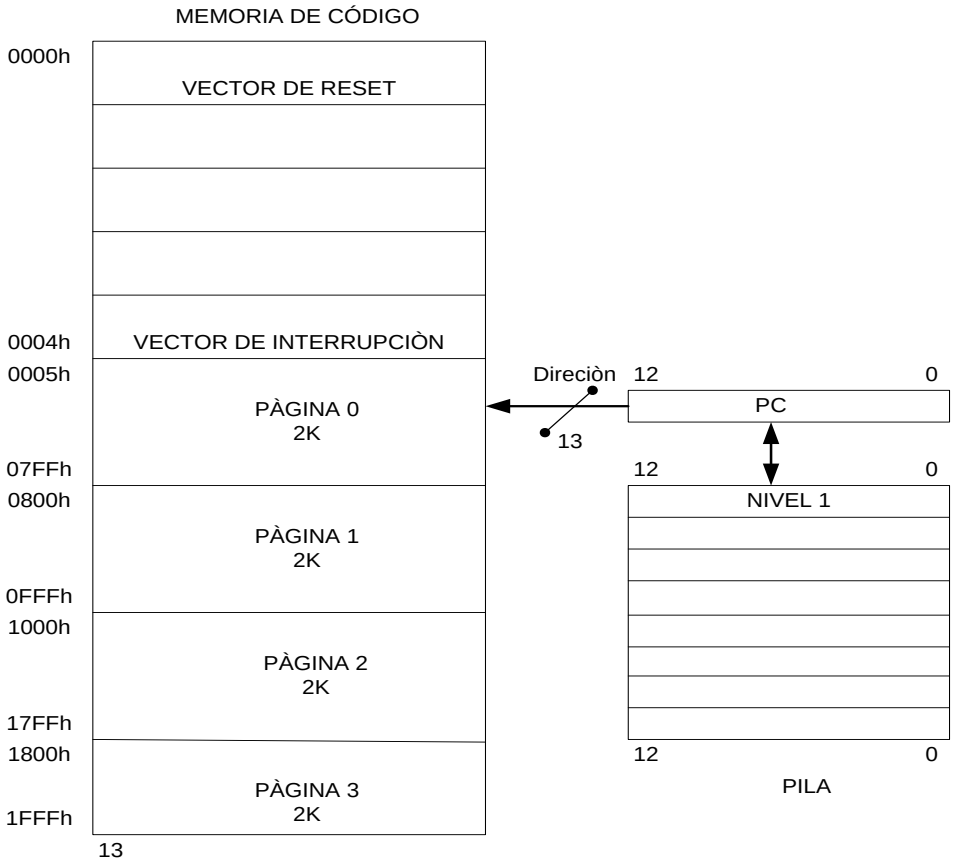


Figura 2.3. Memoria de programa tipo flash en el PIC 16F877

2.3.1.1. REGISTROS ESPECÍFICOS PARA EL CONTROL DE LA MEMORIA DE PROGRAMA

Los 13 bits contenidos en el PC, que direccionan la memoria de código, están guardados en dos registros específicos. El registro PCL guarda los 8 bits de menos peso y se puede leer y escribir. Los bits (12 al 8) del PC se alojan en el registro PCH, que al no poder ser leído ni escrito, se accede a él a través del PCLATH.

Las instrucciones de salto CALL y GOTO sólo proporcionan 11 bits de la dirección a saltar. Esto limita el salto dentro de cada banco de 2K. Cuando se desea salir del banco actual hay que programar correctamente los bits PCLATH (4y3) que seleccionan al banco. Es labor del programador modificar el valor de dichos bits en las instrucciones CALL Y GOTO.

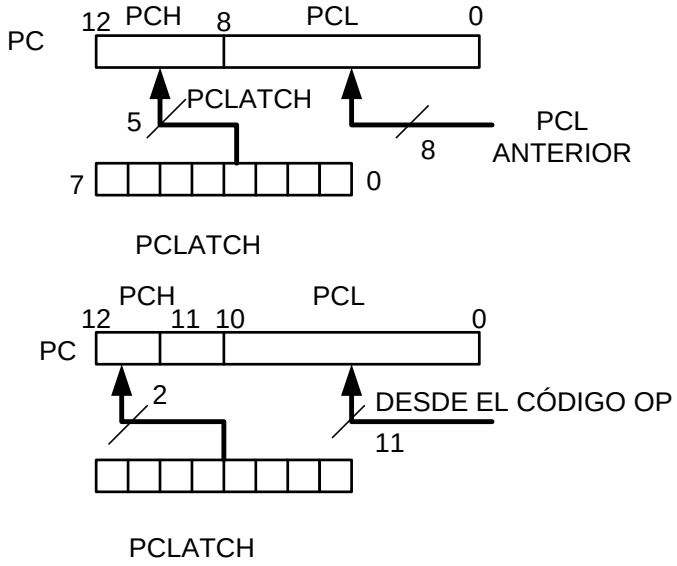


Figura 2.4. En la figura superior se muestra como se carga el PC. Abajo se muestra la carga del PC con las instrucciones CALL y GOTO

2.3.2. MEMORIA DE DATOS

La memoria de datos tiene posiciones implementadas en RAM y otras en EEPROM. En la sección RAM, se alojan los registros operativos fundamentales del funcionamiento del procesador y del manejo de todos sus periféricos, además de registros que el programador puede usar para información de trabajo propia de la aplicación. La memoria EEPROM sirve para guardar datos de forma no volátil, a la que se considera como un dispositivo especial.

La RAM estática consta de 4 bancos con 128 bytes cada uno. En las posiciones iniciales de cada banco se ubican los Registros Específicos que gobiernan al procesador y sus recursos.
Para seleccionar el banco al que se desea acceder en la RAM se emplean los bits 6 y 5 del Registro de Estado, denominados RP1 y RP0 respectivamente, según el código siguiente:

Tabla 2.2. Selección de los bancos de memoria RAM con RP0 y RP1

BANCO	RP1	RP0
0	0	0
1	0	1
2	1	0
3	1	1

2.3.2.1. CONTROL DE LA MEMORIA DE DATOS

Para direccionar la memoria RAM de datos estructurada en 4 bancos de 128 bytes cada uno, existen dos modos diferentes**:**

- Direccionamiento directo.
- Direccionamiento indirecto.

En el modo de direccionamiento directo, los bits RP1 y RP0 del Registro de Estado (6y5) se encargan de seleccionar el banco, mientras que la dirección dentro del banco la determinan 7 bits procedentes del código OP de la instrucción. Para el direccionamiento indirecto se usa el registro FSR, en el que sus 7 bits de menos peso señalan la dirección, y el banco lo determina el bit de más peso de FSR concatenado con el bit IRP del Registro de Estado (7), mostrado en la figura 2.5.

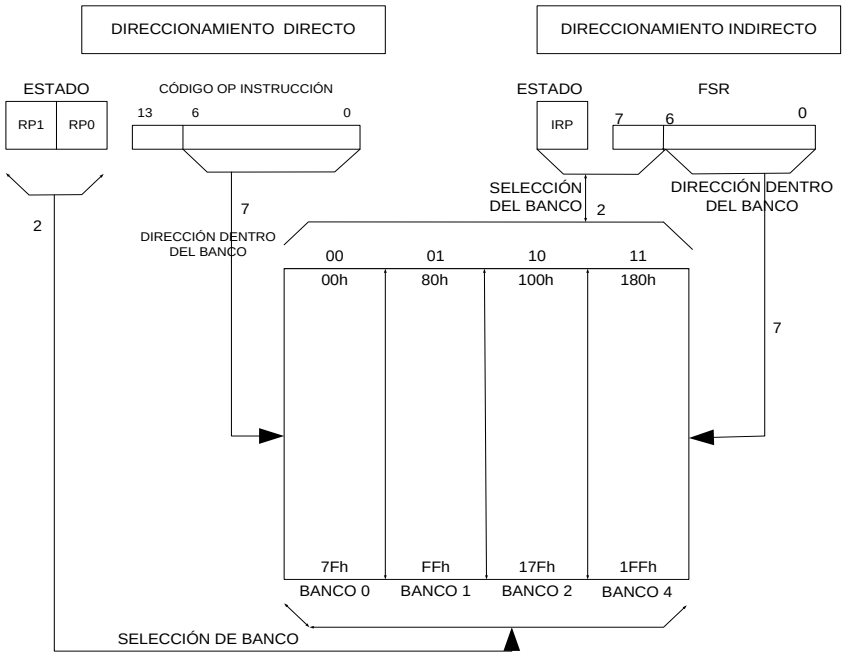


Figura 2.5. Formas de seleccionar el banco y la dirección de la memoria RAM en los direccionamientos directo y indirecto

2.4. PUERTOS DE ENTRADA Y SALIDA

Los microcontroladores PIC 16F877 encapsulados disponen de cinco Puertas de E/S (A, B, C, D y E). Todas las líneas de estas puertas son multifuncionales, es decir, realizan diversas funciones según estén programadas. Sin embargo, todas ellas tienen la capacidad de trabajar como líneas de E/S digitales.

2.4.1. PUERTA A.

Dispone de 6 líneas, denominadas RA0-RA5. Son bidireccionales y su sentido queda configurado según la programación de los bits del registro TRISA. Si el bit 0 del registro TRISA se pone a 1, la línea 0 (RA0) de la Puerta A funciona como entrada. Si se pone a 0 funciona como salida y el contenido de la báscula de salida se aplica a la patita correspondiente, según puede apreciarse en el esquema de la Figura 2.7

Al leer el registro PORTA de la Puerta A se lee el estado de las patitas, que es el que se halla escrito en la báscula de datos de la

Figura 2.6. La escritura entraña una operación de (lectura-modificación-escritura), o sea, se leen las patitas, luego se modifica su valor y finalmente se escribe en la báscula de datos.

Los pines RA0/AN0, RA1/AN1 y RA2/AN2, además de líneas de E/S digitales también pueden actuar como los canales 0, 1 y 2 por los que se puede aplicar una señal analógica al conversor A/D. La patitas RA3/AN3/Vref+ también puede actuar como entrada de la Tensión de Referencia para los periféricos que la precisan. La patita RA4/T0CK1 actúa como E/S digital y como entrada de la señal de reloj para el Timer 0. Por último, la patita RA5/AN4/SS# tiene multiplexadas tres funciones: E/S digital, canal 4 para el conversor A/D y selección del modo esclavo cuando se trabaja con la comunicación serie síncrona.

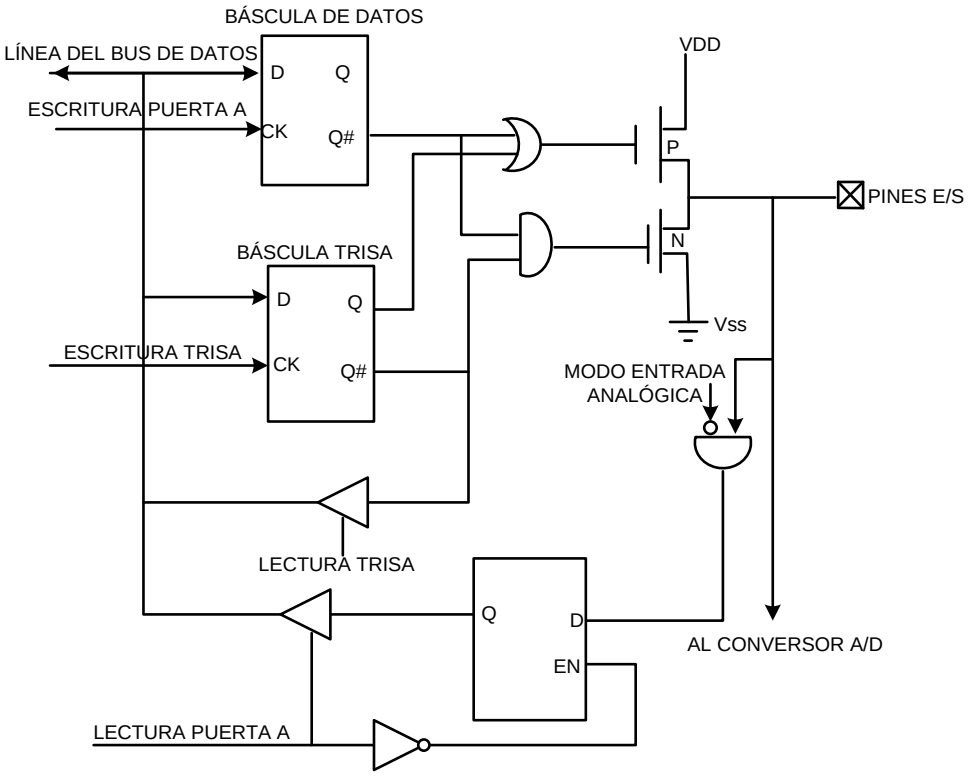


Figura 2.6. Esquema de conexionado de las patitas RA0 – RA3 Y RA5 que multiplexan la función de E/S

2.4.2. PUERTA B

Dispone de 8 líneas bidireccionales cuya función se elige mediante la programación del registro TRIS B, igual que sucedía en la Puerta A con TRIS A. En el siguiente programa, se configuran como entradas las líneas RB(3-O) y RB(7:6), mientras que RB(5:4) quedan asignadas como salidas.

bcf	ESTATUS,RP0	:Selección del banco 0
bcf	ESTATUS,RP1	:Inicializa la puerta B, borrando la
clrf	PORTB	báscula de datos
bsf	STATUS,RP0	: Selección del banco 1.
movlw	0xCF	:Valor para configurar las líneas
movwf	TRISB	RB(3-0) y RB(7:6) de la puerta B como entradas y como salida RB(5:4)

Todas las patitas de la Puerta B disponen de una resistencia interna de pull-up conectada al positivo de la alimentación, que queda habilitada cuando el bit RBPU#, que es el bit 7 del registro OPTION, tiene valor 0. La resistencia de pull-up es un transistor CMOS tipo P, como se aprecia en la figura 2.7, se conecta automáticamente siempre que la línea esté configurada como salida. Cuando se produce un Reset por conexión de la alimentación se desconectan todas las resistencias pull-up.

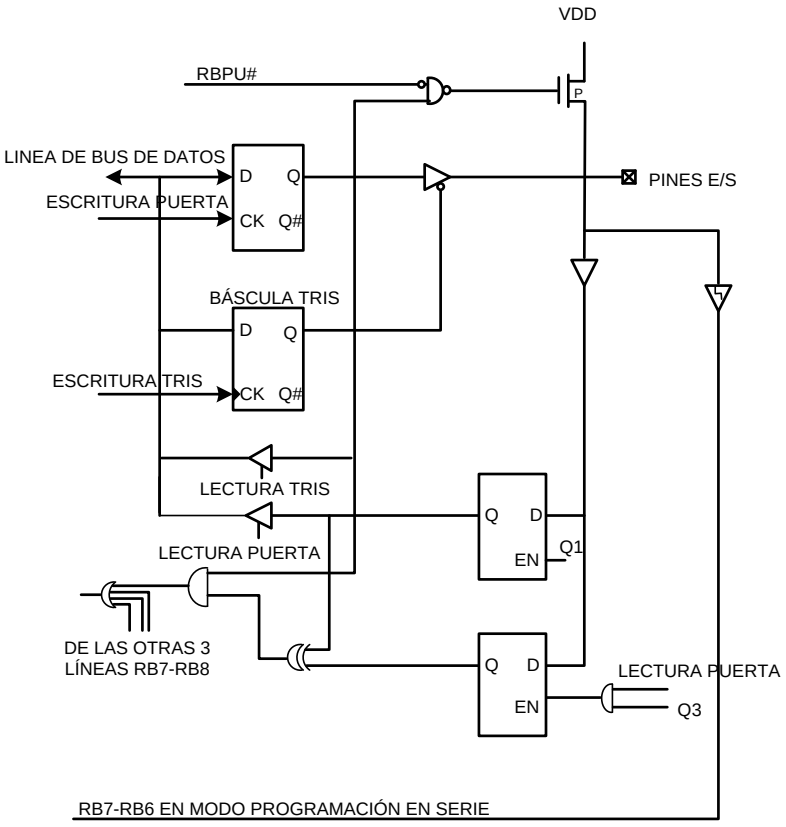


Figura 2.7. Conexionado interno de las patitas RB(7-4)de la puerta B

2.4.3. PUERTA C

Consta de 8 líneas bidireccionales cuyo sentido se configura mediante el registro TRISC. Todas las patitas de esta puerta tienen multiplexadas diferentes funciones.

- **RC0/T1OSO/T1CKI**. Esta línea puede actuar como E/S digital, como salida del Timer 1 o como entrada de impulsos para el Timer1.
- **RC1/T1OS1/T1CK2**. E/S digital, entrada al oscilador del Timer 1 ,entrada del módulo de Captura 2, salida del Comparador 2, Salida del PWM 2.
- **RC2/CCP1**. E/S digital, entrada Captura1, salida Comparador 1, Salida PWM 1.
- **RC3/SCK/SCL**. E/S digital, señal de reloj en modo SPI, señal de reloj en modo I2C.

- **RC4/SDI/SDA.** E/S digital, entrada de datos de modo SPl, línea de datos en modo I2C.
- **RC5/SDO**. E/S digital, salida de datos en modo SPI.
- **RC6/TX/CK**. E/S digital, línea de transmisión en USART, señal de reloj síncrona en transmisión serie.
- **RC7/RX/DT.** E/S digital, línea de recepción deL USART, línea de datos en transmisión serie síncrona.

2.4.4. PUERTA D

Esta puerta de 8 líneas bidireccionales, ocupa la dirección 08h, mientras que su registro de configuración TRISD ocupa la dirección 88h. Todas las patitas disponen de entrada tipo Schmidt Trigger.
Además de usarse como líneas de E/S digitales normales, implementan una puerta paralela esclava de 8 líneas (PSP), que sirve para permitir la comunicación en paralelo con otros elementos del sistema.
Las patitas se denominan RD0/PSP0-RD7/PSP7 y para que funcionen como puerta de comunicación esclava en paralelo es preciso poner el bit PSPMODE= 1. Este bit es el 4 del registro TRISE, que se comentará en la Puerta E.

2.4.5. Puerta E

Ocupa la dirección 09h. Dispone de 3 patitas multifunción, que se configuran como entrada o salida, según el valor de los tres bits de menos peso del registro TRISE, que está ubicado en la dirección 89h.
RE0/RD3/AN5. E/S digital, señal de lectura en el modo de puerta paralela esclava, canal 5 del conversor A/D.
RE1/WR#/AN6. E/S digital, señal de escritura en modo PSP, canal 6 del conversor A/D.
RE2/CS#/AN7. E/S digital, selección de chip en el modo PSP, canal 7del conversor A/D.

El PSP actúa como un puerto de comunicación en paralelo de 8 líneas y para su activación hay que poner el bit PSPMODE a 1. Dicho bit es el 4 del registro TRISE. Además de las 8 líneas de transferencia de datos, se precisan 3 señales de control, que determinan si la operación es de lectura, escritura y de permiso de funcionamiento (RD#, WR# y CS#). Estas tres líneas de control están implementadas en la Puerta E.

2.5. MÓDULOS ESPECIALES

A continuación se indica los tipos y características de los temporizadores

2.5.1. TEMPORIZADORES

El PIC 16F877 disponen de un potente conjunto de temporizadores para manejar eficientemente todas las operaciones que involucran al tiempo y conteo. Dichos temporizadores son tres y se denominan técnicamente TMRO, TMR1 y TMR2.

2.5.1.1. El TMR0.

Es idéntico al que tiene el P1C16F84 y sus funciones más representativas son:

- TMR0 es un contador y un temporizador de 8 bits.
- Leíble y escribible.
- Disparado por un reloj interno y externo.
- Selección del flanco en el reloj externo.
- Predivisor de la frecuencia de reloj programable.
- Generación de interrupción opcional en el desbordamiento.

El registro OPTION, es el encargado del control del TIMER 0 cuyas funciones son las siguientes:

- Asigna el divisor de frecuencias al timer0 (TIMER 0).
- Elige el rango en el que trabaja el divisor de frecuencia.

- Selecciona el tipo de reloj del TIMER0, que puede ser interno o externo a través de la patita TOK1. También selecciona el flanco activo.

2.5.2. TMR1

Tiene las siguientes características:

- TMR1 es un contador, temporizador de 16 bits.
- Leíble y escribible.
- Selección del reloj interno y externo.
- Interrupción opcional por desbordamiento del FFFFh a 0000h.
- Posible reinicialización desde los módulos CCP.

El TMR1 es el único Temporizador/Contador ascendente con un tamaño de 16 bits por lo que requiere el uso de dos registros concatenados de 8 bits: TMR1H:TMR1L, que son los encargados de guardar el valor del conteo en cada momento. Dicho valor evoluciona desde 0000h hasta FFFFh, instante en el que se activa el señalizador TMR1IF y se regresa al valor inicial 0000h. También, se puede provocar una señal de interrupción.
El valor contenido en TMR1H:TMR1L puede ser leído o escrito y los impulsos de reloj que originan el contaje ascendente pueden provenir del exterior o de la frecuencia de funcionamiento del microcontrolador.
El TMR 1 es capàs de funcionar de tres formas:

- Como temporizador.
- Como contador síncrono.
- Como contador asíncrono.

En el modo Temporizador, el valor concatenado TMR1H:TMR1L se incrementa con cada ciclo de instrucción. En el modo contador, el incremento se puede producir con los flancos ascendentes de un reloj externo, cuya entrada se aplica a las líneas RC0 y RC1 de la puerta C, o por impulsos aplicados en la línea RC0. En la figura 2.8 se muestra el diagrama por bloques del TMR 1, en el que destacan las diversas señales de control y el predivisor de frecuencia.

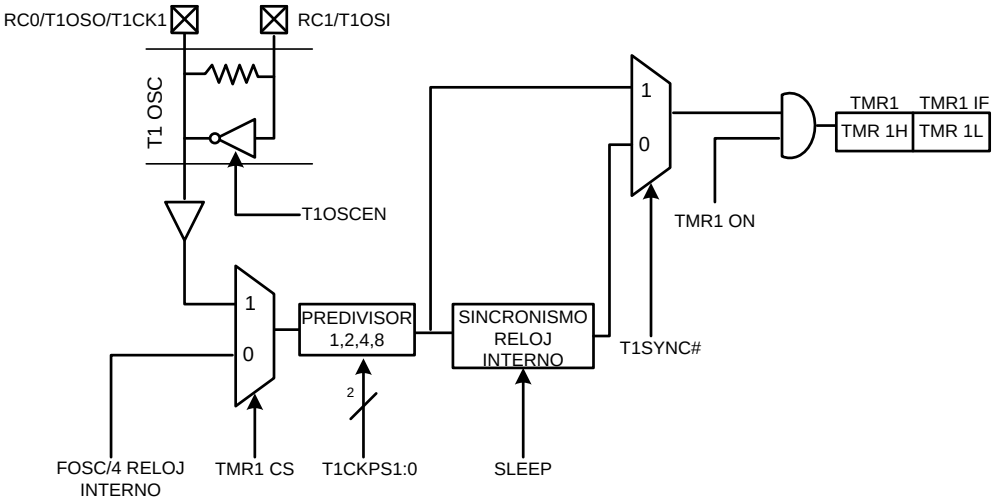


Figura 2.8. Esquema interno de los principales bloques del TMR1 con sus señales de control

El funcionamiento del TMR 1 está gobernado por el valor con el que se programan los bits del registro T1CON, que ocupa la dirección 10h de la memoria RAM y cuya denominación y distribución se presentan en la Figura 2.10

El bit TMR1ON gobierna el permiso o la prohibición de funcionamiento del Timer. En caso de poner un 0 en dicho bit el TMR1 no funciona.

El bit TMR1CS selecciona la fuente de los impulsos de contaje. Si vale 0 elige el reloj interno (Fosc/4) y si vale 1 el reloj externo que se aplica por las patitas RC0 y RC1.

Cuando los impulsos proceden de un reloj externo, es preciso que el bit T1OSCEN tenga valor 1, en cuyo caso las patitas RC0/T1OSO/T1CKI y RC1/T1OSI/CCP2 actúan como entradas del oscilador externo. Si T1OSCEN vale 0 los impulsos vendrán a través

de RCO/T1OSO/T1CKI. En ambos casos, el TMR1 funciona como contador de eventos externos y los bits 1 y 0 utilizados del registro TRISC carecen de significado, ya que una o ambas patitas RC1 y RCO no pueden actuar al mismo tiempo como entrada de impulsos y como líneas de E/S.

Entre las patitas RC1 y RC0 se puede poner un cristal de hasta una frecuencia de 200 KHz. En estas condiciones, el oscilador puede seguir funcionando aunque se force el estado de bajo consumo o SLEEP.

REGISTRO T1CON

-	-	T1CKPS1	T1CKPS0	T10SCEN	T1SYNC#	TMR1CS	TMR1ON

Figura 2.9. Nomenclatura y distribución de los bits del registro de control TMR1

2.5.3. TMR2

Es un Temporizador ascendente de 8 bits, que puede leer, escribir, y realizar funciones especiales para la Puerta Serie Síncrona (SSP) y con los módulos de captura y comparación. Ocupa la dirección 11h del mapa de los registros específicos.

La señal de reloj del TMR2 es interna (Fosc/4), y antes de ser aplicada pasa por un predivisor de frecuencia con rangos de 1:1, 1:4 y 1:16. La salida del TMR2 atraviesa un postdivisor de frecuencia con rangos de división desde 1:1 a 1:16, pasando por los 16 valores posibles.

Al entrar el microcontrolador en modo de reposo o SLEEP, se detiene el oscilador interno y al no existir la señal Fosc/4 deja de funcionar el TMR2.

Para controlar el funcionamiento del TMR2 se usa el registro T2CON, que ocupa la dirección 12h del mapa de los registros específicos y cuya distribución y asignación de bits se muestra en la figura 2.10

REGISTRO T2CON

-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2SCKPSI	T2SCKPS0

FIGURA 2.10. Distribución y asignación de los bits de registro T2CON que sirven para programar el trabajo del TMR2

2.5.4. CONVERSOR ANALÓGICO DIGITAL

Los microcontroladores PIC16F877 poseen un conversor A/D de 10 bits de resolución y 8 canales.

La resolución que tiene cada bit procedente de la conversión tiene un valor que es función de la tensión de referencia Vref, de acuerdo con la fórmula siguiente:

$$\text{Resolución} = (\text{Vref+} - \text{Vref-})/1.024 = \text{Vref}/1.024$$

Así, por ejemplo, si la Vref+ = 5 VDC y la Vref- es tierra, la resolución es de 4,8 mV/bit. Por tanto, a la entrada analógica de 0 V le corresponde una digital de 00 0000 0000 y para la de 5 V una de 1111111111. La tensión de referencia determina los límites máximo y mínimo de la tensión analógica que se puede convertir. El voltaje diferencial mínimo es de 2 V.

A través del canal de entrada seleccionado, se aplica la señal analógica a un condensador de captura y luego se introduce al conversor, el cual proporciona un resultado digital de 10 bits de longitud usando la técnica de «aproximaciones sucesivas».

El conversor A/D es el único dispositivo que puede funcionar en modo reposo (SLEEP), para ello el reloj del conversor deberá conectarse al oscilador RC interno.

2.5.4.1. REGISTRO DE TRABAJO

El funcionamiento del conversor A/D requiere la manipulación de cuatro registros:

- ADRESH. Parte alta del resultado de la conversión.
- ADRESL. Parte baja del resultado de la conversión.
- ADCON0. Registro de control 0.
- ADCON1. Registro de control 1.

En la pareja de registros ADRESH : ADRESL se deposita el resultado de la conversión, que al estar compuesta por 10 bits, sólo son significativos 10 de los bits de dicha pareja.
El registro ADCON0 controla la operación del C A/D, mientras que el ADCON1 sirve para configurar las patitas de la puerta A como entradas analógicas o E/S digitales la figura 2.11 indica la asignación de los bits de los registros de control

REGISTRO
ADCON0

ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE#	_	ADON
7							0

REGISTRO
ADCON1

ADFM	_	_	_	PCFG3	PCFG2	PCFG1	PCFGO
7							0

Figura 2.11. Asignación de los bits de los registros de control del C A/D ADCON0 ADCON1

Los bits ADCON0 (7Y6) sirven para seleccionar la frecuencia de reloj que se emplea en la conversión, con la siguiente asignación:

Tabla 2.3. Valores de frecuencia según el estado de los bits ADCS 1:0

ADCS 1:0	FRECUENCIA
00	Fosc/2
01	Fosc/8
10	Fosc/32
11	Frc (procede del oscilador RC interno)

2.5.4.2 ESTRUCTURA INTERNA Y CONFIGURACIÓN DEL C A/D

El bit de menos peso (ADFM) del registro ADCON1 selecciona el formato del resultado de la conversión. Si vale 1, el resultado está justificado en el registro ADRESH, que tiene 6 bits de más peso a 0; mientras que si vale 0 la justificación se realiza sobre el registro ADRESL, que tiene sus 6 bits de menos peso a 0. Esto significa que los 16 bits que forma la concatenación de ADRESH: ADRESL una vez tiene a 0 los 16 bits de más peso y otras los 6 bits de menos peso, alineación a la derecha o a la izquierda.

En la figura 2.12 se presenta el esquema del conexionado del conversor A/D con las patitas que soportan los canales de entrada y salida de la tensión de referencia.

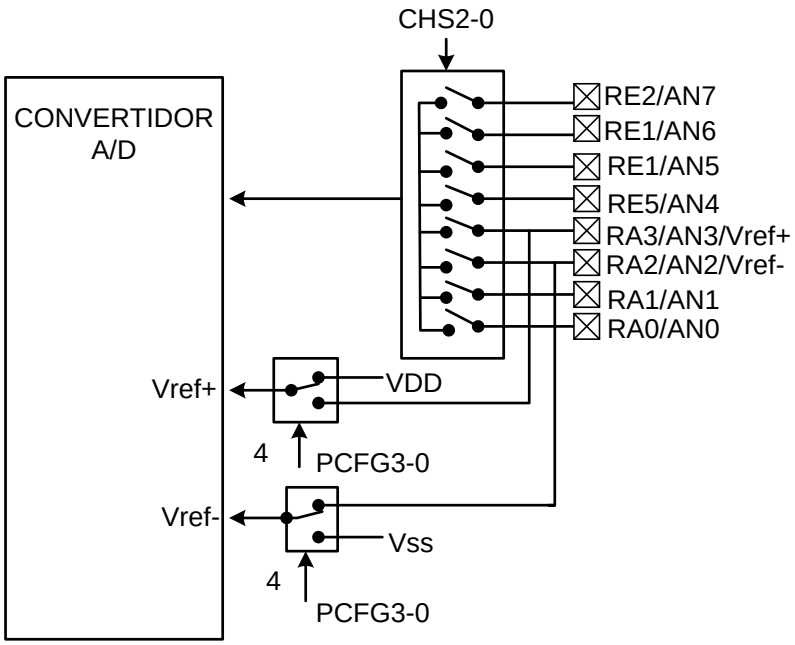


Figura 2.12. Estructura del conexionado del C A/D

2.5.5. MÓDULOS DE COMUNICACIÓN SERIE SÍNCRONA

2.5.5.1. MÓDULO MSSP.

Consta básicamente de dos registros: El SSPSR, que es un registro de desplazamiento que transforma la información serie en paralelo y viceversa, y el registro SSPBUF, que actúa como buffer de la información que se recibe o se trasmite.

El funcionamiento del módulo MSSP es muy sencillo. En transmisión, el byte que se quiere transmitir se carga en el registro SSPBUF a través del bus de datos interno y automáticamente se traspasa al registro SSPSR, que va desplazando bit a bit el dato, sacándolo ordenadamente al exterior al ritmo de los impulsos de reloj. En recepción, los bits van entrando al ritmo de reloj por una patita y se van desplazando en el SSPSR hasta que lo llenan, en cuyo momento la información se traspasa al SSPBUF, donde queda lista para su lectura. Este doble almacenamiento del dato

recibido permite iniciar la recepción de un nuevo dato antes de que se haya leído el último.

Cuando se han recibido 8 bits durante la recepción en SSPSR, se traspasa dicha información a SSPBUF y entonces el bit señalizador BF (Buffer Full) se pone en 1, al igual que la bandera de interrupción SSPIF. Cualquier escritura en el SSPBUF se ignora durante una transferencia de información y se señaliza poniendo a 1 el bit WCOL, que advierte de este tipo de colisiones. Recae en la responsabilidad del programador pasar el bit WCOL a 0 una vez completada la escritura en SSPBUF.

En la figura 2.13 se ofrece un esquema básico sobre la estructura interna del módulo MSSP.

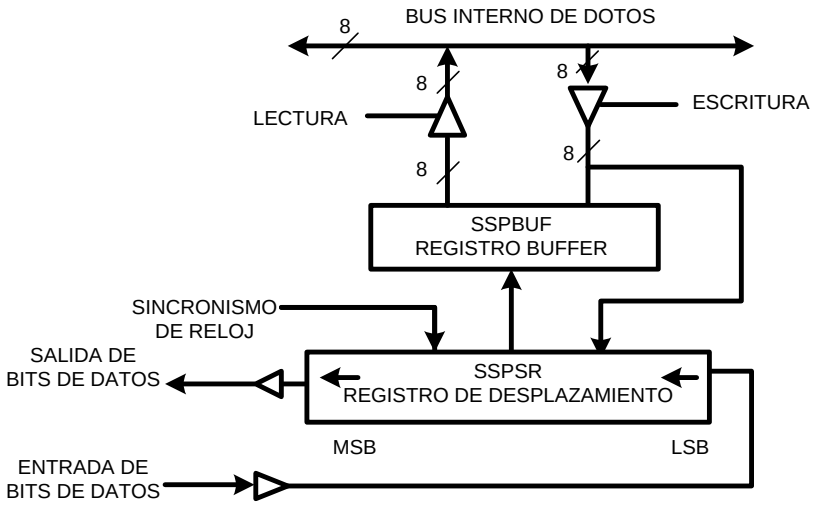


Figura 2.13. Estructura básica del módulo MSSP con los dos registros de trabajo (SSPBUF Y SSPSR)

2.5.5.2. MÓDULO MSSP TRABAJANDO EN MODO I2C

El protocolo de comunicación serie I2C fue desarrollado por Philips para cubrir sus propias necesidades en la implementación de diversos productos electrónicos que

requerían una elevada interconexión de circuitos integrados. El protocolo I2C (Inter-Integrated Circuit) utiliza únicamente dos líneas para la transferencia de información entre los elementos que se acoplan al bus. Una de dichas líneas se dedica a soportar los datos, es bidireccional y se llama SDA; la otra lleva los impulsos de reloj para la sincronización, es unidireccional y recibe el nombre de SCL. Los impulsos de reloj siempre los genera el maestro y tienen la función de sincronizar las transferencias con todos los esclavos colgados a las dos líneas que se puede observar en la figura 2.14

La eficiencia general del sistema depende de la versatilidad de los dispositivos conectados al mismo. Estos circuitos se pueden implementar con una estructura de bus serie. Este tipo de bus no puede alcanzar velocidades similares a las conseguidas a una estructura de bus paralelo, pero requieren mucho menos cables y el hardware es mucho más sencillo.

Los distintos dispositivos conectados al bus serie deben comunicarse entre si mediante un protocolo que evite el bloqueo de la información y garantice la comunicación entre todos ellos,.

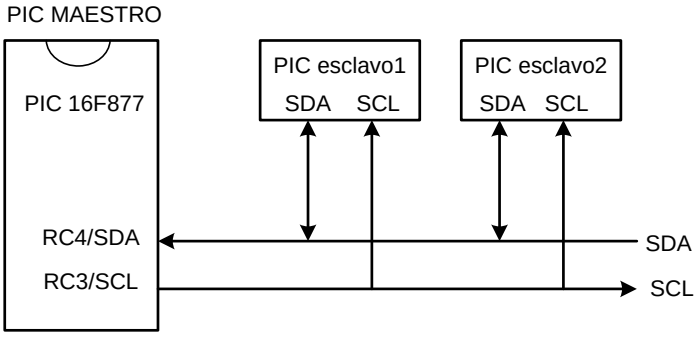


Figura 2.14. El maestro se conecta con todos los esclavos en el BUS I2C con solo dos líneas SDA, SCL.

2.5.6. MÓDULO CCP

El microcontrolador PIC16F877 dispone de dos módulos CCP, llamados CCP1 y CCP2, que son idénticos excepto en lo referente a la modalidad de «Disparo Especial». Dada esta similitud, la descripción

se orienta hacia el módulo CCPI. Estos módulos pueden realizar tres funciones principales:

- Modo captura. Una pareja de registros de un módulo CCPx captura el valor que tiene el TMR1 cuando ocurre un evento especial en la patita RC2/CCP1 (para el módulo CCP1) o en la RC1/T1OSI/CCP2 (para el CCP2).
- Modo comparación. Se compara el valor de 16 bits del TMR1 con otro valor cargado en una pareja de registros de un módulo CCPx y cuando coinciden se produce un evento en las patitas RC2/CCP1 y/o RC1/T10SI/CCP2.
- Modo modulación de anchura de pulsos (PWM). Dentro del intervalo del período de un impulso controla la anchura en que la señal vale nivel alto.

El módulo CCP1 utiliza un registro de trabajo de 16 bits que está formado por la concatenación de los registros CCPR1H-CCPR1L (direcciones 16h y 15h). El registro de control del módulo CCP1 es el CCP1CON, que ocupa la dirección 17h (figura 2.15). El módulo CCP2 tiene como registros de trabajo a CCPR2H-CCPR2L (direcciones 1Ch y 1Bh) y como registro de control a CCP2CON en la dirección 1Dh.

REGISTRO CCPxCON (x puede ser 1 o 2)

_	_	CCPxX	CCPxY	CCPxM2	CCPxM1	CCPxM0

Figura 2.15. Asignación de los bits de los registros CCPXCON para los módulos CCP1 Y CCP2

2.6. SISTEMA DE COMUNICACIÓN

En la actualidad se cuenta con muchos tipos de sistemas de comunicación, que permiten el flujo de información entre circuitos integrados. Por lo cual resulta importante estudiar la comunicación asíncrona, que será utilizada en este proyecto.

2.6.1. COMUNICACIÓN SERIE ASÍNCRONA

El PIC16F877 contiene un módulo MSSP con dos puertas para la comunicación serie «síncrona», o sea, con señal de reloj. Además, también disponen de un módulo USART capaz de soportar la comunicación serie síncrona y asíncrona.

El USART, llamado SCI (Serial Comunications Interface), puede funcionar como un sistema de comunicación full duplex o bidireccional asíncrono, adaptándose a multitud de periféricos y dispositivos que transfieren información de esta forma. También puede trabajar en modo síncrono unidireccional o halt duplex para soportar periféricos como memorias, conversores, etc. En resumen, el USART puede trabajar de tres maneras:

- ASÍNCRONA.(Full duplex, bidireccional).
- SÍNCRONA – MAESTRO (Half duplex, unidireccional).
- SÍNCRONA – ESCLAVO (Half duplex, unidireccional).

En la figura 2.16 se muestra un esquema del comportamiento del USART en modo asíncrono y síncrono. En el primero, las transferencias de información se realizan sobre dos líneas TX (transmisión) y RX (recepción), saliendo y entrando los bits por dichas líneas al ritmo de una frecuencia controlada internamente por el USART. En el modo síncrono, la comunicación se realiza sobre dos líneas, la DT que traslada en los dos sentidos los bits a la frecuencia de los impulsos de reloj que salen por la línea CK desde el maestro. En ambos modos las líneas de comunicación son las dos de más peso de la Puerta C: RC6/TX/CK y RC7/RX/DT.

En esta forma de comunicación serie, se usa la norma RS-232, donde cada palabra de información o dato se envía independientemente de los demás. Suele constar de 8 o 9 bits y van precedidos por un bit de START (inicio) y detrás de ellos se coloca un bit de STOP (parada), de acuerdo con las normas del formato estándar NRZ (NonReturn-to-

Zero). (figura 2.17) Los bits se transfieren a una frecuencia fija y normalizada.

Los cuatro bloques que configuran la arquitectura del USART, en modo asíncrono, son:

- Circuito de muestreo
- Generador de baudios
- Transmisor asíncrono.
- Receptor asíncrono.

El circuito de muestreo actúa sobre la patita RC7/RX/DT, que es por donde se recibe el bit de información o control y se encarga de muestrear tres veces su valor, para decidir éste por mayoría.

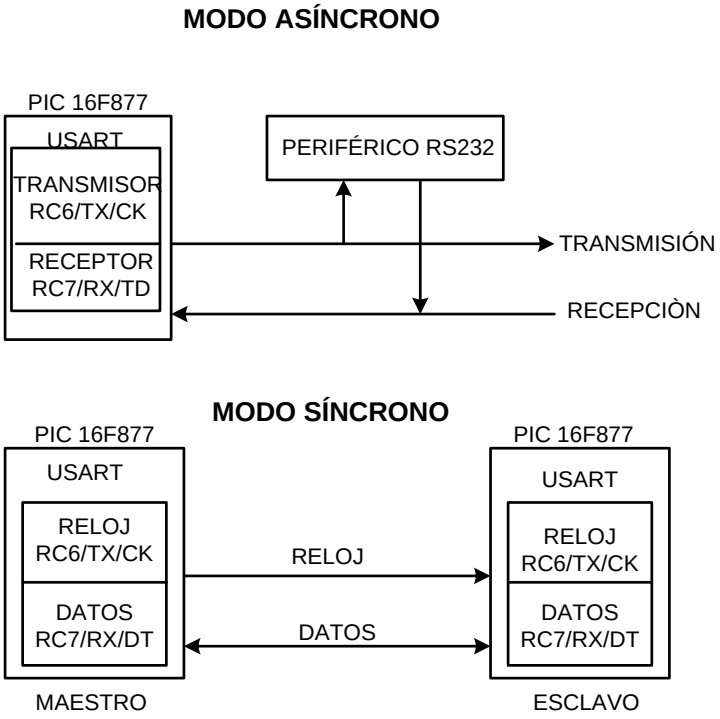


FIGURA 2.16. En el modo asíncrono, la comunicación serie del USART. En el modo síncrono los bits de información circulan en ambas direcciones por la línea DT

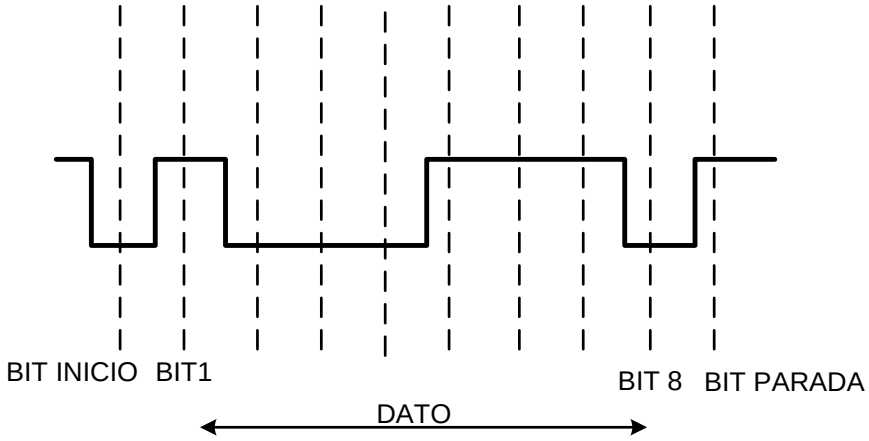


FIGURA 2.17 En el protocolo de comunicación, cada palabra de información consta de 8 o 9 bits, precedidos por un bit de inicio y parada

CAPÍTULO III

3. SOFTWARE DE INSTRUMENTACIÓN VIRTUAL

En la actualidad se cuenta con varios programas de instrumentación virtual, que permiten al operador tener en la pantalla de su ordenador todos los datos que se puedan tomar de un proceso y a la vez realizar el control de dichas variables, sin tener que acudir a los clásicos controles y visualizadores físicos.

3.1. INTRODUCCIÓN A LA INSTRUMENTACIÓN VIRTUAL

Cuando se habla de instrumentos de medida, es normal pensar en una carcasa rígida, en la que destaca su panel frontal lleno de botones, leds y demás tipos de controles y visualizadores. En la cara oculta del panel están los contactos de esos controles que los unen físicamente con la circuitería interna. Esta circuitería interna se compone de circuitos integrados y otros elementos que procesan las señales de entrada en función del estado de los controles, devolviendo el resultado a los correspondientes visualizadores del panel frontal.

Un instrumento virtual es un módulo software que simula el panel frontal, apoyándose en elementos de hardware accesibles por el ordenador (tarjetas de adquisición, tarjetas DSP, instrumentos accesibles vía GPIB, VXI, RS-232), realiza una serie de medidas como si se tratase de un instrumento real.

Cuando se ejecuta un programa que funciona como instrumento virtual o VI (Virtual instrumento), el usuario ve en la pantalla del ordenador un panel cuya función es idéntica a la de un instrumento físico, facilitando la visualización y el control del aparato. A partir de los datos reflejados en el panel frontal, el VI debe actuar recogiendo o generando señales, como lo haría su homólogo físico.

Hasta hace poco, la tarea de construcción de un VI se llevaba a cabo con paquetes software que ofrecían una serie de facilidades, como funciones de alto nivel y la Incorporación de elementos gráficos, que simplificaban la tarea de programación y de elaboración del panel frontal. Sin embargo, el cuerpo del programa seguía basado en texto, lo que suponía mucho tiempo invertido en detalles de programación que nada tienen que ver con la finalidad de un VI. Con la llegada del software de programación gráfica LabVIEW de National lnstruments, Visual Designer de Burr Brown o VEE de Hewlett Packard, el proceso de creación de un VI se ha simplificado notablemente, minimizándose el tiempo de desarrollo de las aplicaciones.

Cuando se crea un VI en LabVIEW se trabaja con dos ventanas: Una en la que se implementará el panel frontal y otra que soportará el nivel de programación. Para la creación del panel frontal se dispone de una librería de controles e indicadores de todo tipo y la posibilidad de crear más, diseñados por el propio usuario.

Cuando un control es ‘pegado” desde la librería en el panel frontal, se crea una variable cuyos valores vendrán determinados por lo que el usuario ajuste desde el panel; inmediatamente, aparece un terminal en la ventana de programación representándolo. El nivel de programación del VI consistirá en conectar estos terminales a bloques funcionales, hasta obtener un resultado que se desea visualizar. Los bloques funcionales son iconos con entradas y salidas que se conectan entre sí mediante cables ficticios por donde fluyen los datos, constituyendo el nivel de programación del VI.

Se puede comparar la ventana de programación con una placa de circuito impreso, donde los terminales del panel frontal se cablean a bloques funcionales (circuito integrado) que se interconectan para generar los datos que se desean visualizar. A su vez, estos circuitos integrados contienen bloques circuitales conectados entre sí, al igual que un icono está formado por la interconexión de otros iconos. La programación gráfica permite diseñar un VI de manera intuitiva, vertiendo las ideas directamente a un diagrama de bloques, como se haría sobre una pizarra.

3.2. CARACTERÍSTICAS GENERALES DEL SOFTWARE LABVIEW.

En este apartado se discuten los aspectos necesarios para familiarizarse con el uso del LabVIEW.

Cada VI tiene dos ventanas separadas, pero relacionadas entre sí. La ventana Panel contiene el panel frontal del VI. La ventana Diagrama es aquella en la cual se construye el diagrama de bloques.

3.2.1. TIPOS DE DATOS EN LABVIEW. CONTROLES E INDICADORES.

LabVIEW ofrece una gran variedad de tipos de datos con los que se puede trabajar respondiendo a las necesidades reales que encontrare. Uno de los aspectos más significativos de LabVIEW es la diferenciación que efectúa en el diagrama de bloques entre los diferentes tipos de Controles o indicadores, basada en que cada uno de ellos tiene un color propio. De esta manera, y como consecuencia de una memorización o asimilación práctica, será muy fácil identificarlos y reconocer inmediatamente si se está trabajando con el tipo de datos adecuado. Se distingue los siguientes tipos, los cuales pueden funcionar tanto como controles como indicadores (entre paréntesis queda reflejado el color con el que queda representado en el diagrama de bloques):

Boolean (verde claro)

Los tipos de datos booleanos son enteros de 16 bits. El bit más significativo contiene el valor Booleano. Si el bit 15 se pone a 1, entonces el valor del control o indicador es **true** (verdadero); por el contrario, si este bit 15 vale 0, el valor de la variable booleana será false (falso).

Extended (naranja)

Los números de coma flotante con precisión extendida presentan el formato:

Windows: 80 bits (formato precisión extendida 80287).

Double (naranja)

Los números en coma flotante de doble precisión cumplen con el formato de doble precisión IEEE de 64 bits. Es el valor por defecto de LabVIEW.

Single (naranja)

Los números en coma flotante de precisión simple cumplen con el formato de precisión simple IEEE de 32 bits.

Long Integer (azul)

Los números enteros largos tienen un formato de 32 bits con o sin signo.

Word Integer (azul)

Estos números tienen un formato de 16 bits, con o sin signo.

Byte Integer (azul)

Tienen un formato de 8 bits, con o sin signo.

Unsigned Long (azul) Entero largo sin signo.

Unsigned Word (azul) Palabra sin signo.

Unsigned Byte (azul) Byte sin signo.

CompleX Extended (naranja) Número complejo con precisión extendida.

CompleX Double (naranja) Complejo con precisión doble.

Complex Single (naranja) Complejo con precisión simple.

3.2.2. PROGRAMACIÓN ESTRUCTURADA

Programar una aplicación en LabVIEW por su carácter de tipo estructurada puede parecer muy diferente a hacerlo en cualquier otro lenguaje de alto nivel. Pero los planteamientos generales deben ser los mismos sea cual sea el lenguaje escogido.

Un programa siempre se basará en la construcción de un Algoritmo y el empleo de unas Estructuras de Datos.

Por Algoritmo se entiende la descripción exacta del orden determinado en que se ha de ejecutar un sistema de operaciones para resolver todos los problemas de un mismo tipo.

Características de los Algoritmos son, pues, Finitud, Definibilidad (de todas las acciones a realizar paso a paso y sin ambigüedad), Generalidad (todos los problemas de un determinado tipo) y Efectividad (funcionamiento correcto en todos los casos).

La implementación del algoritmo lleva a codificar cada una de las acciones que lo constituyen a instrucciones de un lenguaje determinado, en nuestro caso LabVIEW, teniendo de este modo un programa en LabVIEW en forma estructurada.

A la hora de programar, muchas veces es necesario ejecutar un mismo conjunto de sentencias un número determinado de veces, o que éstas se repitan mientras se cumplan ciertas condiciones. También puede ocurrir que sea necesario ejecutar una u otra sentencia dependiendo de las condiciones fijadas o simplemente forzar a que unas se ejecuten siempre antes que otras.

Para ello LabVIEW dispone de cuatro estructuras fácilmente diferenciables por su apariencia y disponibles en la opción Structures del menú Function de la ventana diagrama.

Se usa For Loop cuando se requiera que una operación se repita un número determinado de veces. Su equivalente en lenguaje convencional es:

For i = 0 to N-1

Se usa While Loop cuando se requiera que una operación se repita mientras una determinada condición sea cierta. Su equivalencia en lenguaje convencional es:

Do ejecutar subdiagrama

While condición es **TRUE**

Los registros de desplazamiento o “shift register” son variables locales, disponibles tanto en el For Loop como en el While Loop, que permiten transferir los valores del final de una iteración al principio de la siguiente.

Las estructuras Case y Sequence se diferencia de las iterativas en que puede tener múltiples subdiagramas, de los cuales solamente uno es visible a la vez. En la parte superior de cada estructura existe una pequeña ventana que muestra el identificador del subdiagrama que se está mostrando. A ambos lados de esta ventana existen dos botones que decrementan o incrementan el identificador de forma que se pueda ver el resto de subdiagramas.

3.2.3. VARIABLES LOCALES Y GLOBALES

Las variables son imprescindibles en cualquier tipo de problemas, ya que permiten almacenar la información necesaria para su resolución. En LabVIEW todos los controles introducidos en el Panel Frontal que generan un terminal en la ventana Diagrama van a ser variables,

identificables por el nombre asignado en la etiqueta. Pero puede ocurrir que se requiera utilizar el valor de cierta variable en otro subdiagrama o en otro VI o, simplemente, que se necesite guardar un resultado intermedio. La forma más sencilla de hacerlo es generando variables locales y/o globales dependiendo de la aplicación.

En las variables locales los datos se almacenan en algunos de los controles o indicadores existentes en el Panel Frontal del VI creado; es por eso que estas variables no sirven para intercambiar datos entre VI's. La principal utilidad de estas variables radica en el hecho de que una vez creada la variable local no importa que proceda de un indicador o de un control, ya que se podrá utilizar en un mismo Diagrama tanto de entrada como de salida.

Las variables globales son un tipo especial de VI, que únicamente dispone de Panel Frontal, en el cual se define el tipo de dato de la variable y el nombre de identificación imprescindible para después poder referirse a ella.

3.2.4. ANÁLISIS Y VISUALIZACIÓN DE DATOS

En cualquier proceso es indispensable el poder tener acceso a ciertas medidas o características de los estados de la variable que se está midiendo, es por eso que labVIEW ha estructurado varias formas de análisis de la variable, que brindan al personal que realiza el control una idea clara y precisa de lo que se tiene como resultado de una variable.

El poder visualizar estos datos obtenidos en estructuras creadas para este propósito facilita que el encargado de realizar el control del proceso tenga muchas ventajas sobre el problema a resolverse.

En muchas ocasiones es necesario para una mayor comprensión de los resultados obtenidos representarlos gráficamente. Para ello LabVIEW dispone de cinco tipos de gráficos accesibles desde el menú

Controls del panel Frontal bajo el ítem Graph, divididos en dos grupos: Los indicadores chart y los indicadores graph.

Un indicador graph o indicador gráfico es una representación bidimensional de uno o más gráficos. El graph recibe los datos como un bloque. Un indicador chart o de trazos también muestra gráficas, pero éste recibe los datos y los muestra punto por punto o array por array, reteniendo un cierto número de puntos en pantalla mediante un buffer disponible para ello.

Waveform chart es un tipo especial de indicador numérico que muestra una o mas gráficas, reteniendo en pantalla un cierto número de datos definido por nosotros mismos. Los nuevos datos se añaden al lado de los ya existentes, de forma que se pueden comparar entre ellos.
Los datos se pueden pasar uno a uno al chart o mediante arrays. Evidentemente es conveniente pasar múltiples puntos a la vez ya que de esta manera sólo es necesario redibujar la gráfica una vez y no una por cada punto.

Es posible dibujar varias gráficas en un mismo chart, uniendo los datos de cada gráfica en un cluster de escalares numéricos de forma que cada escalar que contiene el cluster se considera como un punto de cada una de las gráficas para una misma abcisa. Se puede ahorrar tiempo uniendo los clusters en arrays y después pasando todo el array a la gráfica.

Mediante intensity chart se muestra datos tridimensionales colocando bloques de colores sobre planos cartesianos. Para ello se crea arrays bidimensionales de números donde los índices de un elemento corresponderán a las coordenadas X e Y, y el contenido a la coordenada Z, que tendrá asociado un color para cada posible valor.

Cada vez que se envíe un nuevo conjunto de datos, estos aparecerán representados a la derecha de los ya existentes. Intensity chart soporta

los tres modos de visualización de waveform chart y también dispone de un buffer cuyo tamaño es por defecto, de 128 puntos. Las opciones disponibles para Intensity chart son prácticamente las mismas que para waveform chart

Waveform graph representa una serie de valores Y equiespaciados dada siempre una distancia delta de X (AX) comenzando a partir de un valor inicial Xo. A un mismo punto X1 sólo le puede corresponder un valor de Y1. Cuando se representa una nueva serie de datos, al contrario de lo que ocurría en los indicadores chart, estos datos reemplazan a los ya existentes en lugar de añadirse al lado, y pierden los valores representados con anterioridad.

Existen dos posibilidades a la hora de representar una única gráfica en una waveform graph. La primera consiste en unir un array de valores numéricos directamente a la graph de forma que ésta interpreta cada valor como un nuevo punto comenzando en X=ø e incrementando X en 1 para cada punto.

La segunda consiste en crear un cluster en el cual, junto con el array de valores, se indica el valor inicial X0 y el incremento AX.

Existe la posibilidad de representar más de un gráfica en una misma waveform graph. Para ello es necesario unir los datos de las diferentes gráficas en un formato que LabVIEW sepa interpretar. Utilizar un formato u otro vendrá determinado principalmente por las características de las gráficas a mostrar. Así, si todas las gráficas tienen un mismo escalado X y un mismo número de puntos, bastará con crear un array bidimensional de valores numéricos donde cada fila de datos es una única gráfica. LabVIEW interpretará estos datos como puntos en la gráfica comenzando en X=ø e incrementándola en 1. Si interesa cambiar el punto inicial o el incremento de x, se crea un cluster que contendrá el array bidimensional y los valores de x0 y Ax.

En XY Graph un punto X1 puede tener varios valores Y, lo que permite, por ejemplo, dibujar funciones circulares. XY Graph representa una coordenada (X1, Y1) donde los valores de X no tienen porque estar equiespaciados como ocurría en las *waveform graph.* Para representar una única gráfica en una XY Graph existen dos posibilidades. La primera consiste en crear un cluster que contenga un array de datos X y un array de datos Y. La segunda consiste en crear un array de clusters, donde cada cluster contiene un valor de X y un valor de Y.

3.3. LAS COMUNICACIONES SERIE EN LabVIEW

En la actualidad la comunicación serie en labVIEW, es una nueva opción complementaria para crear aplicaciones que puedan cumplir con requerimientos de comunicación con elementos externos de la PC.

3.3.1. EL ESTÁNDAR RS-232

El RS 232 o "Recommended Standard 232 está definido en las especificaciones ANSI (American National Standard Institution) como 'la interface entre un equipo terminal de datos y un equipo de comunicación de datos empleando un intercambio en modo serie de datos binarios". En él se describen las diferentes reglas a seguir para realizar una comunicación serie entre dos dispositivos distantes entre si.

Normalmente, los dispositivos que intervienen en una comunicación serie son el Equipo Terminal de Datos (ETD), que suele ser un PC, y el Equipo de Comunicación de Datos (ECD), generalmente un módem. A pesar de que el estándar RS 232 empezó utilizándose para la comunicación entre un PC y un módem, la gran implantación de los PCs ha derivado en la ampliación del uso del RS 232, convirtiéndose en el estándar más utilizado en aplicaciones de bajo coste que requieran la interconexión serie entre un ETD y un periférico. Como

periféricos serie más usuales se pueden nombrar las impresoras, el ratón, los plotters, los scanners, los digitalizadores, etc.

El estándar ha ido evolucionando a lo largo de los años, durante los cuales ha sufrido diferentes revisiones. La última de estas revisiones ha sido la “E”, realizada en julio de 1991. Ahora, el estándar es conocido como el EIA/TIA-232-E, donde EIA significa “Electronic Industries Association” y TIA significa “Telecommunications Industry Association”.

Las características principales que definen el estándar son:
• Velocidad máxima de transmisión de datos: 20 K bits por segundo (kbps). Ahora bien, existen aplicaciones que se salen de las especificaciones del estándar que llegan a velocidades de hasta 116 kbps.
• Capacidad de carga máxima: 2500 pF. Esto se traduce en una longitud máxima de cable entre el PC y el periférico de 15 a 20 metros. Para distancias mayores se ha de utilizar otro estándar de comunicaciones.

3.3.2. EL CONECTOR DB9S

Dado que es muy utilizado el conector de 9 patillas en las comunicaciones serie basadas en el RS 232, a continuación se muestra una tabla resumen con la función asociada a cada patilla. A la hora de construir un cable para la interconexión serie de dos dispositivos mediante RS 232, esta información es indispensable.

PATILLA	SIGLAS	DESCRIPCION
1	DCD	Data Carrier Detec
6	DSR	Data Set Ready
2	RD	Receive Data Line
7	RTS	Request To Send

Tabla	3	TD	Transmit Data Line
	8	CTS	Clear To Send
	4	DTR	Data Terminal Ready
	9	RI	Ring Indicator
	5	GND	Signal Ground

3.1. Pines del conector DB9

A continuación se describe la función de cada una de las patillas.

• Data Carrier Detect (DCD): El DCE pone a ‘1” esta línea para informar al DTE que está recibiendo ura señal portadora con información.

• Data Set Ready (DSR): Es una señal que el DCE pone a “1” para indicar al DTE que está conectado a la línea.

• Receive Data Line (RD): Las señales que se reciben por la línea RD son en forma de transmisión Serie. Cuando la señal DCD está a “0”, la línea RD se ha de mantener en el estado Mark.

• Request To Send (RTS): Esta señal es puesta a “1’ por el DTE para indicar que está preparado para transmitir datos. Entonces el DCE ha de prepararse para recibir datos. En comunicaciones Half Duplex también se inhibe el modo de recepción de datos. Después de una

cierta espera, el DCE pone a “1” la línea CTS para informar al DTE de que ya está preparado para recibir datos. Una vez que la comunicación ha finalizado y no se transmiten más datos por parte del DTE, RTS pasa do valer “1” a valer “0”. Después de un pequeño tiempo de espera, para asegurarse de que han sido recibidos todos los datos transmitidos, el DCE pone a “ø” la línea CTS.

• Transmit Data Line (TD): Las señales se transmiten por esta línea, en modo serie, del DTE al DCE. Cuando no se está transmitiendo ningún tipo de información, la línea ha de mantenerse en su estado Mark. Para que se puedan transferir datos, las líneas DSR, DTR, RTS y CTS han de encontrarse a “1”.

• Clear To Send (CTS): Esta señal es puesta a “1” por el DCE para indicar al DTE que está preparado para recibir datos. CTS es puesta a “1” como respuesta a un estado “1” simultáneo de las líneas RTS, DSR *y* DTR.

• Data Terminal Ready (DTR): Esta señal, conjuntamente con DSR, indica que los equipos están operativos. DTR es puesta a “1” por el DTE para indicar al DCE que está preparado para recibir o transmitir datos. DTR ha de estar a “1” antes de que el DCE pueda poner a “1” DSR. Cuando DTR es puesta a “ø” por el DTE, el DCE es desconectado del canal de comunicaciones dado que ya ha sido completada la transmisión de la información.

• Ring Indicator (RI): RI es puesta a “1” por el DCE cuando está recibiendo una llamada. Esta línea ha dejado de ser útil al emplearse el estándar en las aplicaciones de modems.

• Signal Ground (pin 5)**:** Esta línea proporciona el común, la referencia de tierra, a todas las líneas antes expuestas. Está eléctricamente separada de la toma de tierra para protección del equipo.

3.3.3. UTILIZACIÓN DEL PUERTO SERIE MEDIANTE LABVIEW

LabView es un sistema de programación de propósito general, pero también incluye librerías de funciones y herramientas de desarrollo diseñadas específicamente para adquisición de datos y control de instrumentos. En esta sección, se analizará una forma de realizar una comunicación por el puerto de comunicaciones RS-232 utilizando las librerías de las que se dispone para tal fin.

- **Property Node:** El Nodo de Propiedad, lee o escribe las propiedades de una referencia. Además adapta automáticamente la clase del objeto al que se hace referencia.

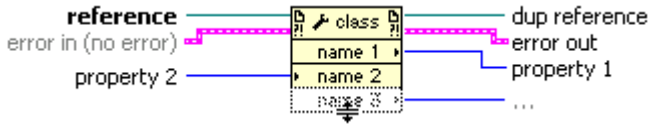


Figura 3.1. Property Node

reference: Es una referencia numérica asociado con un objeto para una conexión TCP.

error in: Describe condiciones de error, que ocurren antes del VI o en el retorno de funciones. El valor predeterminado de la línea no es un error. Si un error ocurriera antes del VI o al retornar una función, constituye un error de salida.

property 2..n: Es un ejemplo de una propiedad, que se desea poner (escribir).

dup reference: Retorno de referencia inalterada.

error out: Contiene la información del error, además describe el estado del error que este VI o la función produce.

property 1..n: Es un ejemplo de una propiedad, que se desea obtener(leer).

- **VISA Configure Serial Port:** Inicializa el puerto serie especificado con el nombre del recurso VISA para las configuraciones.

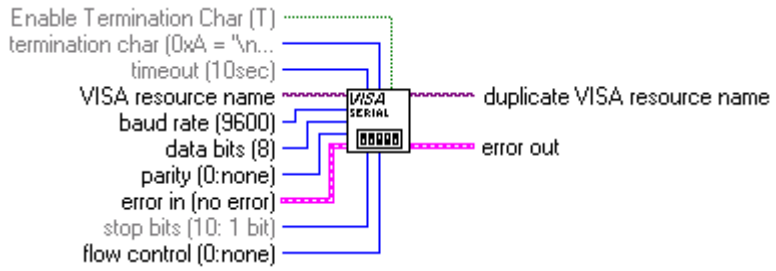


Figura 3.2. VISA Configure Serial Port

Enable Termination Char: Prepara al dispositivo serie para el reconocimiento del Char de finalización.

termination char: Finaliza las llamadas de funciones de lectura.

timeout: fija el valor de la interrupción para escribir y leer las funciones.

VISA resource name: Especifica el recurso a ser abierto.

baud rate: Es la tasa de transmisión de datos, está definido por defecto y es 9600.

data bits: Indica el número de data bits entrantes. El valor de los data bits está entre 5 y 8. El valor por defecto es 8.

parity: Especifica la paridad usada para cada conjunto de datos a ser transmitidos o recibidos.

error in: Describe condiciones de error que ocurren antes de este VI o en el corrido de la función.

stop bits: Especifica el número de bits de parada indicados al final de un conjunto de datos**.**

flow control: Fija el tipo de control usado por el mecanismo de transferencia de datos.

duplicate VISA resource name: es una copia del nombre de recurso VISA que es transmitido a la salida de la funcione VISA.

error out: Contiene la información del error.

- **VISA Write:** Escribe los datos desde el write buffer al dispositivo o interfaz especificado por el nombre del recurso del VISA.

Figura 3.3. VISA Write

VISA resource name: Especifica el recurso a ser abierto. Este mando también especifica la sesión y clase.

write buffer: Contiene los datos a ser escritos al dispositivo.

error in: Describe condiciones del error que ocurren antes de este VI o al ejecutar la función.

dup VISA resource name: Es una copia del nombre del recurso del VISA de retorno de la función.

return count: Contiene el número real de bytes escritos.

error out: Contiene la información del error.

- **VISA Read:** Lee el número especificado de bytes desde el dispositivo o la interfaz especificó por el nombre de recurso del VISA e ingresos los datos en el read buffer.

Figura 3.4. VISA Read

VISA resource name: Especifica el recurso a ser abierto. Este mando también especifica la sesión y clase.

byte count: Es el número de bytes para ser leído.

error in: Describe condiciones del error que ocurren antes de este VI o al ejecutar la función. El valor predeterminado no es ningún error.

dup VISA resource name: Es una copia del nombre del recurso del VISA de retorno de la función.

read buffer: Contiene los datos leídos del dispositivo.

return count: Contiene el número de bytes leídos realmente.

error out : Contiene la información del error

- **VISA Close:** Cierra una sesión del dispositivo o el evento. Por cada sesión de VISA que se abra, cuando se ha terminado con èsta, se debe serrar la sesión.

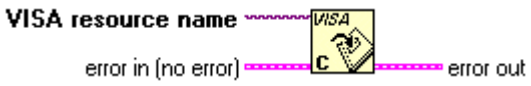


Figura 3.5. VISA Close

VISA resource name: Especifica el recurso a ser abierto. Este control además detalla la sesión y clase.

error in: Describe condiciones de error que ocurren antes de este VI o al ejecutar la función. El valor predeterminado no es ningún error.

error out: Contiene la información del error. Además, describe el estado del error que este VI o la función produce.

- **VISA Serial Break:** Envía un interrupción en el puerto de salida especificado de por lo menos 250 milisegundos.

Figura 3.6. VISA Serial Break

VISA resource name: Especifica el medio a ser abierto. Este mando también especifica la sesión y clase.

error in: Describe condiciones de error que ocurren antes de este VI o al ejecutar una función. El valor predeterminado no es ningún error.

duplicate VISA resource name: Es una copia del nombre del recurso del VISA

error out: Contiene la información del error. Además, describe el estado del error que este VI o la función produce.

CAPÍTULO IV

4. INTERFASE RS232

RS232 es el nombre del interfaz de comunicación serie más utilizado del mundo. La norma serie está disponible en prácticamente el 99% de los ordenadores. Entre ellos el IBM PC compatible que habitualmente está equipado con un puerto serie RS232, utilizado para conectar la impresora.

La norma RS232 fue originalmente diseñada para conectar terminales de datos con dispositivos de comunicación. Desde un principio, fue también utilizada para conectar casi cualquier dispositivo imaginable. Los usos de la RS232 en el entorno doméstico son muchos y ampliamente conocidos. Desde la conexión del ratón, el modem/fax, agendas electrónicas de bolsillo, impresoras serie, digitalizadores de vídeo, radios de AM/FM, etc. La lista sólo está limitada por la imaginación de los diseñadores.

En el entorno industrial el peso de la RS232 es también muy importante. Si bien existen soluciones de comunicación serie más robustas y versátiles, como la RS422 o la RS485, la RS232 sigue siendo por su sencillez, su diseño económico y, sobre todo, por su gran difusión, la norma más frecuente. Así, es fácil ver cómo robots industriales, manipuladores, controles de todo tipo, utilizan la RS232.

4.1. GENERALIDADES DE LA COMUNICACIÓN SERIE.

El uso del estándar RS232 brinda muchas ventajas, que serán aprovechadas en este proyecto, ya sea por la facilidad en el modo de utilización de la interfase o por el costo que representa.

4.1.1. TRANSMISIÓN SERIE/PARALELO.

Conceptualmente una transmisión paralelo consiste en utilizar simultáneamente varios circuitos de transmisión serie. Dejando al margen problemas específicos de una transmisión en paralelo, como puede ser interferencia inducida de símbolos, la transmisión paralelo es el recurso lógico cuando un solo circuito no proporciona un ancho de banda suficiente. Si en un diseño, un problema de transmisión puede

resolverse con una transmisión serie, esta opción es en principio deseable frente a una paralelo.

En una transmisión con múltiples circuitos, la probabilidad de fallo de línea y la necesidad de mantenimiento es proporcional al número de líneas utilizadas.

4.1.2. TRANSMISIÓN SÍNCRONA /ASÍNCRONA

Independientemente de si la transmisión es serie o paralelo, ésta puede ser síncrona o asíncrona. Para entender la diferencia es interesante fijarse en la etimología de las palabras.

Ambas vienen del griego cronos --tiempo (reloj)--. Síncrona significa "mismo reloj" y asíncrona lo contrario, es decir, relojes distintos.

Entre dos equipos, emisor y receptor, existe un problema básico en la identificación de los distintos símbolos (bits en este caso) que se transmiten por una línea de transmisión.

Supongamos dos computadores A y B, y una línea de transmisión por la que se comunican.

Supongamos que A manda a B 50 bits a una velocidad de 1000 bits/segundo. Esto quiere decir que cada bit estará en la línea de transmisión una milésima de segundo. La máquina B necesita conocer este dato y necesita un reloj, o base de tiempos, que le permita medir con precisión esa milésima de segundo para saber cuándo ésta en la línea el segundo bit, el tercer bit, etcétera. Se debe conocer que la forma normal en que el equipo receptor decide si un bit es "0" o "1" es muestreando durante el intervalo del bit, preferiblemente a mitad del intervalo.

Es evidente que si el reloj utilizado por el receptor no mide el tiempo con precisión y la secuencia de bits es lo suficientemente larga, entonces cometerá un error en el muestreo de la línea e identificará una secuencia

de bits incorrecta. Si, por ejemplo, el reloj receptor atrasa y cuando indica al sistema que ha pasado 1 mseg. en realidad ha pasado 1,1 mseg. (un error del 10%) entonces se producirá un primer error de muestreo en el 6º a 7º bits transmitido (si se asume que el primer bit lo muestreó correctamente en el centro del intervalo del bits).

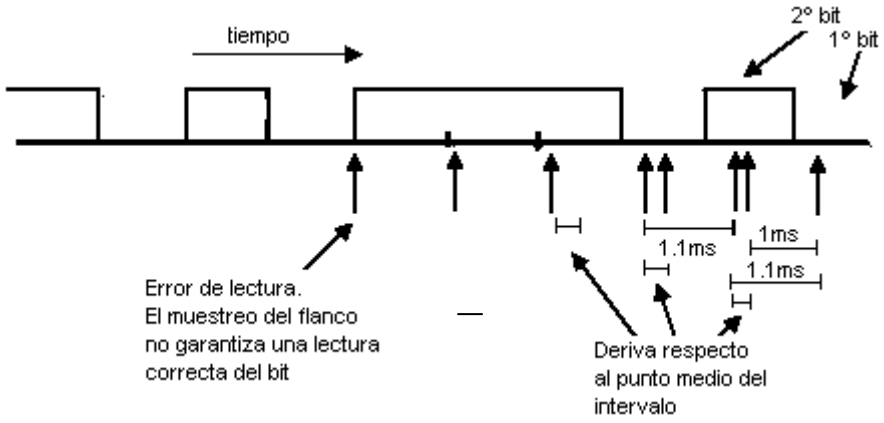


Figura 4.1. Secuencia De Reloj

La figura 4.1 muestra claramente cuál es el problema. Debe quedar claro que aunque el planteamiento del ejemplo hace culpable al reloj del receptor, en una situación real se encuentra que, de usar dos relojes, es imposible garantizar que ambos midan el tiempo exactamente igual. Y aunque el error entre ambos sea mucho menor, si la secuencia de bits es lo suficientemente larga, el error de muestreo terminará por ocurrir.

De esta situación se pueden deducir dos conclusiones:

1.- La transmisión síncrona (emisor y receptor comparten el mismo reloj) permite que el receptor pueda interpretar (muestrear) sin error de sincronísmo una secuencia arbitrariamente larga de bits.

2.- La transmisión asíncrona, a diferencia de la síncrona, está orientada a la transmisión de caracteres (bloque mínimo de tan sólo siete u ocho bits).

4.2. CARACTERÍSTICAS DE LA INTERFASE RS232

La norma serie RS232 fue diseñada para conectar DTEs (equipos terminales de datos) como un terminal, un ordenador, con DCEs (equipos de comunicación de datos), como modems, codecs,AITs, etcétera.

La RS232 permite la transmisión síncrona y asíncrona. La subnorma asíncrona es sin duda la más frecuente.

La transmisión asíncrona se lleva a cabo en concreto además del bit de arranque se utilizan:

- 5, 6, 7 o 8 bits de datos,

- 0 o 1 bit de paridad (la paridad puede ser "par"(Even), impar"(Odd), "siempre a cero"(Reset) y "siempre a uno"(Set).

- 1, 1.5 o 2 bits de parada.

Para agilizar el lenguaje se suele emplear una nomenclatura abreviada como, por ejemplo, "8N1" que indica que la transmisión serie RS232 se ha configurado para transmitir 8 bits de datos, No paridad y 1 bit de STOP. Otro ejemplo sería "6E2" que indica 6 bits de datos, paridad par y 2 bits de STOP.

4.2.1. NORMALIZACIÓN DE LOS ASPECTOS MECÁNICOS.

La RS232 utiliza un conector Cannon DB-25 (ISO 2110) macho para el DTE y hembra para el DCE (ver figura 4.2).

Tabla 4.1. Conector DB 9 y DB 25

PINES Conector DB 9	PINES Conector DB 25	Nombre	Tipo de señal.	Función
	1	P.G.	--	Tierra de seguridad
3	2	TD	Datos	Salida datos DTE
2	3	RD	Datos	Entrada de datos DTE
7	4	RTS	Control	Petición de emisión DTE
8	5	CTS	Control	Listo para trasmitir DCE
6	6	DSR	Control	Listo set de datos

5	7	GND	Tierra	Masa común del circuito
1	8	CD	Control	Detección de portadora
4	20	DTR	Control	Señal de Terminal disponible
9	22	RI	Control	Hay una llamada (sólo modem)
	23	DSRD	<-->	Indicador de velocidad de Tx.

La conexión entre DTE y DCE es simple. Cada pin conecta con su par (el 1 con el 1, el N con el N) existen versiones de DB-25 para cable plano que simplifica el mecanizado de las conexiones. Cada pin tiene asignado una función tal y como se muestra en la tabla 4.1. Los nombres de las líneas están puestos desde el punto de vista del DTE. Así, el pin 2 es la línea TxD (transmisión de datos) pero obviamente eso no es cierto en ambos equipos, sólo en el DTE. En el DCE, por el contrario, es la línea por la que recibe los datos del DTE.

Cuando sólo se utiliza la transmisión asíncrona, sólo es necesario utilizar nueve líneas. Se puede utilizar el conector Cannon-DB-9. Igualmente el macho es el DTE y la hembra el DCE. (Desgraciadamente esta norma, y otras de la RS232, no siempre es seguida por todos los fabricantes, razón por la cual no siempre es fácil manejar esta interface).

La longitud máxima del cable entre DTE y DCE depende de la calidad de éste y de la velocidad de transmisión utilizada. En principio la norma recomienda que no sea superior a 15 metros para una velocidad de 20 Kbits/seg.

4.2.2. NORMALIZACIÓN DE LOS ASPECTOS ELÉCTRICOS.

La subnorma eléctrica de la RS232 es la V28. La norma fija una transmisión en modo común (cada circuito tienen una referencia a tierra y esta es común para todos los circuitos). Los circuitos son punto a punto, es decir, un *driver* con un sólo receptor de la señal.

La señal es bipolar con lógica invertida, utilizando los siguientes valores:

1 lógico =-3 a - 15 voltios

0 lógico = + 3 a + 15 voltios

La ausencia de señal (0 voltios) queda diferenciado del 0 y 1 lógicos.

La RS232 es cortocircuitable. Esto quiere decir que, al menos teóricamente, los *drivers* de salida de las puertas disponen de un mecanismo de auto-protección contra sobrecalentamientos. La tensión máxima de operación es +/-25voltios y la carga máxima es de 3 Kohm. a 7 Kohm., con una corriente máxima de 500 mA.

4.2.3. NORMALIZACIÓN DE LOS ASPECTOS FUNCIONALES.

La norma asíncrona la forman nueve líneas.

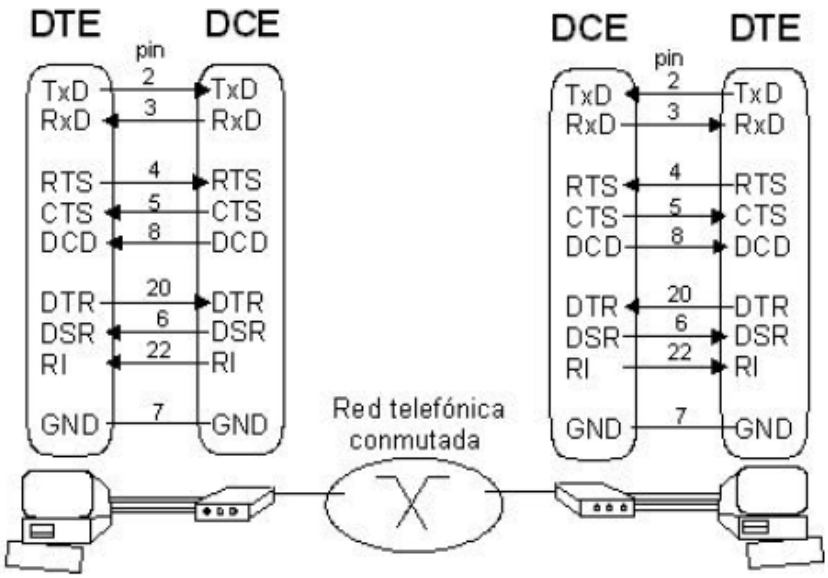


Figura 4.2. Bloques Funcionales De La Rs232

La línea GND conecta la masa de ambos equipos y no merece mayor comentario. Las restantes ocho líneas pueden ser agrupados en tres bloques funcionales:

Primer bloque: Se denomina "de establecimiento de conexión". Está formado por las líneas:

DTR (Data Terminal Ready). Terminal de datos preparado. (El PC y su RS232 están listos).

DSR (Data Set Ready). Equipo de comunicación preparado. (El modem está listo).

RI (Ring Indicator). Indicador de llamada. (El modem indica a su PC que ha recibido una llamada).

El objetivo es que ambos PCs sepan que se ha establecido un canal de comunicación (normalmente a través de la línea telefónica).

Las líneas DTR y DSR del equipo local y del remoto deben estar activas (set) durante todo el proceso. De hecho cuando un PC desea dar por

terminada una conexión basta con que, momentáneamente, desactive (reset) su DTR.

La conexión se inicia manualmente y se gestiona en los modems que negocian, de forma automática, los parámetros de transferencia como la velocidad, compresión, etc.

Se asume que el usuario del PC que llama activará el proceso que va a utilizar la conexión. En el PC llamado, se asume que el proceso homólogo está ya activo o se puede activar automáticamente al recibir de su modem la señal de RI. A partir de este momento los PCs pueden intercambiar información.

Segundo bloque: "Control de flujo".

Estas líneas tienen sentido en el caso de que el canal de comunicación establecido tenga una gestión half-duplex.

Si el canal está establecido, el protocolo software de nivel de enlace de datos que se esté utilizando fijará cuál de los dos DTEs debe comenzar a hablar/transmitir.

Las líneas en este bloque son usadas de la siguiente manera:

RTS (Request To Send). Petición de transmisión. El PC indica a su modem que quiere transmitir a la máquina remota.

CTS (Clear To Send). Canal libre para la transmisión. El modem indica a su PC que puede transmitir. Previamente habrá transmitido una señal portadora por el canal de comunicación para avisar al otro modem que ocupa el canal.

DCD (Data Carrier Detected). Detectada portadora. El modem indica a su PC que el canal de comunicación está ocupado por el equipo remoto.

El PC que quiere transmitir activa RTS, entonces su modem manda una señal portadora (sin modular, sin datos) para avisar al modem remoto

que se reserva el canal. Una vez reservado el canal comunica a su DCE que ya puede transmitir activando la línea CTS.

Cuando un PC haya terminado de transmitir, desactivará RTS, el modem quitará la portadora y desactivará CTS. Entonces el otro modem podrá reservar el canal si su PC desea transmitir.

En caso de que la gestión del canal sea full-duplex todo es más sencillo. Cuando un PC quiere transmitir activa su RTS. Automáticamente su modem le da paso activando CTS.

Tercer Bloque: "Transmisión/recepción de datos".

El funcionamiento de las líneas de este bloque es obvio. Cuando un PC puede transmitir, lo hace por la línea:

TxD. Transmisión de datos y si está recibiendo datos lo hace por RxD. Recepción de datos

4.3. GESTIÓN SIMPLEX, HALF-DUPLEX Y FULL-DUPLEX DE UN CANAL DE COMUNICACIÓN.

Un canal de comunicación puede ser gestionado de tres maneras: simplex, semi-duplex (o half-duplex) y duplex (o full-duplex).

4.3.1. COMUNICACIÓN SIMPLEX

La transmisión simplex (sx) o unidireccional es aquella que ocurre solamente en una dirección, deshabilitando al receptor de responder al transmisor. Normalmente la transmisión simplex no se utiliza donde se requiere interacción humano-máquina. Ejemplos de transmisión simplex son: La radiodifusión de TV y radio, etc.

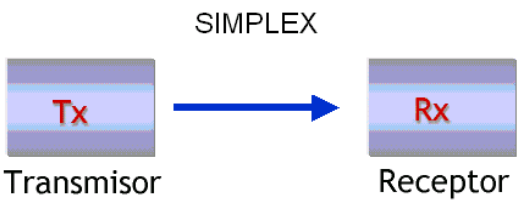


Figura 4.3. Comunicación Siplex

4.3.2. COMUNICACIÓN HALF-DUPLEX

La transmisión half-duplex (hdx) permite transmitir en ambas direcciones; sin embargo, la transmisión puede ocurrir solamente en una dirección a la vez. Tanto transmisor y receptor comparten una sola frecuencia. Un ejemplo típico de half-duplex es el radio de banda civil (CB) donde el operador puede transmitir o recibir, pero no puede realizar ambas funciones simultáneamente por el mismo canal. Cuando el operador ha completado la transmisión, la otra parte debe ser avisada que puede empezar a transmitir.

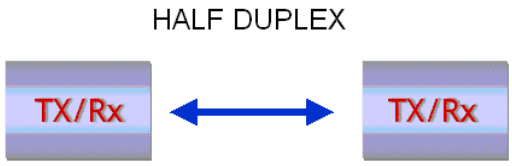


Figura 4.4. Comunicación Half Duplex

4.3.3. COMUNICACIÓN FULL-DUPLEX

La transmisión full-duplex (fdx) permite transmitir en ambas direcciones, pero simultáneamente por el mismo canal. Existen dos frecuencias una para transmitir y otra para recibir. Ejemplos de este tipo abundan en las telecomunicaciones, el caso más típico es la telefonía, donde el transmisor y el receptor se comunican simultáneamente utilizando el mismo canal, pero usando dos frecuencias.

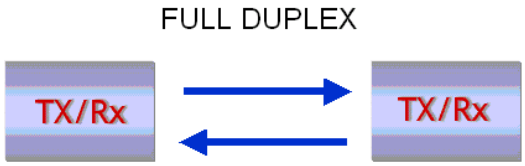


Figura 4.5. Comunicación Full Duplex

4.4. ASPECTOS ESPECIALES DE LA RED RS232.

A continuación se describe los aspectos más relevantes en la comunicación serie, los cuales se debe considerar al utilizar este tipo de interfase en la implementación del diseño.

4.4.1. CONTROL DE FLUJO DE DATOS CON RS232

El control de flujo de datos, es utilizado para prevenir la pérdida de los datos durante la transmisión de la información, con la ayuda de este control se indica si los dispositivos están listos para enviar o recibir datos.

Existen dos posibilidades de control de flujo de datos con la RS232: Una hardware mediante las líneas RTS/CTS y otro software XON/XOFF.

RTS/CTS: la línea CTS indica al PC si puede transmitir o no. En aplicaciones como la conexión de un PC a una impresora serie, la línea CTS está gobernada por la impresora para impedir que el PC desborde su buffer de entrada.

XON/XOFF: Otra posibilidad es usar el protocolo software XON/XOFF que consiste en lo siguiente:

Cuando la impresora está dispuesta para recibir datos (buffer de entrada vacío o casi vacío) transmite al PC la marca XON (XON y XOFF son códigos ASCII predefinidos).

Si el PC transmite demasiado rápido para la impresora y el buffer está próximo a llenarse, entonces se manda la marca XOFF.

El PC transmite sólo si la última marca recibida fue XON.

Dependiendo de las características de los equipos a conectar se puede hacer un control de flujo RTS/CTS, XON/XOFF, ambos o ninguno.

Cuando se utilizan ambos, normalmente es porque hay que controlar dos buffers de recepción, el del dispositivo físico (UART), que se hace

por RTS/CTS, y el buffer de la aplicación que está recibiendo los datos, que se hace con XON/XOFF.

4.4.2. CONEXIONADO DTE-DTE: NULL-MODEM.

Es frecuente que la norma RS232 se utilice para otros propósitos distintos de los originales. Uno de los más frecuentes es conectar un DTE con otro DTE. En este caso el cableado normal DTE a DCE no tiene sentido y se estaría cortocircuitando las líneas de salida. Existe una solución, un cableado cruzado que se conoce como Null-modem.

Es fácil de entender si se analiza usando los tres bloques funcionales.

Bloque 1:

Conectar DTR con DSR remoto y RI remoto.

De esta manera cuando un PC activa su RS232 se lo comunica al remoto.

Bloque 2:

RTS con CTS local y DCD remoto.

Para entender la lógica de este cableado debe observarse la figura 4.4 que hay dos líneas independientes de transmisión de datos, una en cada sentido. Por lo tanto la comunicación es potencialmente Full-duplex. Esto implica que cada DTE puede transmitir cuando lo desee, independientemente de que el otro DTE lo esté haciendo o no. Por lo tanto cuando un PC quiere transmitir activa su RTS, esto activa también su propia CTS lo que le permite transmitir inmediatamente. Además indica que está transmitiendo a la máquina remota activando el DCD remoto.

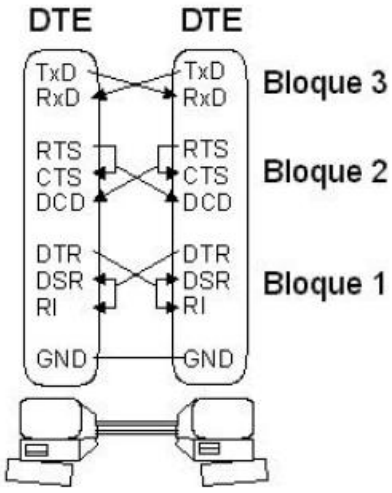


Figura 4.6. Conexionado Dte-Dte: Null-Modem.

Bloque 3:

TxD con RxD remoto.

La comunicación es potencialmente Full-Duplex.

4.4.3. LOS REGISTROS DEL RS232

El sistema operativo MS-DOS soporta hasta cuatro puertos serie denominados COM1 a COM4. A cada uno de ellos se le debe asignar una zona del espacio de direcciones de entrada/salida. Para ello el DOS dispone de una zona de memoria donde guarda las direcciones base de cada uno de estos puertos (si existen en el sistema). Estas direcciones son las que se reflejan en la tabla 4.2.

Tabla 4.2. Dirección de los puertos serie

segmento	offset	puerto
0040	0000	COM1
0040	0002	COM2
0040	0004	COM3
0040	0006	COM4

El almacenamiento en memoria de estas direcciones se realiza en formato byte bajo - byte alto. Así, si las direcciones de los puertos COM1 y COM2 son, respectivamente, 03F8H y 02F8H, la memoria contendrá:

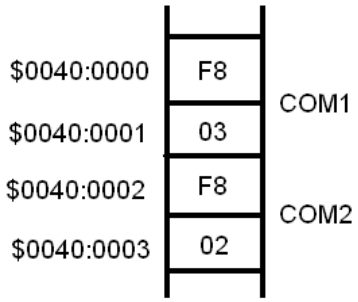


Figura 4.7. Contenido de la memoria

La dirección base del puerto COM1 es 3F8H. Esto quiere decir que a partir de esta dirección se puede acceder a todos los registros del UART asociados a COM1. En concreto, éste ocupará las direcciones desde 3F8H hasta 3FFH, inclusive. Planteando estas direcciones de forma binaria tenemos:

3F8H	0000	0011	1111	1000
3F9H	0000	0011	1111	1001
-				-
-				-
3FEH	0000	0011	1111	1110
3FFH	0000	0011	1111	1111

Figura 4.7. Dirección en forma binaria

4.5. EL PUERTO SERIE: UART 8250

La transmisión de datos en serie es una de las más comunes para aquellas aplicaciones en las que la velocidad no es demasiado importante, o no es posible conseguirla. La línea que transmite los datos en serie está inicialmente en estado alto. Al comenzar la transferencia, se envía un bit a 0 ó **bit de inicio**. Tras él irán los 8 **bits de datos** a transmitir (en ocasiones son 7, 6 ó 5): estos bits están espaciados con un intervalo temporal fijo y preciso, ligado a la velocidad de transmisión que se esté empleando. Tras ellos podría venir o no un bit de paridad generado automáticamente por la UART. Al final, aparecerá un bit (a veces un bit y medio ó dos bits) a 1, que son los bits de parada o **bits de stop**. Para una transmisión en serie básica bastan tres hilos. Sin embargo, el software que controla el puerto serie a través de la interfaz RS-232-C podría requerir más señales de control para establecer la comunicación.

4.5.1. REGISTROS DEL 8250

El 8250 dispone de 11 registros, pero sólo 3 líneas de dirección para seleccionarlos.

Tabla 4.3. Registros del 8250

A2	A1	A0	DLAB	MODO	NOMBRE	SIGNIFICADO
0	0	0	0	R	RBR	Receiver Buffer Register (Registro buffer de recepción)
0	0	0	1	R/W	DLL	Divisor Latch LSB (Divisor de velocidad, parte baja)
0	0	0	0	W	THR	Transmitter Holding Register (Registro de retención de transmisión)
0	0	1	0	R/W	IER	Interrupt Enable Register (Registro de habilitación de interrupciones)
0	0	1	1	R/W	DLM	Divisor latch MSB (Divisor de velocidad, parte alta)
0	1	0	X	R	IIR	Interrupt Identification Register (Registro de identificación de interrupciones)
0	1	1	X	R/W	LCR	Line Control Register (Registro de control de línea)
1	0	0	X	R/W	MCR	Modem Control Register (Registro de control del modem)
1	0	1	X	R/W	LSR	Line Status Register (Registro de estado de la línea)
1	1	0	X	R/W	MSR	Modem Status Register (Registro de estado del modem)

1	1	1	X	R/W	SCR	Scratch Register (Registro residual)

- **LCR (Registrador de control de línea).** Controla el formato del carácter de datos.

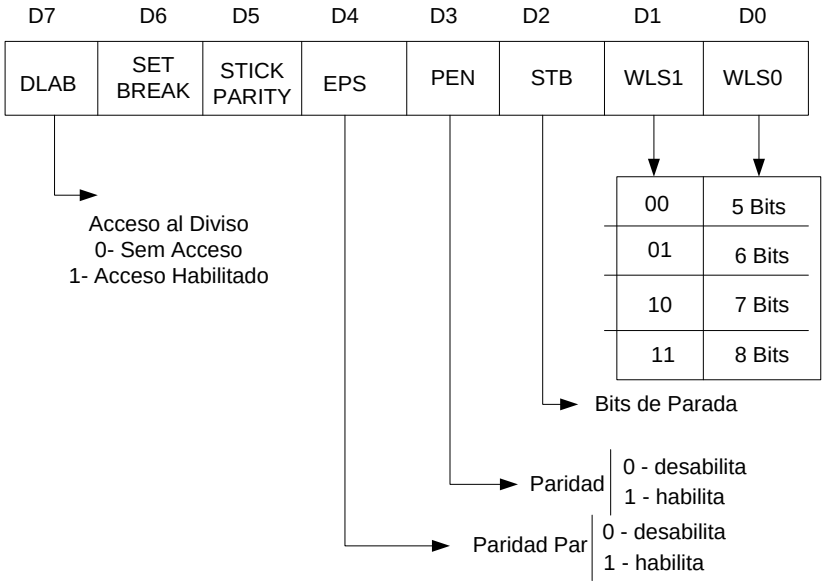


Figura 4.8. Descripción de los registros LCR

Los bits WLS seleccionan el tamaño del dato empleado. STB indica el número de bits de stop, que pueden ser 1 (STB=0) ó 2 (STB=1), al trabajar con datos de 5 bits STB=1 implica 1.5 bits de stop. PEN (Parity Enable) permite habilitar o no la generación de bit de paridad, EPS (Even Parity Select) selecciona paridad par si está a 1 (o impar en caso contrario). Stick Parity permite forzar el bit de paridad a un estado conocido según el valor de EPS. Cuando Break Control es puesto a 1, la salida SOUT se pone en estado espacio (a 0), sólo afecta a SOUT y no a la lógica de transmisión. Esto permite a la CPU alertar a un terminal del sistema sin transmitir caracteres erróneos o extraños si se siguen estas fases: 1) cargar un carácter 0 en respuesta a THRE, 2) activar Break Control en respuesta al próximo THRE, 3) esperar a que el transmisor esté inactivo (TEMT=1) y bajar Break Control. Durante el Break, el transmisor puede usarse como un preciso temporizador de carácter.

El bit DLAB (Divisor Latch Access Bit) puesto a 1 permite acceder a los

Latches divisores DLL y DLM del BRG en lectura y escritura. Para acceder al RBR, THR y al IER debe ser puesto a 0.

- **LSR(Registro de estado de línea):** Este es un registro de 8 bits en el que se encuentra la información correspondiente a la transferencia de datos.

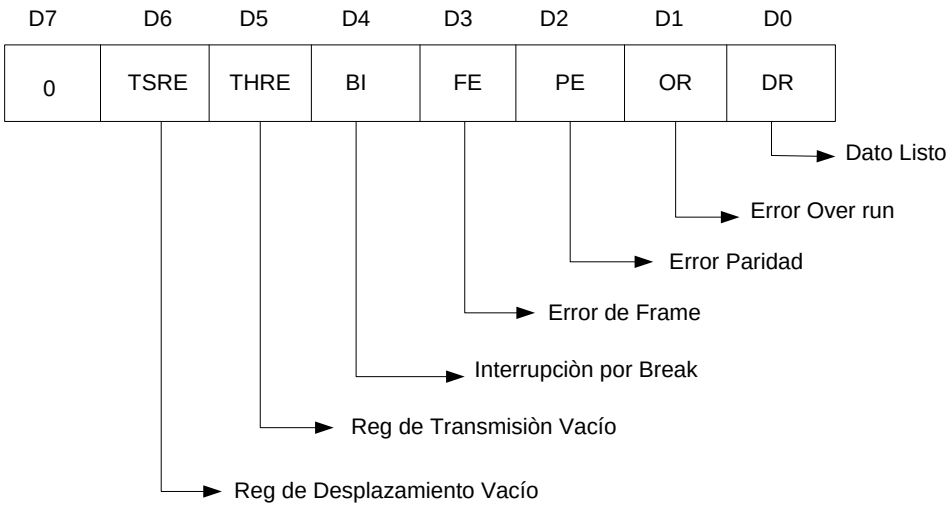


Figura 4.9. Descripción de los registros LSR

DR está activo cuando hay un carácter listo en el RBR y es puesto a 0 cuando se lee el RBR. Los bits 1 al 4 de este registro (OR, PE, FE y BI) son puestos a 0 al consultarlos -cuando se lee el LSR- y al activarse pueden generar una interrupción de prioridad 1 si ésta interrupción está habilitada. OE se activa para indicar que el dato en el RBR no ha sido leído por la CPU y acaba de llegar otro que lo ha sobreescrito. PE indica si hay un error de paridad. FE indica si el carácter recibido no tiene los bit de stop correctos. BI se activa cuando la entrada de datos es mantenida en espacio (a 0) durante un tiempo superior al de transmisión de un carácter (bit de inicio + bits de datos + bit de paridad + bit de parada).

THRE indica que el 8250 puede aceptar un nuevo carácter para la transmisión: este bit se activa cuando el THR queda libre y se desactiva escribiendo un nuevo carácter en el THR. Se puede

producir, si está habilitada; la interrupción THRE. El 8250 emplea un registro interno para ir desplazando los bit y mandar en serie (el Transmitter Shift Register), dicho registro se carga desde el THR. Cuando ambos registros (THR y el Transmitter Shift) están vacíos, TEMT se activa; volverá a desactivarse cuando se deje otro dato en el THR hasta que el último bit salga por SOUT.

- **IIR (Reg. Identificador de interrupciones):** Existen 4 niveles de prioridad en las interrupciones generables por el 8250, por este orden:

    1) Estado de la línea de recepción.
    2) Dato recibido disponible.
    3) Registro de retención de transmisión vacío.
    4) Estado del modem.

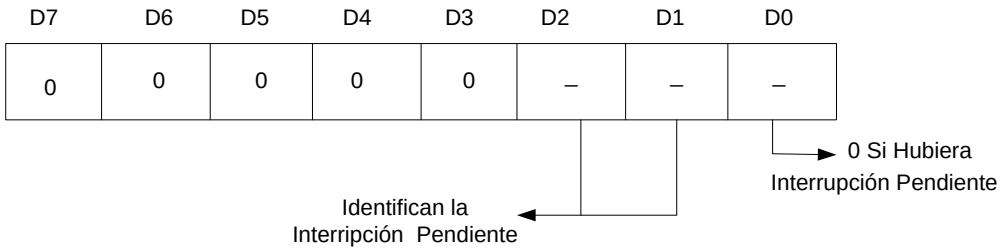


Figura 4.10. Descripción de los registros IIR

La información que indica que hay una interrupción pendiente y el tipo de la misma es almacenada en el IIR. El IIR indica la interrupción de mayor prioridad pendiente. No serán reconocidas otras interrupciones hasta que la CPU envíe la señal de reconocimiento apropiada. En el registro IIR, el bit 0 indica si hay una interrupción pendiente (bit 0=0) o si no la hay (bit 0=1). Los bits 1 y 2 indican el tipo de interrupción. Los restantes están a 0 en el 8250.

- **IER (Reg. Habilitador de interrupción):** Este registro de 8 bits permite individualmente los cuatro tipos de interrupción del 8250. Al

deshabilitar las interrupciones, se inhibe el registro IIR; pero todas las otras funciones operan de manera normal.

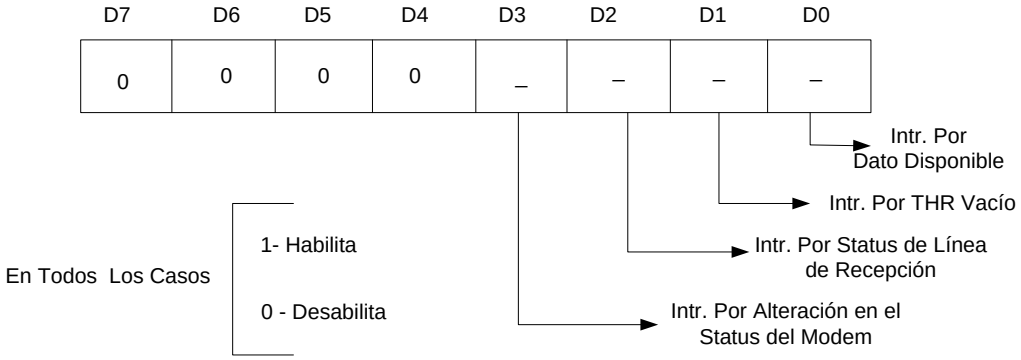


Figura 4.11. Descripción de los registros IER

- **MCR (Reg. De control del módem):** Controla el interface con el modem.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	LOOP	OUT2	OUT1	RTS	DTR

Figura 4.12. Descripción de los registros MCR

Las líneas de salida -DTR, -RTS, -OUT1 y -OUT2 están directamente controladas por estos bits; como se activan a nivel bajo, son puestas a 0 escribiendo un 1 en estos bits y viceversa. Estas líneas sirven para establecer diversos protocolos de comunicaciones.

El bit LOOP introduce el 8250 en un modo lazo (o bucle) de autodiagnóstico. Con LOOP activo, SOUT pasa a estado de marca (a 1) y la entrada SIN es desconectada. Los registros de desplazamiento empleados en la transmisión y la recepción son conectados entre sí. En esta modalidad de operación (modo lazo o bucle), los datos transmitidos son inmediatamente recibidos, lo que permite comprobar el correcto funcionamiento del integrado. Las interrupciones son completamente operativas en este modo, pero la fuente de estas interrupciones son ahora los 4 bits bajos del MCR en lugar de las cuatro entradas de control. Estas interrupciones están aún controladas por el IER.

- **MSR (Reg. De estatus del módem):** Además de la información de estado del modem, los 4 bits bajos (DDCD, TERI, DDSR, DCTS) indican si la línea correspondiente, en los 4 bits superiores, ha cambiado de estado desde la última lectura del MSR; en el caso de TERI sólo indica transiciones bajo-<alto en -RI (y no las de sentido contrario). La línea CTS del modem indica si está listo para recibir datos del 8250 a través de SOUT (en el modo lazo este bit equivale al bit RTS del MCR). La línea DSR del modem indica que está listo para dar datos al 8250 (en el modo lazo -o LOOP- equivale al bit DTR del MCR). RI y DCD indican el estado de ambas líneas (en el modo lazo se corresponden con OUT1 y OUT2 respectivamente). Al leer el MSR, **se borran** los 4 bits inferiores (que en una lectura posterior estarían a 0) pero no los bits de estado (los 4 más significativos).

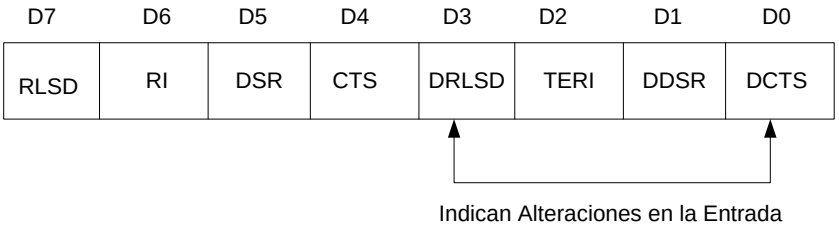


Figura 4.13. Descripción de los registros MSR

CAPÍTULO V

DISEÑO Y CONSTRUCCIÒN

5. DESCRIPCIÓN DEL PROYECTO

En el diseño de la aplicación se debe separar: El trabajo de los microcontroladores y el que realiza LabView, para luego acoplarlos.
Se debe tomar en cuenta las características generales del sistema construido detalladas a continuación:

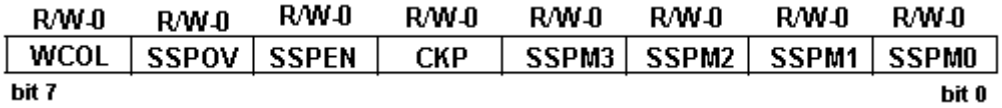
- Velocidad de trabajo en baudios: 9600
- Longitud de los datos en bits: 8
- Dos entradas digitales (0-5V), dos salidas digitales (0-5V).
- Dos entradas analógicas: una de corriente (4 a 20 mA) y otra de voltaje (0 a 5 V).
- Dos salidas analógicas: una de corriente (0 a 20 mA) y otra de voltaje (0 a 5 V).
- Mediante LabView se tiene la Interfase entre la red y el usuario.
- Protocolo de comunicación entre PC y Maestro RS_232.
- Protocolo de comunicación entre Maestro y Esclavo I2C.

Todos los aspectos a considerar con respecto a la interfase RS_232, LabView, microcontroladores y la red I2C, utilizada para este proyecto están detallados en los capítulos anteriores, faltando por detallar los registros del microcontrolador utilizados para la comunicación I2C.

5.1. REGISTROS DEL MICROCONTROLADOR PARA LA RED I2C.

- **MAESTRO:**

- **SSPCON1:** Registro de control 1 para modo I2C:
  "00101000"



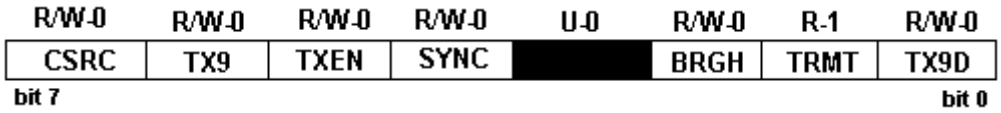
bit 3-0 SSPM3:SSPM0: Selección de los bits para el modo puerto serial sincrónico

1000= Modo maestro de la I2C. .

bit 5 SSPEN: Habilita el puerto serial sincrónico, en donde 1 habilita el puerto serie y configura las líneas SDA y SCL fijándolas en los pines del puerto.

Bit 4,6,7 = bits no habilitados.

- **TXSTA:** Registro de control y estado de transmisión USART: "00100100".



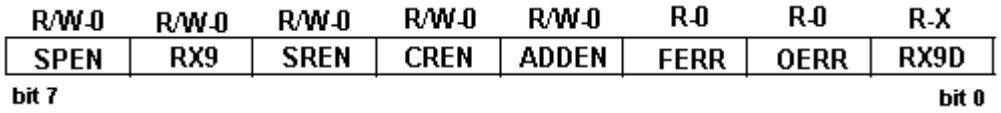
bit 2 BRGH: Permite Seleccionar el bit para alta velocidad en baudios en el modo asíncrono , en donde 1 se utiliza para habilitar alta velocidad de transmisión de datos.

bit 4 SYNC: Bit de selección para el modo USART, en donde 0 habilita el modo asíncrono.

bit 5 TXEN: Permite realizar la habilitación para la transmisión de datos, se coloca en uno para habilitar la transmisión de datos .

- **RCSTA:** Registro de control y estado de recepción USART:

“10010000”

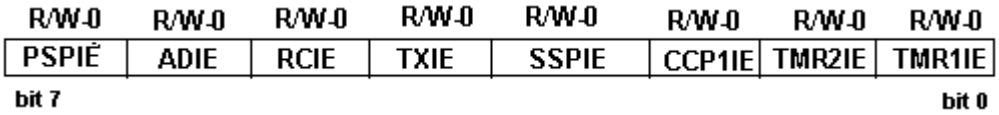


bit 4 CREN: Permite habilitar el bit para la recepción continua en modo asíncrono, con uno se habilita el receptor.

bit 7 SPEN: Mediante este bit se habilita el puerto serie, se coloca en uno para habilita el puerto serie .

- **PIE1:** Habilitación de interrupciones de periféricos:

  : “00100000”.



bit 0 TMR1IE: Permite la habilitación del bit para la interrupción por desbordamiento del TMR1 en donde cero deshabilita la interrupción por desbordamiento del TMR1.

bit 2 CCP1IE: Permite habilitar el bit de interrupción CCP1, se coloca en cero para deshabilitar la interrupción CCP1.

bit 3 SSPIE: Habilita el bit de interrupción del puerto serial sincrónico del maestro, se coloca en cero para deshabilitar la interrupción del MSSP.

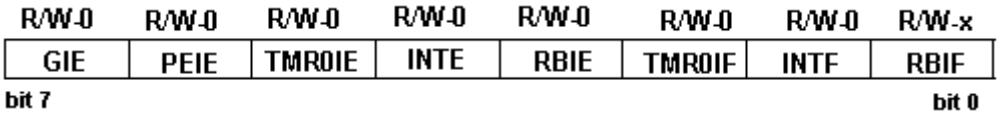
bit 4 TXIE: Permite habilitar el bit de interrupción de la transmisión USART en cero deshabilita la interrupción de la transmisión USART.

bit 5 RCIE: Permite habilitar el bit de interrupción para la recepción USART, se coloca en uno para habilitar la interrupción de recepción USART.

bit 6 ADIE: Habilita el bit de interrupción del conversor A/D, se coloca en cero para deshabilita la interrupción A/D .

- **INTCON:** Habilitación de interrupciones:

  : “11000000”



bit 5 TMR0IE: Permite habilitar el bit para la interrupción por desbordamiento del TMR0, se coloca en cero para deshabilitar la interrupción por desbordamiento del TMR0.

bit 6 PEIE/GIEL: Permite habilitar el bit para la interrupción de periféricos, en uno habilita todas las interrupciones periféricas

bit 7 GIE/GIEH: Mediante este podemos habilitar el bit de Interrupción global, se coloca en uno para habilitar todas las interrupciones

5.1.2. ESCLAVO:

- **SSPCON1:** Registro de control 1 para modo I2C:
  "00110110"

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 3-0 SSPM3:SSPM0: Permite la selección de los bits para el modo puerto serial sincrónico.

0110 = Modo esclavo I2C

bit 4 CKP: Controla el bit de liberación del SCK en el modo Esclavo, con uno se libera al reloj.

bit 5 SSPEN: Permite habilitar el puerto serial sincrónico, con uno se habilita el puerto serie y configura las líneas SDA y SCL fijándolas en los pines del puerto.

- **CCP1CON/CCP2CON:** Registros de control de los módulos CCP
  "00001100"

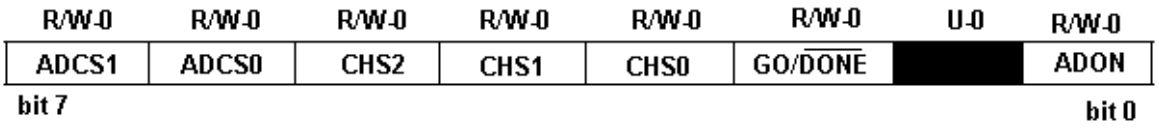
U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
		DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 3-0 CCPxM3:CCPxM0: selección de los bits para el modo CCPx

11xx = modo PWM

- **ADCON0:** Registro de configuración 0 del conversor ADC

  : “11000001”



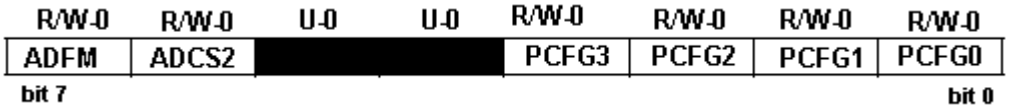
bit 0 ADON: bit de activación del conversor A/D con uno se activa el conversor A/D.

bit 5-3 CHS2:CHS0: Permite la selección del bit para el canal analógico, en donde la cadena 000 habilita el canal 0.

bit 7-6 ADCS1:ADCS0: Permite la selección del bit para el reloj del conversor A/D, donde la cadena 11 habilita la Frc (procede del oscilador RC interno).

- **ADCON1:** Registro de configuración 1 del conversor ADC:

“11000100”



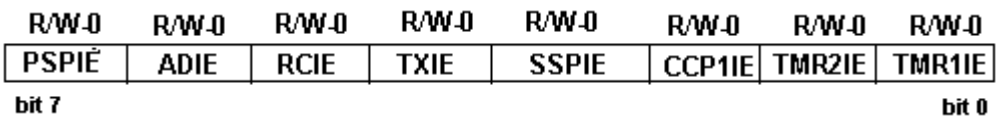
bit 3-0 PCFG3:PCFG0: Bit de control para la configuración del puerto como A/D , se lo habilita con la cadena 0100.

bit 6 ADCS2: Permite seleccionar el bit para el reloj del conversor A/D, con uno se habilita la Frc (procede del oscilador RC interno).

bit 7 ADFM: Selección del bit para el formato de resultado A/D, en donde uno permite justificar a la derecha.

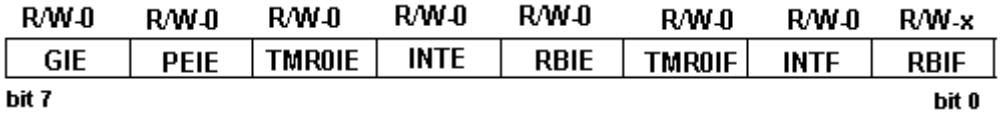
- **PIE1:** Habilitación de interrupciones de periféricos:

  : “00100000”.



bit 5 RCIE: Permite habilitar el bit de interrupción para la recepción lUSART, con uno se habilita la interrupción para recepción USART.

- **INTCON**: Habilitación de interrupciones:
  : “11000000”



bit 5 TMR0IE: Habilitación del bit para la interrupción por desbordamiento del TMR0, con cero se deshabilita la interrupción por desbordamiento del TMR0.

bit 6 PEIE/GIEL: Habilita el bit para la interrupción de periféricos, en donde uno habilita todas las interrupciones periféricas .

bit 7 GIE/GIEH: Permite la habilitación del bit de Interrupción global, en uno se habilita todas las interrupciones.

5.2. HADWARE UTILIZADO PARA CREAR LA RED I2C

La red I2C posee básicamente los siguientes elementos:

- 3 Microcontroladores
- 1 Transceirver MAX232
- 4 Amplificadores Operacionales LM358N
- 3 Osciladores de 10 MHz
- Condensadores 10μF, 33μF, 15nF
- Diodos
- Resistencias

El diagrama esquemático se observa en la figura 5.3 y 5.4, en el Anexo A se encuentra el esquema de las placas diseñadas.

- **ESCLAVO**

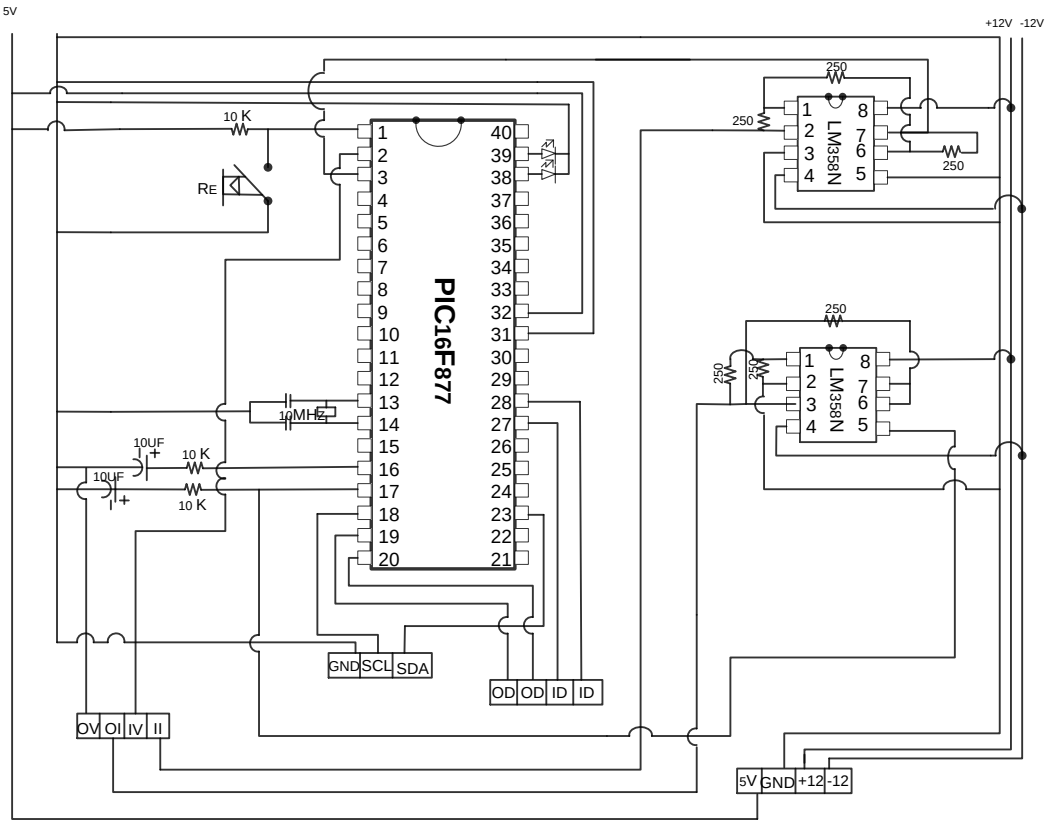


Figura 5.3. Diagrama esquemático para el esclavo del bus I2C.

En este diagrama se muestra al PIC 16F877A configurado como esclavo, con las conexiones y elementos necesarios para su funcionamiento.

En el circuito se utiliza un capacitor de 10μF en serie con una resistencia de 10K conectados al Pin, se usa la misma configuración de elementos para la salida del Pin 17, con el objetivo de filtrar la señal PWM y obtener una señal de voltaje continua. Se usa dos circuitos integrados LM358N, el primero ubicado en la parte superior del circuito, permite la conversión de la

señal de salida del PIC de voltaje a corriente, el segundo ubicado en la parte inferior del circuito, se utiliza para la conversión de la señal de entrada de corriente a voltaje que ingresará al PIC. Por último se tiene un Switch RE conectado al Pin1 del microcontrolador que sirve para resetear al sistema en caso de ser necesario.

Tanto las salidas como entradas de señales utilizadas por el microcontrolador, así como los voltajes de polarización, se muestran en el circuito mediante cuadriculas.

- **MAESTRO**

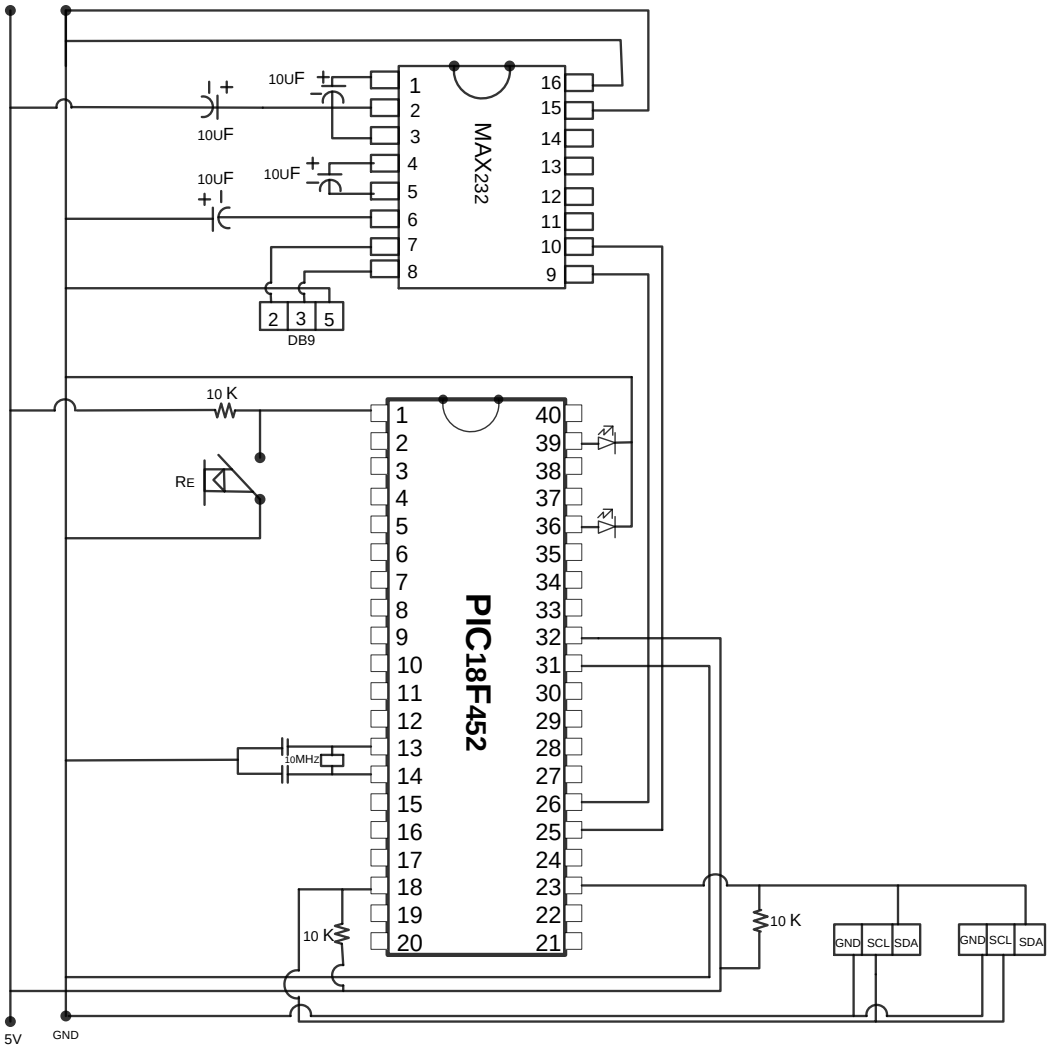


Fig Figura 5.4. Diagrama esquemático para el maestro del bus I2C.

En el diagrama de la figura 5.4. se emplean dos circuitos integrados, un PIC 18F452 utilizado como maestro en el sistema en el cual se encuentra conectado dos resistencias de 10K una desde el Pin 18 a 5V y otra de

igual valor desde el Pin 23 a 5V con el fin de asegurar la correcta transmisión de los datos y el reloj de la red I2C, de igual manera que en el circuito anterior se conecta un switch RE al Pin 1 que permite realizar el reset del sistema en caso de ser necesario.

Se utiliza un MAX 232 para la comunicación serie entre el micro controlador y la PC este integrado hace las conversiones de TTL a RS-232 y de RS-232 a TTL

5.3. SOFTWARE UTILIZADO PARA LA CREACIÓN DE LA RED I2C

Para el desarrollo del software de la red I2C se creó básicamente dos programas, uno para el Máster y otro para los Esclavos, estos se basan en Interrupciones.

5.3.1 Programa del Maestro:

En este se realizaron todas las configuraciones iniciales necesarias para el funcionamiento de la red las mismas que se detallaron anteriormente, además desde aquí se envían las funciones de escritura o lectura de los datos a los esclavos siempre que se cumplan las condiciones para ello.

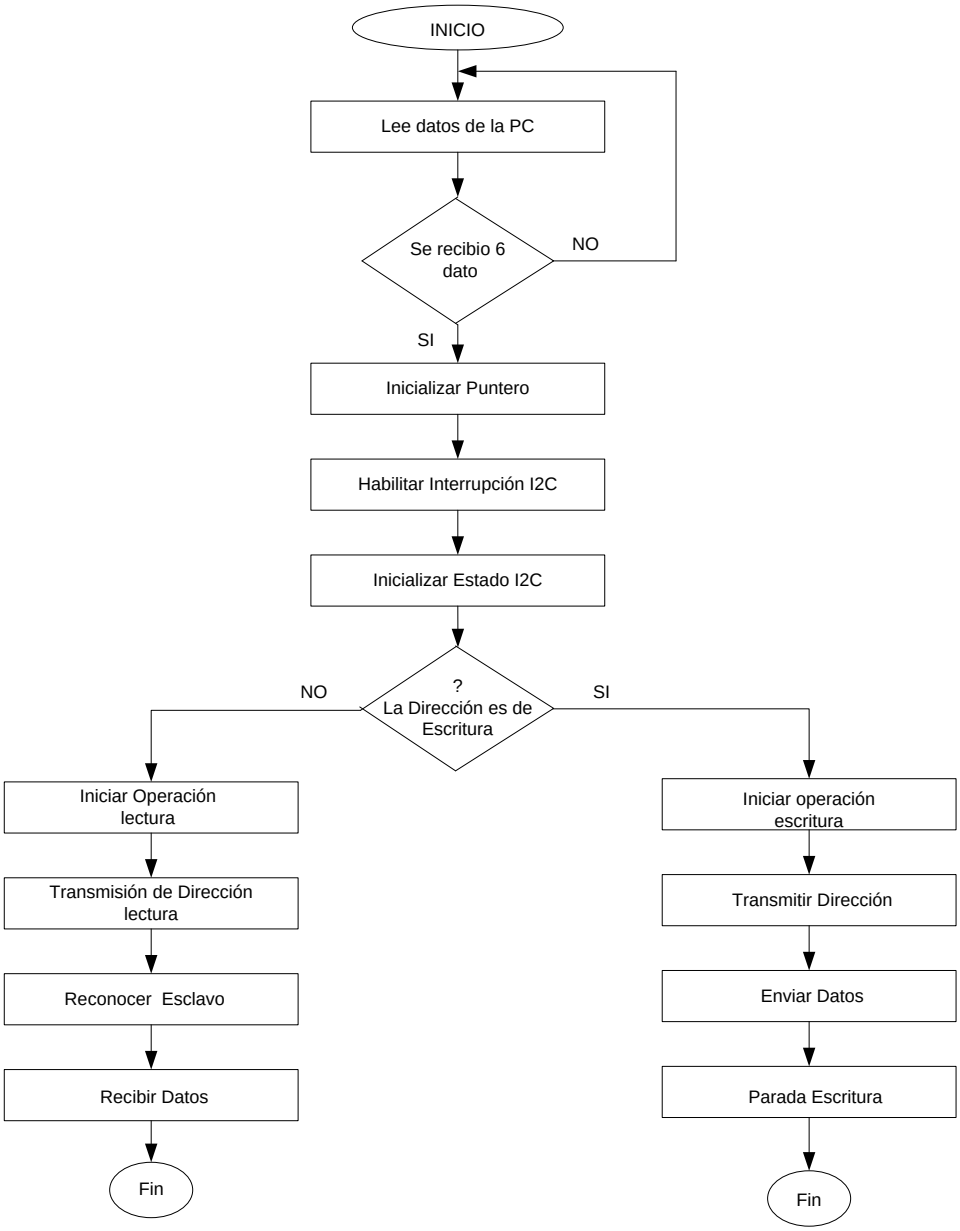


Figura 5.5. Diagrama de flujo principal del maestro del bus I2C.

En este diagrama de flujo se muestra el programa base o fundamental del maestro, en el que se realiza llamadas a funciones que a continuación se detallan.

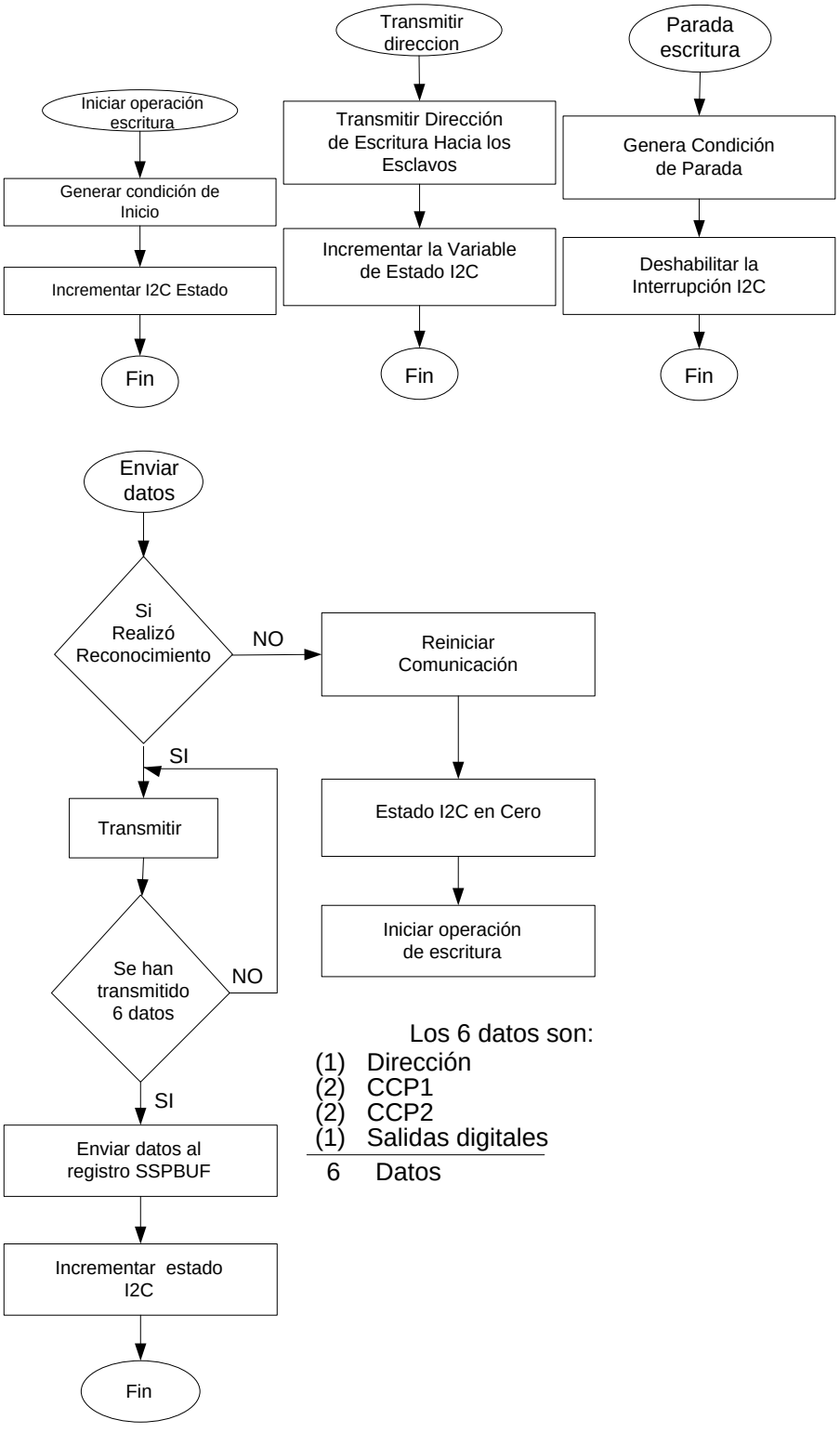
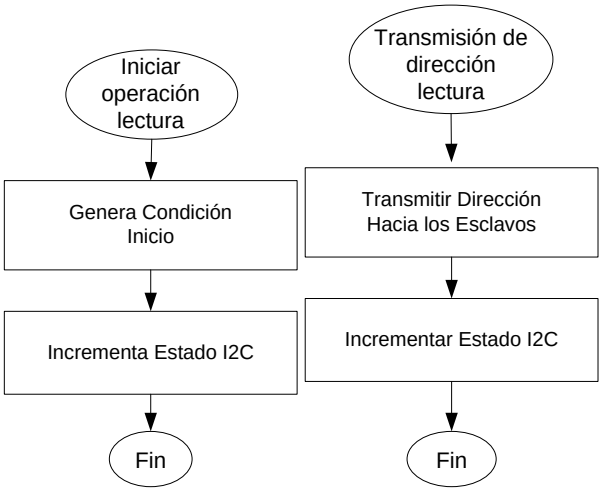


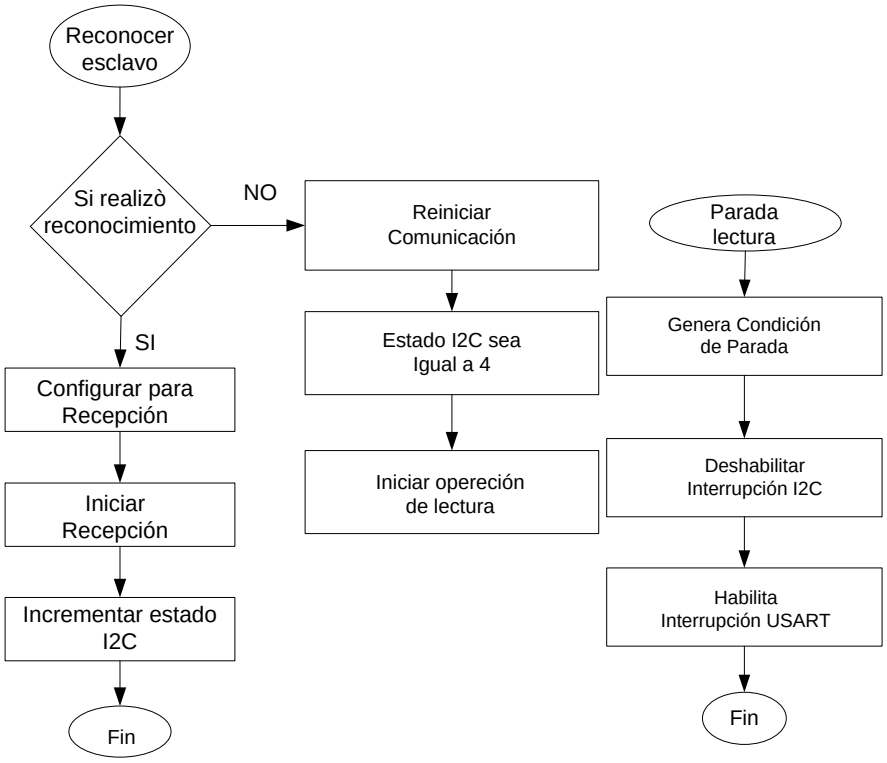
Figura 5.6. Diagramas de flujo para la operación de escritura del maestro del bus I2C.

En este conjunto de diagramas de Flujo se muestra las operaciones de escritura del maestro, por cuestión de manejabilidad se ha divido la problemática de la operación de escritura del maestro en las

siguientes subrutinas, cada una representada con un diagrama de Flujo:

- ✓ **Inicio de Operación de Escritura:** En este diagrama se realiza una condición de inicio para empezar con la operación de escritura del Maestro.
- ✓ **Transmitir dirección**: En este diagrama se realiza la Trasmisión de la dirección de escritura hacia los esclavos.
- ✓ **Enviar datos**: En este Diagrama se realiza en primer lugar verificaciones para asegurar que la trasmisión sea correcta, y por último se realiza la trasmisión de los datos.
- ✓ **Parada de Escritura**: En este diagrama se realiza una condición de parada que indica que la trasmisión ha concluido.




Reconocer esclavo
Si realizò reconocimiento
NO
SI
Reiniciar Comunicación
Estado I2C sea Igual a 4
Iniciar opereción de lectura
Configurar para Recepción
Iniciar Recepción
Incrementar estado I2C
Fin
Parada lectura
Genera Condición de Parada
Deshabilitar Interrupción I2C
Habilita Interrupción USART
Fin

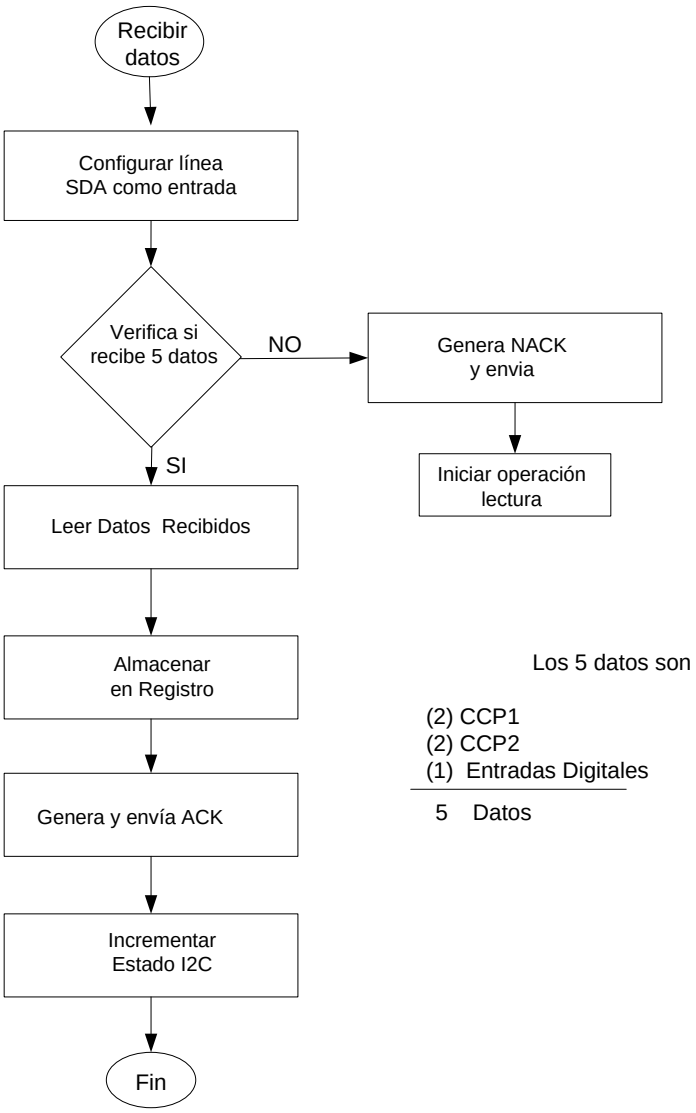


Figura 5.7. Diagramas de flujo para la operación de lectura del maestro del bus I2C.

En este conjunto de diagramas de Flujo se muestra las operaciones de lectura del maestro, por cuestión de manejabilidad se ha divido la problemática de la operación de lectura del maestro en las siguientes subrutinas, cada una representada con un diagrama de Flujo:

- ✓ **Inicio de Operación Lectura:** En este diagrama se realiza una condición de inicio para empezar con la operación de lectura del Maestro.
- ✓ **Reconocer Esclavo:** En este diagrama se ejecuta una validación para asegurar el reconocimiento del esclavo.

- ✓ **Recibir Datos**: En este Diagrama se realiza verificaciones para asegurar que reciban todos los datos enviados desde el esclavo, y luego se realiza configuraciones necesarias para ejecutar la operación de lectura.

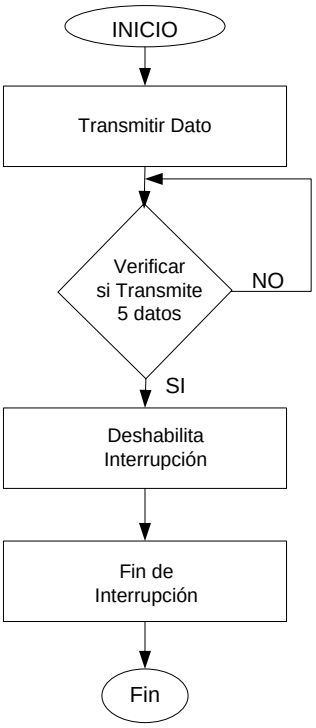
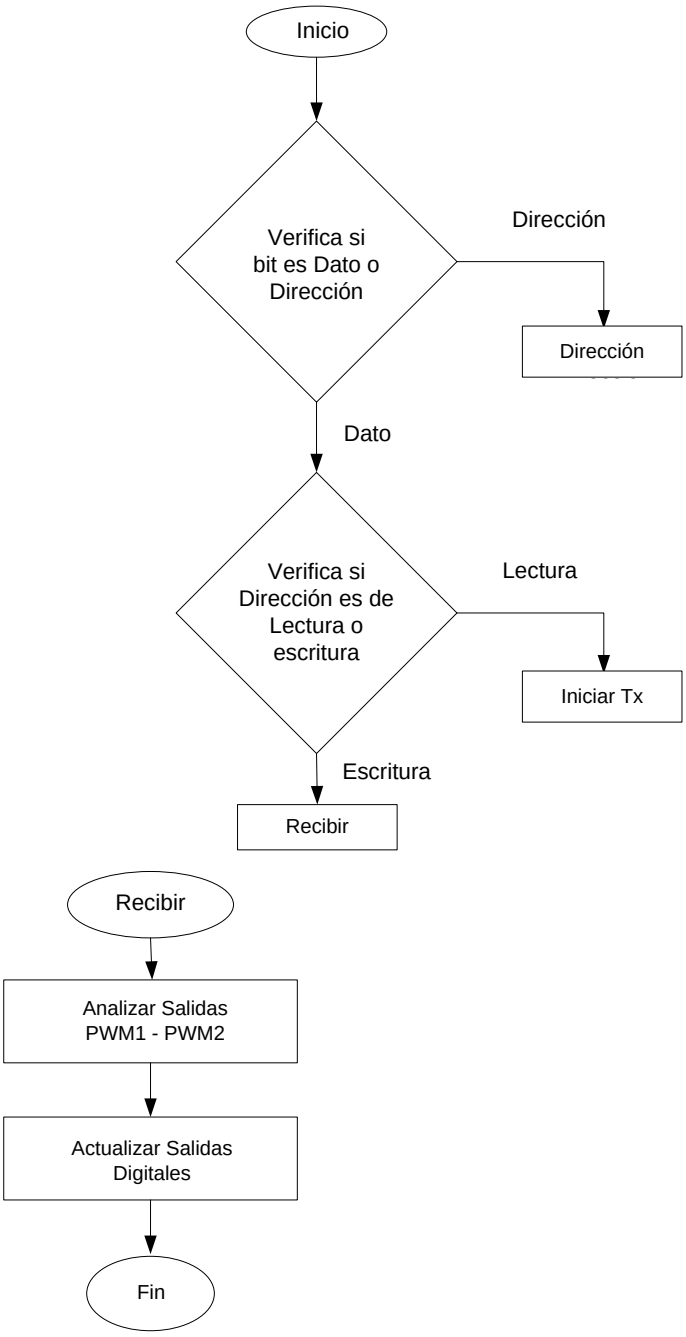


Figura 5.8. Diagrama de Flujo de la Interrupción par Transmisión Serie del Bus I2C

En este diagrama se indica los pasos a realizarse para la transmisión de datos desde el maestro al Pc.

5.3.2. PROGRAMA DEL ESCLAVO:

Al igual que en el maestro se realizaron todas las configuraciones iniciales necesarias para el funcionamiento de la red las mismas que se detallaron anteriormente.


Inicio
Verifica si bit es Dato o Dirección
Dirección
Dirección
Dato
Verifica si Dirección es de Lectura o escritura
Lectura
Iniciar Tx
Escritura
Recibir
Recibir
Analizar Salidas PWM1 - PWM2
Actualizar Salidas Digitales
Fin

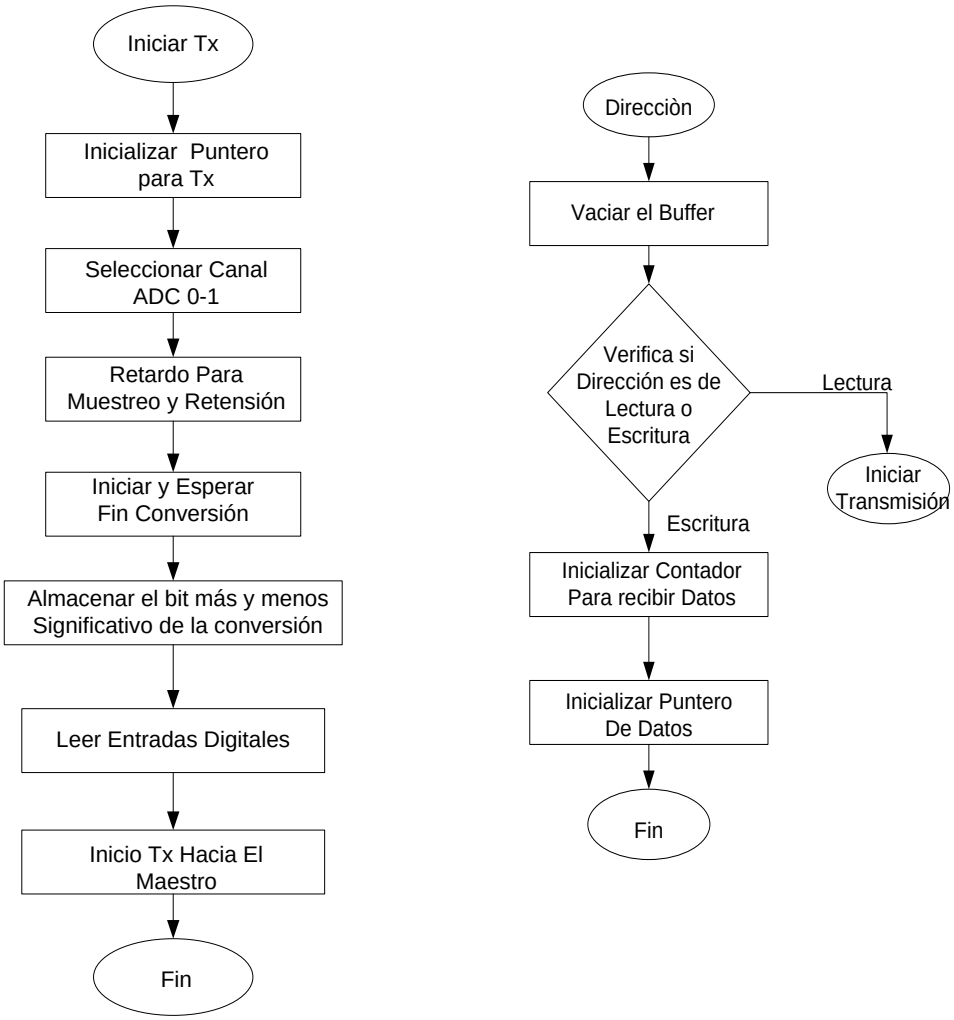


Figura 5.9. Diagrama de Flujo del esclavo del Bus I2C

En este conjunto de diagramas de Flujo se muestra las operaciones tanto de lectura como de escritura realizadas por el esclavo, se usa las siguientes subrutinas, cada una representada con un diagrama de Flujo:

- ✓ **Inicio:** En este diagrama se realiza todas las llamadas a funciones, siempre y cuando las condiciones se cumplan.
- ✓ **Dirección:** En este diagrama se ejecuta una validación para determinar si la dirección es de lectura o escritura.
- ✓ **Inicio de Transmisión**: En este Diagrama se realiza la operación de transmisión de datos hacia el maestro
- ✓ **Recibir**: En este diagrama se realiza la operación de recepción de datos desde el maestro.

Para cada uno de estos puntos se desarrollaron subrutinas independientes, con el objetivo de brindar una mayor facilidad en el mantenimiento y depuración del programa final.

En el anexo B se muestra el programa de los microcontroladores para el maestro y esclavo.

5.3.3. VENTANA PRINCIPAL:

En esta ventana que se implementó en el PC, se indica tanto los controles como los visualizadores a utilizarse en la red I2C:

- 2 Controles de salida de Voltaje
- 2 Controles de salida de Corriente
- 2 Visualizadores de entrada de voltaje
- 2 Visualizadores de entrada de corriente
- 4 Controles para salida digital
- 4 Visualizadores de entradas digitales
- Una caja de texto de aviso de error

El diseño de esta pantalla se muestra el la figura 5.8.

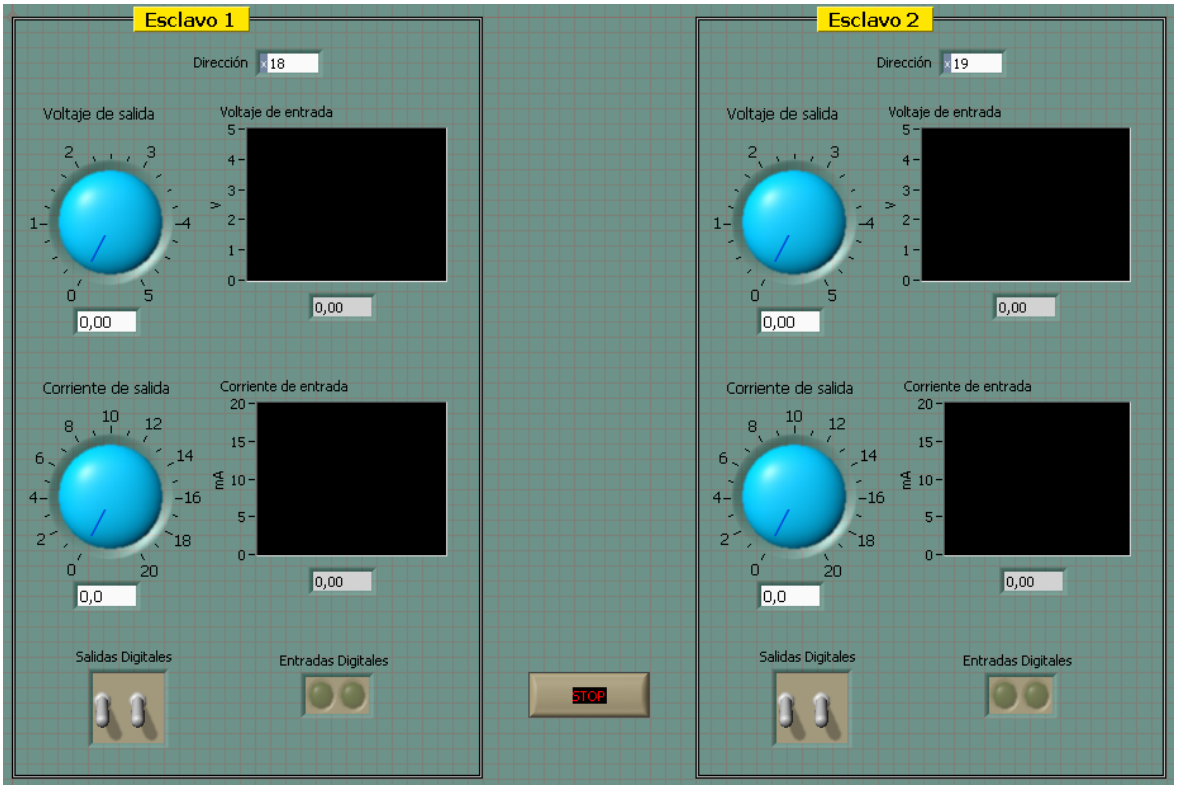


Figura 5.8. Panel de control.

En esta figura se muestra los controles y visualizadores de la red I2C, a la izquierda para el esclavo con dirección 18 y a la derecha para el esclavo con dirección 19.

5.3.4. SOFTWARE DE COMUNICACIÓN DE LabView

Para la comunicación entre la red I2C y el usuario mediante el uso de LabView se requiere de los siguientes pasos, previo a lo cual en el capítulo III se explica con detalle las propiedades de cada icono a emplearse:

- **CONFIGURACIÓN DEL MÓDULO USART PARA LA COMUNICACIÓN SERIE.**

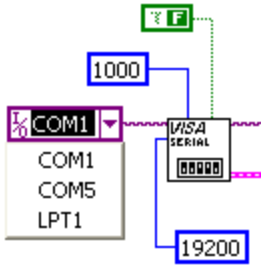


Figura 5.9. Programación del modulo USART.

En la figura se muestra la programación en LabView para configurar el modulo Usarte, con el fin de establecer los parámetros de la comunicación serie.

- **PROGRAMACIÓN DE LA DIRECCIÓN Y ENVÍO DE DATOS HACIA EL PRIMER ESCLAVO.**

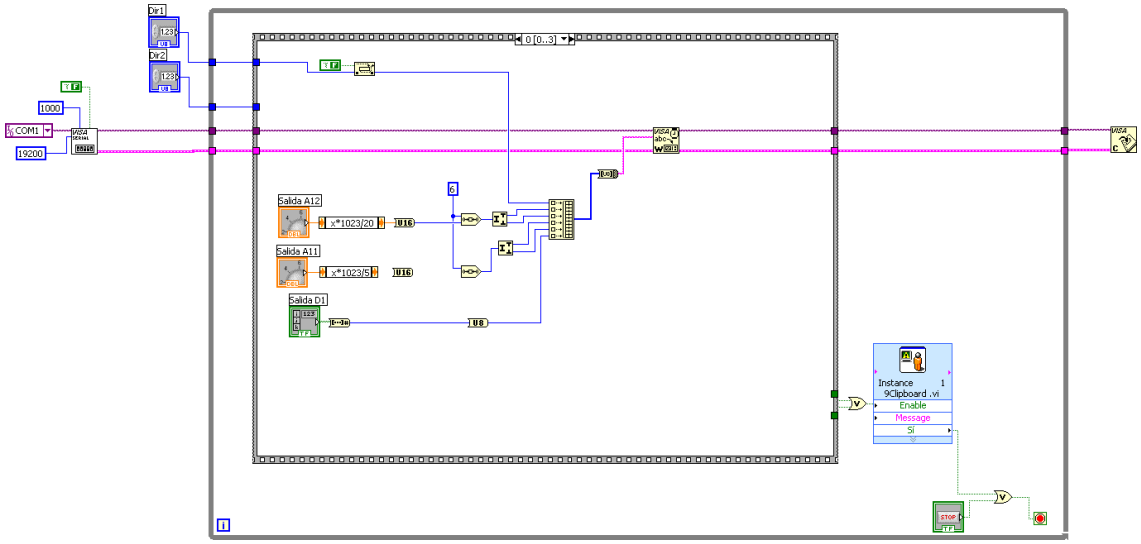


Figura 5.10. Programación de la dirección y envió de datos hacia el primer esclavo.

En la figura se muestra el primer Case Structure dentro de un While Loop, que se emplea para programar el envío de datos hacia el primer esclavo.

- **PROGRAMACIÓN DE LA DIRECCIÓN Y ENVÍO DE DATOS HACIA EL SEGUNDO ESCLAVO.**

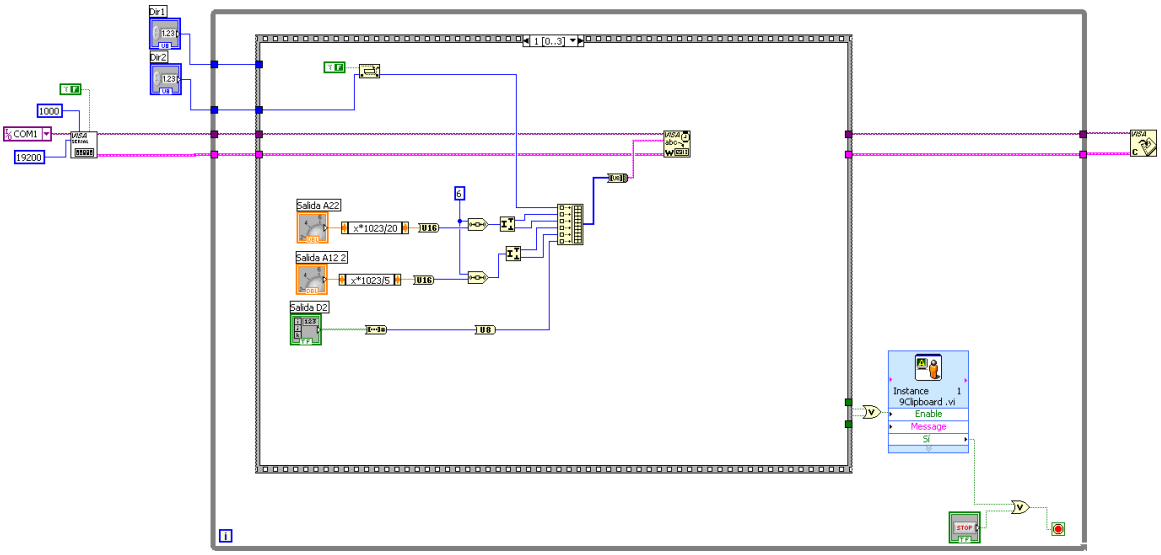


Figura 5.11. Programación de la dirección y envió de datos hacia el segundo esclavo.

En la figura se muestra el segundo Case Structure dentro de un While Loop, que se emplea para programar el envío de datos hacia el segundo esclavo.

- **PROGRAMACIÓN PARA LA RECEPCIÓN DE DATOS DESDE EL PRIMER ESCLAVO.**

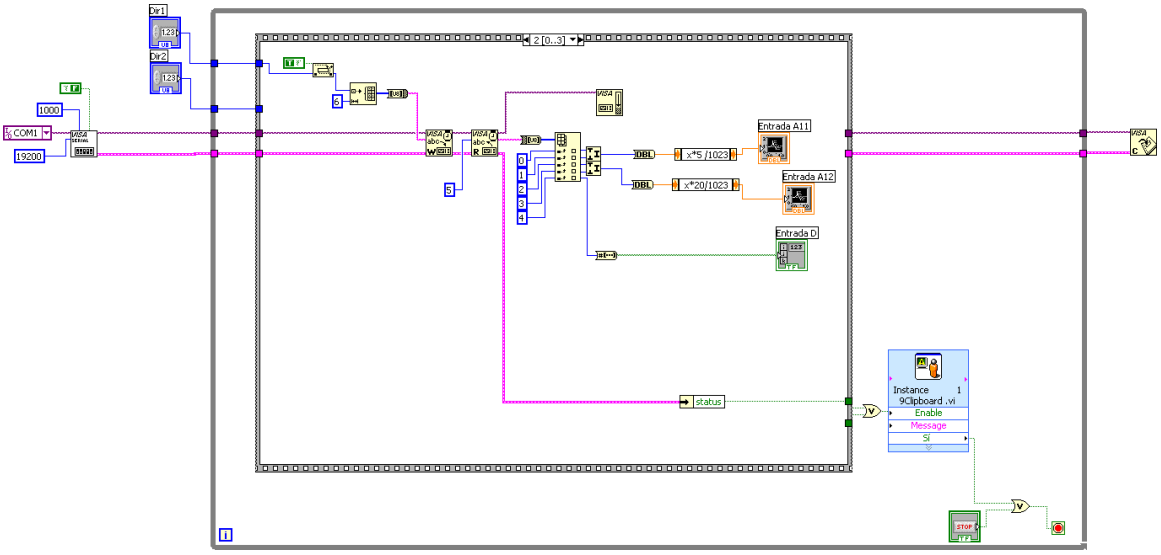


Figura 5.12. Programación para la recepción de datos desde el primer esclavo.

En la figura se muestra el tercer Case Structure dentro de un While Loop, que se utiliza para programar la recepción de los datos provenientes del primer esclavo.

- **PROGRAMACIÓN PARA LA RECEPCIÓN DE DATOS DESDE EL SEGUNDO ESCLAVO.**

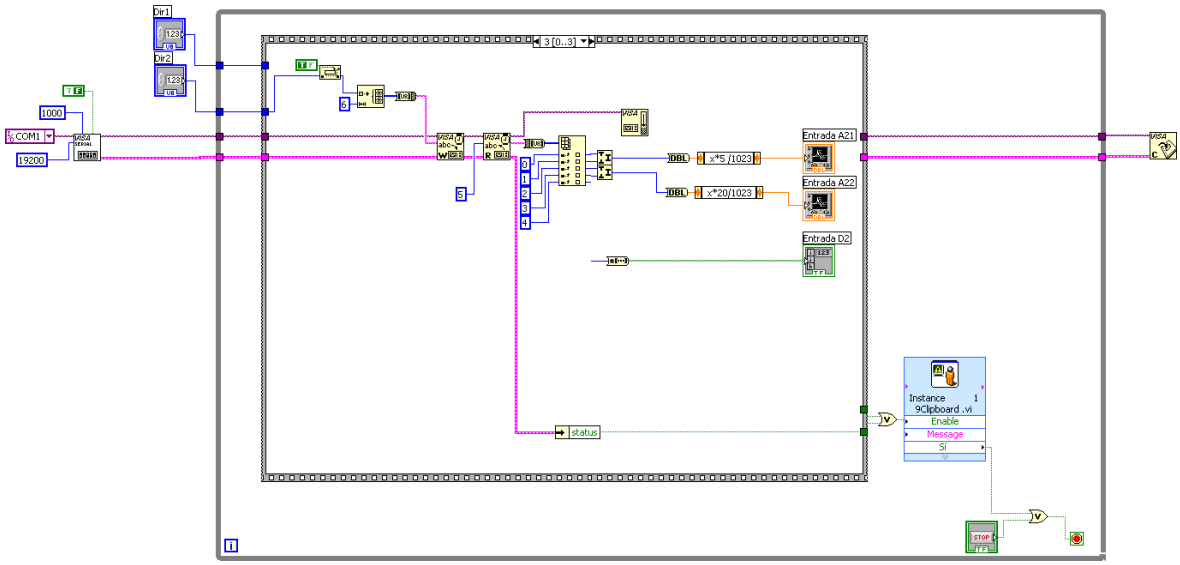


Figura 5.13. Programación para la recepción de datos desde el segundo esclavo.

En la figura se muestra el cuarto Case Structure dentro de un While Loop, que se utiliza para programar la recepción de los datos provenientes del segundo esclavo

CAPÍTULO VI

6. ANÁLISIS DE RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

6.1. ANÁLISIS DE RESULTADOS

Para el funcionamiento de la aplicación se requiere conectar la fuente de corriente continua al circuito (ANEXO A), el puerto serie de la computadora se conecta al circuito mediante el cable de tres hilos (TX, RX, GND) que fue construido para la aplicación y las señales a adquirir tanto analógicas como digitales. En cuanto al software, la aplicación debe ser instalada en labView 7.0 siguiendo los pasos indicados en el (ANEXO C).

Para el análisis del funcionamiento del diseño se considera los siguientes puntos:

- Parámetros de comunicación.
- Adquisición de señales Analógicas y Digitales.
- Salidas de señales Analógicas y Digitales.

6.1.1. PARÁMETROS DE COMUNICACIÓN

La velocidad de la aplicación está en función del valor especificado en el Microcontrolador, y no por LabView, debido a que este último tiene una capacidad de velocidad para la transmisión y recepción de datos muy elevada.

Las pruebas de funcionamiento del sistema se realizó para la velocidad más alta permitida de transmisión y recepción de datos entre la PC y el maestro que es de 19200 baudios, esto en cuanto a la transmisión serie.

En el bus I2C la velocidad de comunicación en que opera la red es de 100KHz.

6.1.2. ADQUISICIÓN DE SEÑALES ANALÓGICAS Y DIGITALES

Para el análisis de la adquisición de señales analógicas tanto de corriente como de voltaje de diferentes valores, se utilizó un multímetro digital, y la aplicación de la red I2C, al comparar estas mediciones se obtuvo resultados muy favorables, según se demuestra en la tabla 6.1.
Este sistema se armó colocando en la entrada analógica, de uno de los esclavos una señal de voltaje. Y en el otro esclavo una señal de corriente que se obtuvo a partir de una fuente, estos datos fueron enviados desde los esclavos empleando la red I2C hacia el maestro para luego ser transmitidos a la PC. En la figura 6.2. se muestra el diagrama de bloques para este circuito.
En la tabla 6.1. Se indican los valores obtenidos de la medición del multÍmetro y de la red I2C. En la figura 6.1. se muestran la formas de ondas de una señal triangular, una cuadrada, dos señal de corriente y se visualizan dos señales digitales que son activadas desde el exterior.

Tabla 6.1. Ejemplo de la adquisición de señales analógicas.

		MULTÍMETRO	RED I2C	ERROR
Voltaje	2 V	2 V	1.96 V	2%
	5 V	5 V	4.97 V	0.6%
Corriente	4 mA	4 mA	4.11 mA	-0.2%
	6mA	6 mA	6.06 mA	-0.1%

En la tabla 6.1. se indica los valores medidos con un multímetro de las señales que ingresan a los esclavos de la red. En la columna con nombre RED I2C se muestra el valor de las mediciones obtenidas en los visualizadores del panel principal, empleado para la red I2C.

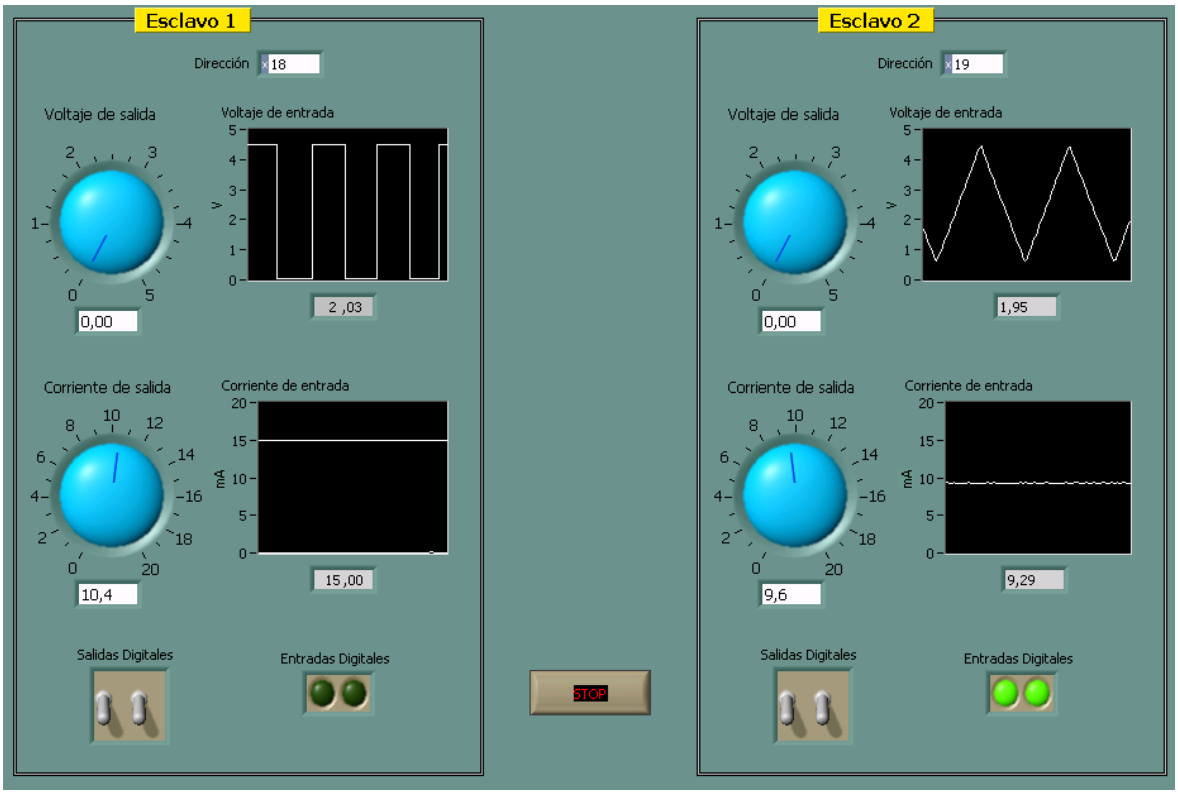


Figura 6.1. Adquisición de señales analógicas y digitales.

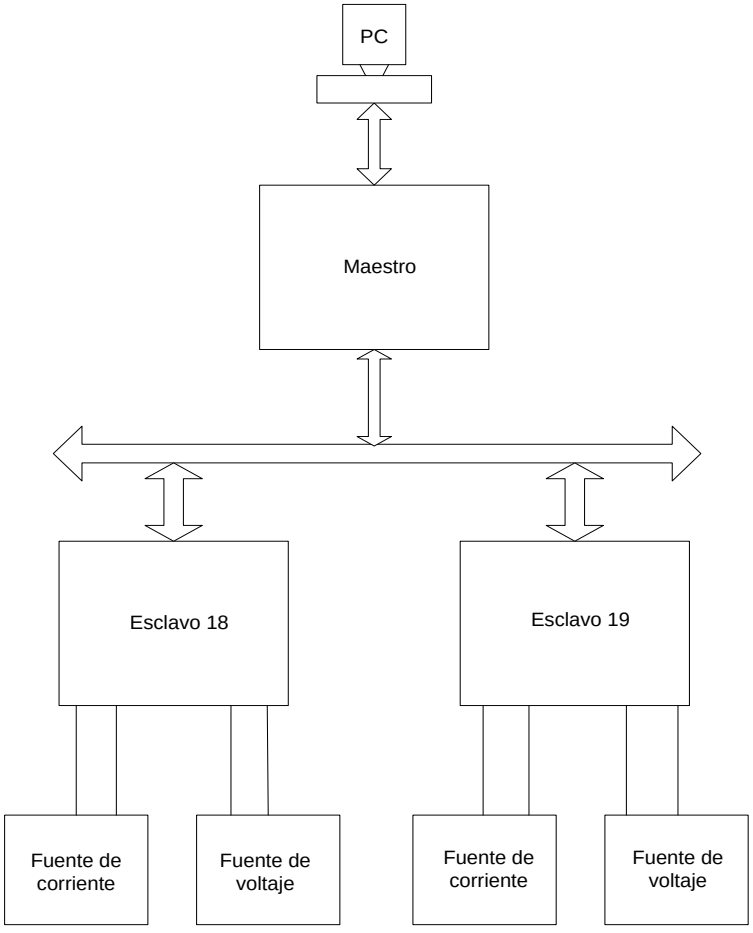


Figura 6.2. Se muestra el diagrama de bloques del circuito

6.1.3. SALIDAS DE SEÑALES ANALÓGICAS Y DIGITALES.

Para el análisis de las salidas de señales analógicas tanto de corriente como de voltaje con diferentes valores, se utilizó un multímetro digital, y la aplicación de la red I2C, al comparar estas mediciones se obtuvo resultados muy favorables.

En la tabla 6.2. Se indican los valores obtenidos de la medición del multÍmetro al enviar una señal de corriente y de voltaje desde la pantalla principal para ser medida en las salidas de los esclavos de la red, además se comprobó el correcto funcionamiento de las salidas digítales con el empleo de leds conectados en las salidas digítales de los esclavos.

Tabla 6.2. Ejemplo de la medición de salidas analógicas.

	RED I2C	**MULTÍMETRO**
Voltaje	1 V 4 V	1V 4.05 V
Corriente	4 mA 20 mA	3.9 mA 20.2 mA

6.2. CONCLUSIONES

- Se cumplió con los objetivos propuestos en el plan como es el del Estudio, Diseño e Implementación de una red I2C para Instrumentación Industrial.
- La implementación permite interconectar varios circuitos integrados con relativa facilidad y alta confiabilidad.
- El sistema creado para el efecto cumple en forma correcta con los requerimientos de entradas y salidas analógicas y digitales, tanto de corriente como de voltaje requeridas en los estándares de señales industriales.
- El programa implementado en LabView permite monitorear, supervisar y enviar una señal en forma individual y aleatoria a cada una de los esclavos conectados a la red.

- Dado que no siempre se requiere alta velocidad de transferencia de datos este bus es ideal para sistemas donde es necesario manejar información entre muchos dispositivos y, al mismo tiempo, se requiere poco espacio. Por ello es común ver dispositivos I2C en video grabadoras, sistemas de seguridad, electrónica automotriz, televisores, equipos de sonido y muchas otras aplicaciones.
- Cabe recalcar que el Maestro es el dispositivo que determina la temporización y la dirección del tráfico de datos en el bus. Es el único que aplica los pulsos de reloj en la línea SCL.
- El bus I2C es un sistema de intercambio de información a través de tan solo dos cables que permite a circuitos integrados interactuar entre sí a velocidades relativamente lentas. Además emplea comunicación serie, utilizando un conductor para manejar los pulsos de reloj y otro para intercambiar datos.
- Cada dispositivo conectado a la red como esclavo es reconocido por su código (dirección) de 7 bits y puede operar como transmisor o receptor de datos.
- El bus I2C es una red diseñada para operar a distancias cortas empleando para ello un cable trenzado como medio de transmisión de la red y una masa común para la interconexión de los circuitos. Si se desea extender la distancia se podría emplear circuitos que permitan amplificar las señales SDA y SCL de la red.
- El número máximo de componentes de la red es de 128 elementos, debido a que el direccionamiento de los esclavos se lo realiza utilizando 7 bits.
- Empleando el circuito integrado MAX232 se logra integrar correctamente el sistema de la red I2C al computador, utilizando la comunicación serie.
- La velocidad de la comunicación serie influye en la velocidad de muestreo para el envió y la recepción de datos desde la PC al maestro y viceversa, provocando que se disminuya la velocidad de transmisión entre ellos.
- Con la comunicación serial asincrónico se logra disminuir a un número de tres líneas el cable empleado para la comunicación serial. Una para

la transmisión, otra para la recepción de los datos y una tercera para tierra.

- El programa LabView, tiene instrucciones para manejar el puerto serial en forma amigable con el usuario, y facilita la programación que permite adquirir y enviar datos a la red I2C.
- Con la ayuda de LabView se logra mostrar un entorno bastante sencillo de operación del sistema.
- El protocolo de transferencia de datos y direcciones de la red I2C nos da la ventaja de diseñar sistemas completamente definidos por software.
- La red I2C logra agrupar a varios elementos como esclavos al sistema, a distancias relativamente cortas y a velocidades de trasmisión de datos lentas. En el campo industrial se ha puesto énfasis en construir redes que permitan controlar a grandes distancias sensores y actuadores, siendo ésta la principal limitación de la red I2C frente a las redes industriales.
- Al realizar este proyecto, hemos logrado consolidado en nosotros el estudio de los microcontroladores empleados para generar una red serial sincrónica, la interfase RS232 como medio de comunicaron entre la PC y el maestro y LabView para la interfase entre el usuario y el sistema.

6.3. RECOMENDACIONES

- En la implementación del circuito se recomienda utilizar capacitares de aproximadamente 100 pF en la entrada de voltaje de las placas, con el fin de eliminar ruido.
- El conductor que se emplea como medio de transmisión de la red I2C debe ser de preferencia UTP o un cable trenzado que permita eliminar ruido en la red.
- Se recomienda que la longitud del cable entre maestro y esclavo no supere los 5 metros de distancia, debido a que se tiene pérdidas de información.

- Se debe tener cuidado con la correcta polarización de los microcontroladores para evitar daños irreversibles en estos.
- Se recomienda colocando dos leds por cada placa, que muestren el instante en que el microcontrolador recibe y envía datos.
- Se recomienda que las herramientas a emplearse dentro del sistema sean conocidas por el programador con el fin de evitar demoras y posibles errores en la construcción del sistema.
- Al soldar los elementos en las placas se debe evitar el provocar una suelda fría, ya que resulta muy difícil detectar esta falla.
- Se debe colocar a tierra las líneas que no sean utilizadas, para evitar tener niveles flotantes, que generen una información considerada como basura en la red.
- Fomentar nuevos trabajos con esta red en el área de la instrumentación, superando las limitaciones del sistema.
- Al realizare un proyecto de grado, incentivar a los alumnos por investigar y crear nuevos sistemas, que cubran las necesidades de la industria moderna.

BIBLIOGRAFÍA

- ANGULO, ROMERO y ANGULO, “Microcontroladores PIC 16F87X. Diseño práctico y aplicación”.
- MANUAL DE LabView.
- MANUAL DEL PIC16F877 – Microchip
- ZELENOVSKY Ricardo, “IBM PC para Ingenieros”
- MANUAL ECG
- ANTONIO MÁNUEL LÁZARO, LabView, ”Programación gráfica para el control de instrumentación”

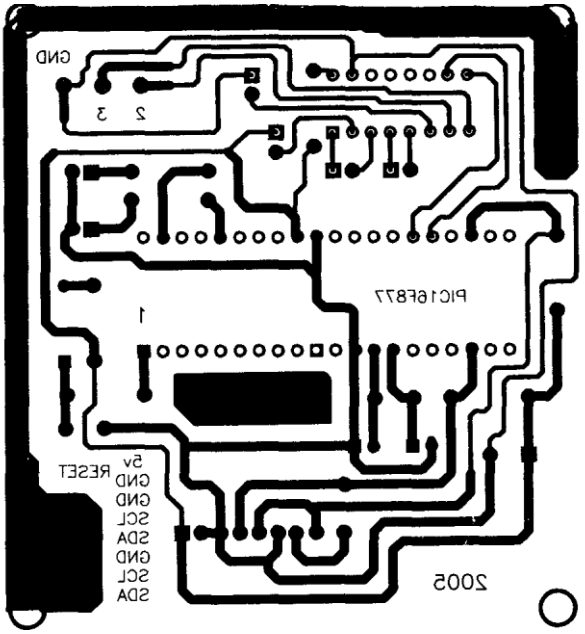
ENLACES

- http://www.redes.upv.es/rc1/practicas/rs232.pdf
- http://mundobot.com/tecnica/tecnica.htm
- http://www.pablin.com.ar/electron/cursos/i2c/
- http://www.comunidadelectronicos.com/articulos/i2c.htm
- http://www.arcelect.com/rs232.htm
- http://www.winpicprog.co.uk/pic_tutorial7.htm

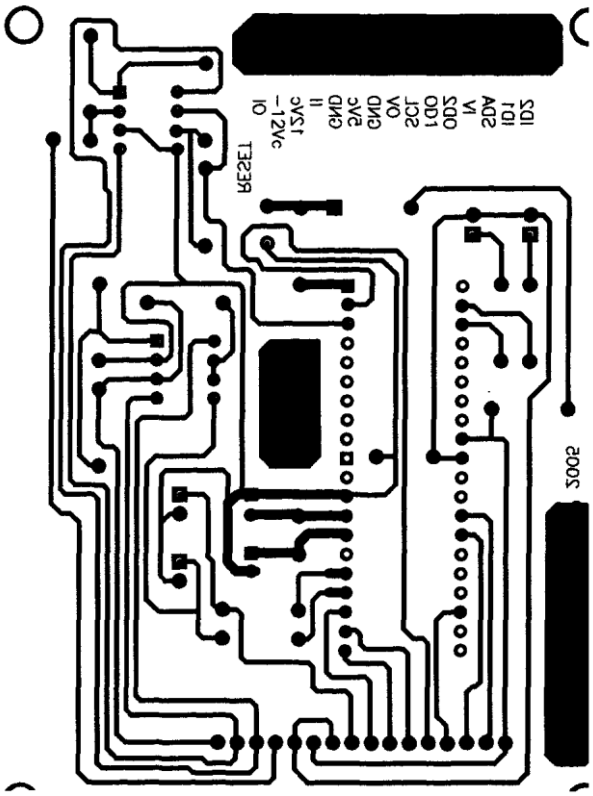
ANEXO A

TARJETAS DISEÑADAS

TARJETA PARA EL MAESTRO DE LA RED I2C



TARJETA PARA EL ESCLAVO DE LA RED I2C



ANEXO B

PROGRAMAS

PROGRAMA PARA EL MAESTRO

```
        LIST P=18F452
        #include <P18F452.INC>
*****************************************************************************

;       *** Interrupciones principales ***
                btfss   PIE1, RCIE
                bra     interrup2
                btfsc   PIR1, RCIF
                bra     interrup_rc

        interrup2
                btfss   PIE1, TXIE
                bra     interrup3
                btfsc   PIR1, TXIF
                bra     interrup_tx

        interrup3

                btfss   PIE1, SSPIE
                bra     fin_interrup
                btfsc   PIR1, SSPIF
                bra     interrup_ssp

fin_interrupretfie  FAST
;*****************************************************************************
;       CONFIGURACIÓN DEL MÓDULO MSSP
;*****************************************************************************
        movlw b'10011000'
        movwf TRISC
        clrf    TRISB
        clrf    LATB
```

```
;*****************************************************************************
;       CONFIGURACIÓN DEL MÓDULO SSP
;*****************************************************************************
        movlw b'00101000'
        movwf SSPCON1

        movlw 0x3F
        movwf SSPADD
        movlw b'10000000'
        movwf SSPSTAT

;*****************************************************************************
;               CONFIGURACIÓN DEL MÓDULO USART
;*****************************************************************************
        movlw .32
        movwf SPBRG
        movlw b'00100100'
        movwf TXSTA
        movlw b'10010000'
        movwf RCSTA
;*****************************************************************************
;       CONFIGURACIÓN DE INTERRUPCIONES
;*****************************************************************************
        movlw b'00100000'
        movwf PIE1

        movlw b'11000000'
        movwf INTCON

        movlw b'11000111'
        movwf T0CON
```

```
        lfsr    0, 0x100
        bra     $
;*******************************************************************************
;       INTERRUPCIÓN POR RECEPCIÓN SERIE
;*******************************************************************************
interrup_rc
        movf    RCREG, W
        movwf           INDF0
        incf    FSR0L, f

        movlw   0x06
        cpfseq  FSR0L
        bra     fin_interrup

        lfsr    0, 0x100
;       bcf     PIE1, RCIE
        bsf     PIE1, SSPIE
        bcf     PIR1, SSPIF

        movff   INDF0, Direccion

        btfss   Direccion, 0
        bra     _TX

        btg     LATB, 7

        movlw   0x04
        movwf   Estado_I2C
        call    Operaciones_I2C
        bra     fin_interrup

_TX
        clrf    Estado_I2C
        call    Operaciones_I2C
```

```
        bra     fin_interrup
;*******************************************************************************
;       INTERRUPCIÓN POR TRANSMISIÓN SERIE
;*******************************************************************************
interrup_tx
movff       INDF2, TXREG
        incf    FSR2L, f
        movlw0x05
        cpfseqFSR2L
        bra     fin_interrup
        bcf     PIE1, TXIE
        bra     fin_interrup



;*******************************************************************************
;       INTERRUPCIÓN POR MóDULO SSP (I2C)
;*******************************************************************************
interrup_ssp
        call    Operaciones_I2C

        bcf     PIR1, SSPIF
        bra     fin_interrup

Operaciones_I2C
        movf   Estado_I2C, W
        addwf Estado_I2C, W
        addwf PCL, f
; Operaciones para la transmisión
        bra     Inicio_Escritura
        bra     Tran_Dir_Escritura
        bra     Enviar_Datos
        bra     Parada_Escritura

; Operaciones para la recepción
```

```
        bra     Inicio_Lectura
        bra     Tran_Dir_Lectura
        bra     Reconocer_Dir
        bra     Recibir_Datos
        bra     Configurar_Lectura
        bra     Parada_Lectura

;*******************************************************************************
; Operaciones para la transmisión
;*******************************************************************************
Inicio_Escritura
        bsf     SSPCON2, SEN
        incf    Estado_I2C
        return

Tran_Dir_Escritura
        lfsr    1, 0x101
        movf    Direccion, W
        movwf   SSPBUF
        incf    Estado_I2C
        return

Enviar_Datos
        btfss   SSPCON2, ACKSTAT
        bra     Transmitir
        bsf     SSPCON2, PEN
        clrf    Estado_I2C
        return

Transmitir
        movff   INDF1, SSPBUF
        movlw   0x6
        incf    FSR1L, f
        cpfseq  FSR1L
```

```
        return
        incf    Estado_I2C
        return


Parada_Escritura
        bsf     SSPCON2, PEN
        bcf     PIE1, SSPIE
        return
;*******************************************************************************
; Operaciones para la recepción
;*******************************************************************************
Inicio_Lectura
        bsf     SSPCON2, SEN
        incf    Estado_I2C
        btg     LATB, 6
        return

Tran_Dir_Lectura
        lfsr    1, 0x100
        movf    Direccion, W
        movwf   SSPBUF
        incf    Estado_I2C
        return

Reconocer_Dir
        btfss   SSPCON2, ACKSTAT
        bra     Iniciar_Recepcion
        movlw   0x04
        movwf   Estado_I2C
        bsf     SSPCON2, PEN
        return

Iniciar_Recepcion
```

```
        btg     LATB, 5
        bsf     SSPCON2, RCEN
        incf    Estado_I2C, f
        return

Recibir_Datos
        movlw   0x04
        cpfseq  FSR1L
        bra     enviar_ACK
enviar_NACK
        movf    SSPBUF, W
        movwf   INDF1

        bsf     SSPCON2, ACKDT
        bsf     SSPCON2, ACKEN

;       movff   INDF1, SSPBUF
        movlw   0x02
        addwf   Estado_I2C, f
        btg     LATB, 3
        return
enviar_ACK
        btg     LATB, 4
        movf    SSPBUF, W
        movwf   INDF1

        bcf     SSPCON2, ACKDT
        bsf     SSPCON2, ACKEN

;       movff   INDF1, SSPBUF
        incf    FSR1L, f
        incf    Estado_I2C, f
        return
```

```
Configurar_Lectura
        bsf     SSPCON2, RCEN
        decf    Estado_I2C, f
        return

Parada_Lectura
        bsf     SSPCON2, PEN
        bcf     PIE1, SSPIE
        bsf     PIE1, TXIE
        lfsr    2,0x100
        return
;*********************************************************************************
;Fin del programa

        END
```

PROGRAMA PARA EL ESCLAVO

```
        list            p=16f877A
        #include        <p16f877A.inc>
;*********************************************************************************
;       CONFIGURACIÓN DE LOS PUERTOS
;*********************************************************************************
        BANKSEL     PORTB
        clrf    PORTB
        clrf    PORTD

        BANKSEL     TRISB
        clrf    TRISB
        movlw   b'00011000'
        movwf   TRISC

        movlw   b'11110000'
```

```
        movwf   TRISD
;*******************************************************************************
;       CONFIGURACIÓN DEL MÓDULO SSP
;*******************************************************************************
        BANKSEL     SSPCON
        movlw   b'00110110'
        movwf   SSPCON

        BANKSEL     SSPCON2
        bsf     SSPCON2, SEN
;*******************************************************************************
        BANKSEL     SSPADD
        movlw   0x18<<1
        movwf   SSPADD
;*******************************************************************************
        BANKSEL     SSPSTAT
        clrf    SSPSTAT
;*******************************************************************************
;       CONFIGURACIÓN DE LOS MÓDULOS CCP1 y CCP2
;*******************************************************************************
        BANKSEL     CCP1CON
        movlw   b'00001100'
        movwf   CCP1CON
        movwf   CCP2CON

        movlw   b'00000100'
        BANKSEL     T2CON
        movwf   T2CON

        movlw   0xFF
        BANKSEL     PR2
        movwf   PR2

        movlw   0x7F
```

```
        BANKSEL    CCPR1L
        movwf CCPR1L
        movwf CCPR2L
;*********************************************************************
;       CONFIGURACIÓN DEL CONVERSOR ADC
;*********************************************************************
        BANKSEL    ADCON0
        movlw b'11000001'
        movwf ADCON0

        BANKSEL    ADCON1
        movlw b'11000100'
        movwf ADCON1
;*********************************************************************
;       CONFIGURACIÓN DE INTERRUPCIONES
;*********************************************************************
        BANKSEL    PIE1
        movlw b'00001000'
        movwf PIE1

        movlw b'11000000'
        movwf INTCON

        goto   $

*********************************************************************
;       INTERRUPCIÓN POR MÓDULO MSSP
;*********************************************************************
interrup_ssp
        BANKSEL    SSPSTAT
        btfss   SSPSTAT, D_A
        goto    Direccion

        btfsc   SSPSTAT, R_W
```

```
        goto    Transmitir
        goto    Recibir
fin_ssp
        BANKSEL     PIR1
        bcf     PIR1, SSPIF

        BANKSEL     SSPCON
        bsf     SSPCON, CKP

        goto    fin_interrup

Direccion
        BANKSEL     SSPBUF
        movf    SSPBUF, W

        BANKSEL     SSPSTAT
        btfsc   SSPSTAT, R_W
        goto    Iniciar_Transmision
        movlw   0x05
        BANKSEL     Contador
        movwf   Contador

        movlw   0x40
        movwf   FSR

        goto    fin_ssp

Iniciar_Transmision
        BANKSEL     PORTB
        bsf     PORTB, 7
        movlw   0x40
        movwf   FSR

        BANKSEL     ADCON0
```

```
bcf     ADCON0, CHS0
call    DLY1
BANKSEL     ADCON0
bsf     ADCON0, GO
btfsc   ADCON0, NOT_DONE
goto    $-1
BANKSEL     ADRESH
movf    ADRESH, W
movwfINDF
incf    FSR, f
BANKSEL     ADRESL
movf    ADRESL, W
movwfINDF
incf    FSR, f

BANKSEL     ADCON0
bsf     ADCON0, CHS0
call    DLY1
BANKSEL     ADCON0
bsf     ADCON0, GO
btfsc   ADCON0, NOT_DONE
goto    $-1
BANKSEL     ADRESH
movf    ADRESH, W
movwfINDF
incf    FSR, f
BANKSEL     ADRESL
movf    ADRESL, W
movwfINDF
incf    FSR, f

BANKSEL PORTD
movf    PORTD, W
movwfINDF
```

```
        movlw 0x40
        movwf FSR

        movf   INDF, W
;       movf   FSR, W

        BANKSEL    SSPBUF
movwf       SSPBUF
        incf    FSR, f

        BANKSEL    PORTB
        bsf     PORTB, 6

        goto    fin_ssp
Recibir
        BANKSEL    SSPBUF
        movf   SSPBUF, W
        movwf INDF
        incf    FSR, f
        decfsz Contador, f
        goto    fin_ssp

        movlw 0x40
        movwf FSR
        BANKSEL CCPR1L
        movf   INDF, W
        movwf CCPR1L

        incf    FSR, f
        movlw b'00001111'
        BANKSEL    CCP1CON
        andwf CCP1CON, f
        swapf INDF, W
```

```
        andlw b'00110000'
        addwf CCP1CON

        incf    FSR, f
        BANKSEL CCPR2L
        movf   INDF, W
        movwfCCPR2L

        incf    FSR, f
        movlwb'00001111'
        BANKSEL    CCP2CON
        andwf CCP2CON, f
        swapf INDF, W
        andlw b'00110000'
        addwf CCP2CON

        incf    FSR, f
        movf   INDF, W

        BANKSEL    PORTD
        movwfPORTD

        goto   fin_ssp

    Transmitir
        BANKSEL    PORTB
        bsf    PORTB, 5
        movf   INDF, W

        BANKSEL    SSPBUF
        movwfSSPBUF
        incf    FSR, f
        goto   fin_ssp
    DLY1
```

```
BANKSEL     Reg_dly
movlw.20
movwfReg_dly
decfszReg_dly, f
goto    $-1
return

END
```

ANEXO C

MANUAL DEL USUARIO

DISEÑO E IMPLEMENTACIÓN DE UNA RED I2C PARA INSTRUMENTACIÓN INDUSTRIAL

1. CONEXIÓN DE LA TARJETA DEL MAESTRO A LA PC.

- Insertar el cable diseñado con el conector DB9 a la PC y el otro extremo en la tarjeta del maestro.
- Realizar la conexión de la fuente en los conectores indicados con + 12 V, -12V, 5V y tierra.
- Conectar las entradas analógicas y digitales a ser adquiridas.
- Conectar a las salidas del sistema, instrumentos que permitan la medición de las variables de salida.
- Chequear la correcta ejecución de los pasos anteriores.

2. CONEXIÓN DEL SOFTWARE

- Ejecute el programa con el nombre “Principal” desde LabView.
- Se muestra en el monitor los controles y visualizadores diseñados para el proyecto.
- No corra la aplicación todavía.
- Encienda la fuente de energía, las señales a adquirir y los instrumentos para la medición de las salidas.

3. UTILIZACIÓN DE LA APLICACIÓN

- Ingresar las direcciones de los esclavos en el sitio indicado.
- Ejecute la aplicación.
- Si aparece un aviso de error, verifique que las direcciones de los esclavos seleccionados correspondan a los digitados en la pantalla.
- Coloque los valores deseados a la salida del sistema utilizando las perillas que se muestran.

- Con la ayuda de los visualizadores de LabView se tiene los valores de las señales ingresadas a la red.
- Si desea cambiar las direcciones de los esclavos primero pare la ejecución del programa.
- Para terminar con la aplicación pulse el icono STOP.

ANEXO D

ACRÓNIMOS PRINCIPALES

Hs Módulo de alta velocidad

ICs Circuitos integrados

SDA Datos seriales

SCL Reloj serial

VDD Fuente de voltaje directo

S Condiciones de arranque

P Condicione de parada

FLASH, RAM Tipos de memoria

EEPROM Memoria de programa

A/D Conversor analógico a digital

USART, I2C, SPI Puertos de comunicación serie

E/S Líneas de entrada y salida

RP0 y RP1 Registros de estado

PSP Protocolo de comunicación

MSB bit más significativo

CBUS Compatibilidad del bus

STOP Paso de “0” a “1” con SCL=“1”

tHD.STA Tiempo del sostenimiento mínimo

LSB bit menos significante

R/W Lectura o escritura

ACK Reloj de pulsación de reconocimiento

Sr Condición de arranque repetido

F/S modo Modo Standard

MSBs bits más significativos

SDAH Reconocimiento datos seriales de alta velocidad

SCLH Reloj serial de altas velocidades

Rs Resistores opcionales en serie

Rp Resistores de empuje

Ā bit no reconocido

A bit de reconocimiento

Cb Capacitancia del bus

DTEs Equipos terminales de datos

DCEs Equipos de comunicación de datos

T**RISA, PORTA** Registros de datos de los puertos

OPTON Registro de opciones.

Fosc/4 Reloj interno

SSP Puerta Serie Síncrono

NRZ Protocolo de comunicación asíncrono no retorno a cero

VI Instrumento virtual

ANSI American National Standard Institution

ETD Equipo Terminal de Datos

ECD Comunicación de Datos

EIA Electronic Industries Association

TIA Telecommunications Industry Association

Latacunga, octubre del 2005

……..

Luis Barrionuevo

ALUMNO

CI. 050248157-5

……..

Patricio Barrionuevo

ALUMNO

CI. 050248144-3

……..

Ing. Nancy Guerrón.

DIRECTORA DE CARRERA

……..

Ab. Eduardo Vásquez

SECRETARIO ACADÉMICO