ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES Y REDES

"SISTEMA ELECTRÓNICO DE GESTIÓN DE COBROS DE AGUA POTABLE CON DISPOSITIVO PORTABLE DE DATOS UTILIZANDO EL PROTOCOLO SPI"

TESIS DE GRADO
PREVIA OBTENCIÓN DEL TÍTULO
DE INGENIERO ELECTRÓNICO Y COMPUTACIÓN

AUTORES

ADRIÁN DARÍO MONTESDEOCA TAPIA
DIEGO NOE PUSAY VILLARROEL

RIOBAMBA-ECUADOR

2011

A mis Padres y hermanos
porque gracias a su cariño, guía y apoyo he llegado a realizar uno de mis anhelos más grandes de mi vida, fruto del inmenso apoyo, amor y confianza que en mi se depositó y con los cuales he logrado terminar mis estudios profesionales que constituyen el legado más grande que pudiera recibir y por lo cual les viviré eternamente agradecido.

Adrián Darío Montesdeoca Tapia

A mi familia que constantemente ha depositado
su apoyo y confianza en mis ideales, por su presencia
incondicional en los buenos y malos momentos.

Diego Noé Pusay Villarroel

NOMBRE	FIRMA	FECHA
Ing. Iván Ménes **DECANO**	**.....**	**.....**
Ing. Pedro Infante **DIRECTOR DE LA ESCUELA**	**.....**	**.....**
Ing. Franklin Moreno **DIRECTOR DE TESIS**	**.....**	**.....**
Ing. Wilson Zúñiga **MIEMBRO DEL TRIBUNAL**	**.....**	**.....**
Tec. Carlos Rodríguez **DIRECTOR DPTO DOCUMENTACIÓN**	**.....**	**.....**
NOTA DE LA TESIS	**.....**	

“Yo, Diego Noé Pusay Villarroel soy responsable de las ideas, doctrinas y resultados expuestos en esta tesis; y, el patrimonio intelectual de la Tesis de Grado pertenece a la ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO”.

_____

Diego Noé Pusay Villarroel

“Yo, Adrián Darío Montesdeoca Tapia soy responsable de las ideas, doctrinas y resultados expuestos en esta tesis; y, el patrimonio intelectual de la Tesis de Grado pertenece a la ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO”.

_____

Adrián Darío Montesdeoca Tapia

ABREVIATURAS

ARES	Advancedrouting and editing software o software de edición y ruteo avanzado
CAD	Conversores analógico/digital
CCP	The capture, compare, and pwm
CDA	Conversores digital/analógico
CLK	Pin del reloj
CMD	Comando
CMD (DataIn)	Línea de datos y comandos hacia la tarjeta
CPHA	Clockphase o reloj de fase
CPOL	Clockpolarity o polaridad de reloj
CPRM	Content protection for recordable media
CRC	Cyclicredundancycheck
CS	Chip select
DI	Entrada de datos
DIP	Dual in line package
E	Enable
E/S	Entrada/salida
ECCP	Theenhancedccp
EEPROM	Electrically-erasableprogrammableread-onlymemory o rom programable y borrable eléctricamente
Gb	Gigabyte
GLCD	Graphicliquid cristal display
I²C	Inter-integratedcircuit
int	Entrada
ISIS	Intelligentschematic input system o sistema de enrutado de esquemas inteligente
kΩ	Kilo-omios
mAh/g	Miliamperios hora gramo
MCLR	Borrado maestro
mhz	Mega hertz
MISO	La de salida de datos
MOSI	Conexión para la entrada de datos serie
MSSP	Master syncronous serial port
OSC	Oscilador
out	Salida
pf	Picofaradios
PROM	Programmableread-onlymemory
R1/R2	Respuestas
RAM	Randomaccessmemory

RISC	Reducedinstruction set computer
ROM	Read-onlymemory
RST	Reset
RW	Lectura escritura
SCLK	La señal de reloj
SD	Secure digital
SDI	Entrada de dato serial
SDO	Salida dato serial
SMBus	Systemmanagement bus
SPI	Serial peripheral interface
SPP	Standard parallelport
SS	Slave select o selección de esclavo
SSP	Syncronous serial port
UCP	Unidad central de proceso
USB	Universal serial bus
V	Voltaje
VSM	Sistema virtual de modelado

ÍNDICE

CAPÍTULO I

CAPÍTULO II

CAPÍTULO III

CAPÍTULO IV

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

EL almacenamiento y tratamiento ágil de la información se torna en una necesidad básica para cualquier organización que desee una correcta administración a fin de que su actividad tenga éxito.

En el presente trabajo de investigación abordamos el desarrollo de un Sistema Electrónico para la Gestión de Cobros de Agua Potable, en el cual se integra un Dispositivo Portable con un Software de Gestión de Base de Datos que permita automatizar íntegramente los Cobros por concepto de consumo de agua potable de una parroquia rural de la ciudad de Riobamba.

El desarrollo del documento se basa en la explicación detallada de cada componente tanto físico como lógico necesario para la implementación, empezando por el funcionamiento de protocolo SPI, las características de las memorias SD, el manejo de estos por medio de micro controladores, y el diseño electrónico que permita su integración, y por último se detalla el desarrollo del Software de gestión de los datos.

El trabajo desarrollado constituye un aporte científico y tecnológico para satisfacer las necesidades de gestión de datos de la Junta de Aguas Potable de la parroquia “San Gerardo”

CAPÍTULO I

1.1 MARCO REFERENCIAL

1.1.1 ANTECEDENTES

Tradicionalmente los dispositivos que trabajan con micro controladores utilizan memorias EEPROM para el almacenamiento de datos sin embargo este tipo de memorias requieren módulos adicionales para comunicación con otros dispositivos y no poseen una capacidad de almacenamiento significativa. La memoria flash como por ejemplo la SD-CARD es una forma avanzada de EEPROM creada por el Dr. FujioMasuoka mientras trabajaba para Toshiba, estas memorias tiene la posibilidad de ser manejadas por micro controladores utilizando el protocolo SPI(Serial Peripherical Interface)que es un estándar para controlar casi cualquier electrónica digital que acepte un flujo de bits serie regulado por un reloj permitiendo comunicación Full Duplex, una mayor velocidad de transmisión que con I^2C o SMBus, es un protocolo flexible en que se puede tener un control absoluto sobre los bits transmitidos.

Las memorias Secure Digital (SD) han ganado gran popularidad en dispositivos portátiles tales como teléfonos móviles, cámara digitales, reproductores multimedia entre otros llegando a remplazar en los computadores personales al Flopy Drive por el Card Reader.

La investigación sobre la compatibilidad entre micro controladores y memorias SD nos permitirá desarrollar dispositivos electrónicos portables para dar solución a problemas puntuales como el de la parroquia San Gerardo perteneciente al cantón Guano, que cuenta con un sistema de Agua Potable, manejado por los propios usuarios del sistema a través de una Junta Administradora, el sistema fue donado por el Fondo Ítalo Ecuatoriano y el Consejo Provincial, la gestión de cobros se ha venido llevando de forma manual lo cual ha generado molestias debido a la lentitud con que se realiza los procesos, falta de recaudación y falencias en el servicio, lo que ha llevado al sistema actual al borde del colapso.

1.1.2 JUSTIFICACIÓN

Debido a la versatilidad de las memorias SD respecto a las EEPROM integradas o externas a un micro controlador se plantea la construcción de un Sistema Electrónico de gestión de cobros con un dispositivo portable que servirá para la recolección y procesamiento de datos referentes al consumo de Agua Potable que representa un problema para las Juntas Administradoras de Agua Potable de Chimborazo.

1.1.3 OBJETIVOS

1.1.3.1 OBJETIVO GENERAL

Diseñar un Sistema Electrónico de gestión de cobros de agua potable con dispositivo portable de datos utilizando el Protocolo SPI.

1.1.3.2 OBJETIVOS ESPECIFICOS

- Estudiar el funcionamiento del protocolo SPI
- Investigar el micro controlador adecuado para el diseño electrónico del equipo a implementar.
- Implementar un Dispositivo Electrónico portable de Datos con memoria SD e interfaz a PC capaz de llevar los registros mensuales de consumo de Agua Potable.
- Implementar un módulo para transferir los datos desde el Dispositivo Portable al Sistema de Gestión a ser implementado en un computador.
- Diseñar e Implementar una aplicación de base de datos que permita procesar la información registrada por el dispositivo, generando recibos para el cobro del consumo de agua potable y reportes de las operaciones realizadas.

1.1.4 HIPÓTESIS

La construcción de un Sistema Electrónico de gestión de cobros de agua potable con dispositivo portable de datos, utilizando el protocolo SPI, un micro-controlador, y memoria SD servirá para la gestión de cobros del agua potable.

CAPÍTULO II

2.1 MARCO TEÓRICO

2.1.1 SERIAL PERIPHERAL INTERFACE (SPI)

2.1.1.1 DEFINICIÓN

El Bus SPI (del inglés Serial Peripheral Interface) es un estándar de comunicaciones, usado principalmente para la transferencia de información entre circuitos integrados en equipos electrónicos. El bus de interface de periféricos serie o bus SPI es un estándar para controlar casi cualquier electrónica digital que acepte un flujo de bits serie regulado por un reloj.

Incluye una línea de reloj, dato entrante, dato saliente y un pin de chip select, que conecta o desconecta la operación del dispositivo con el que uno desea comunicarse. De esta forma, este estándar permite multiplexar las líneas de reloj.

Muchos sistemas digitales tienen periféricos que necesitan existir pero no ser rápidos. La ventajas de un bus serie es que minimiza el número de conductores, pines y el tamaño del circuito integrado. Esto reduce el coste de fabricar montar y probar la electrónica. Un bus de periféricos serie es la opción más flexible cuando muchos tipos diferentes de periféricos serie están presentes. El hardware consiste en señales de reloj, data in, data out y chip select para cada circuito integrado que tiene que ser controlado. Casi cualquier dispositivo digital puede ser controlado con esta combinación de señales. Los dispositivos se diferencian en un número predecible de formas. Unos leen el dato cuando el reloj sube otros cuando el reloj baja. Algunos lo leen en el flanco de subida del reloj y otros en el flanco de bajada. Escribir es casi siempre en la dirección opuesta de la dirección de movimiento del reloj. Algunos dispositivos tienen dos relojes. Uno para capturar o mostrar los datos y el otro para el dispositivo interno.

El Bus SPI es una interfaz serial de 4 líneas usada por muchos microprocesadores y chips periféricos, este provee soporte para bajo/medio ancho de banda (1 megabaud) entre un CPU y otro dispositivo que use SPI.

El bus SPI es básica y relativamente una simple interface serial sincrónica para conexiones de baja velocidad usando un mínimo número de líneas, SPI es un estándar definido por Motorola en los micro controladores de la línea MC68HCxx, y es usado frecuentemente en plataformas de sistemas móviles, el bus es un master/slave interface, cuando dos dispositivos se comunican el uno es llamado “master” y el otro “slave”, el dispositivo

master maneja la señal de reloj, cuando se utiliza SPI, los datos son simultáneamente transmitidos y recibidos, haciendo una comunicación full-duplex.

2.1.1.2 VENTAJAS Y DESVENTAJAS DEL BUS SPI

2.1.1.2.1 VENTAJAS

- Comunicación Full Duplex
- Mayor velocidad de transmisión que con I²C o SMBus
- Protocolo flexible en que se puede tener un control absoluto sobre los bits transmitidos
    - No está limitado a la transferencia de bloques de 8 bits
    - Elección del tamaño de la trama de bits, de su significado y propósito.
- Su implementación en hardware es extremadamente simple
    - Consume menos energía que I²C o que SMBus debido que posee menos circuitos (incluyendo las resistencias pull-up) y estos son más simples
    - No es necesario arbitraje o mecanismo de respuesta ante fallos
    - Los dispositivos clientes usan el reloj que envía el servidor, no necesitan por tanto su propio reloj
    - No es obligatorio implementar un transceptor (emisor y receptor), un dispositivo conectado puede configurarse para que solo envíe, sólo reciba o ambas cosas a la vez
- Usa muchos menos terminales en cada chip/conector que una interfaz paralelo equivalente.

- Como mucho una única señal específica para cada cliente (señal SS), las demás señales pueden ser compartidas.

2.1.1.2.2 DESVENTAJAS

- Consume más pines de cada chip que I²C, incluso en la variante de 3 hilos
- El direccionamiento se hace mediante líneas específicas (señalización fuera de banda) a diferencia de lo que ocurre en I²C que se selecciona cada chip mediante una dirección de 7 bits que se envía por las mismas líneas del bus
- No hay control de flujo por hardware
- No hay señal de asentimiento. El servidor podría estar enviando información sin que estuviese conectado ningún cliente y no se daría cuenta de nada
- No permite fácilmente tener varios servidores conectados al bus
- Sólo funciona en las distancias cortas a diferencia de, por ejemplo, RS-232, RS-485, o Bus CAN

2.1.1.3 ESPECIFICACIONES DEL BUS

El sistema está formado por un componente maestro y uno o más componentes esclavos. El maestro, se define normalmente como un microcomputador provisto de un reloj SPI (SPI clock), y los esclavos, como cualquier circuito integrado, reciben el reloj SPI del maestro.

SPI, es una interface serie síncrona de 4 hilos, la comunicación de datos se activa mediante una señal baja, aplicada en la entrada Slave Select (SS) o en el Chip Select (CSB). Los datos, se transmiten mediante 3 conexiones:

1. Entrada de datos serie (MOSI).
2. Salida de datos (MISO).
3. Señal de reloj (SCK).

En el sistema SPI, existe un Master (sistema encargado de la comunicación) y uno o más esclavos, como podemos ver en estos diagramas.

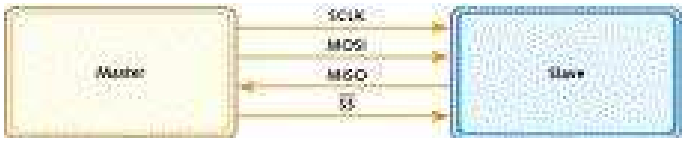


Figura II - 1: Caso Master con un esclavo

SCLK:Clock
MOSI: Master out - Slave in
MISO: Master in - Slave out
SS: Slave select

Tabla II - I: Señales del sistema SPI

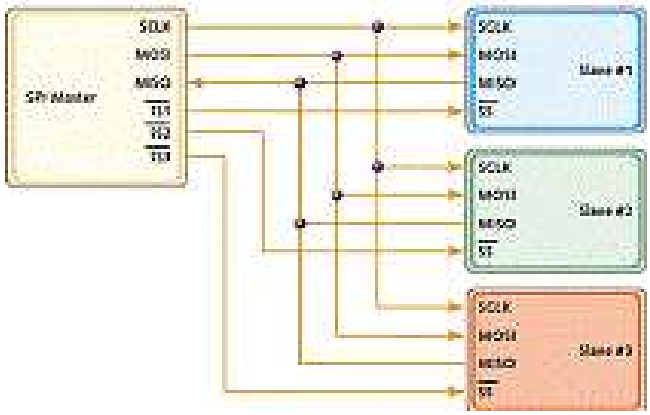


Figura II - 2: Caso de Master con más de un esclavos

Para la comunicación entre Dispositivos el master, genera una señal de reloj y la envía a los esclavos. La línea de selección, slave line, se utiliza para indicar con que esclavo se está intentando comunicar el Master.

Como se puede ver, todas las señales excepto la de selección de esclavo son comunes a todos los esclavos, por eso, el master debe indicar cuál de los esclavos esta activo durante la comunicación.

Podemos imaginar, que con más de uno o dos esclavos, el número de salidas del master y el número de pistas en la placa, serian demasiadas para justificar el uso de este protocolo.

2.1.1.4 MODOS DEL RELOJ

Todos la transferencia de los datos, son sincronizados por la línea de reloj de este bus. Un BIT es transferido por cada ciclo de reloj.

La mayoría de las interfaces SPI tienen 2 bits de configuración, llamados:

- CPOL (ClockPolarity = Polaridad de Reloj)

- CPHA (ClockPhase = Reloj de Fase).

CPOL determina si el estado Idle de la línea de reloj esta en bajo (CPOL=0) o si se encuentra en un estado alto (CPOL=1).

CPHA determina en que filo de reloj losdatos son desplazados hacia dentro o hacia fuera.

Si CPHA=0 los datos sobre la línea MOSI son detectados cada filode bajada y los datos sobre la línea MISO son detectados cada filo de subida.

Cada BIT tiene 2 estados, lo cual permite 4 diferentescombinaciones, las cuales son incompatibles una de la otra. Por lo que si dos dispositivos SPI desean comunicarse entre si, estos debentener el mismo la misma Polaridad de Reloj (CPOL) y la misma Fase de Reloj (CPHA).

Existen cuatro modos de reloj definidos por el protocolo SPI, estos modos son :

- Modo A
- Modo B
- Modo C
- Modo D

Estos determinan el valor de la polaridad del reloj (CPOL = ClockPolarity) y el bit de fase del reloj (CPHA = ClockPhase). La mayoría de los dispositivos SPI pueden soportar al menos 2 modos de los 4 antes mencionados.

Los diferentes modos son ilustrados a continuación.

El BIT de Polaridad del reloj determina el nivel del estado de Idle del reloj y el BIT de Fase de reloj determina que flanco recibe un nuevo dato sobre el bus. El modo requerido para una determinada aplicación, esta dado por el dispositivo esclavo. La capacidad de multi-modo combinada con un simple registro de desplazamiento hace que el bus SPI sea muy versátil.

Si CPOL está en un 0 lógico y ningún dato está siendo transferido (Estado Idle), el maestro mantiene la línea SCLK en bajo.

Si CPOL está en un 1 lógico, el maestro desocupa la línea SCLK alta.

CPHA, conjuntamente con CPOL, controlan cuando los nuevos datos son colocados en el bus. Si CPHA es igual a un '1' lógico, los datos son desplazados sobre la línea MOSI según lo determinado por el valor de CPOL.

Para CPHA = 1:

Si CPOL = 1, los nuevos datos se colocados sobre la línea cuando el flanco del reloj es descendente y se leen cuando el flanco del reloj es ascendente .

Si CPOL = 0, los nuevos datos se ponen en la línea cuando el flanco del reloj es ascendente y se leen cuando el reloj tiene un flanco descendente.

Si CPHA = 0:

El reloj de cambio es la OR de SCLK con la terminal Chip Select.

Tan pronto como el terminal Chip Select se coloca en un nivel lógico 0, los nuevos datos se ponen en la línea y el primer filo de del reloj se leen los datos. Si CPOL se activa a un nivel lógico '1', el primer borde de reloj baja y los bits de datos subsecuentes se leen en cada filo de bajada sobre la línea de reloj.

Cada nuevo bit se pone en la línea cuando el reloj tiene un flanco ascendente de Reloj. Si CPOL es cero, el primer filo de reloj ascendente y los bits de datos subsecuentes se leen en cada filo ascendente de reloj.

Cada nuevo bit se coloca en la línea cuando el filo del reloj baja.

En resumen: Si CPHA=1, la transferencia (datos válidos leídos por el receptor) comienza en el segundo filo de reloj.

Si CPHA=0, la transferencia comienza en el primer filo de reloj. Todas las transferencias subsecuentes dentro del byte ocurren en cada filo de reloj.

Véase las siguientes figuras, en todos los casos, los datos se leen a la mitad del ciclo de reloj después de que se ponen en la línea de datos.

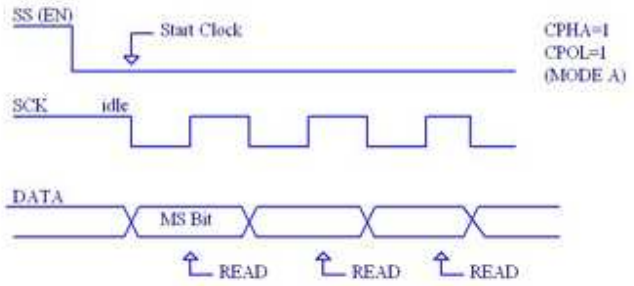


Figura II - 3: Modo A

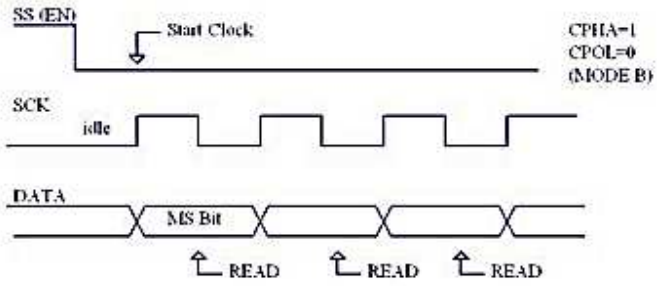


Figura II - 4: Modo B

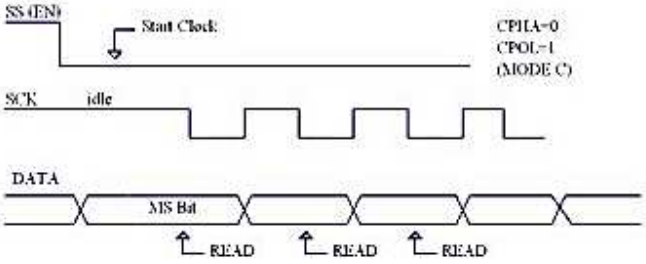


Figura II - 5: Modo C

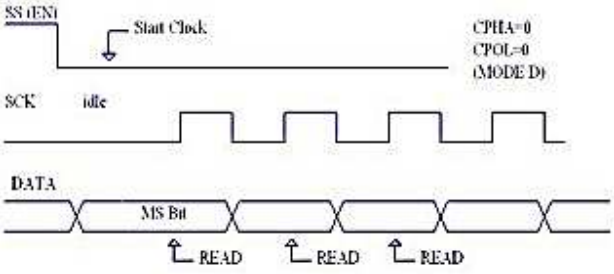


Figura II - 6: Modo D

CAPÍTULO III

3.1 LAS MEMORIAS SD Y SPI

3.1.1 MEMORIAS SD

SD significa Secure Digital, es un formato de tarjeta de memoria flash. Se utiliza en dispositivos portátiles tales como cámaras fotográficas digitales, la Wii, ordenadores PDA, entre otros.

Las tarjetas Secure Digital son utilizadas como soportes de almacenamiento por algunos dispositivos portátiles como:

- Cámaras digitales, para almacenar imágenes.
- La consola Wii, la cual dispone de un lector de este tipo de tarjetas, que ofrece la posibilidad de guardar datos relativos a los juegos.
- Videocámaras, para almacenar tanto imágenes instantáneas como videos.
- PDA, para almacenar todo tipo de datos.

- Teléfonos móviles, para almacenar imágenes, archivos de sonido y otros archivos multimedia.
- Palm.

Existen dos tipos: unos que funcionan a velocidades normales, y otros de alta velocidad que tienen tasas de transferencia de datos más altas. Ya que algunas cámaras fotográficas digitales requieren tarjetas de alta velocidad para poder grabar vídeo con fluidez o para capturar múltiples fotografías en una sucesión rápida.

Las tarjetas SD poseen 9 pines, de los cuales uno es de reloj (CLK), otro es para los comandos, cuatro son de datos y los tres restantes son de alimentación, para la MMC son 7 pines la única diferencia respecto a la SD es que posee sólo dos pines para datos.

Internamente, la tarjeta posee chips de memoria flash como medio de almacenamiento. Además posee un controlador inteligente que maneja los diferentes protocolos de comunicación, algoritmos de seguridad para la protección contra copia no autorizada de información almacenada, algoritmos de corrección de errores de código, diagnósticos y control de potencia

Las especificaciones estándar de la memoria SD son las siguientes:

Area: 768 mm^2

Volumen: 1,613 mm^3

Grosor:2.1 mm

Peso Aprox.: 2g

Núemro de Pines: 9 pines

Voltaje de Operacion: 2.7V - 3.6V

Switch de proteccion de datos: Si

3.1.1.1 CONEXIONES Y SEÑALES

Cuando se utilizan las tarjetas en modo SPI los pins de la tarjeta SD se denominan:

Figura III - 7: Memorias SD

# Pin:	Nombre:	Tipo:	Descripción SPI:
9	DAT2	Reservado	No utilizado en modo SPI
1	!CD (!CS)	Entrada	Chip Select (activo a 0)

2	CMD (DataIn)	Entrada	Línea de datos y comandos hacia la tarjeta.
3	VSS1	Alimentación	Masa
4	VDD	Alimentación	Alimentación
5	CLCK	Entrada	Clock
6	VSS2	Alimentación	Masa
7	DAT0 (DataOut)	Salida	Línea de datos y status de la tarjeta al microcontrolador.
8	DAT1	Reservado	No utilizado en modo SPI

Tabla III - II: Distribución de los pines de la SD

¡CD Permite al controlador seleccionar la tarjeta sobre la cual quiere operar, así cuando CD vale 0 la tarjeta se encuentra seleccionada y lista para operar. Este pin puede ser controlado por cualquier pin de salida del controlador.

CMD (equivalente al DataIn de la MMC) es la entrada de datos serie a la tarjeta y debe estar conectada a la salida MOSI de la interfaz SPI del controlador.

DAT0 (equivalente al DataOut de la MMC) es la salida de datos serie de la tarjeta y debe estar conectada al pin MISO de la interfaz SPI del controlador. **CLCK** es la señal de reloj generada por el controlador y es la que marca el ritmo de transferencia de la información serie entre ambos, así los datos se capturan o transmiten por la tarjeta al ritmo marcado por esta señal.

VDD es el pin de alimentación y **VSS1** y **VSS2** son los pins de masa. **DAT1** y **DAT2** no se utilizan en el modo SPI

3.1.1.2 FUNCIONAMIENTO GENERAL

Básicamente el protocolo SPI consiste en el intercambio de información entre el controlador (master) y la tarjeta (slave). Este intercambio se lleva a cabo mediante el envío de comandos por parte del controlador y de respuestas por parte de la tarjeta. Así, en la lectura, el controlador envía el comando de petición de lectura a la tarjeta y esta le envía la respuesta de confirmación seguida del bloque de datos con la información contenida a partir de la dirección solicitada. En la escritura el proceso es parecido, el controlador indica a la tarjeta mediante el comando de escritura que quiere escribir información en una determinada dirección, esta le responde indicando que esta lista y a continuación el controlador envía el bloque de datos a escribir. Las operaciones que no requieren intercambio de datos funcionan de igual forma pero sin usar bloques de datos.

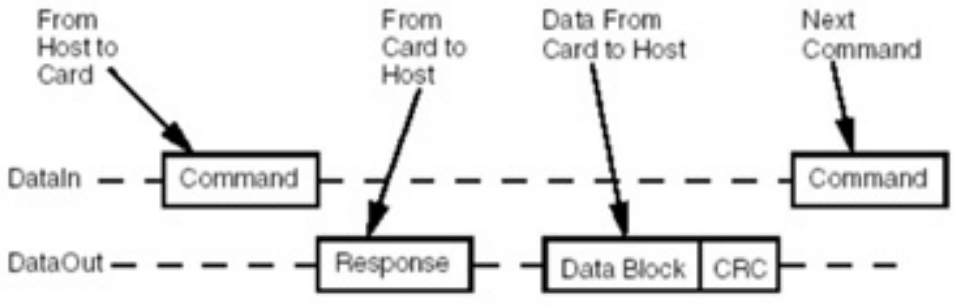


Figura III - 8: Ejemplo de comunicación. Lectura de un bloque

3.1.1.3 COMANDOS MODO SPI

Los comandos son bloques de bytes con un formato fijo:

Byte 1:	b7:0 , b6:1 b5..0:Command
Byte 2-5:	b31..0: Commandargument
Byte 6:	b7..0: CRC 1

Tabla III - III: Comandos SPI

La tarjeta identifica los comandos porque el primer byte de estos (byte 1) siempre comienza por 01, el resto de bits del primer byte contiene el número de comando codificado en binario natural. Así el primer byte correspondiente al comando 0 (CMD0) seria: 01000000, o el primer byte correspondiente al comando 39 (CMD39) sería: 01100111. Los siguientes 4 bytes (bytes 2 –5) contienen los argumentos del comando. En los comandos que no requieren argumentos estos valen 0. El último byte (byte 6) es el byte de CRC para la verificación de errores y en realidad en el protocolo SPI no se utiliza, a no ser que en el registro de configuración se especifique que se desea utilizar CRC.

Toda la comunicación entre la tarjeta y el controlador se realiza según el orden del esquema, de izquierda a derecha, es decir que primero se transmite el bit de mas peso (bit 7) del byte 1, y por último el bit de menos peso (bit 0) del byte 6, es decir es una transferencia More Significant Bit First (MSB First). Estos son algunos de los principales comandos:

Comando:	Argumentos:	Respuesta:	Descripción:
CMD0	No	R1	Resetea la tarjeta
CMD1	No	R1	Inicializa la tarjeta
CMD9	No	R1	Pide a la tarjeta su información CSD
CMD10	No	R1	Pide a la tarjeta su identificación CID
CMD13	No	R2	Consulta el estado de la tarjeta
CMD16	[31..0] Longitud del bloque.	R1	Establece la longitud (en bytes) del bloque para los datos en las operaciones de lectura y escritura.
CMD17	[31..0] Dirección de datos.	R1	Lee un bloque del tamaño indicado por el comando 16.
CMD24	[31..0] Dirección de datos	R1 R1R1	Escribe un bloque del tamaño indicado por el comando 16.

Tabla III - IV: Comandos para memoria SD

3.1.1.4 RESPUESTAS MODO SPI

Las respuestas de la tarjeta son bloques formados por 1 o 2 bytes, dependiendo del tipo de respuesta que se trate. El tipo de respuesta es función del comando, es decir que cada comando tiene asociado un tipo de respuesta.

Bit	Significado
0	Idle State
1	Erase Reset
2	IllegalCommand
3	Com CRC Error

4	Erase_Seq_Error
5	Address Error
6	Parameter Error
7	0

Tabla III - V: Respuesta R1 y RB

Bit	**Significado**
0	0
1	WP Erase Skip
2	Error
3	CC Error
4	Card ECC Failed
5	WP Violation
6	Erase Parameter
7	Out of Range
0	In Idle State
1	Erase Reset
2	IllegalCommand
3	Com CRC error
4	Erase_Seq_Error
5	Address Error
6	Parameter Error

Tabla III - VI: Respuestas en modo SPI

3.1.1.5 BLOQUES DE DATOS

Los bloques de datos comienzan siempre con el byte 0xFE, a este le siguen los bytes de datos y por último los 2 bytes de CRC. El número de bytes de datos depende de la longitud de bloque definida mediante el comando 16, y esta puede ir de 1 hasta 512 bytes (por defecto 512). Por tanto sumando a los bytes de datos el byte de incio y los dos bytes de CRC la longitud total del bloque de datos puede variar entre 4 y 515 bytes. Como por defecto en el protocolo de acceso SPI no se consideran los bytes de CRC, estos pueden tomar cualquier valor.

3.1.1.6 RESET DE LA TARJETA MODO SPI

Por defecto, al arrancar la tarjeta, esta se encuentra en modo MultiMediaCard. Para que entre en modo SPI, hay que enviarle el comando 0 mientras se mantiene activa la señal ¡CS (¡CS=0), pero antes de todo, para poder iniciar la comunicación por el bus hay que enviar como mínimo 74 ciclos de clock a la tarjeta. Así para hacer el reset de la tarjeta y prepararla para trabajar en modo SPI hay que seguir la siguiente secuencia:

- Dar como mínimo 74 ciclos de clock, es decir enviar unos 10 bytes a través de la SPI.
- Activar la señal ¡CS (¡CS=0).
- Enviar el comando 0 con el CRC bien calculado, ya que todavía no estamos en modo SPI, por lo que sí se considera el CRC. De hecho la secuencia del comando 0 siempre es la misma: 0x40,0x00,0x00,0x00,0x00,0x95

3.1.1.7 ACTIVAR LA INICIALIZACIÓN DE LA TARJETA MODO SPI

Una vez reseteada y en modo SPI, hay que hacer la inicialización de la tarjeta, para ello se debe enviar repetidamente el comando 1 hasta que el bit "idle" en la respuesta R1 sea 0. Esto indica que la tarjeta ha completado su inicialización y se encuentra lista para recbir nuevos comandos. La secuencia general es esta:

- Activar el pin ¡CS (¡CS=0).
- Enviar el comando 1: 0x41,0x00,0x00,0x00,0x00, 0xXX . Como la tarjeta ya está en modo SPI el CRC puede tomar cualquier valor.
- Esperar el byte de respuesta, que ha de ser 00000000 (bit "idle" ha de estar 0: tarjeta lista). Si la respuesta no es 0, reenviaremos el comando 1, hasta que lo esté. Puede llevar algunos reintentos.

3.1.1.8 ESCRITURA DE UN BLOQUE EN LA TARJETA MODO SPI

Una vez inicializada la tarjeta, para escribir un bloque en esta, hay que enviar el comando 24 con la dirección de inicio a partir de la cual se desean guardar los datos. Si todo va bien la tarjeta enviará tres respuestas R1 repetidas informando al controlador que ya puede enviar el bloque de datos, que ha de tener una longitud de 512 bytes (en la escritura solo se permiten 512 bytes) más el byte de inicio de bloque de datos y los dos bytes de CRC. La secuencia a seguir es:

- Activar el PIN CS (¡CS=0).
- Enviar el comando 24 0x58, 0xXX,0xXX,0xXX,0xXX,0xYY. Los 4 bytes XX corresponden a la dirección a partir de la cual se quieren guardar los datos. 0xYY corresponde al byte de CRC y como la tarjeta esta en modo SPI pueden tomar cualquier valor ya que no se consideran.
- Si todo va bien la tarjeta responde con el byte de respuesta R1 tres veces consecutivas.
- Enviar a la tarjeta el bloque de datos que consiste en:
    - o - 1 byte de inicio de bloque de datos 0xFE
    - o - 512 bytes con los datos a guardar.
    - o - 2 bytes de CRC
- Mientras la tarjeta está ocupada guardando el valor, irá enviando bytes indicando que está ocupada, y cuando finalice la escritura enviará un byte de confirmación.

3.1.1.9 LECTURA DE UN BLOQUE EN LA TARJETA MODO SPI

Para leer un bloque de la tarjeta hay que enviar a esta el comando 17 con la dirección de inicio de lectura en los bytes de argumento. La dirección puede tomar cualquier valor comprendido dentro del rango de direcciones válidas de la tarjeta pero todo el bloque leído debe estar dentro de un mismo sector físico. A continuación la tarjeta envía un byte de respuesta R1 seguido del bloque de datos, que comienza por 0xFE, continua con los bytes de datos y finaliza con los 2 bytes de CRC que no se usan. El número de bytes de

datos depende del tamaño de bloque que se haya programado mediante el comando 16, y en la lectura puede ir de 1 a 512. La secuencia a seguir es la siguiente:

- Activar el PIN ¡CS (¡CS=0).
- Enviar el comando 17 0x51,0xXX,0xXX,0xXX,0xXX,0xYY. Los 4 bytes XX corresponden a la dirección a partir de la cual se quieren leer los datos. 0xYY corresponde al byte de CRC y como la tarjeta esta en modo SPI puede tomar cualquier valor ya que no se considera.
- Si todo va bien, la tarjeta responde con un byte de respuesta R1, seguido del bloque de datos con la información solicitada y que el controlador tendrá que ir capturando. Esta tiene la misma estructura que los bloques de datos utilizados en la escritura:
    - 1 byte de inicio de bloque de datos 0xFE
    - n bytes con los datos a guardar.
    - 2 bytes de CRC.
- Si se produce un error durante la comunicación, la tarjeta no transmitirá ningún dato y en lugar de estos enviará un byte indicador de error.

CAPÍTULO IV

4.1 MICROCONTROLADORES Y SPI

4.1.1 GENERALIDADES

4.1.1.1 MICROCONTROLADOR

Es un circuito integrado de alta escala de integración diseñado especialmente para controlar sistemas electrónicos, que consta de todos los elementos de una computadora, como procesador o UCP (unidad central de proceso), memoria RAM para Contener los datos, memoria para el programa tipo ROM/PROM/EPROM, Líneas de E/S para comunicarse con el exterior, Generador de impulsos de reloj que sincronizan el funcionamiento de todo el sistema y diversos módulos para el control de periféricos (temporizadores, puertas serie y paralelo, CAD: conversores analógico/digital, CDA: Conversores Digital/Analógico, etc.)

Los microcontroladores más comunes son:

Empresa	**8 bits**	**12 bits**	**14 bits**	**16 bits**	**32 bits**
Atmel AVR	AVR 89Sxx (obsoleto)				AVR32 AT91SAM (ARM)
Freescale (antes Motorola)	68HC05, 68HC08, 68HC11, HCS08	x	x	68HC12, 68HC16	683xx, 68HCS12, 68HCSX12
Hitachi, Ltd	H8	x	x	x	x
Holtek	HT8				
Intel	MCS-48 (familia 8048) MCS51 (familia 8051) 8xC251	x	x	MCS96, MXS296	x
National Semiconductor	COP8	x	x	x	x
Microchip	Familia 10f2xx	Familia 12Cxx de 12 bits	Familia 12Fxx, 16Cxx y 16Fxx de 14 bits (PIC16F87X)	18Cxx y 18Fxx de 16 bits	x
ST	ST 62, ST 7			ST10	STM32 (ARM Cortex-M3) STR7 (ARM7) ATR9 (ARM9)

Texas Instruments	TMS370, MSP430			MSP430 C2000	TMS470 (ARM7)

Tabla IV - VII: Comparación entre distintos fabricantes de microcontroladores

4.1.2 COMPATIBILIDAD

4.1.2.1 SELECCIÓN DEL MICROCONTROLADOR

A la hora de escoger el microcontrolador a emplear en un diseño concreto hay que tener en cuenta multitud de factores, como la documentación y herramientas de desarrollo disponibles y su precio, la cantidad de fabricantes que lo producen y por supuesto las características del microcontrolador (tipo de memoria de programa, número de temporizadores, interrupciones, etc.):

Para que nos hagamos una idea, para el fabricante que usa el microcontrolador en su producto una diferencia de precio en el microcontrolador de algunos dólares es importante (el consumidor deberá pagar además el costo del empaquetado, el de los otros componentes, el diseño del hardware y el desarrollo del software). Si el fabricante desea reducir costos debe tener en cuenta las herramientas de apoyo con que va a contar: emuladores, simuladores, ensambladores, compiladores, etc. Es habitual que muchos de ellos siempre se decanten por microcontroladores pertenecientes a una única familia.

Antes de seleccionar un microcontrolador es imprescindible analizar los requisitos de la aplicación tales como:

- Procesamiento de datos: puede ser necesario que el microcontrolador realice cálculos críticos en un tiempo limitado. En ese caso debemos asegurarnos de seleccionar un dispositivo suficientemente rápido para ello. Por otro lado, habrá que tener en cuenta la precisión de los datos a manejar: si no es suficiente con un microcontrolador de 8 bits, puede ser necesario acudir a microcontroladores de 16 ó 32 bits, o incluso a hardware de coma flotante. Una alternativa más barata y quizá suficiente es usar librerías para manejar los datos de alta precisión.
- Entrada Salida: para determinar las necesidades de Entrada/Salida del sistema es conveniente dibujar un diagrama de bloques del mismo, de tal forma que sea sencillo identificar la cantidad y tipo de señales a controlar. Una vez realizado este análisis puede ser necesario añadir periféricos hardware externos o cambiar a otro microcontrolador más adecuado a ese sistema.
- Consumo: algunos productos que incorporan microcontroladores están alimentados con baterías y su funcionamiento puede ser tan vital como activar una alarma antirrobo. Lo más conveniente en un caso como éste puede ser que el microcontrolador esté en estado de bajo consumo pero que despierte ante la activación de una señal (una interrupción) y ejecute el programa adecuado para procesarla.
- Memoria: para detectar las necesidades de memoria de nuestra aplicación debemos separarla en memoria volátil (RAM), memoria no volátil (ROM, EPROM, etc.) y memoria no volátil modificable (EEPROM). Este último tipo de memoria

puede ser útil para incluir información específica de la aplicación como un número de serie o parámetros de calibración.

- Ancho de palabra: el criterio de diseño debe ser seleccionar el microcontrolador de menor ancho de palabra que satisfaga los requerimientos de la aplicación. Usar un microcontrolador de 4 bits supondrá una reducción en los costes importante, mientras que uno de 8 bits puede ser el más adecuado si el ancho de los datos es de un byte. Los microcontroladores de 16 y 32 bits, debido a su elevado coste, deben reservarse para aplicaciones que requieran sus altas prestaciones (Entrada/Salida potente o espacio de direccionamiento mayor).
- Diseño de la placa: la selección de un microcontrolador concreto condicionará el diseño de la placa de circuitos. Debe tenerse en cuenta que quizá usar un microcontrolador barato encarezca el resto de componentes del diseño.
- Uno de los factores que más importancia tiene a la hora de seleccionar un microcontrolador entre todos los demás es el soporte tanto software como hardware de que dispone. Un buen conjunto de herramientas de desarrollo puede ser decisivo en la elección, ya que pueden suponer una ayuda inestimable en el desarrollo del proyecto.

Es así que para nuestra selección nos basaremos en el compilador de C PCWH de CCS que incluye una herramienta de selección de dispositivo la cual nos permite especificar algunos de los criterios antes expuestos tales como:

- El número de pines que vamos a necesitar, calculados según las necesidades de los a dispositivos a conectar(8 pines para el teclado,5 pines para la memoria SD,8 pines para datos y 6 pines de control además de los pines de alimentación y del oscilador)
- El espacio mínimo de memoria ROM y RAM que requiere el programa principal y sus librerías
- El necesitar que posea una EEPROM que permitan darle funcionalidad al dispositivo

La herramienta de selección inicialmente muestra todos los microcontroladores de hasta 16 bits que soporta el programa (425)

Con estos criterios se ha logrado reducir las posibles elecciones a 13 microcontroladores mostrados en la siguiente figura:

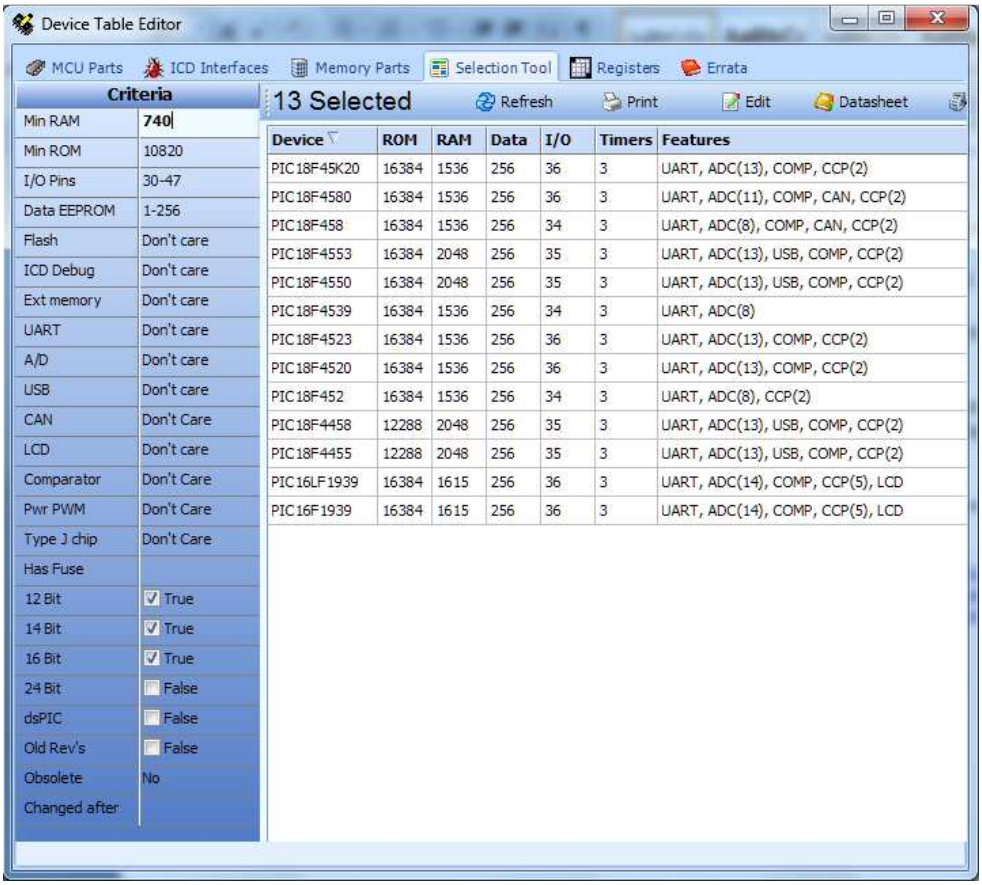


Figura IV - 9: Comparación entre Microcontroladores PIC

Adicionalmente debemos considerar que el microcontrolador cuente con el módulo MSSP para la comunicación SPI, su Package / Case sea DIP(dual in line package) y sudisponibilidad en el mercado nacional.

4.1.2.2CARACTERÍSTICAS DE LOS POSIBLES MICROCONTROLADORES A USAR

Microcontrolador / Parámetros	PIC16F1939 PIC16LF1939	PIC18F4455	PIC18F4458	PIC18F452
ProgramMemoryType	Flash	Flash	Flash	Flash
ProgramMemory (KB)	28	24	24	32
CPU Speed (MIPS)	8	12	12	10
RAM Bytes	1,024	2,048	2,048	1,536
Data EEPROM (bytes)	256	256	256	256
Digital CommunicationPeripherals	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)
Capture/Compare/PWM Peripherals	2 CCP, 3 ECCP	1 CCP, 1 ECCP	1 CCP, 1 ECCP	2 CCP
Timers	4 x 8-bit, 1 x 16-bit	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit
ADC	14 ch, 10-bit	13 ch, 10-bit	13 ch, 12-bit	8 ch, 10-bit
Comparators	2	2	2	
Segment LCD (pixels)	96			
USB (ch, speed, compliance)		1, Full Speed, USB 2.0	1, Full Speed, USB 2.0	
TemperatureRange (C)	-40 a 125	-40 a 85	-40 a 85	-40 a 125
OperatingVoltageRange (V)	1.8 a 5.5/1.8 a 3.6	2 a 5.5	2 a 5.5	2 a 5.5
Pin Count	40	40	40	40
XLP	Yes			
CapTouchChannels	16			
Disponibilidad	No	Si	Si	Si

Microcontrolador / Parámetros	PIC18F4520	PIC18F4523	PIC18F4539	PIC18F4550
ProgramMemoryType	Flash	Flash	Value	Flash
ProgramMemory (KB)	32	32	Flash	32
CPU Speed (MIPS)	10	10	24	12
RAM Bytes	1,536	1,536	10	2,048
Data EEPROM (bytes)	256	256	1,408	256
Digital CommunicationPeripherals	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)	256	1-A/E/USART, 1-MSSP(SPI/I2C)
Capture/Compare/PWM Peripherals	1 CCP, 1 ECCP	1 CCP, 1 ECCP	1-A/E/USART, 1-MSSP(SPI/I2C)	1 CCP, 1 ECCP
Timers	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit
ADC	13 ch, 10-bit	13 ch, 12-bit	8 ch, 10-bit	13 ch, 10-bit
Comparators	2	2		2
Segment LCD (pixels)				
USB (ch, speed, compliance)				1, Full Speed, USB 2.0
TemperatureRange (C)	-40 a 125	-40 a 125	-40 a 125	-40 a 85
OperatingVoltageRange (V)	2 a 5.5	2 a 5.5	2 a 5.5	2 a 5.5
Pin Count	40	40	40	40
XLP				
CapTouchChannels				
Disponibilidad	Si	No	No	Si

Microcontrolador Parámetros	PIC18F4553	PIC18F458	PIC18F4580	PIC18F45K20
ProgramMemoryType	Flash		Flash	Flash
ProgramMemory (KB)	32	32	32	32
CPU Speed (MIPS)	12	10	10	16
RAM Bytes	2,048	1,536	1,536	1,536
Data EEPROM (bytes)	256	256	256	256
Digital CommunicationPeripherals	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)	1-A/E/USART, 1-MSSP(SPI/I2C)
Capture/Compare/PWM Peripherals	1 CCP, 1 ECCP	1 CCP, 1 ECCP	1 CCP, 1 ECCP	1 CCP, 1 ECCP
Timers	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit	1 x 8-bit, 3 x 16-bit
ADC	13 ch, 12-bit	8 ch, 10-bit	11 ch, 10-bit	13 ch, 10-bit
Comparators	2	2	2	2
CAN		1 ECAN	1 ECAN	
USB (ch, speed, compliance)	1, Full Speed, USB 2.0			
TemperatureRange (C)	-40 a 85	-40 a 125	-40 a 125	-40 a 125
OperatingVoltageRange (V)	2 a 5.5	2 a 5.5	2 a 5.5	1.8 a 3.6
Pin Count	40	40	40	40
XLP				Yes
CapTouchChannels				14
Disponibilidad	No	Si	No	No

Los microcontroladores más populares se encuentran, sin duda, entre las mejores elecciones, por lo cual hemos decidido utilizar el microcontrolador**PIC18f4550**

4.2 ANÁLISIS FUNCIONAL

Dado que nuestro dispositivo se basa en la comunicación es necesario entender el funcionamiento del canal de comunicación los pines y registros asociados al modo SPI .

4.2.1 CANAL DE COMUNICACIÓN MSSP

El módulo MSSP o Master Syncronous Serial Port, es una interface serial, muy usada para comunicarse con otros periféricos o microcontroladores. Estos dispositivos pueden ser memorias seriales EEPROM, ShiftRegisters, Displays, conversores ADC. El módulo MSSP puede operar en uno de estos dos modos:

- Serial Peripheral Interface (SPI) o interface serial para periféricos. SPI: Es una Marca Registrada de Motorola Corporation
- Inter-IntegratedCircuit (I2C). I2C: Es una Marca Registrada de Philips

El módulo MSSP posee tres registros de control asociados, SSPSTAT y dos registros de control SSPCON1, SSPCON2. El uso de estos registros y a la configuración individual difieren significativamente dependiendo de cómo va ser usado el módulo MSSP.

En el modo SPI los 8 bits de datos son sincronizados para transmitir o recibir simultáneamente.

Para la comunicación debe existir un dispositivo Maestro (genera la señal de reloj) y uno ó varios esclavos (reciben la señal de reloj)

En los microcontroladores PIC hay 3 pines asociados con la comunicación SPI:

- Salida dato serial (SDO): RC7/RX/DT/SDO;
- Entrada de dato serial (SDI): RB0/AN12/INT0/FLT0/SDI/SDA;
- Reloj serial (SCK): RB1/AN10/INT1/SCK/SCL

De manera adicional, hay un 4º pin que se puede utilizar cuando el microcontrolador se configura como dispositivo Esclavo:

Selección de Esclavo (SS): RA5/AN4/SS/HLVDIN/C2OUT

• La inicialización del Interface SPI se realiza mediante la configuración de los bits de control SSPCON<5:0> y SSPSTAT<7:6>

4.2.2 DIAGRAMA DE BLOQUES DEL MÓDULO SSP EN MODO SPI

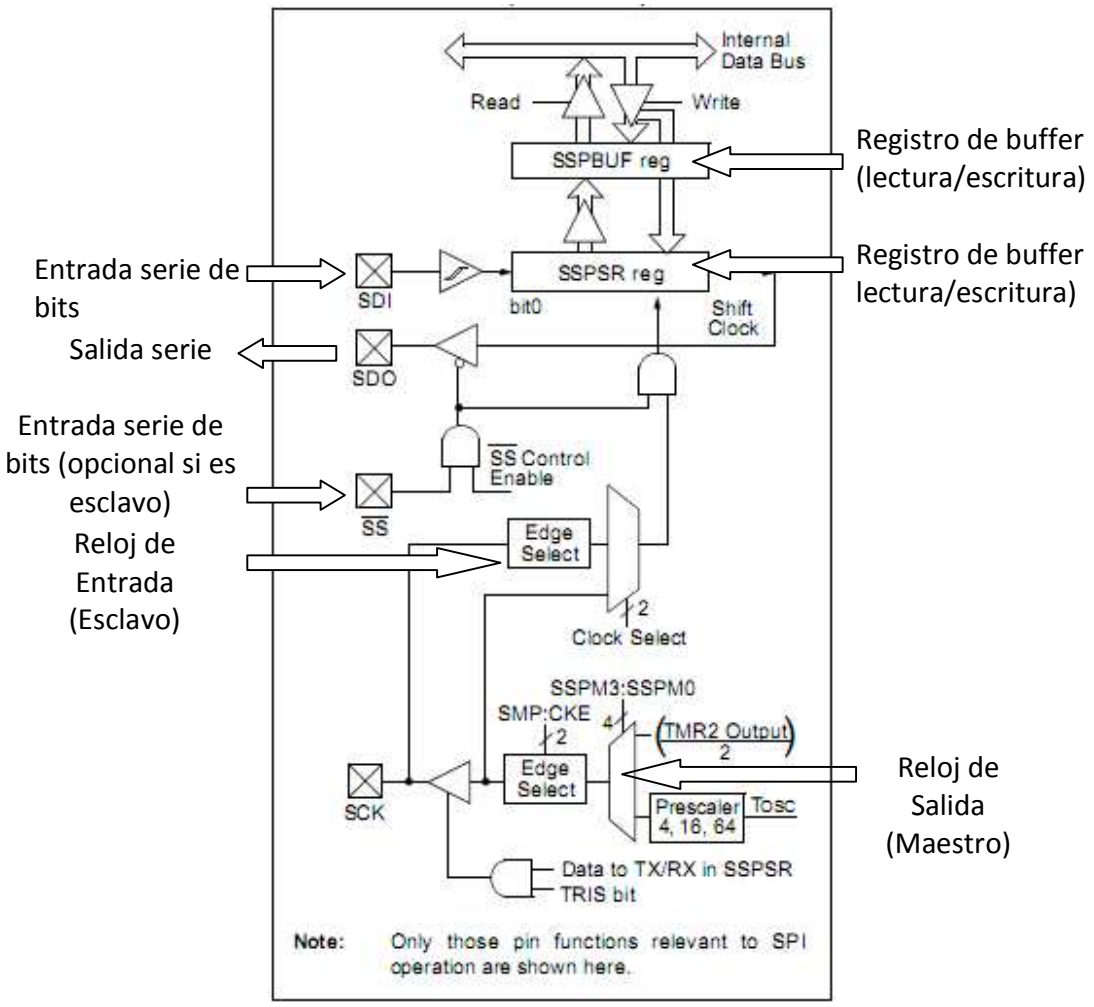


Figura IV - 10: Modo SPI (data sheet)

4.2.3 REGISTROS

El módulo MSSP tiene cuatro registros para la operación en modo SPI. Estos son:

- registro de control: MSSP Control Register 1 (SSPCON1)
- registro de estado: MSSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer Register (SSPBUF)
- Registro de desplazamiento MSSP ShiftRegister (SSPSR) – no directamente accesible

SSPCON1 y SSPSTAT son los registros decontrolyestado para la operación en modoSPI. El registro SSPCON1 permite su lectura y escritura. Los seisbits menores del registro SSPSTATsonde sólo lectura los dos restantes son de lectura / escritura.

- SSPSR (registro de desplazamiento) envía y recoge los bits simultáneamente
- SSPBUF tiene doble función: para “cargar” el registro SSPSR para envío de datos y para “recoger” los datos recibidos en SSPSR

Envío/Recepción Simultánea

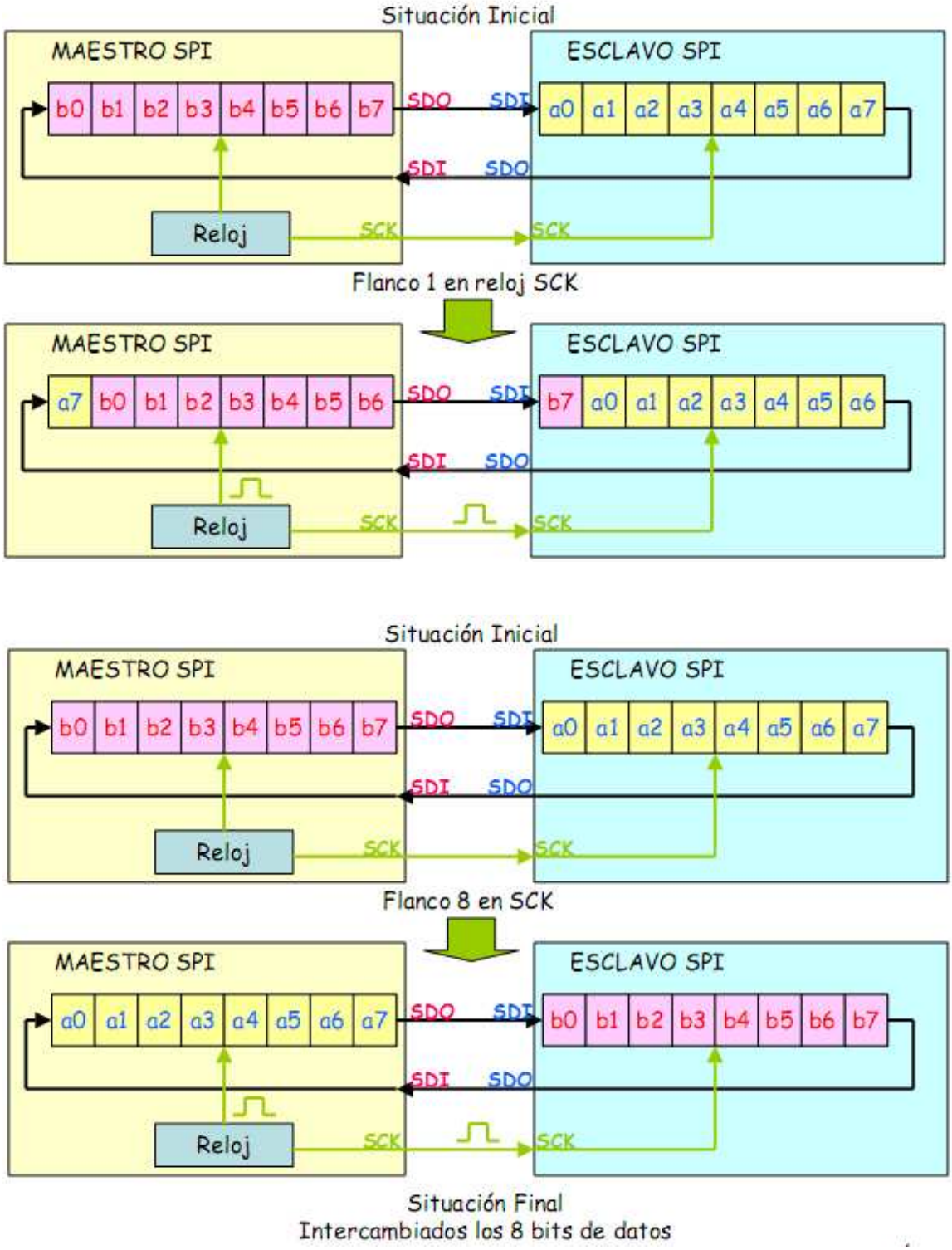


Figura IV - 11: Envió Recepción Simultánea SPI

4.2.4 BANDERAS INDICADORAS EL MODO SPI

SSPIF = PIR1<3> (Completada transmisión) Lectura/escritura.

Indica que se ha completado un envío/recepción en SSPSR, se debe poner a 0 por software, puede generar interrupción

BF = SSPSTAT<0> (Buffer de recepción lleno) Sólo lectura

Se pone a 1 cuando se ha completado la recepción de un dato se pone a 0 por hardware cuando se lee el registro SSPBUF, se empleará normalmente únicamente en modo de recepción

WCOL = SSPCON<7> (Colisión de Escritura) Lectura/escritura

Indica que se ha intentado escribir en SSPBUF mientras se está transmitiendo un dato previo. Si se da tal situación, se debe poner a 0 por software

SSPOV = SSPCON<6> (Desbordamiento en Recepción) Lectura/escritura

Indica que se ha recibido un byte nuevo mientras SSPBUF contiene un dato recibido anteriormente no leído. El nuevo dato se pierde y ya no se actualiza SSPBUF. Se pondrá a 0 por software

El módulo SPI puede utilizarse (independientemente de modo Maestro o Esclavo):

a) Para envío y recepción simultánea (SDI y SDO)

SSPBUF debe leerse antes de cargarlo con el nuevo dato a enviar

b) Sólo para enviar datos (SDO)

SSPBUF se puede cargar con un nuevo dato únicamente después de haberse completado el envío del anterior dato

Si se pretende cargar un dato en SSPBUF durante una transmisión se produce una colisión y dicha carga será ignorada

c) Sólo para recibir datos (SDI)

Sólo en este caso SSPBUF se utilizaría como buffer intermedio de recepción, se puede iniciar la recepción de un nuevo dato antes de leer el dato que se acaba de recibir

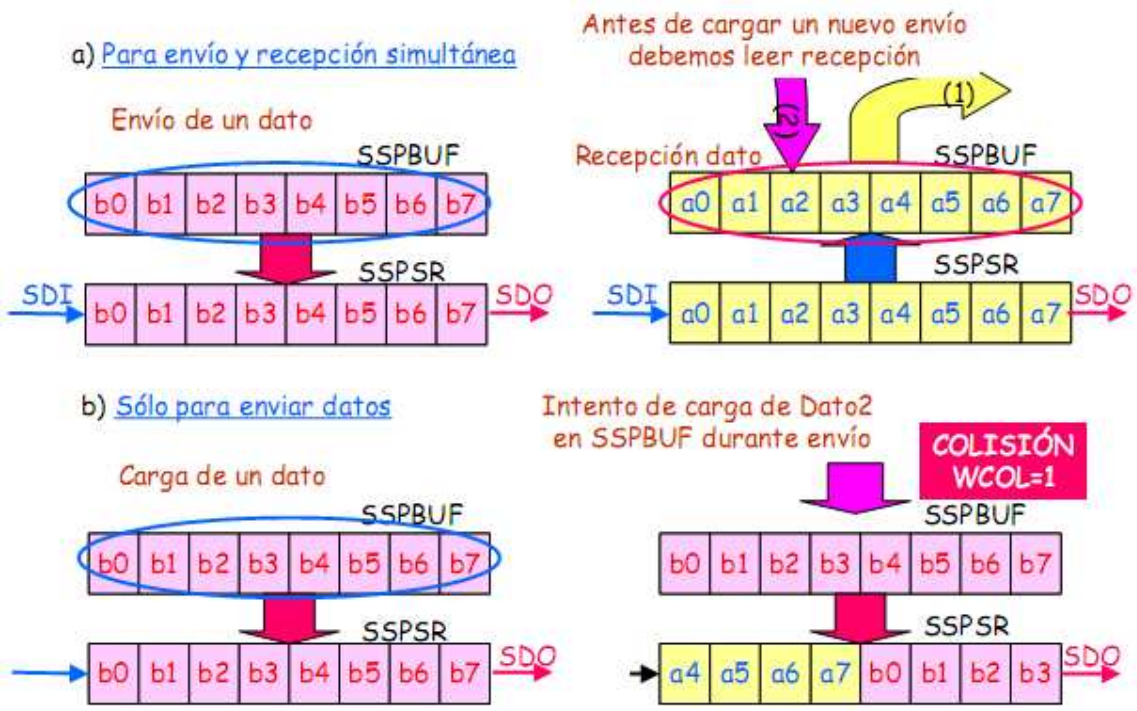


Figura IV - 12: Envío y Recepción Simultánea SSPBUF "continúa"

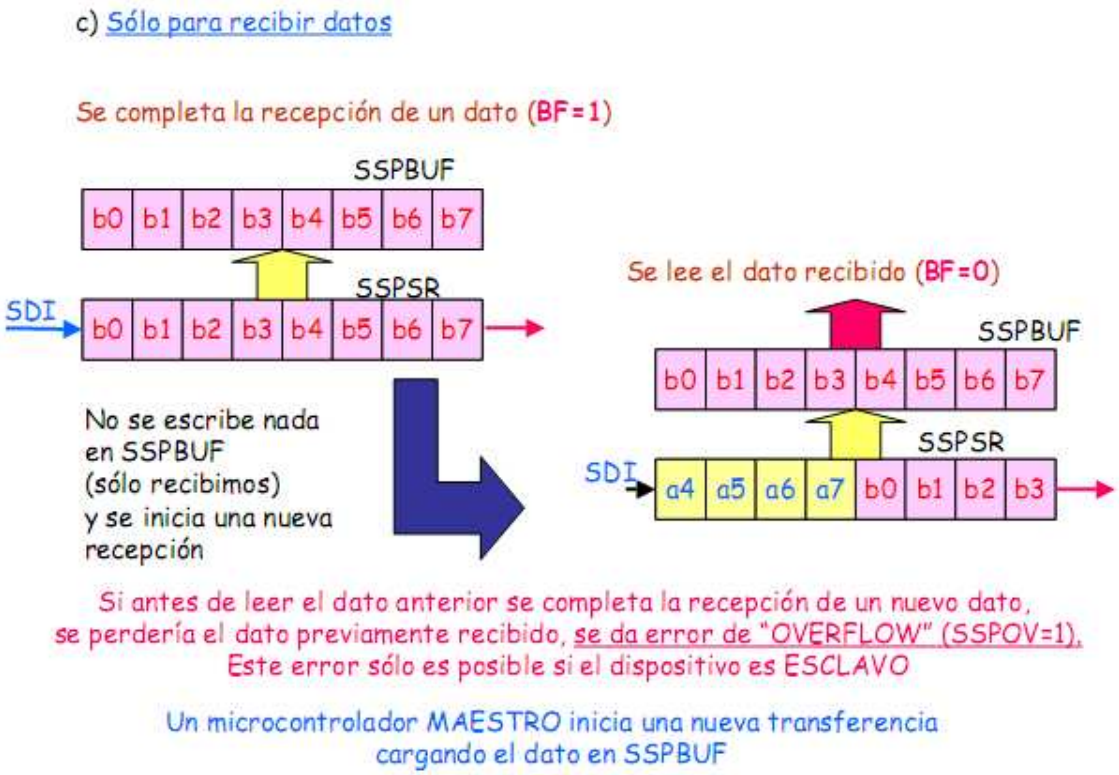


Figura IV - 13: Envío y Recepción Simultánea SSPBUF

4.2.5 ESPECIFICACIÓN DEL MODO SPI EN LA INICIALIZACIÓN

- Modo de funcionamiento: Maestro (SCK salida) o Esclavo (SCK entrada)
- Polaridad del Reloj: Estado Inactivo del Reloj (SCK) a "1" ó a "0"
- Flancos activos del Reloj: Salida de bits en flancos de subida o bajada en SCK
- Muestreo bits de datos: Muestreo de entrada en el "centro" o al "final" del bit
- Frecuencia de Reloj: SÓLO SI ES MAESTRO, frecuencia en salida SCK
- Modo de Selección: SÓLO SI ES ESCLAVO
  - Control externo de SDI y SDO con entrada SS
  - Sin control externo con pin SS

El dispositivo Maestro inicia la transferencia enviando la señal de reloj SCK, los datos salen en los flancos programados y se capturan las entradas en el momento indicado con el bit SMP si es Maestro.

4.2.6 HABILITACIÓN DE LOS PINES DE ENTRADA/SALIDA DE SSP

Para habilitar el módulo SSP, es necesario que el bit SSPEN se encuentre a 1. Si se desea "resetear" el módulo SSP, se debe poner a 0 este bit y luego volverlo a 1

Para que la funcionalidad de los pines SDI, SDO, SCK y /SS sea la determinada por los bits de configuración, es necesario además que los bits de dirección de datos (en TRISA y en TRISC) tengan la dirección adecuada:

SDI (RC4) debe tener TRISC<4> = 1 para ser entrada de datos

SDO (RC5) debe tener TRISC<5> = 0 para que sea salida de datos

SCK (RC3) debe tener TRISC<3> = 0 si el microcontrolador es MAESTRO y TRSC<3> = 1 si se define como ESCLAVO

/SS (RA5) debe ser TRISA<5> = 1 si es ESCLAVO y tiene control externo

4.2.7 FUNCIONAMIENTO COMO MAESTRO

- El Maestro puede iniciar la transferencia en cualquier momento puesto que controla la línea SCK. El protocolo software determinará cuando el Esclavo está en condiciones de enviar datos.
- En modo Maestro, la transmisión/recepción se inicia tan pronto como se escribe en el registro SSPBUF. Si se desea sólo recibir, la línea SDO puede desactivarse

definiéndola como entrada, los datos presentes en la línea SDI irán entrando al registro SSPSR a la velocidad marcada por el reloj de transferencia

- La velocidad de transferencia del modo SPI es programable entre los valores:

  Fosc/4

  Fosc/16

  Fosc/64

  Frecuencia de Salida de TMR2/2 siendo por tanto la máxima velocidad de transferencia: 5Mbps (con Fosc=20MHz)

- La polaridad del reloj y los flancos activos se configuran con los bits CKP y CKE

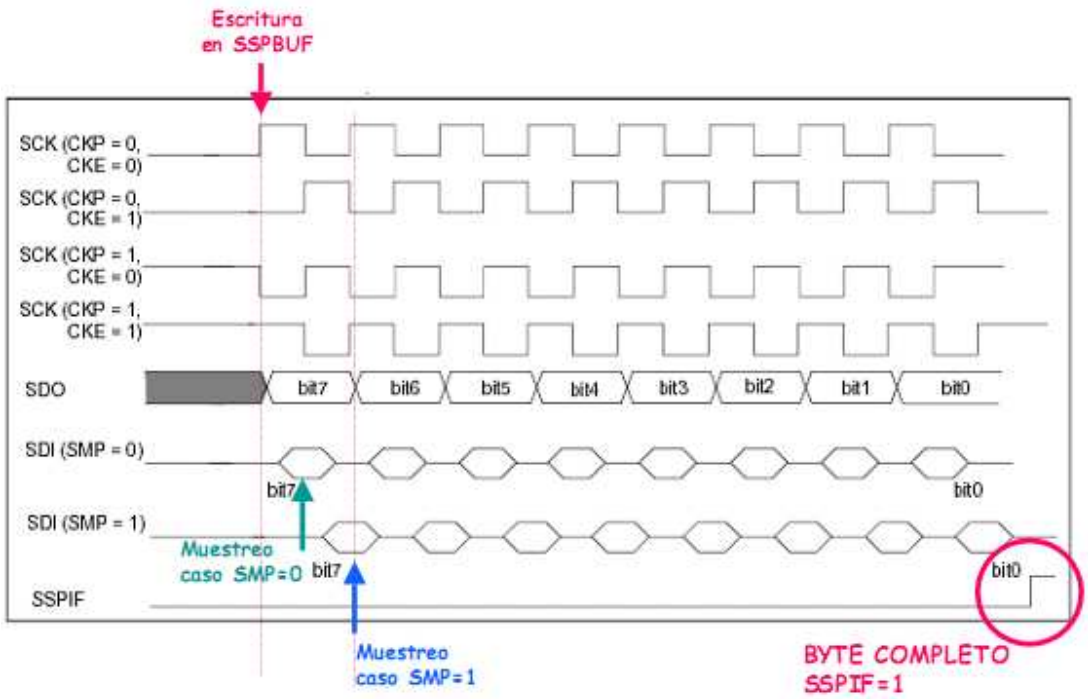


Figura IV - 14: Cronograma SPI maestro

CAPÍTULO V

5.1 DISEÑO ELECTRÓNICO

5.1.1 DESARROLLO DEL HARDWARE

5.1.1.1 ESQUEMA GENERAL

Figura V - 15: Esquema General

En la figura se muestra un diagrama de bloques general del prototipo el cual nos indican las etapas principales que conforman el hardware como se aprecia el usuario tendrá dos interfaces para interactuar con el sistema ya sea a través de software de gestión o del el dispositivo

Las principales características que se quiere para el dispositivo son:

1. Independencia suficiente para realizar los recorridos
2. Proveer una interface amigable para el usuario adecuándose a las necesidades específicas que este tenga
3. Receptar los datos obtenidos de la medición mensual del consumo de agua potable
4. Almacenar dichos datos en un archivo plan con un formato predefinido para que el software de gestión pueda entenderlos y utilizarlos
5. Brindar una gran capacidad de almacenamiento

5.1.1.2 DIAGRAMA DE BLOQUES

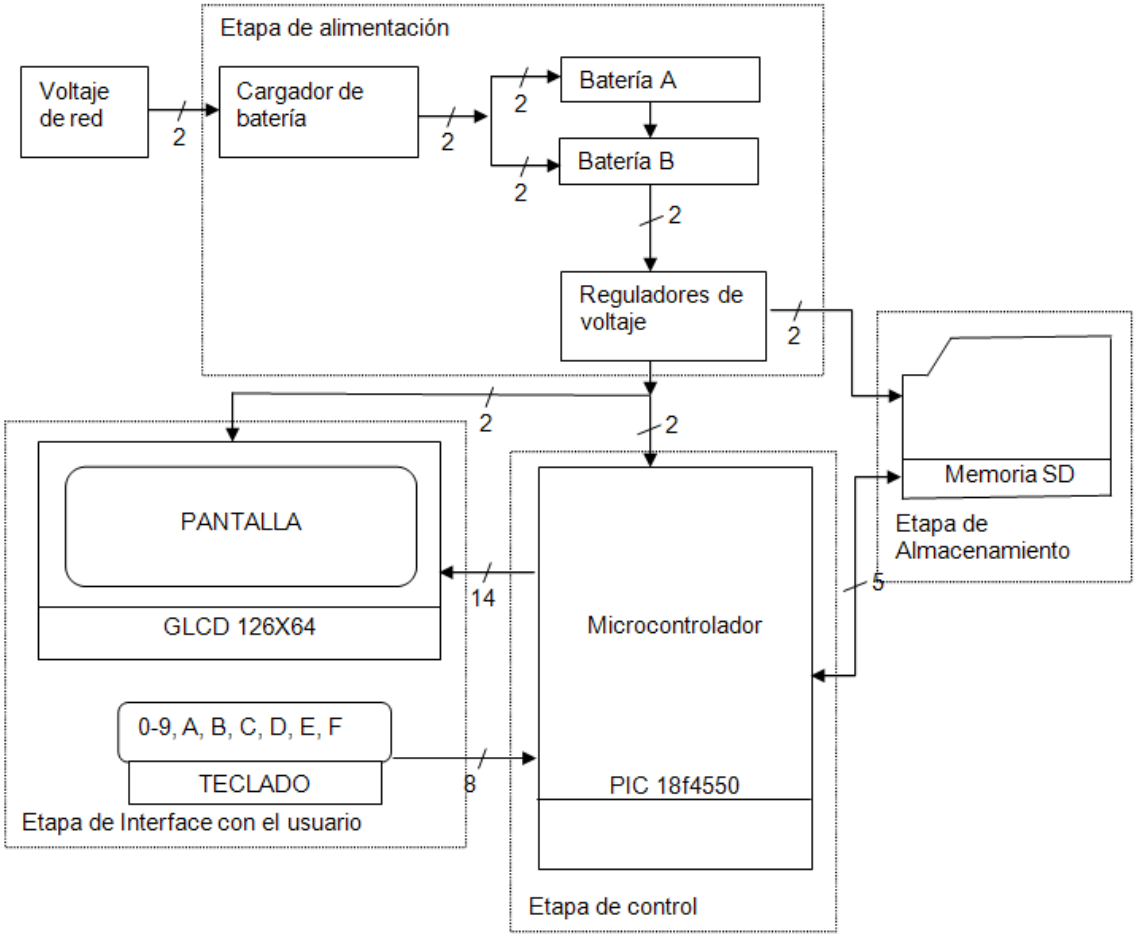


Figura V - 16: Diagrama de Bloques Dispositivo Portable

5.1.1.2.1 ETAPA DE ALIMENTACIÓN

Para alimentar el circuito se ha considerado que:

- Los dispositivos con que vamos a trabajar necesitan 5v a excepción de la memoria SD que trabaja con 3,3v
- Al tratarse de un dispositivo portátil requiere un cierto nivel de autonomía (mayor duración de la batería)

- Estabilidad en los voltajes para evitar perdida de datos o mal funcionamiento

Por lo cual se ha optado por utilizar dos baterías de Li-Ion conectadas en serie nos han permitido obtener un voltaje a plena carga de 8.4V, a carga nominal de 7.4V y a baja carga de 6.6V.

Nivel de voltaje que nos permitirán utilizar un regulador de voltaje positivo LM 7805 para los obtener los 5V y un LM1117 para los 3.3V.

Para cargar las baterías de nuestro dispositivo se optó por un circuito cargador externo recomendado para la marca de nuestras baterías, Al encenderse o al colocar una batería el circuito verifica el estado de carga de la misma y, de ser necesario, efectúa la carga. Una vez completada la carga el circuito entra en modo de espera, controlando periódicamente el estado de la celda por si debe continuar cargando.

El circuito está pensado para una batería con una única celda de Li-Ion. Es importante destacar que este tipo de baterías no pueden ser cargadas ni en serie ni en paralelo, por lo que el proceso de carga se la realiza una batería a la vez

Los dispositivos utilizados son descritos a continuación:

5.1.1.2.2 BATERÍA DE ION DE LITIO

La batería de iones de litio, también denominada batería Li-Ion, es un dispositivo diseñado para almacenamiento de energía eléctrica que emplea como electrolito, una sal de litio

que procura los iones necesarios para la reacción electroquímica reversible que tiene lugar entre el cátodo y el ánodo.

Las propiedades de las baterías de Li-ion, como la ligereza de sus componentes, su elevada capacidad energética y resistencia a la descarga, la ausencia de efecto memoria o su capacidad para operar con un elevado número de ciclos de regeneración, han permitido el diseño de acumuladores livianos, de pequeño tamaño y variadas formas, con un alto rendimiento, especialmente adaptados para las aplicaciones de la industria electrónica de gran consumo. Desde la primera comercialización a principios de los años 1990 de un acumulador basado en la tecnología Li-ion, su uso se ha popularizado en aparatos como teléfonos móviles, agendas electrónicas, ordenadores portátiles y lectores de música.

Sin embargo, su rápida degradación y sensibilidad a las elevadas temperaturas, que pueden resultar en su destrucción por inflamación o incluso explosión, requieren en su configuración como producto de consumo, la inclusión de dispositivos adicionales de seguridad

Los tres participantes en las reacciones electroquímicas en una batería del ion del litio son ánodo, cátodo, y electrólito.

El ánodo y el cátodo son los materiales en de los cuales y de qué litio puede emigrar. El proceso del litio que se mueve en el ánodo o el cátodo se refiere como *inserción* (o *intercalation*), y el proceso reverso, en el cual el litio se mueve del ánodo o del cátodo se refiere como *extracción* (o *deintercalation*). Cuando es una célula el descargar, el litio se

extrae del ánodo y se inserta en el cátodo. Cuando es la célula el cargar, el proceso reverso ocurre: el litio se extrae del cátodo y se inserta en el ánodo.

Cátodos

Material	Voltaje medio	Capacidad gravimétrica
$LiCoO_2$	3.7 V	140 mAh/g
$LiMnO_2$	4.0 V	100 mAh/g
$LiFePO_4$	3.3 V	170 mAh/g
Li_2FePO_4F	3.6 V	115 mAh/g

Tabla V - VIII: Cátodos de las diferentes Baterías

Voltaje proporcionado:

- A Plena carga: Entre 4.2V y 4.3V dependiendo del fabricante

- A carga nominal: Entre 3.6V y 3.7V dependiendo del fabricante

- A baja carga: Entre 2,65V y 2,75V dependiendo del fabricante (este valor no es un límite, se recomienda).

Ventajas

- Una elevada densidad de energía: Acumulan mucha mayor carga por unidad de peso y volumen.

- Poco peso: A igualdad de carga almacenada, son menos pesadas y ocupan menos volumen que las de tipo Ni-MH y mucho menos que las de Ni-Cd y Plomo.
- Gran capacidad de descarga. Algunas baterías de Li-Ión -las llamadas "Lipo" Litio-ión Polímero- que hay en el mercado, se pueden descargar totalmente en menos de dos minutos.
- Poco espesor: Se presentan en placas rectangulares, con menos de 5 mm de espesor. Esto las hace especialmente interesantes para integrarlas en dispositivos portátiles que deben tener poco espesor.
- Alto voltaje por célula: Cada batería proporciona 3,7 voltios, lo mismo que tres baterías de Ni-MH o Ni-Cd (1,2 V cada una).
- Carecen de efecto memoria.
- El efecto memoria es un fenómeno que reduce la capacidad de las baterías con cargas incompletas. Se produce cuando se carga una batería sin haber sido descargada del todo: se crean unos cristales en el interior de estas baterías, a causa de una reacción química al calentarse la batería, bien por uso o por las malas cargas
- Descarga lineal: Durante toda la descarga, el voltaje de la batería varía poco, lo que evita la necesidad de circuitos reguladores. Esto es una ventaja, ya que hace muy fácil saber la carga que almacena la batería.

- Larga vida en las baterías profesionales para vehículos eléctricos. Algunos fabricantes muestran datos de más de 3.000 ciclos de carga/descarga para una pérdida de capacidad del 20% a C/3.
- Facilidad para saber la carga que almacenan. Basta con medir, en reposo, el voltaje de la batería. La energía almacenada es una función del voltaje medido.
- Muy baja tasa de auto descarga: Cuando guardamos una batería, ésta se descarga progresivamente aunque no la usemos. En el caso de las baterías de Ni-MH, esta "auto descarga" puede suponer más de un 20% mensual. En el caso de Li-Ion es de menos un 6% en el mismo periodo. Mucha de ellas, tras seis meses en reposo, pueden retener un 80% de su carga.

Inconvenientes

A pesar de todas sus ventajas, esta tecnología no es el sistema perfecto para el almacenaje de energía, pues tiene varios defectos, como pueden ser:

- Duración media: Depende de la cantidad de carga que almacenen, independientemente de su uso. Tienen una vida útil de unos 3 años o más si se almacenan con un 40% de su carga máxima (En realidad, cualquier batería, independientemente de su tecnología, si se almacena sin carga se deteriora. Basta con recordar el proceso de sulfatación que ocurría en las antiguas baterías de zinc-carbón cuando se almacenaban al descargarse completamente).

- Soportan un número limitado de cargas: entre 300 y 1000, menos que una batería de Ni-Cd e igual que las de Ni-MH, por lo que hoy día ya empiezan a ser consideradas en la categoría de consumibles.
- Son costosas: Su fabricación es más costosa que las de Ni-Cd e igual que las de de Ni-MH, si bien actualmente el precio baja rápidamente debido a su gran penetración en el mercado, con el consiguiente abaratamiento. Podemos decir que se utilizan en todos los teléfonos móviles y ordenadores portátiles del mundo y continúa extendiendo su uso a todo tipo de herramientas portátiles de baja potencia.
- Pueden sobrecalentarse hasta el punto de explotar: Están fabricadas con materiales inflamables que las hace propensas a detonaciones o incendios, por lo que es necesario dotarlas de circuitos electrónicos que controlen en todo momento la batería.
- Peor capacidad de trabajo en frío: Ofrecen un rendimiento inferior a las baterías de Ni-Cd o Ni-MH a bajas temperaturas, reduciendo su duración hasta en un 25%.

5.1.1.2.3 REGULADORES DE VOLTAJE

Existen muchas maneras de lograr un voltaje estable, pero en general utilizan varios componentes discretos, lo que redunda en un costo elevado, un diseño más complicado, y circuitos más grandes. La alternativa es utilizar algún regulador de tensión integrado,

disponibles para casi todos los voltajes que podamos imaginar, y para corrientes desde unas pocas centésimas de Amper hasta varios amperes.

LM7805

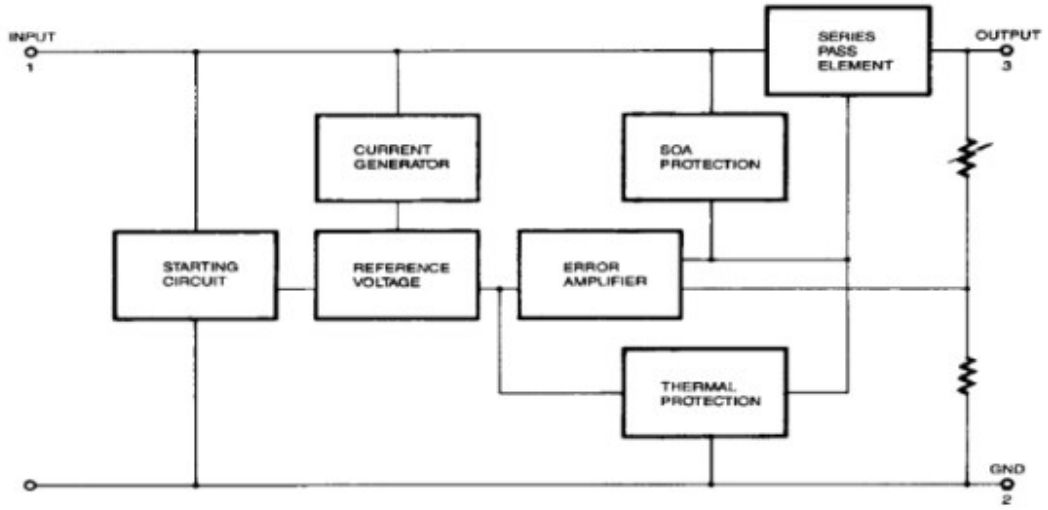


Figura V - 17: Diagrama interno de un regulador 78xx.

Dentro de los reguladores de voltaje con salida fija, se encuentran los pertenecientes a la familia LM78xx, donde “xx” es el voltaje de la salida. Estos son5, 6, 8, 9, 10, 12, 15, 18 y 24V, entregando una corriente máxima de 1 Amperios y soporta consumos pico de hasta 2.2 Amperios. Poseen protección contra sobrecargas térmicas y contra cortocircuitos, que desconectan el regulador en caso de que su temperatura de juntura supere los 125°C.

La capsula que los contiene es una TO-220 (figura V-4), igual a la de muchos transistores de mediana potencia. Para alcanzar la corriente máxima (1 Amperios) es necesario dotarlo

de un disipador de calor adecuado, sin el solo obtendremos una fracción de esta corriente antes de que el regulador alcance su temperatura máxima y se desconecte.

La potencia además depende de la tensión de entrada, por ejemplo, si tenemos un LM7805, cuya tensión de salida es de 5V, con una tensión de entrada de 9v, y una carga en su salida de 0,5A, multiplicando la diferencia entre la tensión de entrada y la tensión de salida por la corriente que circulará por la carga nos da los vatios que va a tener que soportar el integrado:

$$(Vint - Vout) \times Iout = (9 - 5) \times 0.5 = 2W$$

La tensión de entrada es un factor muy importante, ya que debe ser superior en unos 3 voltios a la tensión de salida (es el mínimo recomendado por el fabricante), pero todo el exceso debe ser eliminado en forma de calor. Si en el ejemplo anterior en lugar de entrar con 9 volts solo usamos 8V (los 5V de la salida más el margen de 3V sugerido) la potencia disipada es mucho menor:

$$(Vint - Vout) \times Iout = (8 - 5) \times 0.5 = 1.5W$$

De hecho, con 8v la carga del integrado es de 1,5W, menos que con 9v, por lo que el calor generado será mucho menor y en consecuencia el disipador necesario también menor.

En la figura V-5 vemos la disposición de pines de este reguladores. El pin 1 corresponde a la entrada (input), el pin 2 es el punto común (common) y el pin3 es el correspondiente a la salida (output).

El voltaje máximo que soportan en la entrada es de 35 voltios para los modelos del LM7805 al 7815.

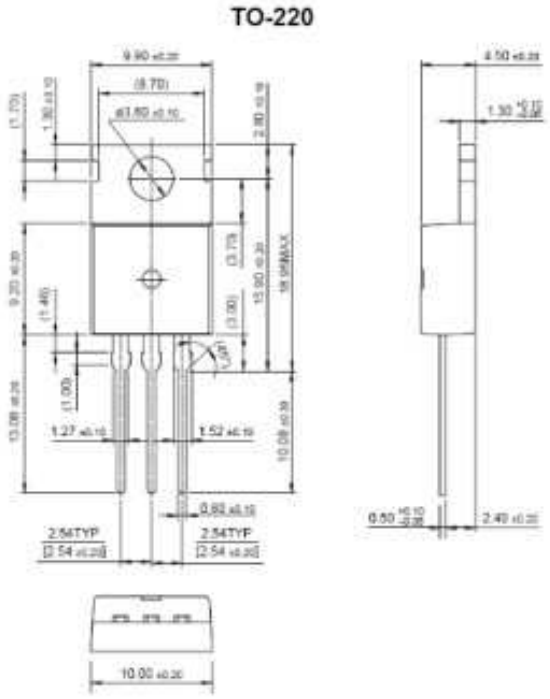


Figura V - 18: Integrado TO220

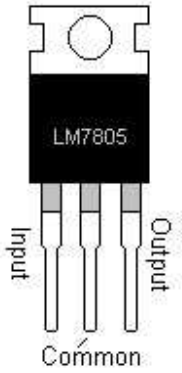


Figura V - 19: Integrado LM7805

LM1117

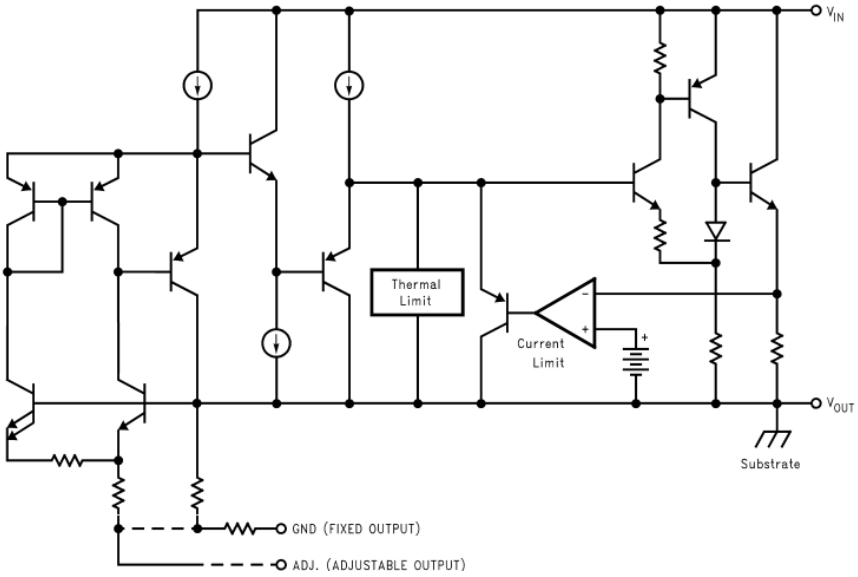


Figura V - 20: Circuito Interno del Regulador LM1117

Dentro de los reguladores de voltaje con salida fija, tenemos al LM1117, disponible en 1.8V, 2.5V, 2.85V ,3.3V ,5V, y regulable, entregando una corriente máxima de 800 miliamperios y soporta consumos pico de hasta 1500 miliamperios. Poseen protección contra sobrecargas térmicas y contra cortocircuitos, que desconectan el regulador en caso de que su temperatura de juntura supere los 125°C.

La capsula que los contiene es una TO-220 (figura V-7), igual a la de muchos transistores de mediana potencia. Para alcanzar la corriente máxima de 800 miliamperio es necesario dotarlo de un disipador de calor adecuado, sin el solo obtendremos una fracción de esta corriente antes de que el regulador alcance su temperatura máxima y se desconecte.

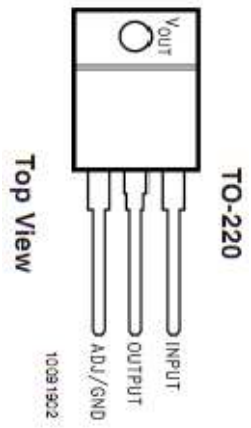


Figura V - 21: Circuito integrado LM1117

5.1.1.2.4 ETAPA DE ALMACENAMIENTO

Para almacenar los datos se utiliza una memoria SD de 2GB por lo expuesto en los capítulos iníciales utilizando para ello un socket SD que consta de de 9 pines de comunicación correspondientes a los 9 pines de la memoria SD y un pin conectado a un interruptor o switch que nos permite detectar es insertada una memoria el socket no dispone de mecanismo de expulsión.

La conexión con el micro controlador PIC18F4550 está basada en la tabla 1-3 del Data Sheet en la cual se especifica los pines que permiten la comunicación SPI y explicado con mayor detalle en el capítulo IV del presente trabajo.

Dado que la SD trabaja con 3.3v y el PIC18F4550 con 5V las líneas de entrada a la memoria deben pasan por resistencias para obtener niveles de voltaje adecuados.

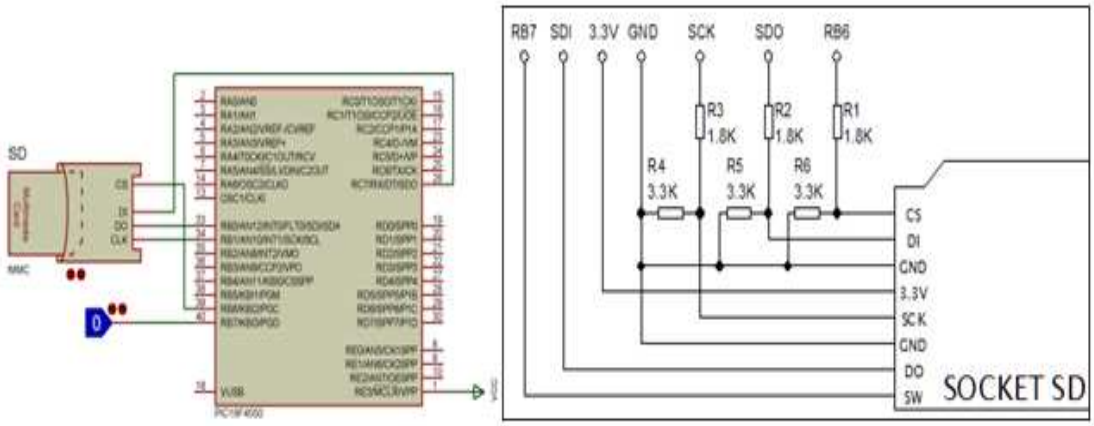


Figura V - 22: Diagrama de conexión de la memoria SD al PIC 18F4550

5.1.1.2.5 MICROCONTROLADOR PIC18F4550

Para el prototipo se decidió utilizar el micro controlador PIC18F4550 de Microchip por sus características

Figura V - 23: PIC18F4550.

El PIC18F4550 es un micro ampliamente utilizado como un microcontrolador "estándar" debido a sus innumerables características y potencia, hay que decir que tiene incluido una memoria Flash USB y control de flujo de datos. Soporta SPI. Posee una arquitectura RISC (reducedinstruction set computer) de 16 bits longitud de instrucciones y 8 bits de datos.

El PIC 18F4550 se caracteriza por:

Memoria Flash:	32Kbytes
Máximo número de instrucciones simples:	16384
Memoria SRAM:	2048 bytes
Memoria EEPROM:	256 bytes
Entradas / Salidas:	35
Número de entradas A/D:	13
Número de CCP:	1
Número de ECCP:	1
Soporta SPP:	Si
Soporta SPI:	Si
Soporta master I2C:	Si
Número de EAUSART:	1
Número de comparadores:	2
Número de temporizadores de 8 bits:	1
Número de temporizadores de 16 bits:	3
Universal Serial Bus (USB) module:	Si

Tabla V - IX: Características del PIC 18F4550

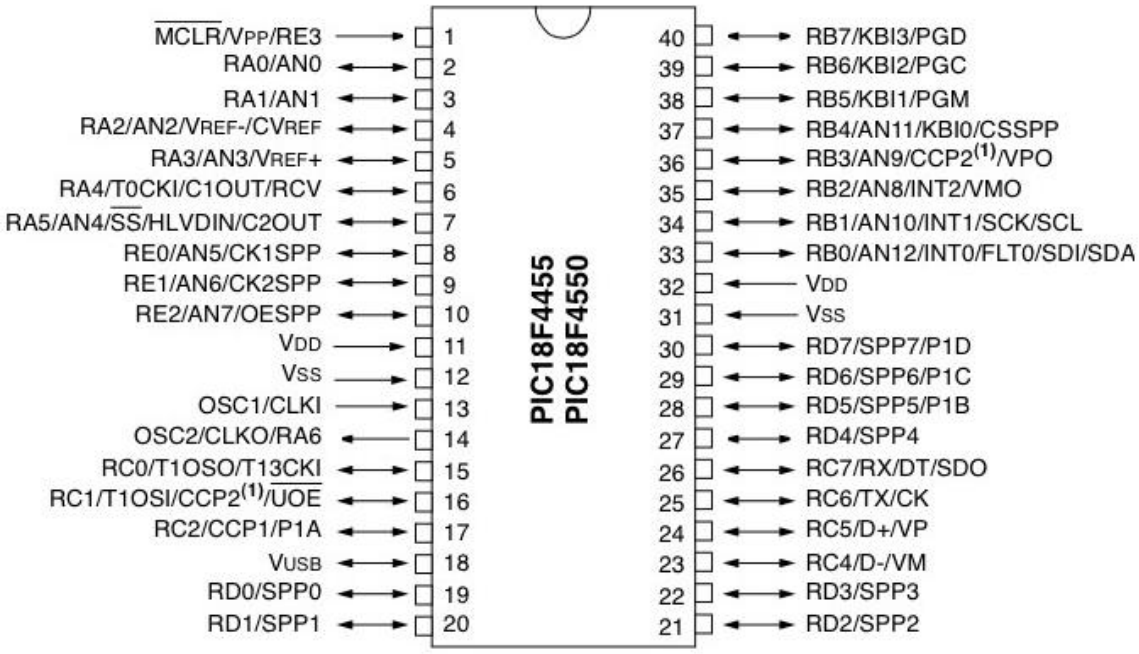


Figura V - 24: Descripción de Pines PIC 18F4550

CONFIGURACIÓN DE PINES DEL MICRO CONTROLADOR PIC 18F4550 PARA EL PROYECTO

PIC 18F4550		Elemento que está conectado	
# PIN	Nombre / Descripción	Nombre / Descripción	# PIN
1	MCLR/RE3	5V DC	
2	RA0	GLCD/RST	17
3	RA1	GLCD/E	6
4	RA2	GLCD/RW	5
5	RA3	GLCD/DI	4
6	RA4	NC	
7	RA5	NC	

8	RE0	GLCD/CS2	16
9	RE1	GLCD/CS1	15
10	RE2	NC	
11	VDD	5V DC	
12	VSS	GND	
13	OSC1/CLKI	Cristal oscilador 20Mhz	
14	OSC2/CLK0	Cristal oscilador 20Mhz	
15	RC0	TECLADO/C	7
16	RC1	TECLADO/B	6
17	RC2	TECLADO/A	5
18	Vusb	NC	
19	RD0	GLCD/DB0	7
20	RD1	GLCD/DB1	8
21	RD2	GLCD/DB2	9
22	RD3	GLCD/DB3	10
23	RC4	NC	
24	RC5	NC	
25	RC6	TECLADO/D	8
26	RC7/SDO	SD/DI	2
27	RD4	GLCD/DB4	11

28	RD5	GLCD/DB5	12
29	RD6	GLCD/DB6	13
30	RD7	GLCD/DB7	14
31	VSS	GND	
32	VDD	5V DC	
33	RB0/SDI	SD/DO	7
34	RB1/SCK	SD/CLK	5
35	RB2	TECLADO/4	4
36	RB3	TECLADO/3	3
37	RB4	TECLADO/2	2
38	RB5	TECLADO/1	1
39	RB6	SD/CS	1
40	RB7	SD/SWITCH	8

Tabla V - X: Configuración de pines del PIC 18f4550 para el proyecto

5.1.1.2.6 ETAPA DE INTERFACE CON EL USUARIO

Para esta etapa se ha dispuesto utilizar

- Un GLCD para la visualización,
- Un teclado 4X4 para ingreso de datos y manejo de Menús

GLCD (GraphicLiquid Cristal Display) CON CONTROLADOR GRÁFICO KS0108

Para presentar la información se ha utilizado un LCD gráfico de 128 x 64 pixeles con el controlador KS0108.

Las principales características de este GLCD son:

- 128 x 64 pixeles.
- 2 controladores gráficos KS0108 (cada uno controla una mitad de la pantalla).
- Tiempo de vida mínimo: 100 000 horas.
- 1024 bytes de memoria.

Un diagrama de bloques del circuito interno se muestra en la Figura, mientras que en la Tabla se indican sus pines y su función.

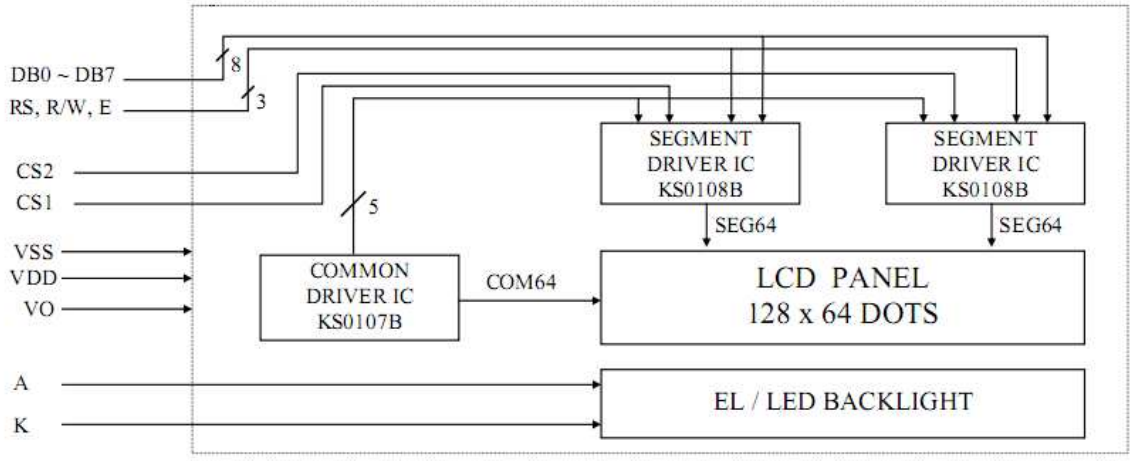


Figura V - 25: Diagrama de bloques del circuito interno del GLCD.

CONFIGURACIÓN DE PINES DEL GLCD 128X64 WINSTAR

PIN	SÍMBOLO	NIVEL	DESCRIPCIÓN
1	VSS	0 V	Tierra
2	VDD	5 V	V (+)
3	VO	-	Voltaje de entrada GLCD
4	RS	H/L H:	Señal de datos. L: señal de instrucción
5	R/W	H/L H:	modo lectura. L: modo escritura
6	E	H, H→L	Chip Enable
7	DB0	H/L	Bit de datos 0
8	DB1	H/L	Bit de datos 1
9	DB2	H/L	Bit de datos 2
10	DB3	H/L	Bit de datos 3
11	DB4	H/L	Bit de datos 4
12	DB5	H/L	Bit de datos 5
13	DB6	H/L	Bit de datos 6
14	DB7	H/L	Bit de datos 7
15	CS1	H	Señal chip select para controlador 1
16	CS2	H	Señal chip select para controlador 2
17	/RES	L	Reset
18	Vee	-9 V	Polarización negativa LCD
19	LED-A	4.2 V	Blacklightánodo
20	LED-K	0 V	Blacklightcátodo

Tabla V - XI:Configuración de pines GLCD

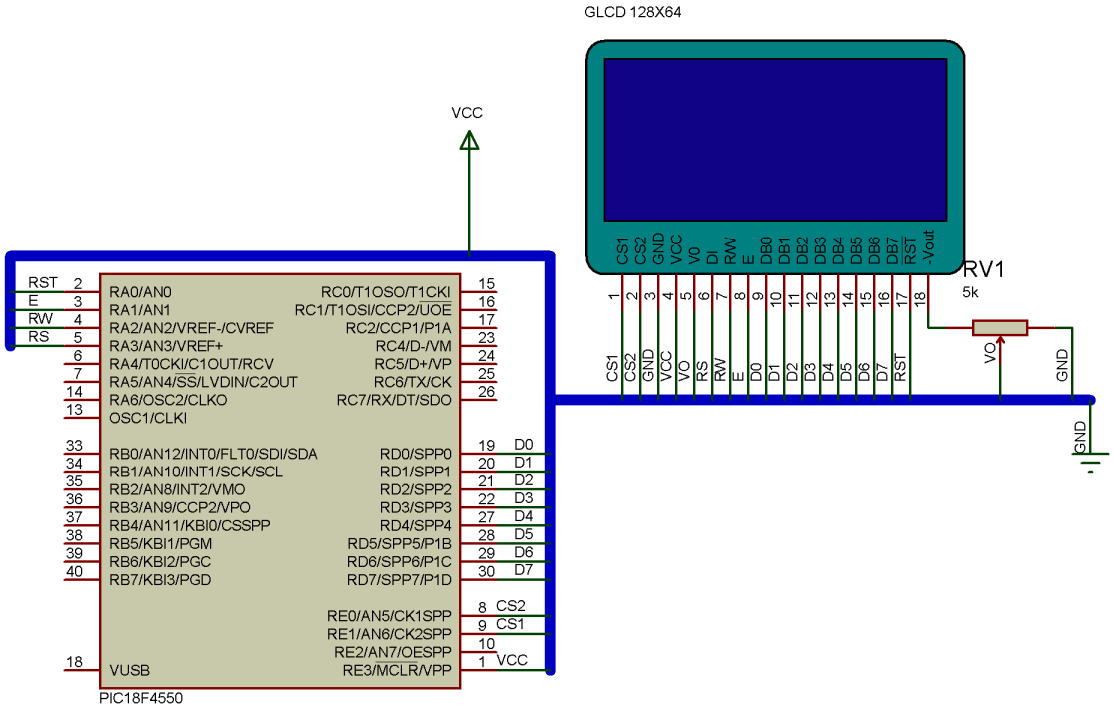


Figura V - 26: Circuito de la etapa de visualización de datos

Teclado 4x4

Configuración del panel de control. Como se puede ver en la Figura V-11. Se asignó solo ocho pines del microcontrolador para dieciséis botones del panel de control, un pin por cada columna de botones y un pin por cada fila de botones, de modo que cada botón del panel se identificará por la posición en la que se encuentra con respecto a filas y columnas, y adquirirá como nombre un número del 0 al 9 y una letra mayúscula entre A y F que será lo que el microcontrolador reconozca. Por ejemplo, si se presiona la tecla correspondiente a la posición Fila A y Columna 2, a la cual se le asignó el nombre “8”, que

es lo que reconocerá el microcontrolador y ejecutará la acción. Esta es una técnica conocida como Barrido de Teclas.

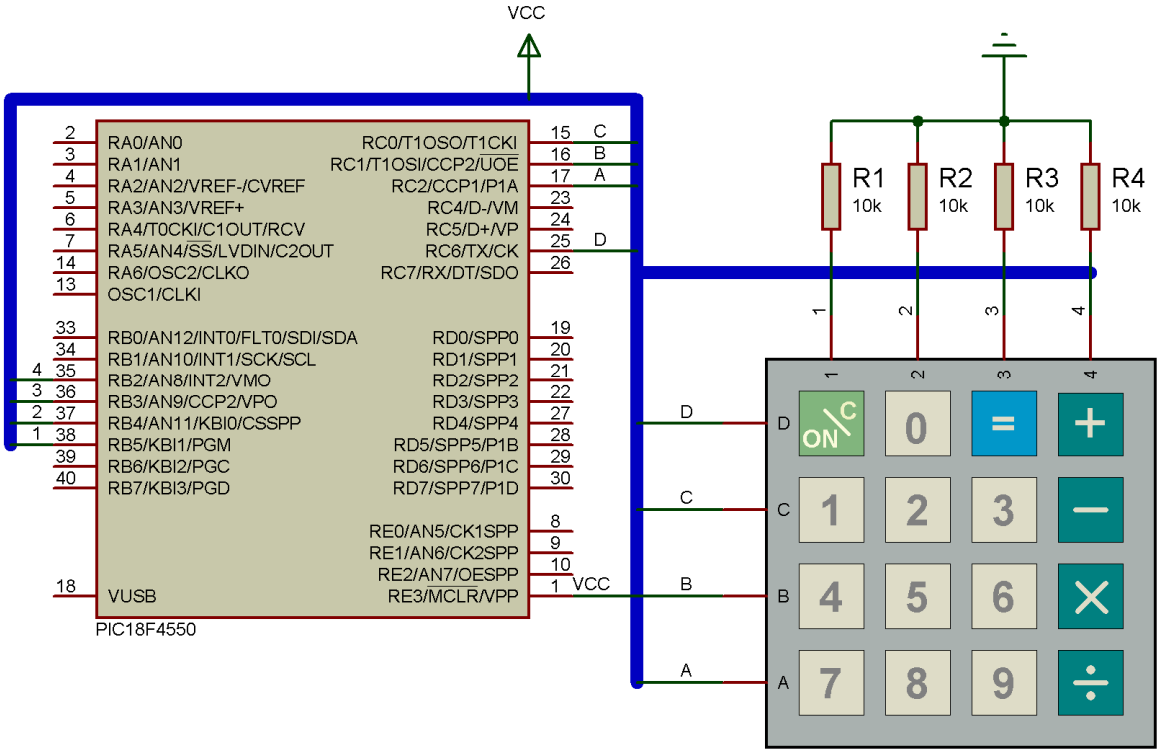


Figura V - 27: Conexión del Teclado al PIC 18f4550

5.1.1.3 LISTADO DE DISPOSITIVOS

- Micro controlador PIC 18F4550
- Teclado 4x4
- GLCD 128x64 WINSTAR
- Dos baterías de Li-Ion
- Regulador De Voltaje Positivo a 5V LM 7805

- Regulador De Voltaje Positivo a 3.3V LM1117
- Slot para inserción de la tarjeta de memoria SD-Card
- Tres resistencias de 1.8k
- Tres resistencias de 3.3k
- Cuatro resistencias de 10k
- Potenciómetro de 5k
- Oscilador de 20mhz
- Dos capacitores de 15pf
- Dos capacitores de 10pf

5.2 DIAGRAMA DEL CIRCUITO GENERAL

La siguiente figura muestra el circuito esquematizo final completo del hardware del prototipo:

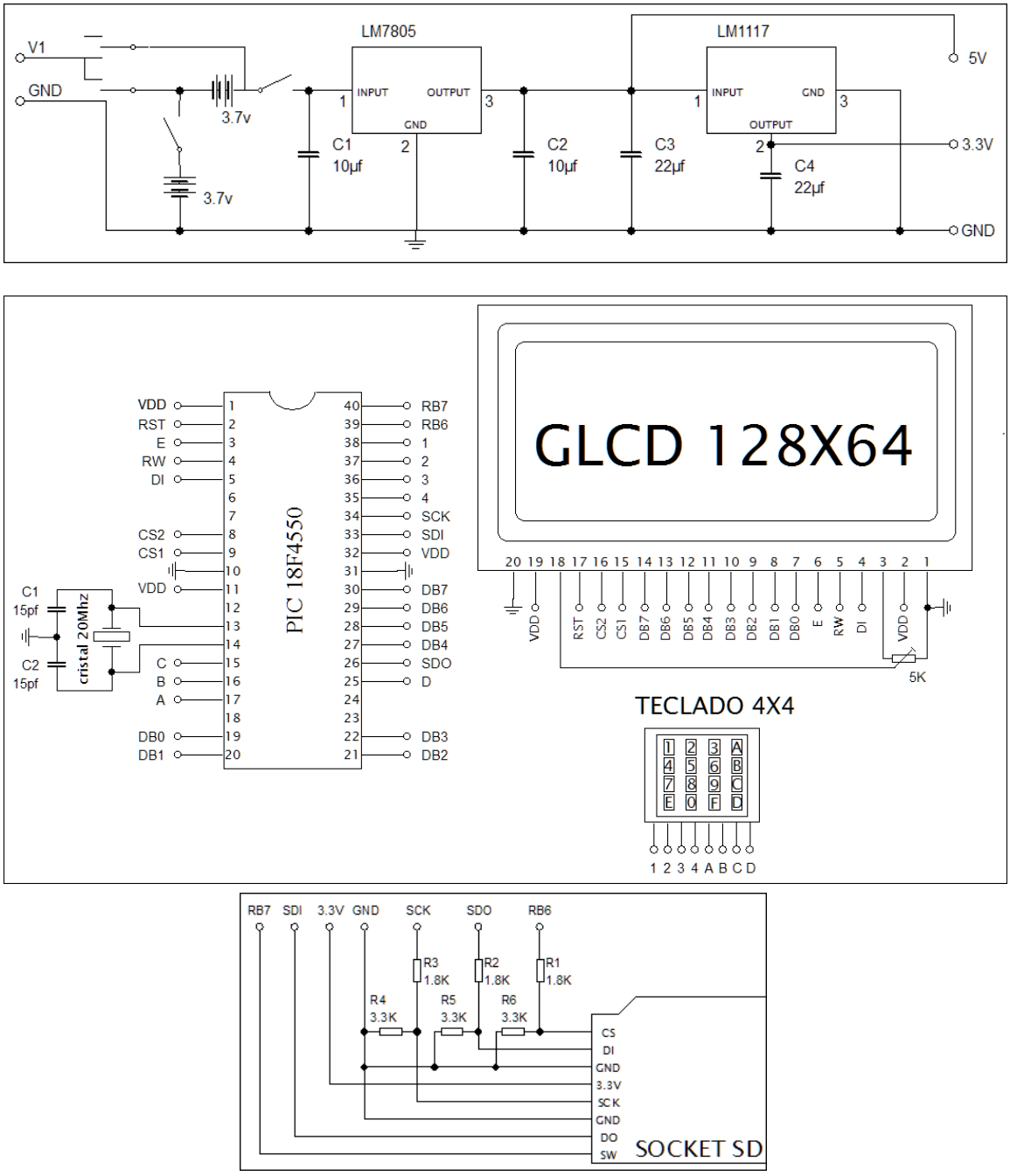


Figura V - 28: Diagrama del Circuito General

5.3 CONSTRUCCIÓN DE LA PLACA DE CIRCUITO IMPRESO

Mediante PROTEUS ARES, o AdvancedRouting and EditingSoftware (*Software de Edición y Ruteo Avanzado*); es la herramienta de enrutado, ubicación y edición de componentes, se utiliza para la fabricación de placas de circuito impreso, permitiendo editar las diferentes capas generalmente, las capas superficial (Top Copper), y de soldadura (BottomCopper).

Se ha creado el circuito impreso del dispositivo utilizando un trazado manual de las rutas al no contar con el empaquetado de la memoria SD ni del GLCD

Para tener una mejor visualización al momento del trazado se optó por asignar una capa diferente tanto a tierra como a los diferentes niveles de voltaje presentes en el diseño garantizando así la correcta alimentación de los distintos dispositivos

Además se debe considerar las dimensiones del circuito en relación a la carcasa que lo va a contener puesto que se ha optado por utilizar una carcasa existente de dimensiones fijas (11x13.5x3 cm) al no disponer de los materiales necesarios para elaborar una carcasa de iguales acabado a la medida del diseño.

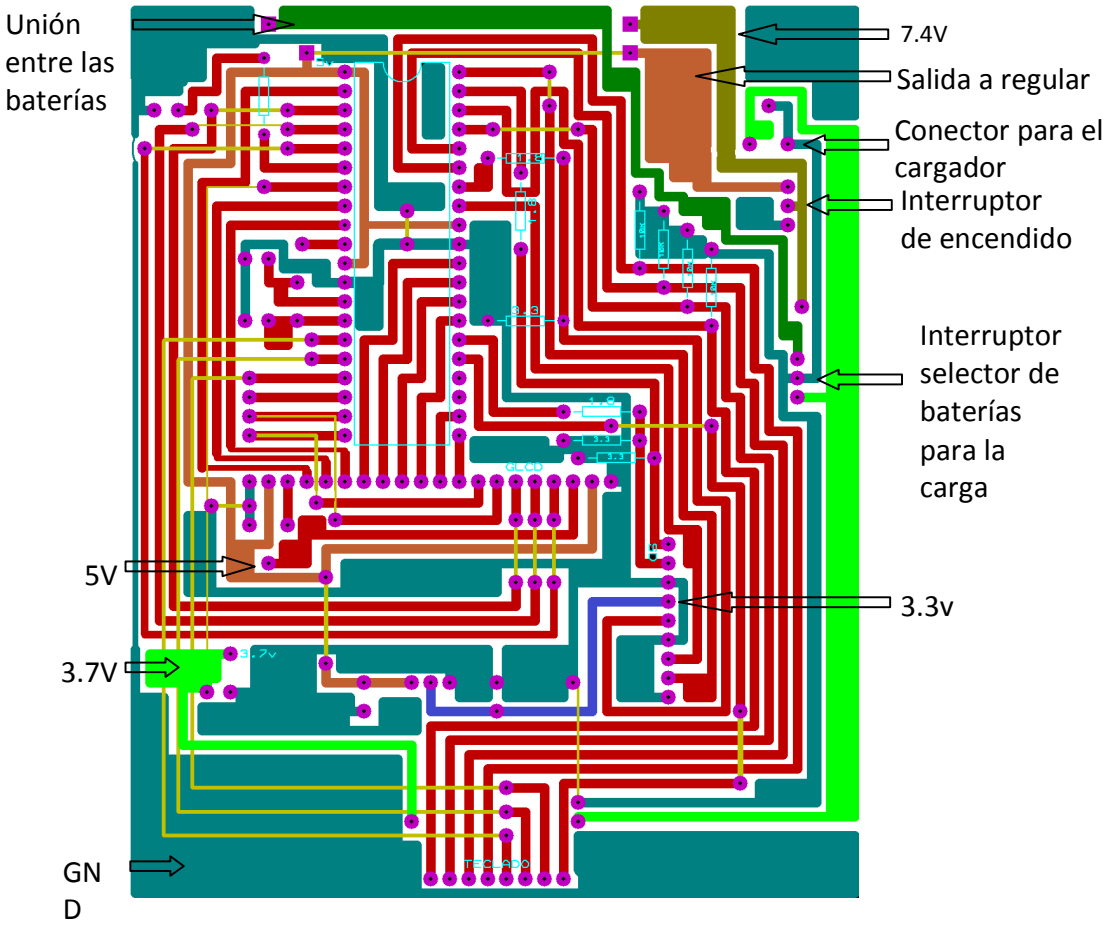


Figura V - 29: Circuito impreso trazado de rutas

Se obtiene como resultado la siguiente distribución de dispositivos, considérese que para el teclado, GLCD y SD se utilizaras socket de conexión

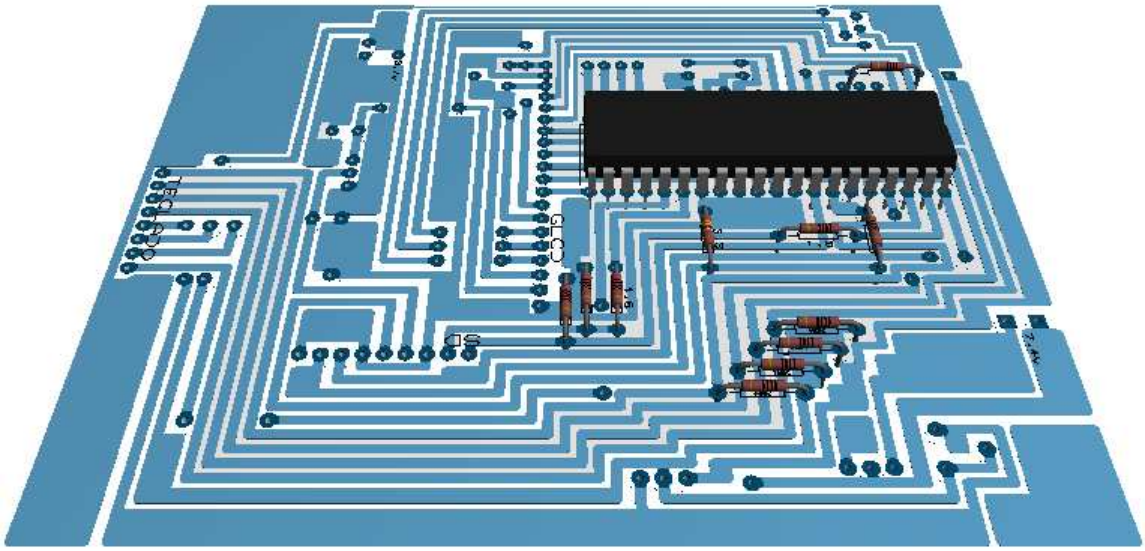


Figura V - 30: Distribución de dispositivos

5.4 DESARROLLO DEL FIRMWARE

Si queremos realizar la programación de los microcontroladores PIC en un lenguaje como el C, es preciso utilizar un compilador de C.

Dicho compilador "traduce" el código C del archivo fuente (.C) a lenguaje máquina para los microcontroladores PIC, generando así un archivo en formato Intel-hexadedimal (.HEX) que es el necesario para programar (utilizando un programador de PIC).

El compilador de C que vamos a utilizar es el PCWH de la casa CCS Inc. A su vez, el compilador lo integraremos en un entorno de desarrollo integrado (IDE) que nos va a permitir desarrollar todas y cada una de las fases que se compone un proyecto, desde la

edición hasta la compilación pasando por la depuración de errores. La última fase, a excepción de la depuración y retoques hardware finales, será programar el PIC.

El CCS PCWH Compiler: es un compilador que nos permite escribir los programas en lenguaje C en vez de assembler, con lo que se logra un menor tiempo de desarrollo, y mucha facilidad en la programación.

La función principal del microcontrolador el almacenar datos en un archivo de texto en la memoria SD para lo cual realiza las siguientes subrutinas:

- Manejo teclado 4x4, su función es activar secuencialmente las filas y censar los cambios producidos en las columnas para obtener el carácter presionado para concatenarlos y ejecutar la acción correspondiente a las teclas de control
- Manejo GLCD, su función es trasformar la información en píxeles para su visualización en el GLCD
- Comunicación SPI, su función es generar la señales digitales requeridas por la memoria SD
- Manejo FAT 16, su función dar estructura a los datos para que sean entendidos por la memoria SD

5.5 DIAGRAMA DE FLUJO

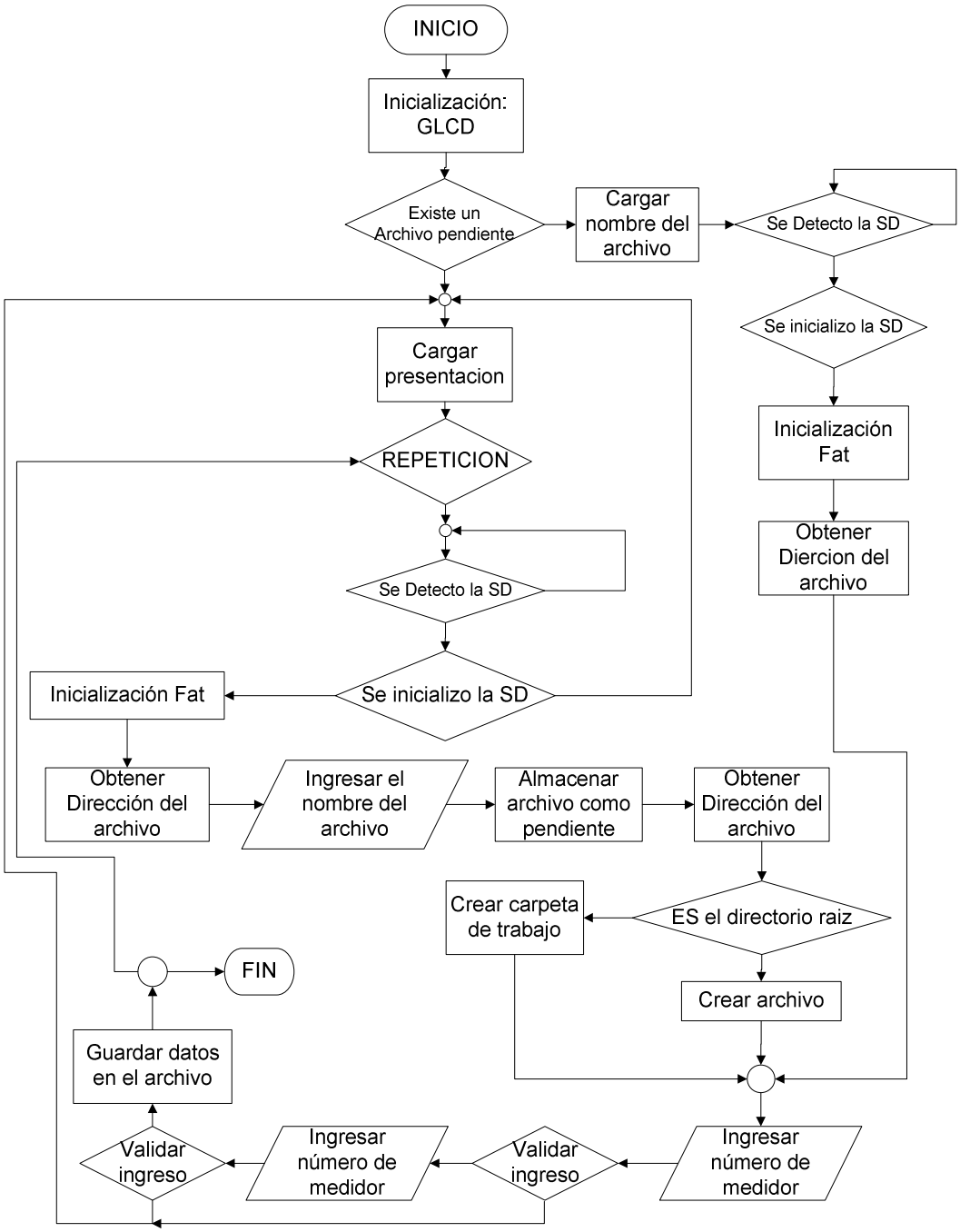


Figura V - 31: Diagrama de Flujo

5.6 LIBRERÍAS Y FUNCIONES UTILIZADAS

void teclado(unsigned short *x,unsigned short *y,char *z,char *datof);

Propósito: manejo del teclado, impresión de caracteres en pantalla y formar la cadena a almacenar

Entradas: (x, y): la coordenada superior izquierda del primer digito

datof: cadena a la que se quiere asignar lo ingresado

Salidas: x: número de caracteres ingresados

datof: cadena a almacenar o cadena de control

Descripción: Realizando el barrido de los puertos para identificar la tecla presionada de ser uno de los dígitos del 0 al 9 los imprime y almacena, la tecla B borra el nombre del archivo con el que está trabajando almacenado en la EEPROM y regresa al inicio del programa, la tecla D permite eliminar el ultimo carácter tanto de la pantalla como de la cadena, la tecla F termina el ingreso y devuelve la cadena endatof y el número de caracteres ingresados en x

GLCD45550.C

Para el manejo del GLCD 128x64 concontrolador KS0108 se ha utilizado como base el archivo **genericglcd.c** publicado en los foros de www.ucontrol.com.ar y creado por Suky (Casanova Alejandro)

Al que se le ha realizado algunas modificaciones como:

- La definición de les pines de control para adecuarse a nuestro microcontrolador.
- La adición de la función para la impresión de imágenes

- La adición de una fuente adicional

El archivo resultante consta de las siguientes funciones:

void glcd_init(int1 mode);

Propósito: Inicializar el lcdgrafico, debe ser llamada antes de cualquier otra función

Entradas: mode: que permite dos valores, OFF(0) – apaga el GLCD y ON(1) -enciende el GLCD

void glcd_pixel(int x, int y, int1 color);

Propósito: Encender o apagar un pixel del GLCD, llamada a través de otros funciones no directamente del programa principal

Entradas: x: coordenadas para el eje x

y: coordenadas para el eje y

color: estado del pixel ono off

void glcd_line(int x1, int y1, int x2, int y2, int1 color);

Propósito: Dibujar una línea en el GLCD

Entradas: (x1, y1): coordenadas iníciales

(x2, y2): coordenadas finales

color: estado del pixel ono off

Dependencia: glcd_pixel();

void glcd_rect(int x1, int y1, int x2, int y2, int fill, int1 color);

Propósito: Dibujar una Rectángulo en el GLCD

Entradas: (x1, y1): coordenadas iníciales

(x2, y2): coordenadas finales

Fill: 0 marca solo el contorno, 1 mara el contorno y el interior

color: estado del pixel ono off

Dependencia: glcd_pixel(), glcd_line()

void glcd_text57(int x, int y, char* textptr, int size, int1 color);

Propósito: Escribir texto en el GLCD

Entradas: (x, y): la coordenada superior izquierda de la primera letra

Textptr: un puntero a un arreglo de caracteres a mostrar

Size: tamaño del teto 1 = 5x7, 2 = 10x14, ...

color: estado del pixel ono off

Dependencia: glcd_pixel(), glcd_line()

void glcd_text35(int8 x, int8 y, char* textptr, int1 color);

Propósito: Escribir texto en el GLCD de 3x5

Entradas: (x, y): la coordenada superior izquierda de la primera letra

Textptr: un puntero a un arreglo de caracteres a mostrar

color: estado del pixel ono off

Dependencia: glcd_pixel(), glcd_line()

void glcd_writeByte(char chip, BYTE data);

Propósito: Escribe un byte de datos en chip especificado

Entradas: Chip: GLCD_CS1 o GLCD_CS2

Data: datos a ser escritos

Dependencia: glcd_writeByte()

void glcd_fillScreen(int1 color);

Propósito: Limpiar la pantalla, trabaja más rápido que dibujar un rectángulo en toda la pantalla

Entradas: color: estado del pixel ono off

Dependencia: glcd_writeByte()

void glcd_imagen(int1 tipo);

Propósito: Imprimir una imagen en pantalla

Entradas: tipo: 1imagen normal, o imagen invertida

Dependencia: glcd_pixel()

CCS SPI

CCSofreceunalibrería de apoyo para aprovechar el hardware ysoftwarebasado enla funcionalidad deSPI. LalibreríaSPIcontiene funciones paraimplementarunbusSPI.

#USE SPI

#USE SPI (options)

Propósito: Configurar todos los parámetros paraimplementarunbusSPI. Los pines decomunespresentesenel hardwareSPIson: DI, DO, yCLK. Estos pinesnoes necesario asignarvaloresa través delas opciones, elcompiladorasignará automáticamente los valoresespecíficosdel hardwarea estospines.

Entrada: Las opciones sonseparadosporcomasypuedenser:

MASTER: establece el dispositivocomo master. (default)

SLAVE: establece el dispositivocomoslave.

BAUD=n: bitsdedestinoporsegundo, **default**es lo másrápidoposible.

CLOCK_HIGH=n: High time of clock in us (no se necesitasi BAUD= está en uso). (default=0)

CLOCK_LOW=n: Low time of clock in us (not needed if BAUD= is used). (default=0)

DI=pin: Opcional pin para entrada de datos.

DO=pin: Opcional pin para salida de datos.

CLK=pin: Pin del reloj.

MODE=n: traducción del inglés al español
ElmodoaponerelbusSPI.

ENABLE=pin: pin opcional a ser activado durante la trasferencia.

LOAD=pin: PinOpcionalaserpulsadoactivadodespués de que los datos setransfieren.

DIAGNOSTIC=pin: PinOpcionalaserpuesto a 1 cuando los datos son los de muestra.

SAMPLE_RISE: Muestreaen el flanco ascendente.

SAMPLE_FALL: Muestreaen el flanco descendente (default).

BITS=n: Máximonúmero debits enunatransferencia. (default=32)

SAMPLE_COUNT=n: Número demuestrasatomar. (default=1)

LOAD_ACTIVE=n: Activael estado para el pin LOAD (0, 1).

ENABLE_ACTIVE=n: Activael estado para el pin ENABLE (0, 1). (default=0)

IDLE=n: Estado inactivopara el pin CLK (0, 1). (default=0)

ENABLE_DELAY=n: Tiempode retraso. (default=0)

DATA_HOLD=n: El tiempoentreel cambiode datosyel cambiodel reloj

LSB_FIRST: LSB esenviadoprimero.

MSB_FIRST: MSB esenviadoprimero. (default)

STREAM=id: especifica un nombrestreampara el protocolo.

SPI1: Utilizalos pinesde hardwareparaunpuertoSPI 1

SPI2: Utilizalos pinesde hardwareparaunpuertoSPI 2

FORCE_HW: UtiliceelhardwareSPI del PIC.

Salida: Ninguna

spi_xfer();

spi_xfer(data);

spi_xfer(stream, data);

spi_xfer(stream, data, bits);

result = spi_xfer(data);

result = spi_xfer(stream, data);

result = spi_xfer(stream, data, bits);

Propósito: Las transferenciasde datosdesde y hacia un dispositivoSPI.

Entrada: Data: eslavariableoconstanteparatransferir vía SPI. a través del pin DO

STREAM: nombre del streampara el protocolo

Bits: esla cantidad de bitsde data que serán transferidos

Salida: Los datos leídos del dispositivo SPI a través del pin DI

Dependencia: #USE SPI

setup_spi( )

setup_spi (mode);

Propósito: Inicializalainterfazde puerto serie(SPI). Esto se utiliza para dispositivos de 2o3 hilos que siguenunrelojcomún. Estafunciónsólo estádisponibleenlos dispositivosconhardwareSPI

Entrada: Mode puedenser:

SPI_MASTER, SPI_SLAVE, SPI_SS_DISABLED

SPI_L_TO_H, SPI_H_TO_L

SPI_CLK_DIV_4, SPI_CLK_DIV_16,

SPI_CLK_DIV_64, SPI_CLK_T2

SPI_SAMPLE_AT_END, SPI_XMIT_L_TO_H

traducción del inglés al español
Las constantesde cadagrupopueden serunidas con|

Salida: traducción del inglés al español
Indefinido

Dependencia: Las constantessedefinenen el archivo dispositivo.h(18F4550.h)

#USE SPI

SD_SPI.C

Para el manejo de la comunicación SPI con la SD se ha utilizado como base el archivo **SD_SPI.C** versión 1.1 publicado en los foros de www.ucontrol.com.ar y creado por Suky (Casanova Alejandro)

Al que se le ha realizado algunas modificaciones como:

- La definición de les pines de control para adecuarse a nuestro microcontrolador.
- La definición del modo SPI para adaptarse al memoria SD a utilizar

El archivo resultante consta de las siguientes funciones:

int1 detectSD(void);

Propósito: Detectar si la memoria es insertada leyendo el pin SD_CD

Entrada: Ninguna

Salida: 0 tarjeta detectada,1 tarjeta no detectada

Dependencia: Ninguna

voidSDSelect(void);

Propósito: Seleccionar la memoria llevando a 0 el pin CS

Entrada: Ninguna

Dependencia: spi_xfer()

voidSDDeselect(void);

Propósito: deseleccionar la memoria llevando a 1 el pin CS

Entrada: Ninguna

Dependencia: spi_xfer()

int1 SDCard_send_command(intcmd, int32 arg, int& res);

Propósito: Envía comando a la memoria SD

Entrada: cmd: Comando a enviar, arg: Argumento del comando

Salida: res: Respuesta al comando, 0 a ocurrido algún error, 1 todo ok

Dependencia: spi_xfer(),SDSelect()

int1 SDCard_init(void);

Propósito: inicialización de la memoria SD

Entrada: Ninguna

Salida: 1 si se ha iniciado correctamente, 0 caso contrario.

Dependencia: SETUP_SPI(),spi_xfer(),SDCard_send_command()

int1 SDCard_read_block(int32 Address,int *Buffer);

Propósito: Realiza lectura de un bloque

Entrada: Address: Dirección del sector

Salida: Buffer: Buffer donde se almacena lectura de memoria, 0 a ocurrido algún error, 1 todo ok

Dependencia: SDCard_send_command();spi_xfer()

int1 SDCard_write_block(int32 Address,int *Buffer);

Propósito: Realiza escritura de un bloque

Entrada: Address: Dirección del sector

Salida: Buffer: Buffer donde se almacena lectura de memoria, 0 a ocurrido algún error, 1 todo ok

Dependencia: SDCard_send_command();spi_xfer()

int1 SDCard_read_CSD(void);

Propósito: Realiza lectura del registro CSD

Entrada: Ninguno

Salida: 0 a ocurrido algún error, 1 todo ok

Dependencia: SDCard_send_command();spi_xfer()

int1 SDCard_read_CID(void);

Propósito: Realiza lectura del registro CID

Entrada: Ninguno

Salida: 0 a ocurrido algún error, 1 todo ok

Dependencia: SDCard_send_command();spi_xfer()

FAT16.C

Para el manejo de la comunicación SPI con la SD se ha utilizado como base el archivo **SD_SPI.C** versión 1.5 publicado en los foros de www.ucontrol.com.ar y creado por Suky

El archivo consta de las siguientes funciones:

int1 FAT_init(void);

Propósito: Inicia FAT, leyendo primer sector de memoria determinando parámetros del sistema de archivos

Entrada: Ninguno

Salida: 1 si la inicialización es correcta, 0 caso contrario

Dependencia: SDCard_read_block()

int1 FAT_FindDirectory(char *ptr,int16 NClusP,int16&NClus);

Propósito: Busca directorio/s retornando posición para luego trabajar en él

Entrada: *ptr: Nombre de directorio/s

NClusP: Nº de Clúster donde comenzar la búsqueda

Salida: NClus: Nº de clúster del último directorio en búsqueda

1 si es exitoso, 0 caso contrario

Dependencia: SDCard_read_block()

void FAT_LoadName(char *ptr, int Type, char *ptr2,int16 AddDir);

Propósito: Carga nombre en el Buffer

Entrada: ptr: Nombre corto del archivo sin tratamiento

Type: Tipo, archivo o directorio

ptr2: Nombre del archivo sacando el punto " . " y colocando espacios adecuados " "

AddDir: Dirección adicional en el sector para crear archivo/directorio

Salida: Ninguna

Dependencia: Ninguna

int1 FAT_FindFileDir(char *Name,int16 NClusP,int Type,int32& SecFileDir,int16& AddDir,int16&NClus, int32&SizeFile);

Propósito: Busca Archivo/Directorio dentro del clúster padre (NClusP).

Entrada: Name: Nombre del archivo/directorio a buscar

NClusP: Numero del cluster donde buscar. Se inicia por DirectorioRaiz->NClusP=0

Type: Archivo=0x20, Directorio=0x10

Salida: SecFileDir: Sector donde está definido archivo

AddDir: Dirección adicional donde se encuentra definido archivo

NClus: Numero de clúster donde está ubicado el archivo/directorio

SizeFile: Tamaño del archivo.-

1 si la inicialización es correcta, 0 caso contrario.-

Dependencia: SDCard_read_block(), FAT_LoadName ()

int1 FAT_FindFreeEnt(int16 NClus,int& NSecFree,int16&AddDir);

Propósito: Dentro del Directorio(NClus) a crear archivo/directorio busca Nº Sector y posición dentro del sector para crear la entrada

Entrada: NClus: Numero de clúster donde crear entrada en el Directorio

Salida: NSecFree: Número de sector adicional del Sector Inicial

AddDir: Dirección adicional dentro del Sector

1 si la inicialización es correcta, 0 caso contrario

Dependencia: SDCard_read_block()

int1 FAT_CreateEntFAT(int16 NClus,int16 EOC);

Propósito: Crea entrada en FAT1 y FAT2.

Entrada: NClus: Numero de clúster donde crear entrada en el Directorio

EOC: Si es clúster final se guarda 0xFFFF o si no se direcciona al próximo clúster

Salida: 1 si la inicialización es correcta, 0 caso contrario

Dependencia: SDCard_read_block(), SDCard_write_block()

void FAT_LoadBuffer(char *Name,int Type,int16 NClus,int32 SizeFile,int16 AddDir,int *BufferFAT2);

Propósito: Carga buffer para crear entrada del archivo/directorio. Coloca nombre, fecha, hora, etc.

Entrada: Name: Nombre corto del archivo

Type: Tipo, archivo o directorio

NClus: Numero de clúster donde estará el archivo/directorio

SizeFile: Tamaño del archivo

AddDir: Dirección adicional en el sector para crear archivo/directorio

BufferFAT: Buffer donde guardar los parámetros

Salida: Ninguna

Dependencia: FAT_LoadName()

int1 FAT_DefineDirectory(char *NameLong,char *NameShort,int16 NClusP,int16 NClus,intNSecFree, int16 AddDir);

Propósito: Define directorio, solo crea la definición del nuevo directorio en el directorio seleccionado xD (NClusP)

Entrada: NameLong: Nombre largo del directorio

NameShort: Nombre corto del directorio

NClusP Número de clúster padre donde se define directorio

NClus: Número de clúster donde se guarda el directorio nuevo

NSecFree: Número de sector en el cluster padre para crear definición

AddDir: Dirección adicional dentro de NSecFree donde comenzar a crear definición

Salida: 0 si ha ocurrido algún error y 1 si ha sido exitoso

Dependencia: SDCard_read_block(), SDCard_write_block(),FAT_LoadBuffer()

int1 FAT_CreateDirectory(char *NameLong,char *NameShort, int16 NClusP);

Propósito: Crea un directorio. Busca un clúster libre en la FAT, crea la definición y crea la entrada en la FAT

Entrada: NameLong: Nombre largo del directorio

NameShort: Nombre corto del directorio

Salida: NClusP Numero de clúster padre donde se define directorio

1 si es exitoso, 0 caso contrario

Dependencia: FAT_FindFreeEnt(),FAT_DefineDirectory(),FAT_CreateEntFAT()

int1 FAT_CreateFile(char *NameLong, char *NameShort,int16 NClusP,char *WriteString);

Propósito: Crea un archivo. Busca un clúster libre en la FAT, crea la definición, crea el archivo y crea la entrada en la FAT

Entrada: NameLong: Nombre largo del directorio

NameShort: Nombre corto del directorio

NClusP: Numero de clúster padre donde se define archivo

WriteString: Buffer del archivo a cargar en memoria

Salida: 0 si ha ocurrido algún error y 1 si ha sido exitoso

Dependencia: FAT_FindFreeEnt(),FAT_DefineDirectory(),FAT_CreateEntFAT()

int1 FAT_WriteAddFile(int32 SecClusEnd,int32 SecClusNew,int16 NSecAddWrite,int16 AddDirWrite, int32 SecFileDir,int16 AddDirSec,int32 Size,char *WriteString);

Propósito: Carga nuevos datos al clúster final del archivo y en el nuevo clúster si es necesario. Además actualiza tamaño del archivo

Entrada: SecClusEnd: Ultimo clúster del archivo

SecClusNew: Nuevo clúster de ser necesario

NSecAddWrite: Ultimo sector del clúster final a donde comenzar agregar datos

AddDirWrite: Dirección adicional del último sector para comenzar agregar datos

SecFileDir: Sector donde está definido archivo

AddDirSec: Dirección adicional donde se encuentra definido archivo

Size: Nuevo tamaño a cargar

WriteString: Datos a cargar

Salida: 0 si ha ocurrido algún error y 1 si ha sido exitoso

Dependencia: SDCard_read_block(), SDCard_write_block()

int1 FAT_OpenAddFile(char *NameShort,int16 NClusP, char *WriteString);

Propósito: Busca un archivo por nombre corto, determina clúster inicial, clúster final, calcula tamaño del archivo y mas el adicional determina si es necesario usar otro Clúster. Si es necesario busca un nuevo clúster libre para adicionar al archivo. Terminado el agregado de datos, actualiza estructura de clúster en FAT y actualiza tamaño del archivo en la definición.

Entrada: NameShort: Nombre corto del directorio

NClusP: Nº de Clúster donde comenzar la búsqueda

WriteString: Datos a agregar al archivo

Salida: 0 si ha ocurrido algún error y 1 si ha sido exitoso

Dependencia: FAT_FindFileDir (),FAT_WriteAddFile ()

5.7 SIMULACIÓN

Para obtener una idea del funcionamiento que se obtendrá de nuestro firmware se ha recurrido un paquete EDA. Hay una gran variedad de paquetes EDA (Herramientas de CAD Electrónico) estas herramientas unas más conocidas que otras entre ellas podemos citar: TangoPCB, Elegance, Livewire, Proteus VSM, ExpresPCB, Eagle, etc. Todas son marcas registradas, algunas de ellas disponen de versiones demo que pueden servir para conocer su funcionamiento.

De entre los cuales hemos optado por utilizar PROTEUS VSM(**S**istema **V**irtual de Modelado) de LabcenterElectronics, Este permite el diseño de circuitos empleando un entorno gráfico en el cual es posible colocar los símbolos representativos de los componentes y realizar la simulación de su funcionamiento sin el riesgo de ocasionar daños a los circuitos.

La simulación puede incluir instrumentos de medición y la inclusión de gráficas que representan las señales obtenidas en la simulación. Lo que más interés ha despertado es la capacidad de simular adecuadamente el funcionamiento de los microcontroladores más populares (PICS, ATMEL-AVR, MOTOROLA, 8051, etc.)

PROTEUS

Consta de los dos programas principales: Ares e Isis, y los módulos VSM y Electra.

ARES

ARES, o AdvancedRouting and EditingSoftware (*Software de Edición y Ruteo Avanzado*); es la herramienta de enrutado, ubicación y edición de componentes, se utiliza para la fabricación de placas de circuito impreso, permitiendo editar generalmente, las capas superficial (Top Copper), y de soldadura (BottomCopper).

ISIS

El Programa ISIS, IntelligentSchematicInput System (*Sistema de Enrutado de Esquemas Inteligente*) permite diseñar el plano eléctrico del circuito que se desea realizar con componentes muy variados, desde simples resistencias, hasta algúno que otro microprocesador o micro controlador, incluyendo fuentes de alimentación, generadores de señales y muchos otros componentes con prestaciones diferentes. Los diseños realizados en Isis pueden ser simulados en tiempo real, mediante el módulo VSM, asociado directamente con ISIS.

El módulo VSM

Una de las prestaciones de Proteus, integrada con ISIS, es VSM, el Virtual SystemModeling (*Sistema Virtual de Modelado*), una extensión integrada con ISIS, con la cual se puede simular, en tiempo real, con posibilidad de más rapidez; todas las características de varias familias de microcontroladores, introduciendo nosotros mismos el programa que controlará el microcontrolador y cada una de sus salidas, y a la vez, simulando las tareas

que queramos que lleve a cabo con el programa. Se pueden simular circuitos con microcontroladores conectados a distintos dispositivos, como motores, lcd´s, teclados en matriz, etc. Incluye, entre otras, las familias de PIC's PIC10, PIC12, PIC16, PIC18, PIC24 y dsPIC33. ISIS es el corazón del entorno integrado PROTEUS. Combina un entorno de diseño de una potencia excepcional con una enorme capacidad de controlar la apariencia final de los dibujos.

A continuación se presenta una captura de pantalla de la simulación realizada en PROTEUS ISIS en el cual se utiliza un microcontrolador PIC18F4550, un teclado 4X4, un GLCD 128x64 con controlador KS0108, un módulo de tarjeta MMC virtual con interface SPI y se han omitido el oscilador y las resistencias que para propósitos de la simulación no son estrictamente requeridos.

Para el PIC18F4550 definimos la frecuencia del procesamiento del reloj (20MHz) y el archivo del programa (\main.hex)

Para el GLCD se define la frecuencia en 300KHz ya que a diferentes frecuencias puede presentar distorsión en la imagen a mostrar

Para la tarjeta MMC se requiere especificar un archivo de imagen que nos permita montar volúmenes virtuales cuya extensión puede ser *.iso,*.mmc o *.ima entre otros.

La conexión de los pines de los dispositivos se la realiza de acuerdo a la definición realizada en el Firmware.

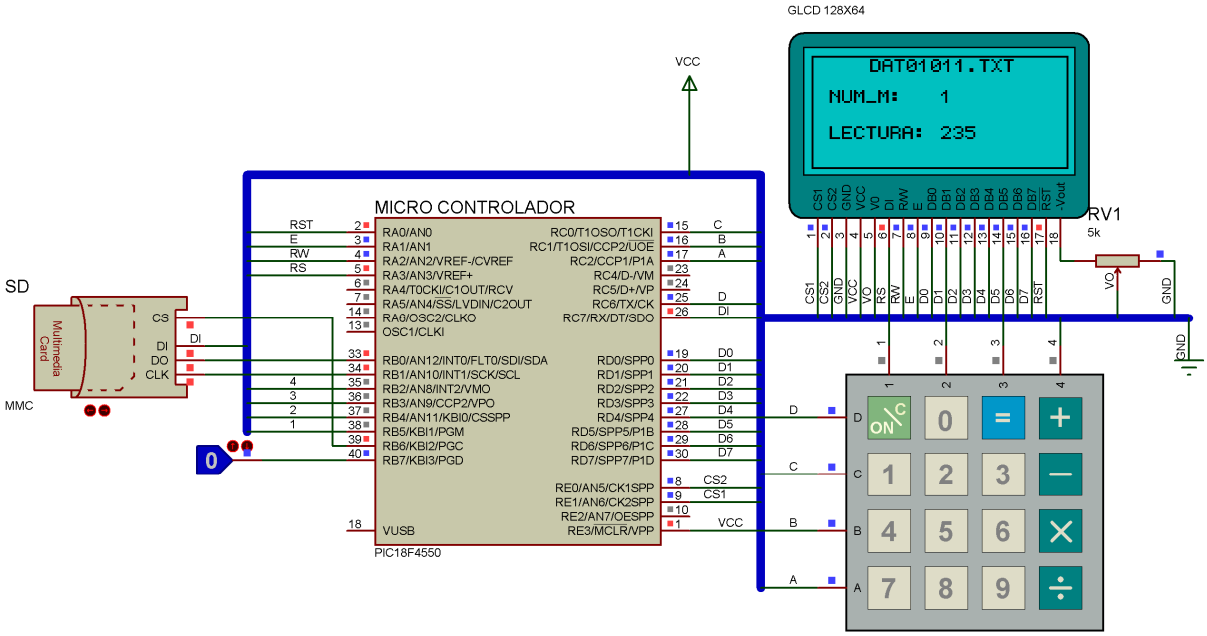


Figura V - 32: Simulación del ingreso de datos

Para visualizar el contenido de la memoria virtual se ha recurrido al software ULTRAISO que nos permite manejar unidades virtuales

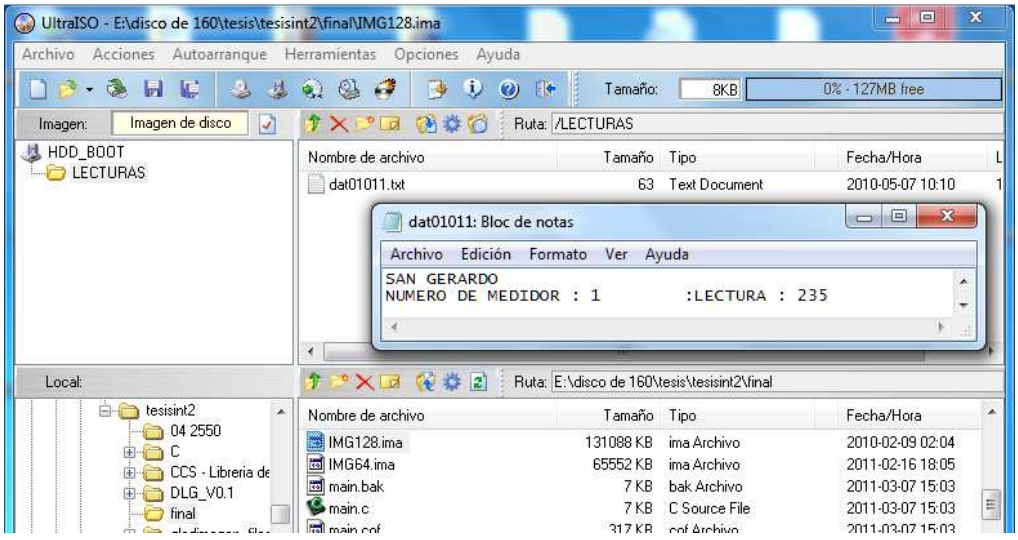


Figura V - 33: Datos que contiene la memoria

Capítulo VI

6.1 DISEÑO DEL SOFTWARE DE GESTIÓN

6.1.1 ESPECIFICACIÓN DEL SOFTWARE

En esta fase se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir.

Es importante señalar que en esta etapa vamos a **consensuar** todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software.

6.1.1.1 DESCRIPCIÓN GENERAL

6.1.1.1.1 PERSPECTIVA DEL PRODUCTO

El Software a desarrollarse es un producto independiente, orientado a satisfacer las necesidades puntuales de la organización.

6.1.1.1.2 FUNCIONALIDAD DEL PRODUCTO

- Emitir ágil y oportunamente comprobantes de cobro por servicios de Agua Potable.
- Gestionar datos referentes a los usuarios del agua potable.
- Comunicarse con un dispositivo portable de datos para obtener mediciones de consumo de agua potable.
- Emitir reportes de los cobros realizados y demás información útil para la gestión de cobros de agua potable.
- Aplicar cobros por multas e intereses por mora.

6.1.1.1.3 CARACTERÍSTICAS DE LOS USUARIOS

Tipo de usuario:	Tesorero
Formación:	Primaria, Secundaria o Superior
Habilidades:	Conocimientos Básicos de Informática.
Actividades:	Realizar los Cobros de Agua Potable y presentar informes económicos

6.1.1.1.4 RESTRICCIONES

El Software será diseñado para el Sistema Operativo Windows XP o superiores, y no se garantiza su funcionamiento en otras plataformas, además para la comunicación con el dispositivo portable es necesario un Lector de Memorias SD.

6.1.1.1.1.5 SUPOSICIONES Y DEPENDENCIAS

Se asume que el personal que va a operar el sistema posee conocimientos básicos de informática y están dispuestos a adaptarse a los requerimientos que supone un sistema automatizado.

6.1.1.1.6 EVOLUCIÓN PREVISIBLE DEL SISTEMA

El sistema deberá adaptarse en un futuro a nuevas políticas y mecanismos de cobro que la organización exija.

6.1.1.1.7 REQUISITOS COMUNES DE LOS INTERFACES

Interfaces de usuario

Las interfaces deben ser amigables, intuitivas y fáciles de utilizar, presentando en todo momento una breve descripción de la tarea que nos permite realizar, además se sugiere el uso de fuentes grandes, y colores contrastados

Interfaces de hardware

Para un correcto funcionamiento se recomienda un computador con procesador de Doble núcleo, por lo menos 1Gb de RAM, 160 Gb de Disco Duro, Monitor, Lector de memorias, y una impresora matricial.

Interfaces de software

Para el funcionamiento de la base de datos se requiere instalar SQL Server 2000 y algunas librerías adicionales para acceso a datos de Microsoft.

Interfaces de comunicación

Para la extracción de datos del Dispositivo Portable se requiere la instalación de un Lector de Tarjetas.

6.1.2DISEÑO DE LA BASE DE DATOS

Para la implementación de la base de datos se hautilizado las siguientes tablas destinadas a ser implementadas en SQL Server 2000 en las que se especifica nombre del campo, tipo y tamaño:

SOCIO		
CI	CHAR	10
APELLIDO	VARCHAR	50
OBS	CAHR	25

BARRIO		
COD_BARRIO	INT	4
NOMBRE_B	CHAR	20
DESCRIPCION	CHAR	25

TARIFA		
COD_BARRIO	INT	4
BASICO	MONEY	8
NOMBRE	VARCHAR	50
LIMITE	INT	4
COSTOXM3	MONEY	8
OBS	VARCHAR	50

SOCIO_BARRIO		
CI	CHAR	10
COD_BARRIO	INT	4
NUM_MEDIDOR	CHAR	10
ACOMETIDA	BIT	1
COD_TARIFA	INT	4

RECIBO		
NUM_REC	INT	4
TOTAL	MONEY	8
FECHA_REC	DATETIME	8
OBS_REC	CHAR	25

COBRO		
COD_COBRO	INT	4
COD_RUBRO	INT	4
MONTO	MONEY	8
FECHA	DATETIME	8

SB_COBRO		
CI	CHAR	10
COD_BARRIO	INT	4
COD_COBRO	INT	4
NUM_REC	INT	4
ESTADO	BIT	1
MORA	MONEY	8
MEDICION	INT	4
CONSUMO	INT	4

RUBRO		
COD_RUBRO	INT	4
DESCRIPCION	VARCHAR	50
COD_MES	INT	4
AÑO	CHAR	4

MES		
COD_MES	INT	4
MES	CHAR	10

6.1.1.1 DIAGRAMA ENTIDAD RELACIÓN

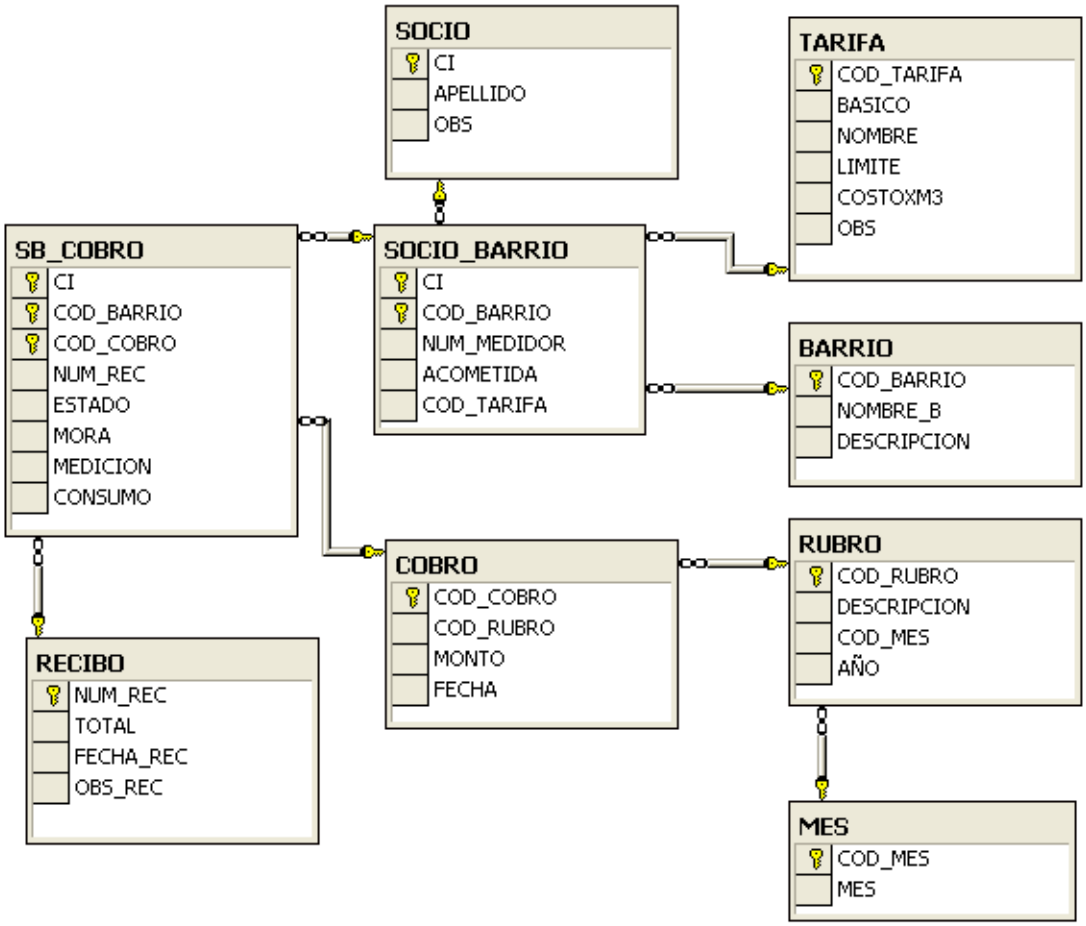


Figura VI - 34: Diagrama entidad relación

6.1.3 DISEÑO DE LA INTERFACES DE USUARIO

Para la aplicación cliente se ha elegido Microsoft Visual 6.0 y el diseño de las interfaces es presentado a continuación:

Figura VI - 35: Ventana para Generar Recibos de cobro

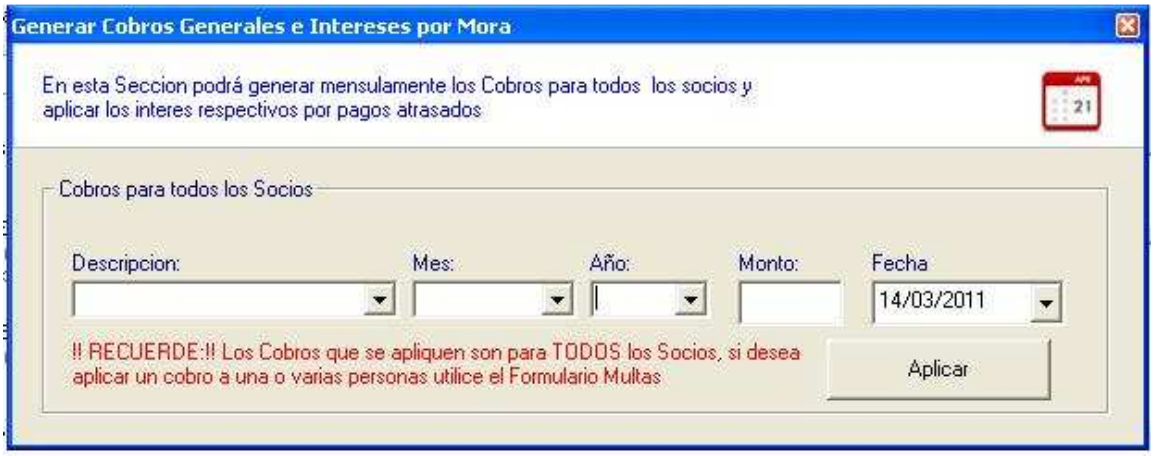


Figura VI - 36: Ventana para Generar Cobros generales a los socios

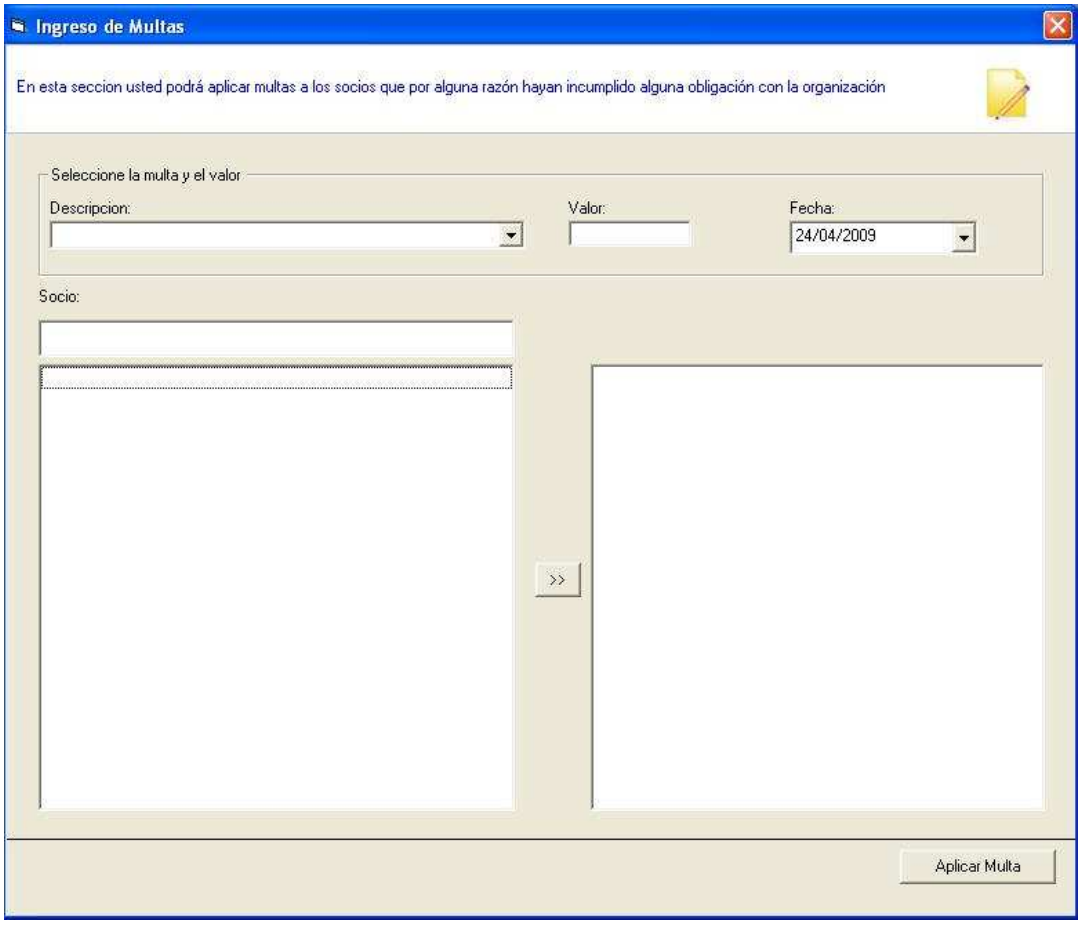


Figura VI - 17: Ventana para Aplicar Multas a uno o varios socios

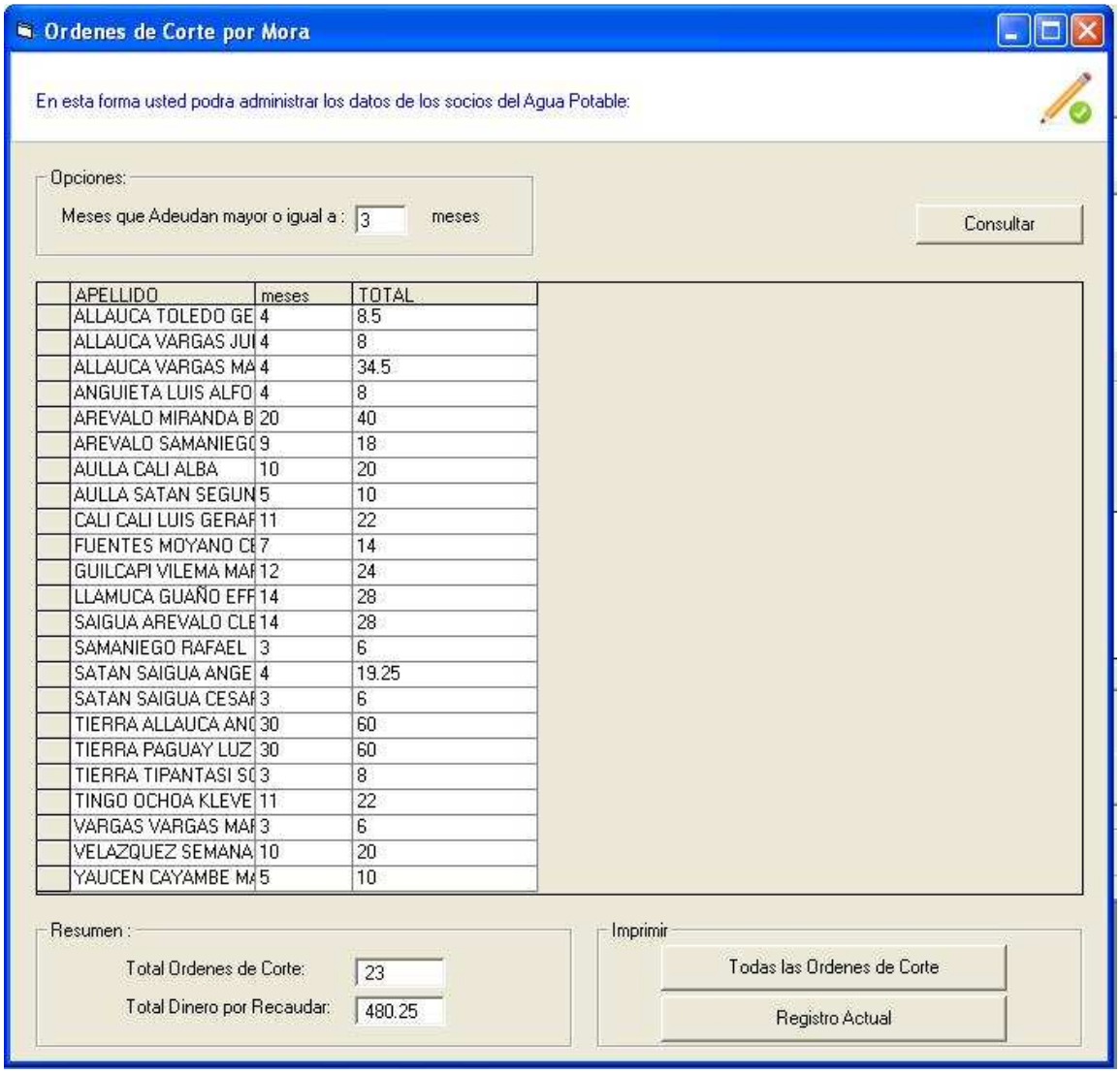


Figura VI - 38: Ventana para Consultar y generar ordenes de corte por mora

Figura VI - 39: Selección de opciones para los informes de cobros realizados

Figura VI - 40: Interface para visualizar los informes de cobros realizados.

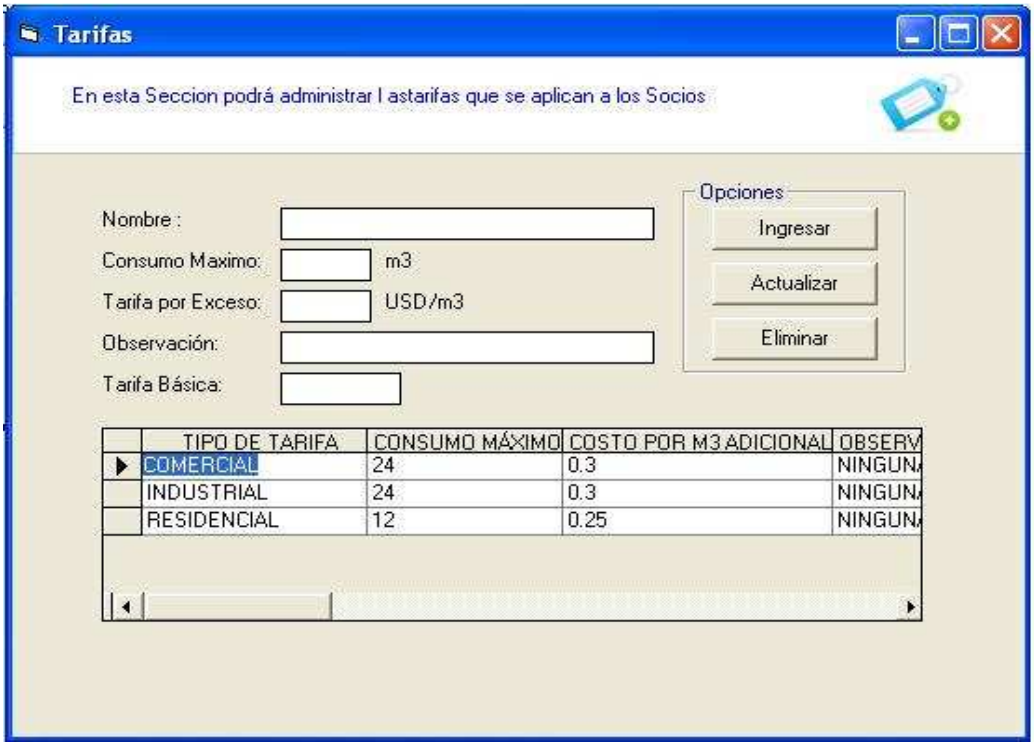


Figura VI - 41: Ventana para Gestionar las tarifas de cobro de agua potable

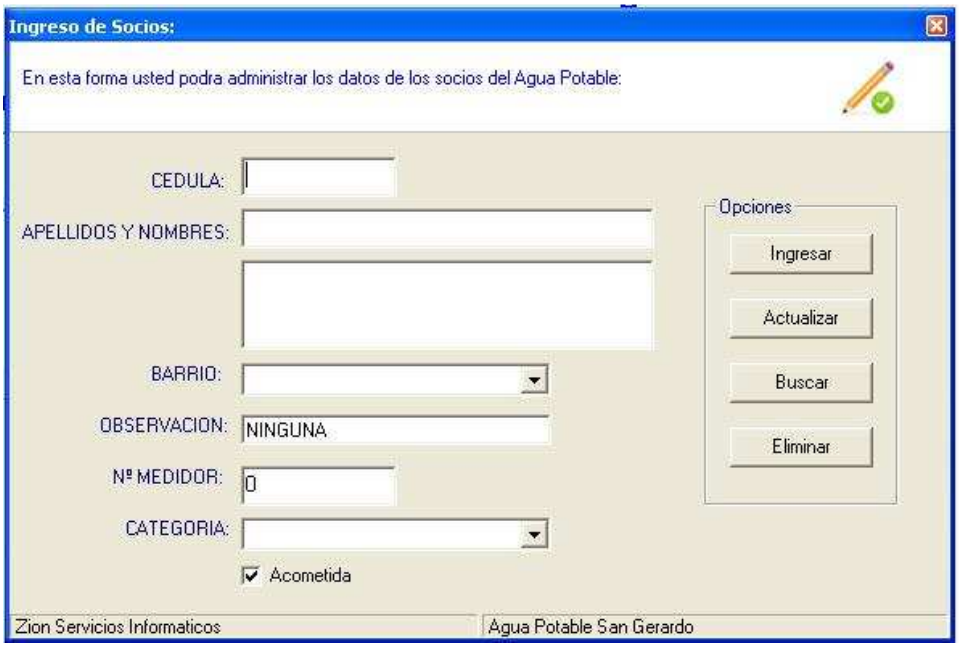


Figura VI - 42: Ventana para Gestionar los datos de los socios.

6.1.4 CODIFICACIÓN

6.1.4.1 PROCEDIMIENTOS ALMACENADOS

A Continuación se detalla los procedimientos almacenados programados en el Sistema de Base de Datos:

BORRAR_REC: Permite borrar un recibo emitido previamente.

```
CREATE PROCEDURE BORRAR_REC
@NUM_REC INT
AS
UPDATE SB_COBRO
SET NUM_REC=0, ESTADO=0
WHERE NUM_REC=@NUM_REC
DELETE FROM RECIBO
WHERE NUM_REC=@NUM_REC
GO
```

C_CI_SOCIO: Permite borrar un recibo emitido previamente.

```
CREATE PROCEDURE C_CI_SOCIO
@CI_I CHAR(10),
@CI_F CHAR(10)
AS
DECLARE @COD_BARRIO INT
INSERT INTO SOCIO (CI)
VALUES (@CI_F)
UPDATE SOCIO
SET APELLIDO = (SELECT APELLIDO FROM SOCIO WHERE CI=@CI_I), OBS = (SELECT OBS
FROM SOCIO WHERE CI=@CI_I)
WHERE CI=@CI_F

INSERT INTO SOCIO_BARRIO (CI,COD_BARRIO)
VALUES (@CI_F,5)

UPDATE SOCIO_BARRIO
SET COD_BARRIO = (SELECT  COD_BARRIO FROM SOCIO_BARRIO WHERE CI=@CI_I),
NUM_MEDIDOR = (SELECT NUM_MEDIDOR FROM SOCIO_BARRIO WHERE CI=@CI_I),
ACOMETIDA = (SELECT ACOMETIDA FROM SOCIO_BARRIO WHERE CI=@CI_I)
WHERE CI=@CI_F

UPDATE SB_COBRO
SET CI=@CI_F
WHERE CI=@CI_I
```

```
SET @COD_BARRIO =(SELECT COD_BARRIO FROM SOCIO_BARRIO WHERE CI=@CI_I)
DELETE FROM SOCIO_BARRIO WHERE CI=@CI_I AND COD_BARRIO=@COD_BARRIO
DELETE FROM SOCIO WHERE CI=@CI_I
GO
```

C_COD_BARRIO : Permite cambiar el código de Barrio de un usuario.

```
CREATE PROCEDURE C_COD_BARRIO
@CI CHAR(10),
@NOMBRE_B_I CHAR(20),
@NOMBRE_B_F CHAR(20)
AS
DECLARE @COD_BARRIO_I INT
DECLARE @COD_BARRIO_F INT
SET @COD_BARRIO_I =(SELECT COD_BARRIO FROM BARRIO WHERE
@NOMBRE_B_I=NOMBRE_B)
SET @COD_BARRIO_F =(SELECT COD_BARRIO FROM BARRIO WHERE
@NOMBRE_B_F=NOMBRE_B)
INSERT INTO SOCIO_BARRIO (CI,COD_BARRIO)
VALUES (@CI,@COD_BARRIO_F)

UPDATE SOCIO_BARRIO
SET NUM_MEDIDOR = (SELECT NUM_MEDIDOR FROM SOCIO_BARRIO WHERE CI=@CI
AND COD_BARRIO=@COD_BARRIO_I),ACOMETIDA = (SELECT ACOMETIDA FROM
SOCIO_BARRIO WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO_I),COD_TARIFA =
(SELECT COD_TARIFA FROM SOCIO_BARRIO WHERE CI=@CI AND
COD_BARRIO=@COD_BARRIO_I)
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO_F
UPDATE SB_COBRO
SET COD_BARRIO=@COD_BARRIO_F
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO_I
DELETE FROM SOCIO_BARRIO WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO_I
GO
```

ELIMINAR_COBRO_SOCIO: Permite borrar un recibo emitido previamente.

```
CREATE PROCEDURE ELIMINAR_COBRO_SOCIO
@CI CHAR(10),
@COD_COBRO INT
AS
DELETE FROM SB_COBRO
WHERE CI=@CI AND COD_COBRO=@COD_COBRO
DELETE FROM COBRO
WHERE COD_COBRO NOT IN (SELECT COD_COBRO FROM SB_COBRO)
DELETE FROM RUBRO
WHERE COD_RUBRO NOT IN (SELECT COD_RUBRO FROM COBRO)
GO
```

ELIMINAR_SOCIO: Permite borrar los registros de un socio pero solamente se la tabla socio.

```
CREATE PROCEDURE ELIMINAR_SOCIO
@CI CHAR(10)
AS
DELETE
FROM SOCIO WHERE CI=@CI
GO
```

ELIMINAR_SOCIO_TODO: Permite borrar todos los registros de un socio inclusive en las tablas relacionadas.

```
CREATE PROCEDURE ELIMINAR_SOCIO_TODO
@CI CHAR(10),
@COD_BARRIO INT
AS
DELETE FROM SB_COBRO
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO

DELETE FROM SOCIO_BARRIO
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO
GO
```

GENERAR_COBRO: Permite generar cobros a todos los socios del agua potable.

```
CREATE PROCEDURE GENERAR_COBRO
@DESCRIPCION VARCHAR(50),
@MES CHAR(25),
@AÑO INT,
@MONTO INT,
@FECHA DATETIME
AS
DECLARE @COD_COBRO INT
DECLARE @COD_RUBRO INT
DECLARE @COD_MES INT
SET @COD_MES=(SELECT COD_MES FROM MES WHERE @MES=MES)
SET @COD_RUBRO=(SELECT COD_RUBRO FROM RUBRO WHERE
@DESCRIPCION=DESCRIPCION AND @COD_MES=COD_MES AND @AÑO=AÑO)

IF @COD_RUBRO IN (SELECT COD_RUBRO FROM RUBRO)
BEGIN
IF (SELECT MAX(COD_COBRO) FROM COBRO)=NULL
BEGIN
SET @COD_COBRO=1
INSERT INTO COBRO
VALUES (@COD_COBRO,@COD_RUBRO,@MONTO,@FECHA)
```

```
END
ELSE
BEGIN
IF (SELECT COD_COBRO FROM COBRO WHERE COD_RUBRO=@COD_RUBRO AND
@MONTO=@MONTO AND @FECHA=FECHA) IS NULL
BEGIN
SELECT @COD_COBRO=MAX(COD_COBRO)+1 FROM COBRO
INSERT INTO COBRO
VALUES (@COD_COBRO,@COD_RUBRO,@MONTO,@FECHA)
END
ELSE
SET @COD_COBRO =  (SELECT TOP 1 COD_COBRO FROM COBRO WHERE
COD_RUBRO=@COD_RUBRO AND  @MONTO=@MONTO AND @FECHA=FECHA)
END
END
INSERT INTO SB_COBRO (CI,COD_BARRIO,COD_COBRO)
SELECT CI,COD_BARRIO,COD_COBRO FROM SOCIO_BARRIO,COBRO WHERE
COD_COBRO=@COD_COBRO AND ACOMETIDA='1'
GO
```

GENERAR_LECTURA: Permite genera un registro con las lecturas de consumo de agua potable extraídas del dispositivo portable.

```
CREATE PROCEDURE GENERAR_LECTURA
@CI CHAR(10),
@COD_BARRIO INT,
@COD_COBRO INT,
@MEDICION INT,
@CONSUMO INT
AS
UPDATE SB_COBRO
SET MEDICION=@MEDICION,CONSUMO=@CONSUMO
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO AND COD_COBRO=@COD_COBRO
GO
```

GENERAR_MULTA: Genera multas a una lista de socios del agua potable, por alguna obligación incumplida.

```
CREATE PROCEDURE GENERAR_MULTA
@NUM INT,
@APELLIDO CHAR(50),
@COD_RUBRO INT,
@MONTO FLOAT,
@FECHA DATETIME
AS
DECLARE @COD_COBRO INT
```

```
DECLARE @COD_MES INT
DECLARE @COD_BARRIO INT
DECLARE @CI CHAR(10)
SET @CI = (SELECT CI FROM SOCIO WHERE APELLIDO=@APELLIDO)
IF @COD_RUBRO IN (SELECT COD_RUBRO FROM RUBRO) AND @NUM=1
BEGIN
IF (SELECT MAX(COD_COBRO) FROM COBRO)=NULL
BEGIN
SET @COD_COBRO=1
INSERT INTO COBROVALUES (@COD_COBRO,@COD_RUBRO,@MONTO,@FECHA)
END
ELSE
BEGIN
SELECT @COD_COBRO=MAX(COD_COBRO)+1 FROM COBRO
INSERT INTO COBRO
VALUES (@COD_COBRO,@COD_RUBRO,@MONTO,@FECHA)
END
END
IF @NUM > 1
BEGIN
SET @COD_COBRO=(SELECT MAX(COD_COBRO) FROM COBRO)
END
INSERT INTO SB_COBRO (CI,COD_BARRIO,COD_COBRO)
SELECT CI,COD_BARRIO,COD_COBRO FROM SOCIO_BARRIO,COBRO WHERE CI=@CI AND
COD_COBRO= @COD_COBRO
GO
```

GENERAR_REC: Genera un numero de recibo para los pagos que hace un socio dl agua potable.

```
CREATE PROCEDURE GENERAR_REC
@CI CHAR(10),
@NOMBRE_B VARCHAR(50),
@NUM_REC INT,
@OBS_REC VARCHAR(50),
@TOTAL MONEY,
@COD_COBRO INT
AS
DECLARE @COD_BARRIO  INT
SET @COD_BARRIO=(SELECT COD_BARRIO FROM BARRIO WHERE
@NOMBRE_B=NOMBRE_B)
IF @NUM_REC NOT IN (SELECT NUM_REC FROM RECIBO )
BEGIN
INSERT INTO RECIBO
VALUES (@NUM_REC,@TOTAL,GETDATE(),@OBS_REC)
END
UPDATE SB_COBRO
```

```
SET NUM_REC=@NUM_REC, ESTADO=1
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO AND COD_COBRO=@COD_COBRO
GO
```

INGRESO_MEDIDOR: Permite ingresar el número de medidor correspondiente a un socio.

```
CREATE PROCEDURE INGRESO_MEDIDOR
@CI CHAR(10),
@COD_BARRIO INT,
@NUM_MEDIDOR INT
ASIF @CI IN (SELECT CI FROM SOCIO)
BEGIN
IF @COD_BARRIO IN (SELECT COD_BARRIO FROM BARRIO)
BEGIN
INSERT INTO SOCIO_BARRIO
VALUES (@CI,@COD_BARRIO,@NUM_MEDIDOR)
END
END
GO
```

INGRESO_RUBRO: Permite ingresar nuevos rubros que serán cobrados.

```
CREATE PROCEDURE INGRESO_RUBRO
@DESCRIPCION VARCHAR(50),
@MES CHAR(25),
@AÑO INT
AS
DECLARE @COD_RUBRO INT
DECLARE @COD_MES INT
IF @MES IN (SELECT MES FROM MES)
BEGIN
SET @COD_MES=(SELECT COD_MES FROM MES WHERE @MES=MES)
IF (SELECT MAX(COD_RUBRO) FROM RUBRO)=NULL
BEGIN
SET @COD_RUBRO=1
INSERT INTO RUBRO
VALUES (@COD_RUBRO,@DESCRIPCION,@COD_MES,@AÑO)
END
ELSE
BEGIN
SELECT @COD_RUBRO=MAX(COD_RUBRO)+1 FROM RUBRO
INSERT INTO RUBRO
VALUES (@COD_RUBRO,@DESCRIPCION,@COD_MES,@AÑO)
END
END
GO
```

INGRESO_SOCIO: Permite ingresar nuevos socios.

```
CREATE PROCEDURE INGRESO_SOCIO
@CI CHAR(10),
@APELLIDO VARCHAR(50),
@NOMBRE_B VARCHAR(50),
@OBS CHAR(25),
@ACOMETIDA BIT,
@NUM_MEDIDOR INT,
@NOMBRE VARCHAR(50)
AS
DECLARE @COD_BARRIO INT
SET @COD_BARRIO=(SELECT COD_BARRIO FROM BARRIO WHERE
@NOMBRE_B=NOMBRE_B)
DECLARE @COD_TARIFA INT
SET @COD_TARIFA =(SELECT COD_TARIFA FROM TARIFA WHERE NOMBRE=@NOMBRE)
IF @CI NOT IN (SELECT CI FROM SOCIO)
BEGIN
INSERT INTO SOCIO
VALUES (@CI,@APELLIDO,@OBS)
END
INSERT INTO SOCIO_BARRIO
VALUES (@CI,@COD_BARRIO,@NUM_MEDIDOR,@ACOMETIDA,@COD_TARIFA)
GO
```

INGRESO_TARIFA: Permite ingresar tarifas de cobro de agua potable.

```
CREATE PROCEDURE INGRESO_TARIFA
@NOMBRE VARCHAR(50),
@LIMITE INT,
@COSTOXM3 MONEY,
@OBS CHAR(25),
@P_BASICA MONEY
AS
DECLARE @COD_TARIFA INT

IF (SELECT COUNT(COD_TARIFA) FROM TARIFA)=0
SET @COD_TARIFA = 1
ELSE
(SELECT @COD_TARIFA =MAX (COD_TARIFA)+1 FROM TARIFA)
IF @NOMBRE NOT IN (SELECT NOMBRE FROM TARIFA)
BEGIN
INSERT INTO TARIFA
VALUES (@COD_TARIFA,@P_BASICA,@NOMBRE,@LIMITE,@COSTOXM3,@OBS)
SELECT 'TARIFA INGRESADA '
END
ELSE
BEGIN
```

```
SELECT 'TARIFA YA EXISTE '
END
GO
```

MODF_SOCIO: Permite modificar los datos de un socio.

```
CREATE PROCEDURE MODF_SOCIO
@CI CHAR(10),
@APELLIDO VARCHAR(50),
@NOMBRE_B VARCHAR(50),
@OBS VARCHAR(50),
@NUM_MEDIDOR CHAR(10),
@ACOMETIDA BIT,
@NOMBRE VARCHAR(50)
AS

DECLARE @COD_BARRIO INT
DECLARE @COD_TARIFA INT
SET @COD_BARRIO= (SELECT COD_BARRIO FROM BARRIO WHERE
NOMBRE_B=@NOMBRE_B)
SET @COD_TARIFA =(SELECT COD_TARIFA FROM TARIFA WHERE NOMBRE=@NOMBRE)

IF @CI IN (SELECT CI FROM SOCIO)
BEGIN
IF (SELECT MAX(CI) FROM SOCIO_BARRIO
WHERE CI=@CI AND COD_BARRIO=@COD_BARRIO)= NULL
BEGIN
UPDATE SOCIO_BARRIO
SET COD_BARRIO =@COD_BARRIO
WHERE CI=@CI
END
UPDATE SOCIO
SET APELLIDO=@APELLIDO,
OBS=@OBS
WHERE CI=@CI
UPDATE SOCIO_BARRIO
SET NUM_MEDIDOR=@NUM_MEDIDOR,
ACOMETIDA=@ACOMETIDA,
COD_TARIFA=@COD_TARIFA
WHERE CI=@CI
AND COD_BARRIO =@COD_BARRIO
SELECT 'DATOS MODIFICADOS CON EXITO'
END
GO
```

MODIFICAR_TARIFA: Permite modificar los registros relacionados a una tarifa.

```
CREATE PROCEDURE MODIFICAR_TARIFA
@NOMBRE VARCHAR(50),
@LIMITE INT,
@COSTOXM3 MONEY,
@OBS CHAR(25),
@BASICO MONEY
AS
DECLARE @COD_TARIFA INT
SET @COD_TARIFA = (SELECT COD_TARIFA FROM TARIFA WHERE NOMBRE=@NOMBRE)
UPDATE TARIFA
SET LIMITE=@LIMITE,COSTOXM3=COSTOXM3,OBS=@OBS,BASICO=@BASICO
WHERE COD_TARIFA=@COD_TARIFA
GO
```

PENDIENTE: Permite consultar todas las deudas pendientes de un socio.

```
CREATE PROCEDURE PENDIENTE
@CI VARCHAR(10),
@NOMBRE_B VARCHAR(50)
AS
DECLARE @COD_BARRIO  INT
SET @COD_BARRIO=(SELECT COD_BARRIO FROM BARRIO  WHERE
@NOMBRE_B=NOMBRE_B)
SELECT R.DESCRIPCION,M.MES,R.AÑO,(C.MONTO+SB.MORA)
,C.COD_COBRO,SB.COD_COBRO,medicion,consumo
FROM SOCIO AS S,COBRO AS C,RUBRO AS R,MES AS M,SB_COBRO AS SB, BARRIO AS B
WHERE S.CI = @CI  AND R.COD_MES=M.COD_MES AND @COD_BARRIO=B.COD_BARRIO
AND
R.COD_RUBRO=C.COD_RUBRO AND S.CI=SB.CI AND SB.COD_BARRIO=B.COD_BARRIO AND
C.COD_COBRO=SB.COD_COBRO AND ESTADO=0
GO
```

VER_LISTADOXB: Permite listar a los socios del agua potable agrupados por barrios.

```
CREATE PROCEDURE VER_LISTADOXB
@MESACT INT,
@AÑO INT,
@NOMBRE_B VARCHAR(20)
AS
DECLARE @COD_BARRIO INT
SET @COD_BARRIO =(SELECT COD_BARRIO FROM BARRIO WHERE NOMBRE_B LIKE
@NOMBRE_B)
SELECT APELLIDO as '                    NOMBRES Y APELLIDOS
',NUM_MEDIDOR,SB.CI
FROM SOCIO AS SC, SOCIO_BARRIO AS S,RUBRO AS R,MES AS M,COBRO AS C,SB_COBRO
AS SB
```

```
WHERE  SC.CI=S.CI AND R.COD_MES=M.COD_MES AND R.COD_RUBRO=C.COD_RUBRO
AND C.COD_COBRO=SB.COD_COBRO
AND S.CI=SB.CI  AND DESCRIPCION LIKE 'CONSUMO AGUA POTABLE'
AND (M.COD_MES= @MESACT) And sb.cod_barrio= @COD_BARRIO and año= @AÑO
and s.acometida=1
ORDER BY APELLIDO,AÑO DESC,M.COD_MES DESC
GO
```

VER_PAGOXF: Permite listar los cobros realizados en un intervalo de fechas.

```
CREATE PROCEDURE VER_PAGOXF
@FECHA_I VARCHAR(20),
@FECHA_F VARCHAR(20)
AS
SELECT FECHA,APELLIDO,DESCRIPCION,MES,AÑO,(MONTO+MORA)AS TOTALFROM SOCIO
AS S,RUBRO AS R,MES AS M,COBRO AS C,SB_COBRO AS SB,RECIBO AS RC
WHERE  R.COD_MES=M.COD_MES AND R.COD_RUBRO=C.COD_RUBRO AND
C.COD_COBRO=SB.COD_COBRO
AND S.CI=SB.CI AND RC.NUM_REC= SB.NUM_REC AND ESTADO=1 AND
RC.FECHA_REC BETWEEN @FECHA_I AND @FECHA_F
GO
```

6.1.4.2 CODIGOS RELEVANTES APLICACIÓN CLIENTE

CONEXIÓN A LA BASE DE DATOS

```
If conexion.State = ADODB.adStateOpenThenconexion.CloseEnd If
conexion.Open "Provider=MSDASQL.1;Persist Security Info=False;Data Source=AGUA"
conexion.CursorLocation = adUseClient
```

LLAMADA A PROCEDIMIENTOS ALMACENADOS

```
If conexion.State = ADODB.adStateOpen Then
  Set REG = conexion.Execute("ULTIMO_REC")
EndIf
```

CARGAR DATOS EN UN COMPONENTE LIST

```
If conexion.State = ADODB.adStateOpen Then
  Set REG = conexion.Execute("VER_SOCIO('" & TEXT1.Text & "')")
End If
If REG.RecordCount<> 0 Then
REG.MoveFirst
   Do While Not REG.EOF
      List1.AddItem REG.Fields(1)
REG.MoveNext
Loop
EndIf
```

6.1.4.3 PRUEBAS

El Sistema ha sido probado por el lapso de 6 meses tiempo durante el cual se ha realizado numerosas correcciones y ajustes sugeridos por el Tesorero de la Junta Administradora de Agua Potable de San Gerardo Msc. Martín QuisniaPaguay, centrándose especialmente en la interfaz y los informes.

Durante la etapa de pruebas se ha obtenido los siguientes datos:

- Autonomía del Dispositivo Portable: aproximadamente 4 horas
- Peso del archivo de lecturas de consumo: 40 kb.
- Peso de la aplicación: 468 kb.
- Número de usuarios del agua potable: 353
- Número de recibos emitidos: 3287
- Total Recaudado: $33525.25

La recolección de lecturas de consumo se realiza de domicilio en domicilio en un periodo promedio de 2 días previo al cobro que se realizan el último domingo de cada mes.

CONLUSIONES

- El funcionamiento del protocolo SPI, es muy versátil, permite comunicar toda clase de dispositivos, no necesita hardware especial y ocupa pocas líneas de comunicación para su funcionamiento, ideal para aplicaciones portátiles.
- Los Microcontroladores PIC de gama alta en su mayoría soportan el manejo del protocolo SPI, y poseen el suficiente número de pines para trabajar con Teclado y visualizadores GLCD.
- La implementación de un Dispositivo Electrónico Portable de Datos con memoria SD, que almacene los registros de consumo de agua potable y sea capaz de exportar información a un PC, es una solución económica, viable y funcional para agilitar la recolección y digitalización de datos.
- La memoria SD permite transferir rápidamente los datos desde el dispositivo portable a un computador, sin que sea necesario una conexión física entre estos, debido a que los datos almacenados en la memoria ya cuentan con un formato apropiado para ser leídos desde un computador equipado con Lector de Memorias.
- La Implementación de una aplicación de base de datos que permita procesar la información registrada por el dispositivo, generando recibos para el cobro del consumo de agua potable y reportes de las operaciones realizadas automatiza la mayor parte del trabajo administrativo realizado por la Junta.

RECOMEDACIONES

- Asegurar el voltaje adecuado para el funcionamiento correcto del circuito con los reguladores LM7805 y LM1117.
- Utilizar todas las herramientas de diseño y programación que se tenga disponible para facilitar la obtención de resultados.
- Realizar la simulación del circuito para comprobar el correcto funcionamiento del circuito.
- Dentro del simulador para visualizar todos los pasos importantes ejecutados en nuestro código se recure al terminal virtual y a la comunicación rs232 como depurador.
- Ir comprobando el correcto funcionamiento de los dispositivos implementando módulos individuales en el protoboard para asegurar que se los comprende y utiliza Adecuadamente.
- Participar activamente en foros de internet relacionados con el tema del desarrollo que permita identificar problemas que podrías tener en tu desarrollo y las diferentes soluciones que se les pueden dar.
- Para el correcto desarrollo es necesario entender el funcionamiento de los diferentes dispositivos con respecto al protocolo SPI a nivel de hardware.
- Utilizar el cargador recomendado para la marca y modelo de batería para evitar posibles fallas de funcionamiento.

- Optimizar los recursos de nuestros de nuestro micro controladora través de una depuración continua de nuestro programa.

RESUMEN

Se ha desarrollado un sistema informático y un dispositivo Electrónico portable de recolección de datos para la gestión de cobros del agua potable de la parroquia San Gerardo del cantón Riobamba

Para la construcción del hardware se ha utilizado un micro controlador de gama media, memoria SD, pantalla LCD y teclado numérico cuyo firmware ha sido escrito y compilado con la herramienta PCWH de la empresa CCS Inc. y el Software de gestión con SQL-server2000 y Visual Basic 6.0 empleando técnica de programación orientada a objetos bajo la arquitectura cliente servidor

El resultado del desarrollo ha sido un dispositivo de 11x13.5x3 cm con cuatro horas de autonomía, recargable, económico y de fácil construcción con soporte para memorias SD de hasta 2GB, gestionadas a través del protocolo SPI. Producto de la toma de datos de 353 usuarios del agua potable se ha obtenido un archivo texto plano de aproximadamente 40 kb, el programa de gestión instalado ocupa 468 kb, teniendo como requisito un computador de bajo costo con sistema operativo Windows XP o superior, SQL-server2000 y un lector de memorias pre instalado. Se ha probado el sistema durante el último semestre con periodos de recolección de tres días por mes, emitiéndose 3287 recibos de cobro y recaudándose $ 33525.25 dólares a satisfacción de la dirigencia y los usuarios.

Mediante el sistema informático y el dispositivo electrónico portable se ha mejorado notoriamente le gestión de cobros de agua potable por lo que se recomienda su uso y mantenimiento.

SUMMARY

Has been developed a computer system and aportableelectronicdeviceto collectdata forcollection managementof drinking water from parroquia SanGerardo, cantonRiobamba.

To build thehardwarehas been used a midrange micro-controller,SD Memory Card, LCD display and keypadwhose firmwarehas been written andcompiled withCCSPCWHCompanyInc.and the SoftwareManagement was build with SQL-Server2000 andVisualBasic 6.0 using object-oriented programming technique and underclient-server architecture.

The result of thedevelopment has been adevice of 11x13.5x3cmwith fourhours of battery life, rechargeable, economical and easy to buildwith support forSD memory cardsup to 2GB,managed through theSPIprotocol. Productdata collectionof 353users ofdrinking water hasreceived aplain textfileof approximately 40kb,the management softwareinstalled needs468kb, also as requirement is necessary alow-costcomputerrunning WindowsXPor higher, SQL -Server2000 and amemory readerpreinstalled.the system has been testingduring the lastsemester withcollection periodsof threedays per month, delivering 3287 paymentreceipts and collected$ 33,525.25dollars tothe satisfaction of theleadersand users.

Using thecomputer system andportableelectronic device the management of collectionsofwater improved greatly so it isrecommended for useand maintenance.

BIBLIOGRAFÍA

- **BOYLESTAD, R. y NASHELSKY, L**. Electrónica: teoría de circuitos y dispositivos electrónicos. 8va ed. México DF.- México, pearson educación, 2003.
- **FORO DE HARDWARE**

  http://www.ucontrol.com.ar/forosmf/programacion-en-c/sd-card-libreria-fat16-libreria-a-nivel-hardware-%28ccs-c18-c30-ect-%29/

  14/01/2011
- **INFORMACION COMPARATIVA DE MICRO CONTROLADORES**

  http://www.monografias.com/trabajos34/microcontroladores-genericos/microcontroladores-genericos.shtml#intro

  13/10/2010

  http://www.forosdeelectronica.com/wiki/microcontroladores

  16/12/2010

  http://es.scribd.com/doc/44197510/PIC-18F4550

  18/12/2010
- **INFORMACION DETALLADA SOBRE MICROCONTROLADORES**

  http://www.microchip.com

  13/10/2010

- **MARTIN CUENCA, E.; ANGULO J. M y ANGULO**. Microcontroladores PIC: lasolución está en un chip. España.- Madrid, paraninfo-ITP, 2001
- **PALLÁS ARENEY, RAMÓN y VALDÉS PÉREZ, FERNANDO E.**Microcontroladores:fundamentos y aplicaciones con PIC. Madrid, Marcombo, 2007
- **PROTOCOLO SPI**

  http://es.wikipedia.org/wiki/Serial_Peripheral_Interface

  10/10/2010

ANEXOS

ANEXO 1

Datasheet de los componentes del

dispositivo portable

PIC18F2455/2550/4455/4550
Data Sheet

28/40/44-Pin, High-Performance, Enhanced Flash, USB Microcontrollers With nano Watt Technology

Preliminary DS39632C

PIC18F2455/2550/4455/4550

28/40/44-Pin, High-Performance, Enhanced Flash, USB Microcontrollers with nanoWatt Technology

Universal Serial Bus Features:

- USB V2.0 Compliant
- Low Speed (1.5 Mb/s) and Full Speed (12 Mb/s)
- Supports Control, Interrupt, Isochronous and Bulk Transfers
- Supports up to 32 Endpoints (16 bidirectional)
- 1-Kbyte Dual Access RAM for USB
- On-Chip USB Transceiver with On-Chip Voltage Regulator
- Interface for Off-Chip USB Transceiver
- Streaming Parallel Port (SPP) for USB streaming transfers (40/44-pin devices only)

Power-ManagedModes:

- Run: CPU on, peripheralson
- Idle: CPU off, peripheralson
- Sleep: CPU off, peripherals off
- Idle mode currents down to 5.8 µA typical
- Sleep mode currents down to 0.1 µA typical
- Timer1 Oscillator: 1.1 µA typical, 32 kHz, 2V
- WatchdogTimer: 2.1 µA typical
- Two-SpeedOscillatorStart-up

Flexible OscillatorStructure:

- Four Crystal modes, including High Precision PLL for USB
- Two External Clock modes, up to 48 MHz
- InternalOscillator Block:
  - 8 user-selectable frequencies, from 31 kHz to 8 MHz
  - User-tunable to compensate for frequency drift
- SecondaryOscillatorusing Timer1 @ 32 kHz
- Dual Oscillator options allow microcontroller and USB module to run at different clock speeds
- Fail-SafeClock Monitor:
  - Allows for safe shutdown if any clock stops

PeripheralHighlights:

- High-Current Sink/Source: 25 mA/25 mA
- ThreeExternalInterrupts
- Four Timer modules (Timer0 to Timer3)
- Up to 2 Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 5.2 ns (TCY/16)
  - Compare is 16-bit, max. resolution 83.3 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Enhanced Capture/Compare/PWM (ECCP) module:
  - Multiple output modes
  - Selectablepolarity
  - Programmabledead time
  - Auto-shutdown and auto-restart
- Enhanced USART module:
  - LIN bus support
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I^2C™ Master and Slave modes
- 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D) with Programmable Acquisition Time
- Dual Analog Comparators with Input Multiplexing

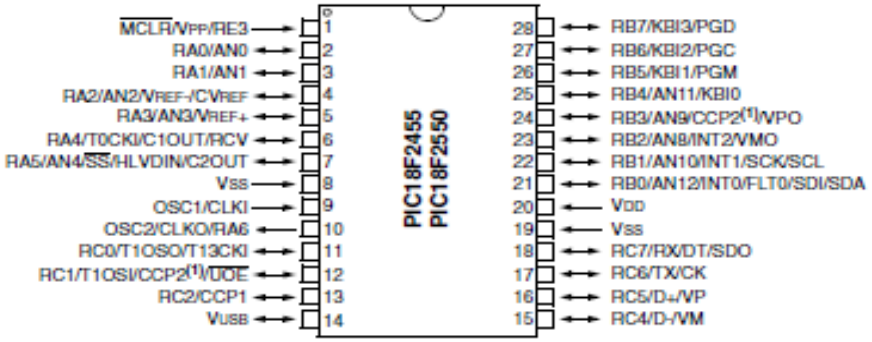
SpecialMicrocontrollerFeatures:

- C Compiler Optimized Architecture with optional Extended Instruction Set
- 100,000 Erase/Write Cycle Enhanced Flash Program Memory typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory typical
- Flash/Data EEPROM Retention: > 40 years
- Self-Programmableunder Software Control
- PriorityLevelsforInterrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended WatchdogTimer (WDT):
  - Programmable period from 41 ms to 131s
- ProgrammableCodeProtection
- Single-Supply 5V In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins
- Optional dedicated ICD/ICSP port (44-pin devices only)
- Wide Operating Voltage Range (2.0V to 5.5V)

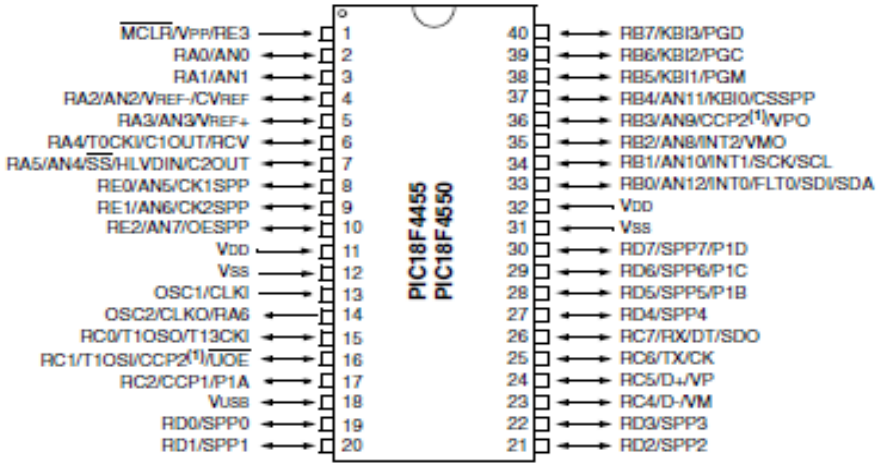
Device	ProgramMemory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EAUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I^2C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

Pin Diagrams

28-Pin PDIP, SOIC



40-Pin PDIP



Note 1: RB3 is the alternate pin for CCP2 multiplexing.

TABLE 1-2: PIC18F2455/2550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	PDIP, SOIC			
				PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/AN12/INT0/FLT0/ SDI/SDA	21			
RB0		I/O	TTL	Digital I/O.
AN12		I	Analog	Analog input 12.
INT0		I	ST	External interrupt 0.
FLT0		I	ST	PWM Fault input (CCP1 module).
SDI		I	ST	SPI data in.
SDA		I/O	ST	I^2C™ data I/O.
RB1/AN10/INT1/SCK/ SCL	22			
RB1		I/O	TTL	Digital I/O.
AN10		I	Analog	Analog input 10.
INT1		I	ST	External interrupt 1.
SCK		I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL		I/O	ST	Synchronous serial clock input/output for I^2C mode.
RB2/AN8/INT2/VMO	23			
RB2		I/O	TTL	Digital I/O.
AN8		I	Analog	Analog input 8.
INT2		I	ST	External interrupt 2.
VMO		O	—	External USB transceiver VMO output.
RB3/AN9/CCP2/VPO	24			
RB3		I/O	TTL	Digital I/O.
AN9		I	Analog	Analog input 9.
CCP2(1)		I/O	ST	Capture 2 input/Compare 2 output/PWM 2 output.
VPO		O	—	External USB transceiver VPO output.
RB4/AN11/KBI0	25			
RB4		I/O	TTL	Digital I/O.
AN11		I	Analog	Analog input 11.
KBI0		I	TTL	Interrupt-on-change pin.
RB5/KBI1/PGM	26			
RB5		I/O	TTL	Digital I/O.
KBI1		I	TTL	Interrupt-on-change pin.
PGM		I/O	ST	Low-Voltage ICSP™ Programming enable pin.
RB6/KBI2/PGC	27			
RB6		I/O	TTL	Digital I/O.
KBI2		I	TTL	Interrupt-on-change pin.
PGC		I/O	ST	In-Circuit Debugger and ICSP programming clock pin.
RB7/KBI3/PGD	28			
RB7		I/O	TTL	Digital I/O.
KBI3		I	TTL	Interrupt-on-change pin.
PGD		I/O	ST	In-Circuit Debugger and ICSP programming data pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTC is a bidirectional I/O port.
RC0/T1OSO/T13CKI	15	34	32			
RC0				I/O	ST	Digital I/O.
T1OSO				O	—	Timer1 oscillator output.
T13CKI				I	ST	Timer1/Timer3 external clock input.
RC1/T1OSI/CCP2/$\overline{UOE}$	16	35	35			
RC1				I/O	ST	Digital I/O.
T1OSI				I	CMOS	Timer1 oscillator input.
CCP2(2)				I/O	ST	Capture 2 input/Compare 2 output/PWM 2 output.
$\overline{UOE}$				O	—	External USB transceiver $\overline{OE}$ output.
RC2/CCP1/P1A	17	36	36			
RC2				I/O	ST	Digital I/O.
CCP1				I/O	ST	Capture 1 input/Compare 1 output/PWM 1 output.
P1A				O	TTL	Enhanced CCP1 PWM output, channel A.
RC4/D-/VM	23	42	42			
RC4				I	TTL	Digital input.
D-				I/O	—	USB differential minus line (input/output).
VM				I	TTL	External USB transceiver VM input.
RC5/D+/VP	24	43	43			
RC5				I	TTL	Digital input.
D+				I/O	—	USB differential plus line (input/output).
VP				I	TTL	External USB transceiver VP input.
RC6/TX/CK	25	44	44			
RC6				I/O	ST	Digital I/O.
TX				O	—	EUSART asynchronous transmit.
CK				I/O	ST	EUSART synchronous clock (see RX/DT).
RC7/RX/DT/SDO	26	1	1			
RC7				I/O	ST	Digital I/O.
RX				I	ST	EUSART asynchronous receive.
DT				I/O	ST	EUSART synchronous data (see TX/CK).
SDO				O	—	SPI data out.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the $\overline{DEBUG}$ Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTD is a bidirectional I/O port or a Streaming Parallel Port (SPP). These pins have TTL input buffers when the SPP module is enabled.
RD0/SPP0	19	38	38			
RD0				I/O	ST	Digital I/O.
SPP0				I/O	TTL	Streaming Parallel Port data.
RD1/SPP1	20	39	39			
RD1				I/O	ST	Digital I/O.
SPP1				I/O	TTL	Streaming Parallel Port data.
RD2/SPP2	21	40	40			
RD2				I/O	ST	Digital I/O.
SPP2				I/O	TTL	Streaming Parallel Port data.
RD3/SPP3	22	41	41			
RD3				I/O	ST	Digital I/O.
SPP3				I/O	TTL	Streaming Parallel Port data.
RD4/SPP4	27	2	2			
RD4				I/O	ST	Digital I/O.
SPP4				I/O	TTL	Streaming Parallel Port data.
RD5/SPP5/P1B	28	3	3			
RD5				I/O	ST	Digital I/O.
SPP5				I/O	TTL	Streaming Parallel Port data.
P1B				O	—	Enhanced CCP1 PWM output, channel B.
RD6/SPP6/P1C	29	4	4			
RD6				I/O	ST	Digital I/O.
SPP6				I/O	TTL	Streaming Parallel Port data.
P1C				O	—	Enhanced CCP1 PWM output, channel C.
RD7/SPP7/P1D	30	5	5			
RD7				I/O	ST	Digital I/O.
SPP7				I/O	TTL	Streaming Parallel Port data.
P1D				O	—	Enhanced CCP1 PWM output, channel D.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the DEBUG Configuration bit is cleared.

TABLE 1-3: PIC18F4455/4550 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	QFN	TQFP			
						PORTE is a bidirectional I/O port.
RE0/AN5/CK1SPP	8	25	25			
RE0				I/O	ST	Digital I/O.
AN5				I	Analog	Analog input 5.
CK1SPP				O	—	SPP clock 1 output.
RE1/AN6/CK2SPP	9	26	26			
RE1				I/O	ST	Digital I/O.
AN6				I	Analog	Analog input 6.
CK2SPP				O	—	SPP clock 2 output.
RE2/AN7/OESPP	10	27	27			
RE2				I/O	ST	Digital I/O.
AN7				I	Analog	Analog input 7.
OESPP				O	—	SPP output enable output.
RE3	—	—	—	—	—	See $\overline{MCLR}$/VPP/RE3 pin.
VSS	12, 31	6, 30, 31	6, 29	P	—	Ground reference for logic and I/O pins.
VDD	11, 32	7, 8, 28, 29	7, 28	P	—	Positive supply for logic and I/O pins.
VUSB	18	37	37	O	—	Internal USB 3.3V voltage regulator output.
NC/ICCK/ICPGC(3)	—	—	12			No Connect or dedicated ICD/ICSP™ port clock.
ICCK				I/O	ST	In-Circuit Debugger clock.
ICPGC				I/O	ST	ICSP programming clock.
NC/ICDT/ICPGD(3)	—	—	13			No Connect or dedicated ICD/ICSP port clock.
ICDT				I/O	ST	In-Circuit Debugger data.
ICPGD				I/O	ST	ICSP programming data.
NC/$\overline{ICRST}$/ICVPP(3)	—	—	33			No Connect or dedicated ICD/ICSP port Reset.
$\overline{ICRST}$				I	—	Master Clear (Reset) input.
ICVPP				P	—	Programming voltage input.
NC/ICPORTS(3)	—	—	34	P	—	No Connect or 28-pin device emulation.
ICPORTS						Enable 28-pin device emulation when connected to Vss.
NC	—	13	—	—	—	No Connect.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: Alternate assignment for CCP2 when CCP2MX Configuration bit is cleared.

2: Default assignment for CCP2 when CCP2MX Configuration bit is set.

3: These pins are No Connect unless the ICPRT Configuration bit is set. For NC/ICPORTS, the pin is No Connect unless ICPRT is set and the $\overline{DEBUG}$ Configuration bit is cleared.

2.0 OSCILLATOR CONFIGURATIONS

2.1 Overview

Devices in the PIC18F2455/2550/4455/4550 family incorporate a different oscillator and microcontroller clock system than previous PIC18F devices. The addition of the USB module, with its unique requirements for a stable clock source, make it necessary to provide a separate clock source that is compliant with both USB low-speed and full-speed specifications.

To accommodate these requirements, PIC18F2455/2550/4455/4550 devices include a new clock branch to provide a 48 MHz clock for full-speed USB operation. Since it is driven from the primary clock source, an additional system of prescalers and postscalers has been added to accommodate a wide range of oscillator frequencies. An overview of the oscillator structure is shown in Figure 2-1.

Other oscillator features used in PIC18 enhanced microcontrollers, such as the internal oscillator block and clock switching, remain the same. They are discussed later in this chapter.

2.1.1 OSCILLATOR CONTROL

The operation of the oscillator in PIC18F2455/2550/4455/4550 devices is controlled through two Configuration registers and two control registers. Configuration registers, CONFIG1L and CONFIG1H, select the oscillator mode and USB prescaler/postscaler options. As Configuration bits, these are set when the device is programmed and left in that configuration until the device is reprogrammed.

The OSCCON register (Register 2-2) selects the Active Clock mode; it is primarily used in controlling clock switching in power-managed modes. Its use is discussed in **Section 2.4.1 "Oscillator Control Register"**.

The OSCTUNE register (Register 2-1) is used to trim the INTRC frequency source, as well as select the low-frequency clock source that drives several special features. Its use is described in **Section 2.2.5.2 "OSCTUNE Register"**.

2.2 Oscillator Types

PIC18F2455/2550/4455/4550 devices can be operated in twelve distinct oscillator modes. In contrast with previous PIC18 enhanced microcontrollers, four of these modes involve the use of two oscillator types at once. Users can program the FOSC3:FOSC0 Configuration bits to select one of these modes:

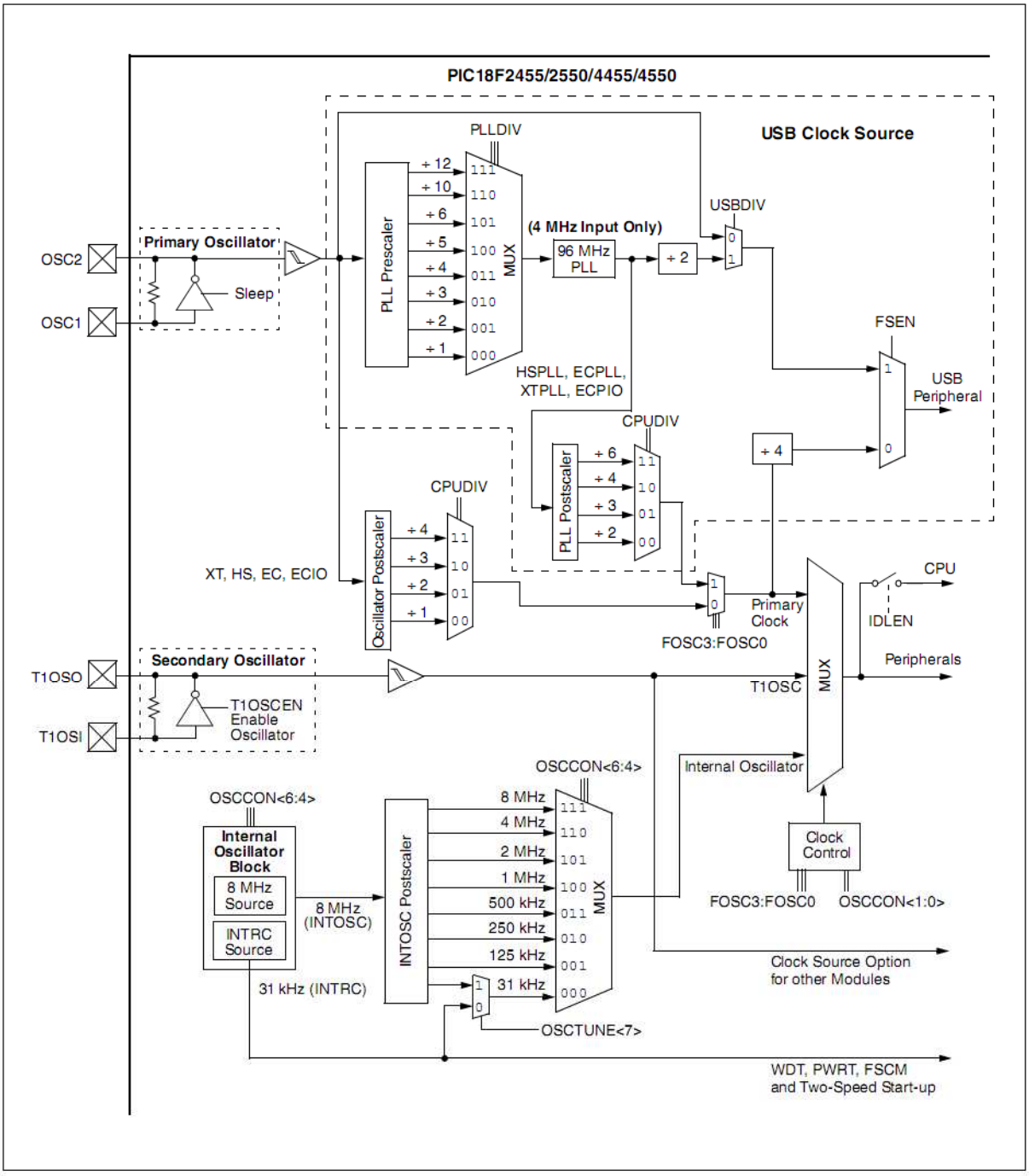
1. XT Crystal/Resonator
2. XTPLL Crystal/Resonator with PLL enabled
3. HS High-Speed Crystal/Resonator
4. HSPLL High-Speed Crystal/Resonator with PLL enabled
5. EC External Clock with FOSC/4 output
6. ECIO External Clock with I/O on RA6
7. ECPLL External Clock with PLL enabled and FOSC/4 output on RA6
8. ECPIO External Clock with PLL enabled, I/O on RA6
9. INTHS Internal Oscillator used as microcontroller clock source, HS Oscillator used as USB clock source
10. INTXT Internal Oscillator used as microcontroller clock source, XT Oscillator used as USB clock source
11. INTIO Internal Oscillator used as microcontroller clock source, EC Oscillator used as USB clock source, digital I/O on RA6
12. INTCKO Internal Oscillator used as microcontroller clock source, EC Oscillator used as USB clock source, FOSC/4 output on RA6

2.2.1 OSCILLATOR MODES AND USB OPERATION

Because of the unique requirements of the USB module, a different approach to clock operation is necessary. In previous PICmicro® devices, all core and peripheral clocks were driven by a single oscillator source; the usual sources were primary, secondary or the internal oscillator. With PIC18F2455/2550/4455/4550 devices, the primary oscillator becomes part of the USB module and cannot be associated to any other clock source. Thus, the USB module must be clocked from the primary clock source; however, the microcontroller core and other peripherals can be separately clocked from the secondary or internal oscillators as before.

Because of the timing requirements imposed by USB, an internal clock of either 6 MHz or 48 MHz is required while the USB module is enabled. Fortunately, the microcontroller and other peripherals are not required to run at this clock speed when using the primary oscillator. There are numerous options to achieve the USB module clock requirement and still provide flexibility for clocking the rest of the device from the primary oscillator source. These are detailed in **Section 2.3 "Oscillator Settings for USB"**.

FIGURE 2-1: PIC18F2455/2550/4455/4550 CLOCK DIAGRAM



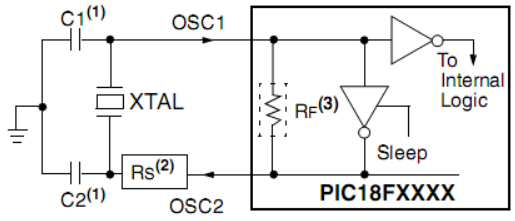
2.2.2 CRYSTAL OSCILLATOR/CERAMIC RESONATORS

In HS, HSPLL, XT and XTPLL Oscillator modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation. Figure 2-2 shows the pin connections.

The oscillator design requires the use of a parallel cut crystal.

Note:	Use of a series cut crystal may give a frequency out of the crystal manufacturer's specifications.

FIGURE 2-2: CRYSTAL/CERAMIC RESONATOR OPERATION (XT, HS OR HSPLL CONFIGURATION)



Note 1: See Table 2-1 and Table 2-2 for initial values of C1 and C2.
2: A series resistor (RS) may be required for AT strip cut crystals.
3: RF varies with the oscillator mode chosen.

TABLE 2-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

Typical Capacitor Values Used:			
Mode	**Freq**	**OSC1**	**OSC2**
XT	4.0 MHz	33 pF	33 pF
HS	8.0 MHz 16.0 MHz	27 pF 22 pF	27 pF 22 pF

Capacitor values are for design guidance only.

These capacitors were tested with the resonators listed below for basic start-up and operation. **These values are not optimized**.

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following Table 2-2 for additional information.

Resonators Used:
4.0 MHz
8.0 MHz
16.0 MHz

TABLE 2-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	**Crystal Freq**	**Typical Capacitor Values Tested:**	
		C1	**C2**
XT	4 MHz	27 pF	27 pF
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

Capacitor values are for design guidance only.

These capacitors were tested with the crystals listed below for basic start-up and operation. **These values are not optimized.**

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following this table for additional information.

Crystals Used:
4 MHz
8 MHz
20 MHz

Note 1: Higher capacitance increases the stability of oscillator but also increases the start-up time.
2: When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use the HS mode or switch to a crystal oscillator.
3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.
4: Rs may be required to avoid overdriving crystals with low drive level specification.
5: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

An internal postscaler allows users to select a clock frequency other than that of the crystal or resonator. Frequency division is determined by the CPUDIV Configuration bits. Users may select a clock frequency of the oscillator frequency, or 1/2, 1/3 or 1/4 of the frequency.

An external clock may also be used when the microcontroller is in HS Oscillator mode. In this case, the OSC2/CLKO pin is left open (Figure 2-3).

FAIRCHILD
SEMICONDUCTOR®

March 2008

LM78XX/LM78XXA
3-Terminal 1A Positive Voltage Regulator

Features

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

General Description

The LM78XX series of three terminal positive regulators are available in the TO-220 package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

Ordering Information

Product Number	Output Voltage Tolerance	Package	Operating Temperature
LM7805CT	±4%	TO-220	-40°C to +125°C
LM7806CT			
LM7808CT			
LM7809CT			
LM7810CT			
LM7812CT			
LM7815CT			
LM7818CT			
LM7824CT			
LM7805ACT	±2%		0°C to +125°C
LM7806ACT			
LM7808ACT			
LM7809ACT			
LM7810ACT			
LM7812ACT			
LM7815ACT			
LM7818ACT			
LM7824ACT			

Block Diagram

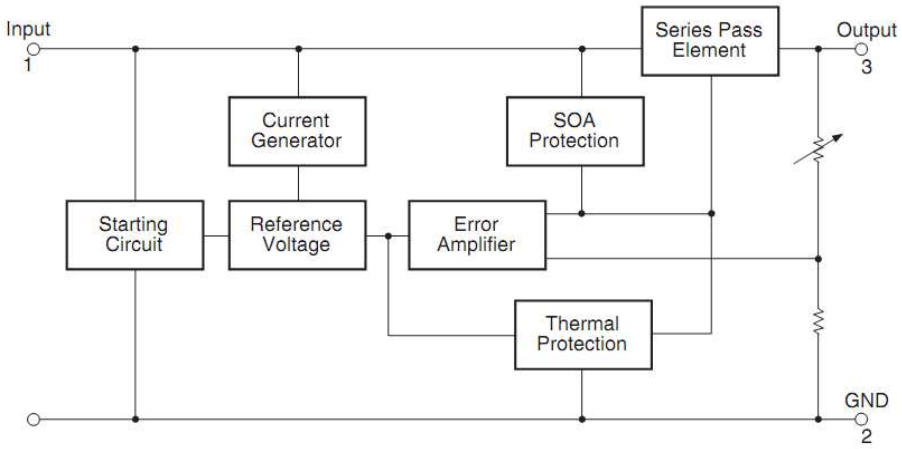


Figure 1.

Pin Assignment

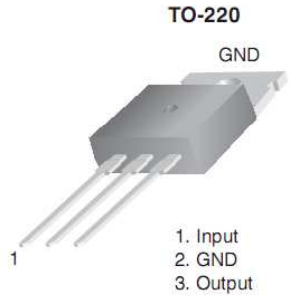


Figure 2.

Absolute Maximum Ratings

Absolute maximum ratings are those values beyond which damage to the device may occur. The datasheet specifications should be met, without exception, to ensure that the system design is reliable over its power supply, temperature, and output/input loading variables. Fairchild does not recommend operation outside datasheet specifications.

Symbol	Parameter		Value	Unit
V_I	Input Voltage	V_O = 5V to 18V	35	V
		V_O = 24V	40	V
$R_{\theta JC}$	Thermal Resistance Junction-Cases (TO-220)		5	°C/W
$R_{\theta JA}$	Thermal Resistance Junction-Air (TO-220)		65	°C/W
T_{OPR}	Operating Temperature Range	LM78xx	-40 to +125	°C
		LM78xxA	0 to +125	
T_{STG}	Storage Temperature Range		-65 to +150	°C

Electrical Characteristics (LM7805)

Refer to the test circuits. -40°C < T_J < 125°C, I_O = 500mA, V_I = 10V, C_I = 0.1µF, unless otherwise specified.

Symbol	Parameter	Conditions		Min.	Typ.	Max.	Unit
V_O	Output Voltage	T_J = +25°C		4.8	5.0	5.2	V
		5mA ≤ I_O ≤ 1A, P_O ≤ 15W, V_I = 7V to 20V		4.75	5.0	5.25	
Regline	Line Regulation(1)	T_J = +25°C	V_O = 7V to 25V	–	4.0	100	mV
			V_I = 8V to 12V	–	1.6	50.0	
Regload	Load Regulation(1)	T_J = +25°C	I_O = 5mA to 1.5A	–	9.0	100	mV
			I_O = 250mA to 750mA	–	4.0	50.0	
I_Q	Quiescent Current	T_J = +25°C		–	5.0	8.0	mA
ΔI_Q	Quiescent Current Change	I_O = 5mA to 1A		–	0.03	0.5	mA
		V_I = 7V to 25V		–	0.3	1.3	
$\Delta V_O/\Delta T$	Output Voltage Drift(2)	I_O = 5mA		–	-0.8	–	mV/°C
V_N	Output Noise Voltage	f = 10Hz to 100kHz, T_A = +25°C		–	42.0	–	µV/V_O
RR	Ripple Rejection(2)	f = 120Hz, V_O = 8V to 18V		62.0	73.0	–	dB
V_{DROP}	Dropout Voltage	I_O = 1A, T_J = +25°C		–	2.0	–	V
r_O	Output Resistance(2)	f = 1kHz		–	15.0	–	mΩ
I_{SC}	Short Circuit Current	V_I = 35V, T_A = +25°C		–	230	–	mA
I_{PK}	Peak Current(2)	T_J = +25°C		–	2.2	–	A

Notes:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.
2. These parameters, although guaranteed, are not 100% tested in production.

Typical Performance Characteristics

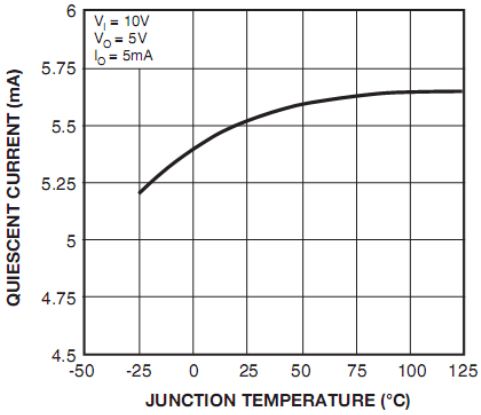


Figure 3. Quiescent Current

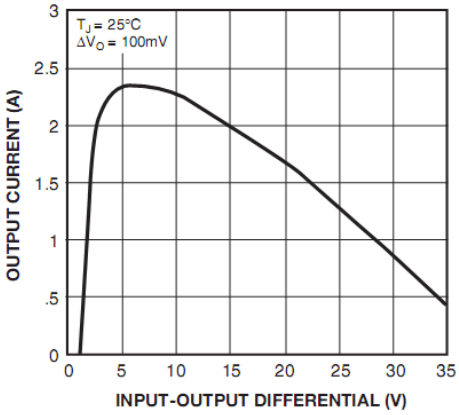


Figure 4. Peak Output Current

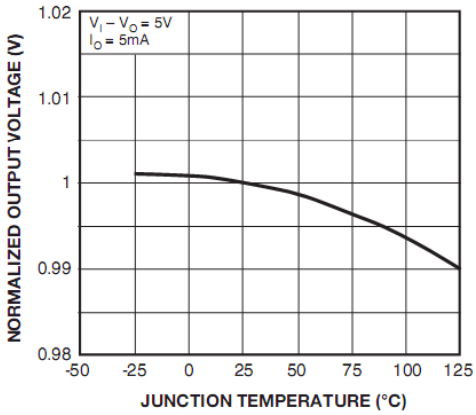


Figure 5. Output Voltage

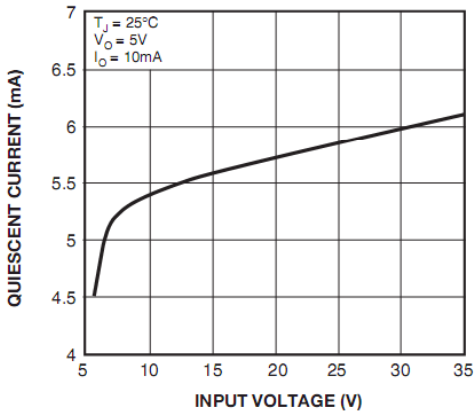


Figure 6. Quiescent Current

Typical Applications

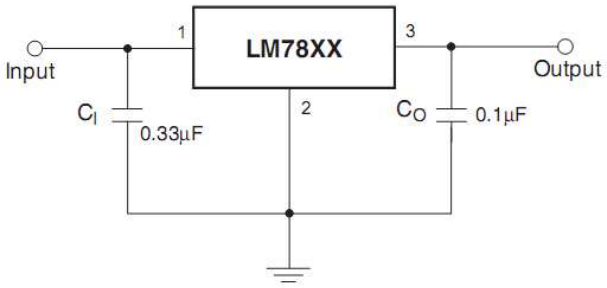


Figure 7. DC Parameters

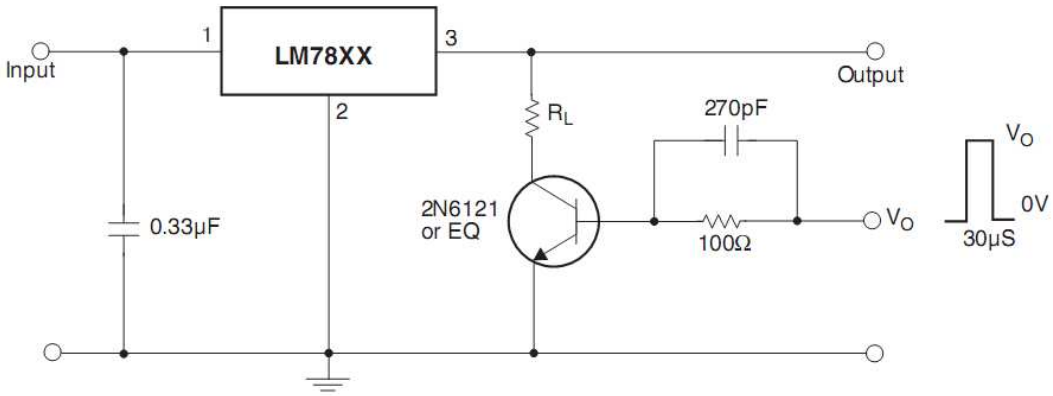


Figure 8. Load Regulation

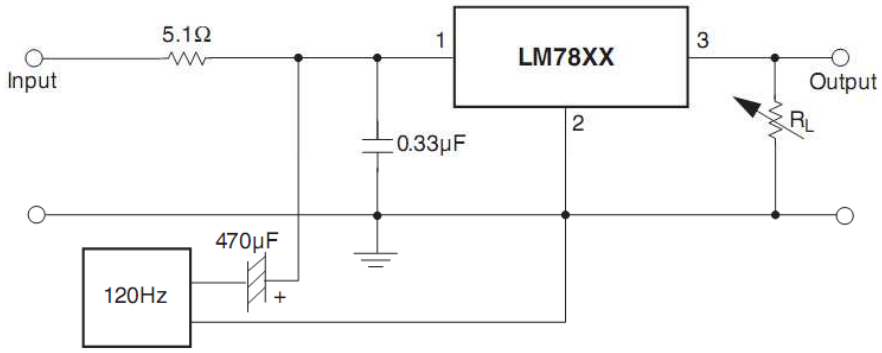


Figure 9. Ripple Rejection

National Semiconductor

April 2006

LM1117/LM1117I
800mA Low-Dropout Linear Regulator

General Description

The LM1117 is a series of low dropout voltage regulators with a dropout of 1.2V at 800mA of load current. It has the same pin-out as National Semiconductor's industry standard LM317.

The LM1117 is available in an adjustable version, which can set the output voltage from 1.25V to 13.8V with only two external resistors. In addition, it is also available in five fixed voltages, 1.8V, 2.5V, 2.85V, 3.3V, and 5V.

The LM1117 offers current limiting and thermal shutdown. Its circuit includes a zener trimmed bandgap reference to assure output voltage accuracy to within ±1%.

The LM1117 series is available in LLP, TO-263, SOT-223, TO-220, and TO-252 D-PAK packages. A minimum of 10µF tantalum capacitor is required at the output to improve the transient response and stability.

Features

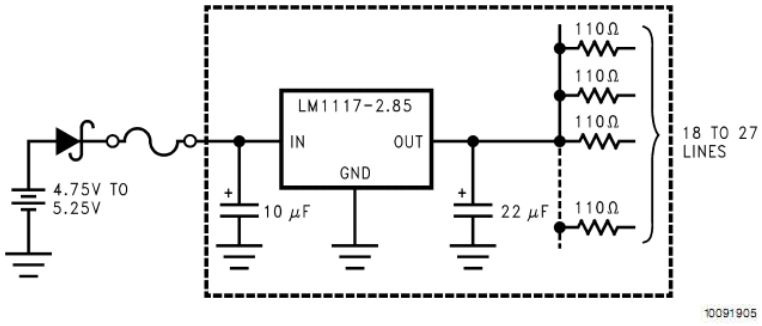
- Available in 1.8V, 2.5V, 2.85V, 3.3V, 5V, and Adjustable Versions
- Space Saving SOT-223 and LLP Packages
- Current Limiting and Thermal Protection
- Output Current 800mA
- Line Regulation 0.2% (Max)
- Load Regulation 0.4% (Max)
- Temperature Range
  - — LM1117 0°C to 125°C
  - — LM1117I –40°C to 125°C

Applications

- 2.85V Model for SCSI-2 Active Termination
- Post Regulator for Switching DC/DC Converter
- High Efficiency Linear Regulators
- Battery Charger
- Battery Powered Instrumentation

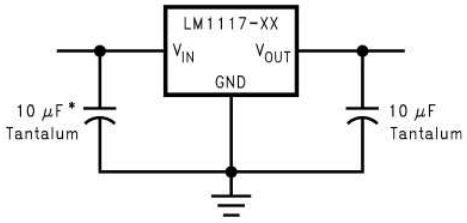
Typical Application

Active Terminator for SCSI-2 Bus



10091905

Fixed Output Regulator



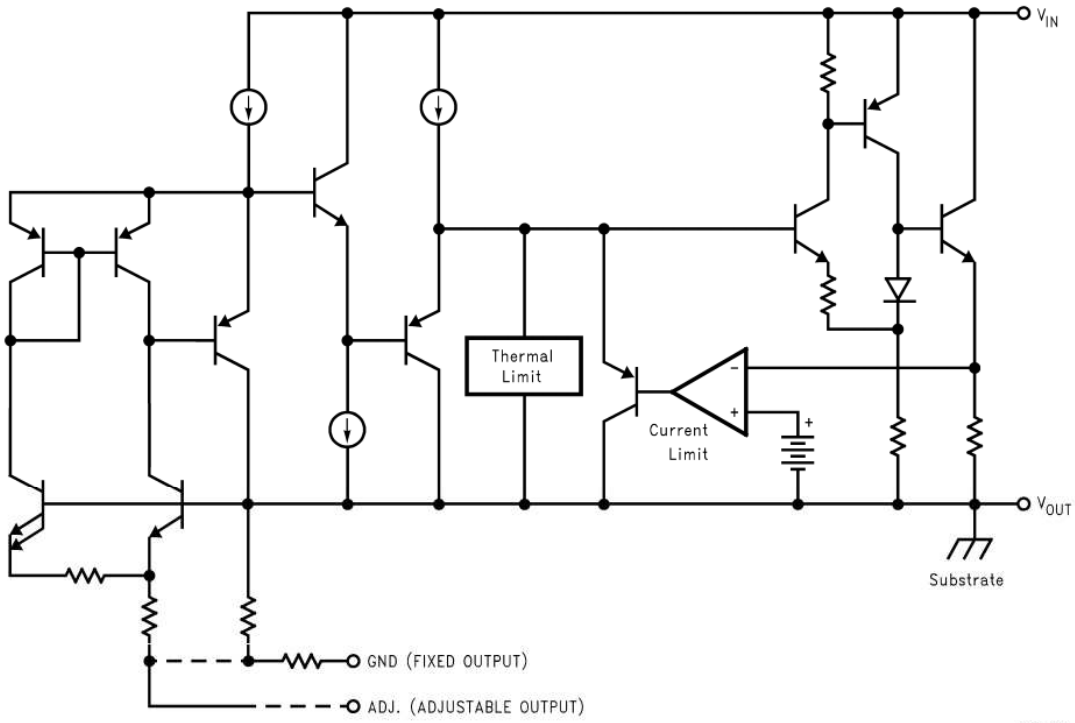
* Required if the regulator is located far from the power supply filter.

10091928

Ordering Information

Package	Temperature Range	Part Number	Packaging Marking	Transport Media	NSC Drawing
3-lead SOT-223	0˚C to +125˚C	LM1117MPX-ADJ	N03A	Tape and Reel	MP04A
		LM1117MPX-1.8	N12A	Tape and Reel	
		LM1117MPX-2.5	N13A	Tape and Reel	
		LM1117MPX-2.85	N04A	Tape and Reel	
		LM1117MPX-3.3	N05A	Tape and Reel	
		LM1117MPX-5.0	N06A	Tape and Reel	
	–40˚C to +125˚C	LM1117IMPX-ADJ	N03B	Tape and Reel	
		LM1117IMPX-3.3	N05B	Tape and Reel	
		LM1117IMPX-5.0	N06B	Tape and Reel	
3-lead TO-220	0˚C to +125˚C	LM1117T-ADJ	LM1117T-ADJ	Rails	T03B
		LM1117T-1.8	LM1117T-1.8	Rails	
		LM1117T-2.5	LM1117T-2.5	Rails	
		LM1117T-2.85	LM1117T-2.85	Rails	
		LM1117T-3.3	LM1117T-3.3	Rails	
		LM1117T-5.0	LM1117T-5.0	Rails	
3-lead TO-252	0˚C to +125˚C	LM1117DTX-ADJ	LM1117DT-ADJ	Tape and Reel	TD03B
		LM1117DTX-1.8	LM1117DT-1.8	Tape and Reel	
		LM1117DTX-2.5	LM1117DT-2.5	Tape and Reel	
		LM1117DTX-2.85	LM1117DT-2.85	Tape and Reel	
		LM1117DTX-3.3	LM1117DT-3.3	Tape and Reel	
		LM1117DTX-5.0	LM1117DT-5.0	Tape and Reel	
	–40˚C to +125˚C	LM1117IDTX-ADJ	LM1117IDT-ADJ	Tape and Reel	
		LM1117IDTX-3.3	LM1117IDT-3.3	Tape and Reel	
		LM1117IDTX-5.0	LM1117IDT-5.0	Tape and Reel	
8-lead LLP	0˚C to +125˚C	LM1117LDX-ADJ	1117ADJ	Tape and Reel	LDC08A
		LM1117LDX-1.8	1117-18	Tape and Reel	
		LM1117LDX-2.5	1117-25	Tape and Reel	
		LM1117LDX-2.85	1117-28	Tape and Reel	
		LM1117LDX-3.3	1117-33	Tape and Reel	
		LM1117LDX-5.0	1117-50	Tape and Reel	
	–40˚C to 125˚C	LM1117ILDX-ADJ	1117IAD	Tape and Reel	
		LM1117ILDX-3.3	1117I33	Tape and Reel	
		LM1117ILDX-5.0	1117I50	Tape and Reel	
TO-263	0˚C to +125˚C	LM1117SX-ADJ	LM1117SADJ	Tape and Reel	TS3B
		LM1117SX-2.85	LM1117S2.85	Tape and Reel	
		LM1117SX-3.3	LM1117S3.3	Tape and Reel	
		LM1117SX-5.0	LM1117S5.0	Tape and Reel	

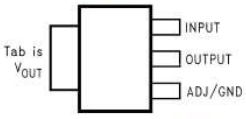
Block Diagram



10091901

Connection Diagrams

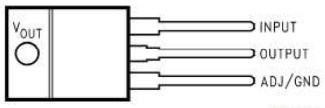
SOT-223



10091904

Top View

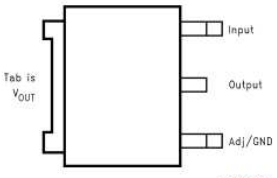
TO-220



10091902

Top View

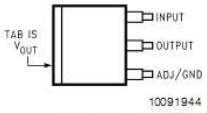
TO-252



10091938

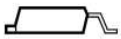
Top View

TO-263



10091944

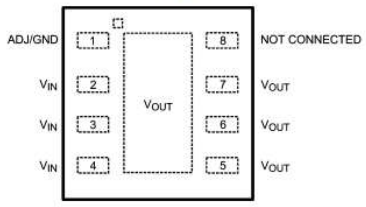
Top View



10091945

Side View

LLP



10091946

When using the LLP package
Pins 2, 3 & 4 must be connected together and
Pins 5, 6 & 7 must be connected together

Top View

Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Maximum Input Voltage (V_{IN} to GND)	20V
Power Dissipation (Note 2)	Internally Limited
Junction Temperature (T_J) (Note 2)	150˚C
Storage Temperature Range	-65˚C to 150˚C
Lead Temperature	
TO-220 (T) Package	260˚C, 10 sec
SOT-223 (IMP) Package	260˚C, 4 sec
ESD Tolerance (Note 3)	2000V

Operating Ratings (Note 1)

Input Voltage (V_{IN} to GND)	15V
Junction Temperature Range (T_J)(Note 2)	
LM1117	0˚C to 125˚C
LM1117I	–40˚C to 125˚C

LM1117 Electrical Characteristics

Typicals and limits appearing in normal type apply for T_J = 25˚C. Limits appearing in **Boldface** type apply over the entire junction temperature range for operation, 0˚C to 125˚C.

Symbol	Parameter	Conditions	Min (Note 5)	Typ (Note 4)	Max (Note 5)	Units
V_{REF}	Reference Voltage	LM1117-ADJ				
		I_{OUT} = 10mA, V_{IN}-V_{OUT} = 2V, T_J = 25˚C	1.238	1.250	1.262	V
		10mA ≤ I_{OUT} ≤ 800mA, 1.4V ≤ V_{IN}-V_{OUT} ≤ 10V	**1.225**	1.250	**1.270**	V
V_{OUT}	Output Voltage	LM1117-1.8				
		I_{OUT} = 10mA, V_{IN} = 3.8V, T_J = 25˚C	1.782	1.800	1.818	V
		0 ≤ I_{OUT} ≤ 800mA, 3.2V ≤ V_{IN} ≤ 10V	**1.746**	1.800	**1.854**	V
		LM1117-2.5				
		I_{OUT} = 10mA, V_{IN} = 4.5V, T_J = 25˚C	2.475	2.500	2.525	V
		0 ≤ I_{OUT} ≤ 800mA, 3.9V ≤ V_{IN} ≤ 10V	**2.450**	2.500	**2.550**	V
		LM1117-2.85				
		I_{OUT} = 10mA, V_{IN} = 4.85V, T_J = 25˚C	2.820	2.850	2.880	V
		0 ≤ I_{OUT} ≤ 800mA, 4.25V ≤ V_{IN} ≤ 10V	**2.790**	2.850	**2.910**	V
		0 ≤ I_{OUT} ≤ 500mA, V_{IN} = 4.10V	**2.790**	2.850	**2.910**	V
		LM1117-3.3				
		I_{OUT} = 10mA, V_{IN} = 5V T_J = 25˚C	3.267	3.300	3.333	V
		0 ≤ I_{OUT} ≤ 800mA, 4.75V≤ V_{IN} ≤ 10V	**3.235**	3.300	**3.365**	V
		LM1117-5.0				
		I_{OUT} = 10mA, V_{IN} = 7V, T_J = 25˚C	4.950	5.000	5.050	V
		0 ≤ I_{OUT} ≤ 800mA, 6.5V ≤ V_{IN} ≤ 12V	**4.900**	5.000	**5.100**	V
ΔV_{OUT}	Line Regulation (Note 6)	LM1117-ADJ				
		I_{OUT} = 10mA, 1.5V ≤ V_{IN}-V_{OUT} ≤ 13.75V		0.035	**0.2**	%
		LM1117-1.8		1	**6**	mV
		I_{OUT} = 0mA, 3.2V ≤ V_{IN} ≤ 10V				
		LM1117-2.5		1	**6**	mV
		I_{OUT} = 0mA, 3.9V ≤ V_{IN} ≤ 10V				
		LM1117-2.85				
		I_{OUT} = 0mA, 4.25V ≤ V_{IN} ≤ 10V		1	**6**	mV
		LM1117-3.3				
		I_{OUT} = 0mA, 4.75V ≤ V_{IN} ≤ 15V		1	**6**	mV
		LM1117-5.0				
		I_{OUT} = 0mA, 6.5V ≤ V_{IN} ≤ 15V		1	**10**	mV

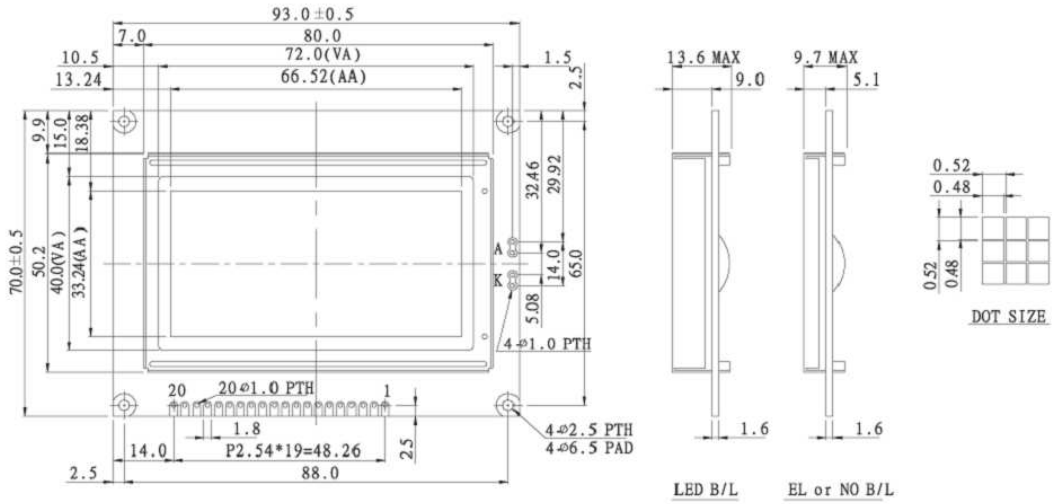
LM1117 Electrical Characteristics (Continued)

Typicals and limits appearing in normal type apply for T_J = 25˚C. Limits appearing in **Boldface** type apply over the entire junction temperature range for operation, 0˚C to 125˚C.

Symbol	Parameter	Conditions	Min (Note 5)	Typ (Note 4)	Max (Note 5)	Units
ΔV_{OUT}	Load Regulation (Note 6)	LM1117-ADJ V_{IN}-V_{OUT} = 3V, 10 ≤ I_{OUT} ≤ 800mA		0.2	**0.4**	%
		LM1117-1.8 V_{IN} = 3.2V, 0 ≤ I_{OUT} ≤ 800mA		1	**10**	mV
		LM1117-2.5 V_{IN} = 3.9V, 0 ≤ I_{OUT} ≤ 800mA		1	**10**	mV
		LM1117-2.85 V_{IN} = 4.25V, 0 ≤ I_{OUT} ≤ 800mA		1	**10**	mV
		LM1117-3.3 V_{IN} = 4.75V, 0 ≤ I_{OUT} ≤ 800mA		1	**10**	mV
		LM1117-5.0 V_{IN} = 6.5V, 0 ≤ I_{OUT} ≤ 800mA		1	**15**	mV
V_{IN}-V_{OUT}	Dropout Voltage (Note 7)	I_{OUT} = 100mA		1.10	**1.20**	V
		I_{OUT} = 500mA		1.15	**1.25**	V
		I_{OUT} = 800mA		1.20	**1.30**	V
I_{LIMIT}	Current Limit	V_{IN}-V_{OUT} = 5V, T_J = 25˚C	800	1200	1500	mA
	Minimum Load Current (Note 8)	LM1117-ADJ V_{IN} = 15V		1.7	**5**	mA
	Quiescent Current	LM1117-1.8 V_{IN} ≤ 15V		5	**10**	mA
		LM1117-2.5 V_{IN} ≤ 15V		5	**10**	mA
		LM1117-2.85 V_{IN} ≤ 10V		5	**10**	mA
		LM1117-3.3 V_{IN} ≤ 15V		5	**10**	mA
		LM1117-5.0 V_{IN} ≤ 15V		5	**10**	mA
	Thermal Regulation	T_A = 25˚C, 30ms Pulse		0.01	0.1	%/W
	Ripple Regulation	f_{RIPPLE} =1 20Hz, V_{IN}-V_{OUT} = 3V V_{RIPPLE} = $1V_{PP}$	**60**	75		dB
	Adjust Pin Current			60	**120**	µA
	Adjust Pin Current Change	10 ≤ I_{OUT}≤ 800mA, 1.4V ≤ V_{IN}-V_{OUT} ≤ 10V		0.2	**5**	µA
	Temperature Stability			0.5		%
	Long Term Stability	T_A = 125˚C, 1000Hrs		0.3		%
	RMS Output Noise	(% of V_{OUT}), 10Hz ≤ f ≤10kHz		0.003		%
	Thermal Resistance Junction-to-Case	3-Lead SOT-223		15.0		˚C/W
		3-Lead TO-220		3.0		˚C/W
		3-Lead TO-252		10		˚C/W
	Thermal Resistance Junction-to-Ambient (No air flow)	3-Lead SOT-223 (No heat sink)		136		˚C/W
		3-Lead TO-220 (No heat sink)		79		˚C/W
		3-Lead TO-252 (Note 9) (No heat sink)		92		˚C/W
		3-Lead TO-263		55		˚C/W
		8-Lead LLP(Note 10)		40		˚C/W

WG12864A Graphic 128x64dots

Dimension drawing



Graphic type

Feature

1. Built-in controller Samsung-(KS0107、KS0108 or Erquivalent)
2. +5V power supply
3. 1/64 duty cycle
4. N.V. Built-in

Pin NO.	Symbol	Function
1	Vss	GND
2	Vdd	Power supply (+5V)
3	Vo	Contrast Adjustment
4	D/I	Data/instruction
5	R/$\overline{W}$	Data read/write
6	E	H→L Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	CS1	Chip select for IC1
16	CS2	Chip select for IC2
17	$\overline{RST}$	Reset
18	Vee	Negative voltage output
19	A	Power supply for LED+(4.2V)RA=OΩ
20	K	Power supply for LED (0V)

Mechanical Data

Item	Standard Value	Unit
Module Dimension	93.0x70.0	mm
Viewing Area	72.0x40.0	mm
Mounting hole	88.0x65.0	mm
Dot Pitch	0.52x0.52	mm

Absolute Maximum Rating

Item	Symbol	Standard Value			Unit
		min.	typ.	max.	
Power Supply	VDD-VSS	4.75	5.0	5.25	V
Input Voltage	VI	-0.3	---	VDD	V

Note：VSS=0 Volt, VDD=5.0 Volt.

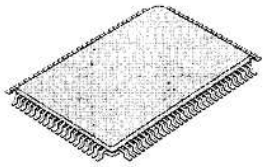
Electronical Characteristics

Item	Symbol	Condition		Standard Value			Unit
				min.	typ.	max.	
Input Voltage	VDD	L level		0.7V_{DD}	---	V_{DD}	V
	VIO	H level		0	---	0.3V_{DD}	V
Supply Current	IDD	VDD=5V		---	2.5	7.5	m
Recommended LC Driving Voltage for Normal Temp. Version module	VDD-V0	0℃		9.7	10.2	10.7	V
		25℃		8.9	9.4	9.9	
		50℃		8.6	9.1	9.6	
LED Forward Voltage	VF	25℃		---	4.2	4.6	V
LED Forward Current	IF	25℃	Array	---	330	660	mA
			Edge	---	120	240	
EL Power Supply Current	IEL	Vel=110VAC;400Hz		---	---	5.0	m

KS0108B 64CH SEGMENT DRIVER FOR DOT MATRIX LCD

INTRODUCTION

100 QFP



The KS0108B is a LCD driver LSI with 64 channel output for dot matrix liquid crystal graphic display system. This device consists of the display RAM, 64 bit data latch 64 bit drivers and decoder logics. It has the internal display RAM for storing the display data transferred from a 8 bit micro controller and generates the dot matrix liquid crystal driving signals corresponding to stored data.The KS0108B composed of the liquid crystal display system in combination with the KS0107B (64 common driver)

FEATURES

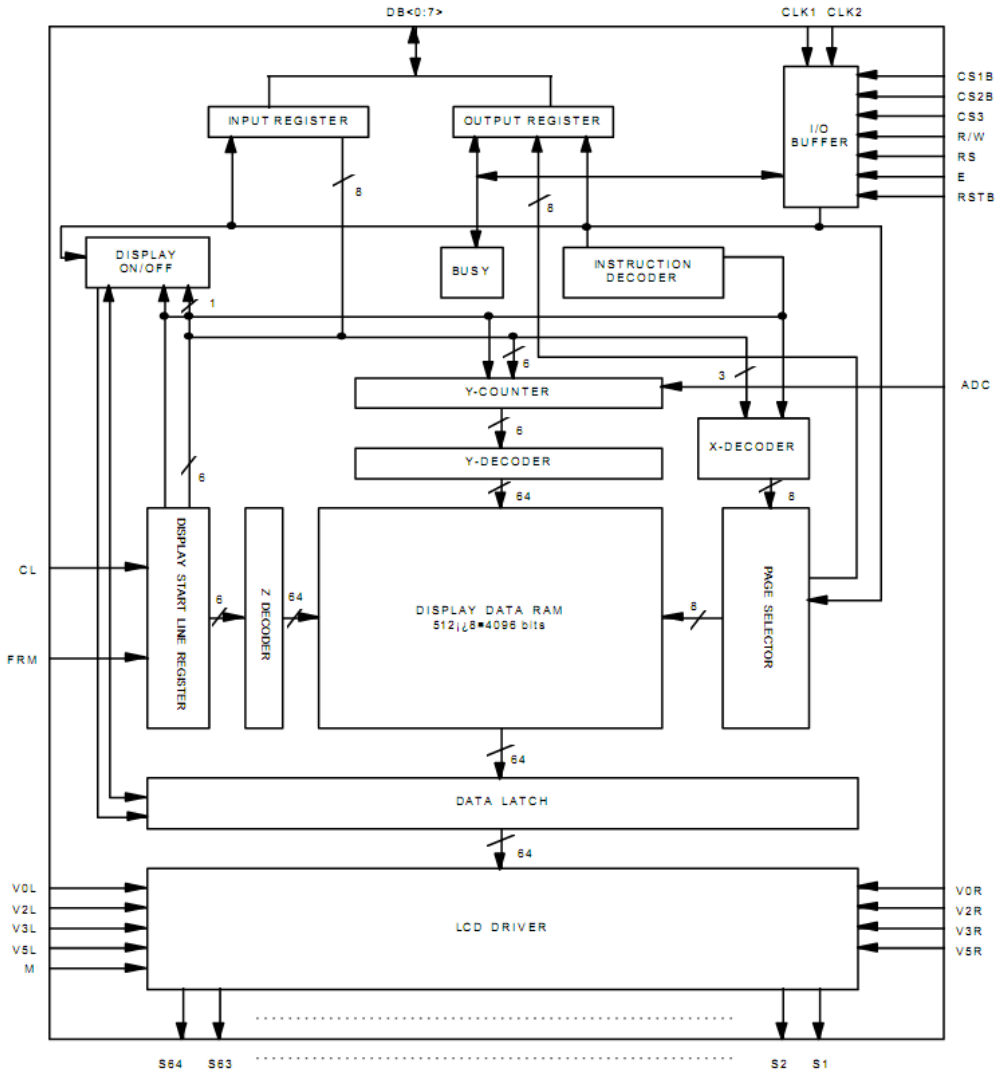
- Dot matrix LCD segment driver with 64 channel output
- Input and Output signal
  - Input: 8 bit parallel display data
    Control signal from MPU
    Splitted bias voltage (V1R, V1L, V2R, V2L, V3R. V3L, V4R, V4L)
  - Output: 64 channel waveform for LCD driving.
- Display data is stored in display data RAM from MPU.
- Interface RAM
  - Capacity: 512 bytes (4096 bits)
  - RAM bit data: RAM bit data = 1:ON
    RAM bit data- = 0:OFF
- Applicable LCD duty: 1/32~1/64
- LCD driving voltage: 8V~17V(V_{DD}-V_{EE})
- Power supply voltage: + 5V±10%

Driver		Controller
COMMON	SEGMENT	
KS0107B	Other KS0108B	MPU

- High voltage CMOS process.
- 100QFP and bare chip available.

SAMSUNG ELECTRONICS

BLOCK DIAGRAM



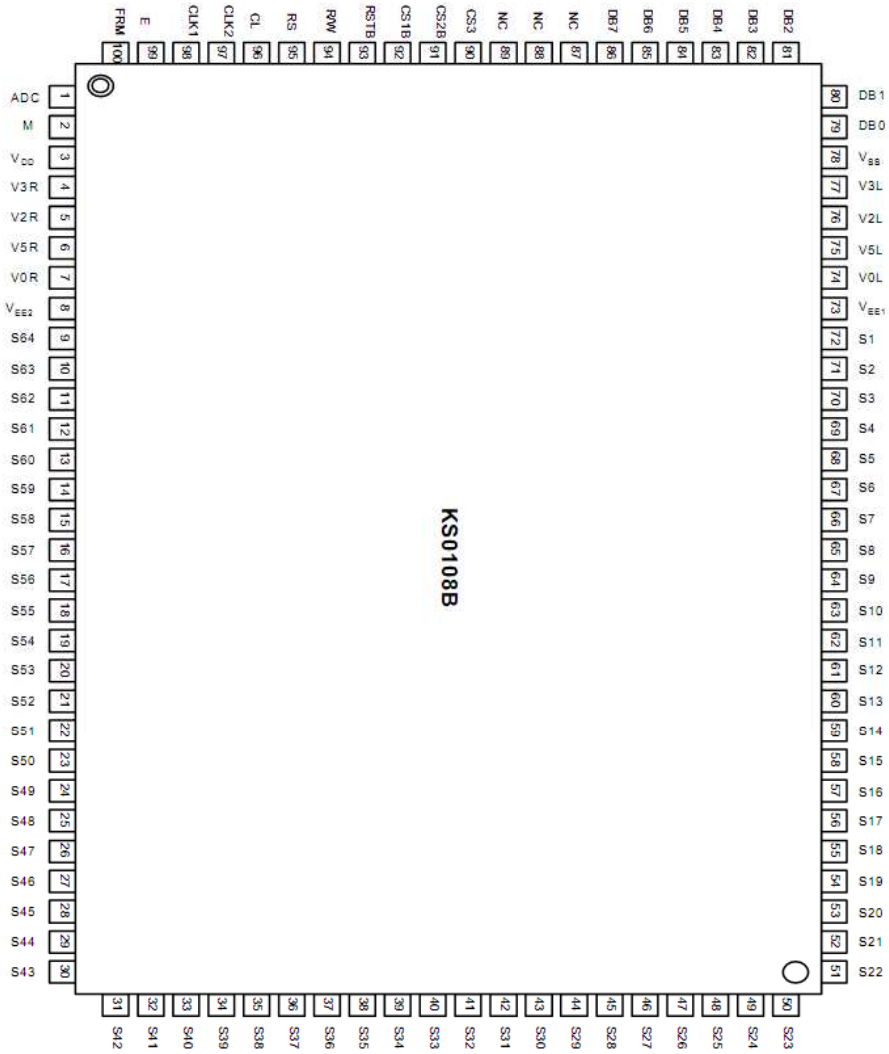


Fig.2. 100QFP Top View

PIN DESCIPTION

PIN (NO)	SYMBOL	INPUT/OUTPUT	DESCRIPTION
3 78 73, 8	V_{DD} V_{SS} $V_{EE1.2}$	Power	For internal logic circuit (+5V±10%) GND (0V) For LCD driver circuit V_{SS}=0V, V_{DD}=5V¡¾10% V_{DD}-V_{EE}=8V~17V V_{EE1} and V_{EE2} is connected by the same voltage.
74, 7 76, 5 77, 4 75, 6	V0L, V0R V2L, V2R V3L, V3R V5L, V5R	Power	Bias supply voltage terminals to drive the LCD. Select Level: V0L(R), V5L(R) / Non-Select Level: V2L(R), V3L(R)
92 91 90	CS1B CS2B CS3	Input	Chip selection In order to interface data for input or output The terminals have to be CS1B=L, CS2B=L, and CS3=H.
2	M	Input	Alternating signal input for LCD driving.
1	ADC	Input	Address control signal of Y address counter. ADC=H→DB<0:7>=0→Y0→S1 DB<0:7>=63→Y63→S64 ADC=L→DB<0:7>=0→Y63→S64 DB<0:7>=63→Y0→S1
100	FRM	Input	Synchronous control signal. Presets the 6-bit Z counter and syncronizes the common signal with the frame signal when the frame signal becomes high.
99	E	Input	Enable signal. write mode (R/W=L) → data of DB<0:7> is latched at the falling edge of E. read mode (R/W=H) → DB<0:7> appears the reading data while E is at high level.
98 97	CLK1 CLK2	Input	2 phase clock signal for internal operation. Used to execute operations for input/output of display RAM data and others.
96	CL	Input	Display synchronous signal. Display data is latched at rising time of the CL signal and increments the Z-address counter at the CL falling time.
95	RS	Input	Data or Instruction. RS=H→DB<0:7> : Display RAM Data RS=L→DB<0:7> : Instruction Data
94	R/W	Input	Read or Write. R/W=H → Data appears at DB<0:7> and can be read by the CPU while E=H, CS1B=L, CS2B=L and CS3=H. R/W=L¡æDisplay data DB<0:7> can be written at falling of E when CS1B=L, CS2B=L and CS3=H.
79~86	DB0~DB7	Input/Output	Data bus. There state I/O common terminal.

PIN DESCRIPTION(continued)

<table>
<tr><th>PIN (NO)</th><th>NAME</th><th>INPUT/OUTPUT</th><th>DESCRIPTION</th></tr>
<tr><td>72~9</td><td>S1~S64</td><td>Output</td><td>LCD Segment driver output.
Display RAM data 1:ON
Display RAM data 0:OFF
(Relation of display RAM data & M)
<table>
<tr><th>M</th><th>DATA</th><th>Output Level</th></tr>
<tr><td rowspan="2">L</td><td>L</td><td>V_2</td></tr>
<tr><td>H</td><td>V_0</td></tr>
<tr><td rowspan="2">H</td><td>L</td><td>V_3</td></tr>
<tr><td>H</td><td>V_5</td></tr>
</table></td></tr>
<tr><td>93</td><td>RSTB</td><td>Input</td><td>Reset signal.
When RSTB=L,
(1) ON/OFF register becomes set by 0. (display off)
(2) Display start line register becomes set by 0
(Z-address 0 set, display from line 0)
After releasing reset, this condition can be changed only by instruction.</td></tr>
<tr><td>87~89</td><td>NC</td><td></td><td>No connection.(open)</td></tr>
</table>

MAXIMUM ABSOLUTE LIMIT

Characteristic	Symbol	Value	Unit	Note
Operating Voltage	V_{DD}	-0.3~+7.0	V	*1
Supply Voltage	V_{EE}	V_{DD}-19.0~V_{DD}+0.3	V	*4
Driver Supply Voltage	V_B	-0.3~V_{DD}+0.3	V	*1,3
	V_{LCD}	V_{EE}-0.3~V_{DD}+0.3	V	*2
Operating Temperature	T_{OPR}	-30~+85	°C	
Storage Temperature	T_{STG}	-55~+125	°C	

*1. Based on V_{SS}=0V.
*2. Applies the same supply voltage to V_{EE1} and V_{EE2}. V_{LCD}=V_{DD}-V_{EE}.
*3. Applies to M, FRM, CL, RSTB, ADC, CLK1, CLK2, CS1B, CS2B, CS3, E, R/W, RS and DB0~DB7.
*4. Applies V0L(R), V2L(R), V3L(R) and V5L(R).
Voltage level: V_{DD}≥V0L=VOR≥V2L=V2R≥V3L=V3R≥V5L=V5R≥V_{EE}.

ELECTRICAL CHARACTERISTICS

DC Characteristics(V_{DD}=4.5~5.5V, V_{SS}=0V, V_{DD}-V_{EE}=8~17V, T_a=-30~+85°C)

Characteristic	Symbol	Condition	Min	Typ	Max	Unit	Note
Input High Voltage	V_{IH1}	-	0.7V_{DD}	-	V_{DD}	V	*1
	V_{IH2}	-	2.0	-	V_{DD}	V	*2
Input Low Voltage	V_{IL1}	-	0	-	0.3V_{DD}	V	*1
	V_{IL2}	-	0	-	0.8	V	*2
Output High Voltage	V_{OH}	I_{OH}=-200μA	2.4	-	-	V	*3
Output Low Voltage	V_{OL}	I_{OL}=1.6mA	-	-	0.4	V	*3
Input Leakage Current	I_{LKG}	V_{IN}=V_{SS}~V_{DD}	-1.0	-	1.0	μA	*4
Three-state(OFF) Input Current	I_{TSL}	V_{IN}=V_{SS}~V_{DD}	-5.0	-	5.0	μA	*5
Driver Input Leakage Current	I_{DIL}	V_{IN}=V_{EE}~V_{DD}	-2.0	-	2.0	μA	*6
Operating Current	I_{DD1}	During Display	-	-	100	μA	*7
	I_{DD2}	During Access Access Cycle=1MHz	-	-	500	μA	*7
On Resistance	R_{ON}	V_{DD}-V_{EE}=15V ¡¾I_{LOAD}=0.1mA	-	-	7.5	KΩ	*8

*1. CL, FRM, M, RSTB, CLK1, CLK2
2. CS1B, CS2B, CS3, E, R/W, RS, DB0~DB7
3. DB0~DB7
4. Excepted DB0~DB7
5. DB0~DB7 at High Impedance
6. V0L(R), V2L(R), V3L(R), V5L(R)
7. 1/64 duty, FCLK=250KHZ, Frame Frequency=70HZ, Output: No Load
8. V_{DD}~V_{EE}=15.5V
   V0L(R)>V2L(R)=V_{DD}-2/7 (V_{DD}-V_{EE})>V3L(R)=V_{EE}+2/7(V_{DD}-V_{EE})>V5L(R)

AC Characteristics(V_{DD}=5V±10%, V_{SS}=0V, T_a=-30°C~+85°C)

(1) Clock Timing

Characteristic	Symbol	Min	Typ	Max	Unit
CLK1, CLK2 Cycle Time	t_{CY}	2.5	-	20	μS
CLK1 'LOW' Level Width	t_{WL1}	625	-	-	ns
CLK2 'LOW' Level Width	t_{WL2}	625	-	-	
CLK1 'HIGH' Level Width	t_{WH1}	1875	-	-	
CLK2 'HIGH' Level Width	t_{WH2}	1875	-	-	
CLK1-CLK2 Phase Difference	t_{D12}	625	-	-	
CLK2-CLK1 Phase Difference	t_{D21}	625	-	-	
CLK1, CLK2 Rise Time	t_R	-	-	150	
CLK1, CLK2 Fall Time	t_F	-	-	150	

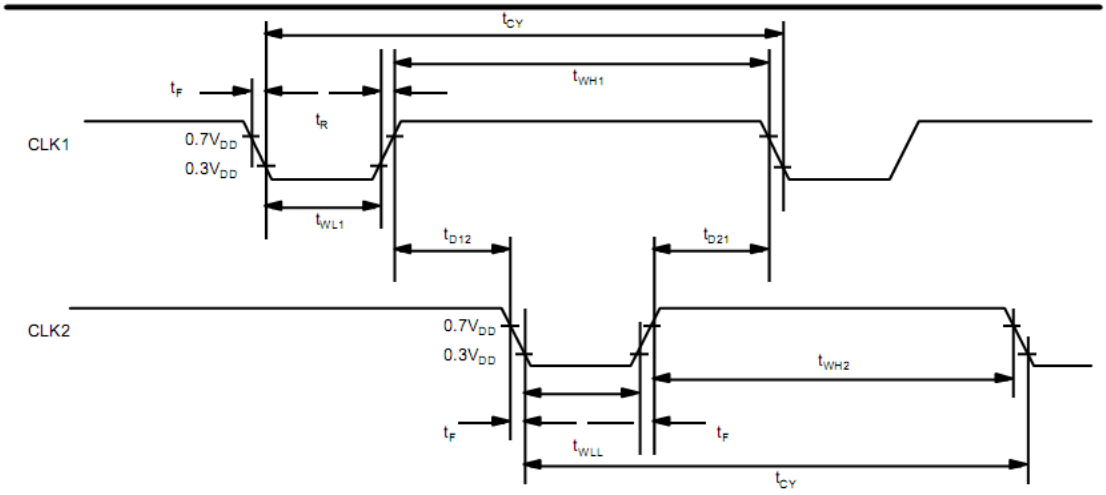


Fig 1. External clock waveform

(2) Display Control Timing

Characteristic	Symbol	Min	Typ	Max	Unit
FRM Delay Time	t_{DF}	-2	-	+2	us
M Delay Time	t_{DM}	-2	-	+2	us
CL 'LOW' Level Width	t_{WL}	35	-	-	us
CL 'HIGH' Level Width	t_{WH}	35	-	-	us

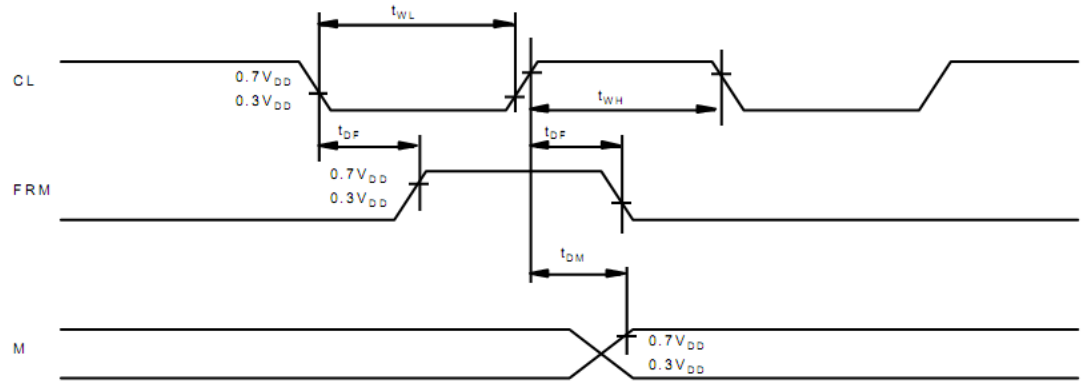


Fig 2. Display control signal waveform