ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

CONSTRUCCIÓN E IMPLEMENTACIÓN DE UN SISTEMA AUTOMÁTICO DE CONTROL DEL HORARIO DE CLASES, ALERTANDO EL CAMBIO DE HORA Y VISUALIZACIÓN DE UN MENSAJE DE BIENVENIDA PARA LA ESCUELA "VENCEDORES"

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO EN ELECTRÓNICA Y TELECOMUNICACIONES

ANDREA PAOLA GUERRA MERINO
paogm21@hotmail.com

DIRECTOR: ING. CARLOS ORLANDO ROMO HERRERA
cromo@hotmail.com

Quito, Diciembre 2014

DECLARACIÓN

Yo, Andrea Paola Guerra Merino, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Andrea Paola Guerra Merino

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Andrea Paola Guerra Merino, bajo mi supervisión.

Ing. Carlos Romo

DIRECTOR DE PROYECTO

AGRADECIMIENTOS

A Dios por haberme guiado en cada momento de mi vida, por cada una de las bendiciones, por haberme regalado una familia muy linda, y puesto en mi camino a personas que han sido mi apoyo, mi fuerza y mi alegría.

Mi profundo y eterno agradecimiento a una mujer que ha sido mi ejemplo, un ejemplo de fuerza, de entrega, de sabiduría, y sobre todo de amor, a mi mami Carmita, que a pesar de los obstáculos supo ser fuerte y vencerlos gracias mamita.

A mi papi que ha pesar de que por voluntad de Dios no pudo estar junto a mi físicamente, he sentido su presencia en cada segundo de mi vida, y sé que desde donde está me bendice y guía siempre.

A mis hermanas María Salomé y Gaby que han sido mi apoyo, y mis fieles amigas, por haber estado conmigo en todo momento, porque han sido mi refugio y también con quienes he podido contar incondicionalmente.

A mi hijita Melissa por ser mi fortaleza y mi aliento para luchar cada día, por hacer hermoso cada segundo de mi vida, gracias por llegar a mi vida y dejar en mi corazón y en mi mente grabados miles de momentos inolvidables y traerme tanta alegría.

A mí querida ESFOT y a los profesores de la carrera de Electrónica y Telecomunicaciones por su apoyo y sus conocimientos que serán los cimientos de mi vida profesional, y porque en sus aulas pude encontrar buenos amigos quienes fueron mi gran apoyo y me brindaron una amistad sincera, gracias a todos ellos.

Al Ing. Carlos Romo gracias por dirigir este trabajo, por su apoyo, y por ayudarme a culminar uno de mis objetivos.

DEDICATORIA

Para la persona que ha dejado en mi un vacío profundo y un dolor insuperable pero a la vez un recuerdo hermoso y un amor infinito, sé que tan grande fue su amor que desde el cielo seguirá guiándome, bendiciéndome y siempre junto a mí, quiero dedicar este trabajo a mi abuelita Blanqui por enseñarme con su amor y con su ejemplo a luchar, a ser una buena mujer, hija y madre.

CONTENIDO

ÍNDICE DE FIGURAS

CAPÍTULO 1

CAPÍTULO 2

CAPÍTULO 3

ÍNDICE DE TABLAS

CAPÍTULO 2

CAPÍTULO 3

RESUMEN

En el presente proyecto se desarrollara la construcción e implementación de un sistema automático de control del horario de clases, alertando el cambio de hora y visualización de un mensaje de bienvenida para la escuela “Vencedores”, que está compuesto por un reloj en tiempo real es decir que permite visualizar la hora constantemente sin pérdida de datos al momento de cortar el suministro eléctrico.

Cada vez que detecte el cambio de hora de materias, sonará la sirena de aviso a los estudiantes y profesores que se tiene que dar el cambio de materia.

Posee un pulsador para activar manualmente la sirena, que si en un momento fuera necesario tocar la alarma en caso de emergencia o reunión se lo pueda realizar.

El proyecto utiliza diferentes dispositivos eléctricos y electrónicos para conseguir el correcto funcionamiento del sistema, con la única finalidad de proporcionar un servicio, para los alumnos y profesores de la escuela Vencedores.

Para presentar detalladamente cómo se logró el objetivo de este proyecto, la parte escrita se ha conformado de cuatro capítulos, presentados de la siguiente manera:

En el Capítulo 1: Se describe de una manera general los principales componentes que se utilizaron.

En el Capítulo 2: Se explica principalmente la construcción del hardware del sistema, es decir se describen las etapas que conforman el circuito con sus respectivos diagramas, conjuntamente se encuentra el programa para el microcontrolador.

En el Capítulo 3: Se detalla la implementación del sistema y las pruebas de funcionamiento realizadas, obteniendo así resultados satisfactorios una vez que el sistema de control de hora se ha implementado completamente en la escuela Vencedores, comprobando así la validez del sistema que se construyó.

En el Capítulo 4: se mencionan las conclusiones obtenidas durante el proceso de elaboración y las recomendaciones del trabajo realizado.

PRESENTACIÓN

Se plantea el desarrollo de un sistema automatizado de mensajería y alerta para las escuelas y colegios, debido a que se torna molestoso el hecho de tener un timbre, el cual a su vez necesita de una persona que este al pendiente de la hora para poder timbrar y de esa manera anunciar a los miembros de la institución educativa el correspondiente cambio de hora.

Tomando en cuenta esta necesidad se ha pensado en realizar un proyecto para la escuela Vencedores, la misma que está ubicada en la calle Azogues y Jorge Piedra en el sector de Andalucía.

El proyecto utilizará un microcontrolador que manejará un RTC, el cual permitirá visualizar en los displays el tiempo real, y cuando sea la hora actual igual a las horas programadas se proceda a activar la sirena. Ayudando al personal y a los estudiantes de la unidad educativa a saber qué hora es, ya que se lo ubicará en un lugar que facilite la visibilidad de todos sus ocupantes.

CAPÍTULO 1

FUNDAMENTOS TEÓRICOS DEL PROYECTO

1.1 INTRODUCCIÓN

En el capítulo I se detalla de una manera concreta los principales elementos, que se van a utilizar para la construcción del presente proyecto, teniendo en cuenta que se debe cumplir los objetivos planteados.

1.2 CONCEPTOS GENERALES

1.2.1 RELOJ EN TIEMPO REAL DS1307

1.2.1.1 Introducción

Un reloj de tiempo real es un dispositivo muy útil en cualquier aplicación donde se quiera dejar constancia de fecha y hora de algún evento y de manera exacta. Para el presente proyecto se decidió incorporar el C. I. DS1307, para que por medio de programa nos permita dar la fecha y hora exacta.

1.2.1.2 Características del DS1307 [1]

El DS1307 es un reloj calendario en BCD, cuyas características más destacadas son las siguientes:

• Reloj de tiempo real que cuenta los segundos, los minutos, las horas, la fecha, el mes, el día de la semana, y el año, con compensación de años bisiestos, válido hasta el año 2100

• Formato de 12 Horas con indicador AM/PM ó de 24 horas

• Protocolo I2C

• 56 bytes de RAM no volátil, para almacenamiento de datos

• Señal de onda cuadrada programable

---

1 http://www.bolivar.udo.edu.ve/microinternet/articulos/Reloj%20de%20Tiempo%20Real%20DS1307

• Circuitos internos de respaldo para la alimentación automática

• Bajo consumo de potencia: menor a 500nA en modo respaldo, a 25 grados Centígrados

1.2.1.3 Descripción de los Pines

La figura 1.1, muestra la asignación de los pines del DS1307 los cuales se describen a continuación:

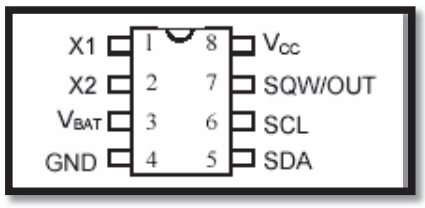


Figura 1. 1 Asignación de pines del DS 1307

VCC, GND: La alimentación DC es proporcionada a este circuito a través de estos pines. VCC es la entrada de +5VDC, mientras que GND es la referencia.

VBAT: Entrada de alimentación de una pila estándar de litio de 3 Voltios. El voltaje debe está entre 2.5 y 3.5 voltios para una operación apropiada.

SCL: Entrada de reloj para sincronizar la transferencia de datos en la interfaz serial.

SDA: Entrada/salida de datos para la interfaz I2C. Este pin es de drenaje abierto, por lo que requiere de una resistencia pull-up externa.

X1, X2: Conexiones para un cristal de cuarzo estándar de 32.768 Hz.

SQW/OUT: Salida para generar una de cuatro posibles frecuencias de salida: 1Hz, 4KHz, 8KHz ó 32KHz. Este pin también es de drenaje abierto, por lo que requiere de una resistencia pull-up externa.

1.2.1.4 Circuito Típico de aplicación[2]

La figura 1.2, muestra un diagrama esquemático de una aplicación típica para este circuito. Se observan las resistencia pull-up en los pines SDA, SCL y SQW/OUT que normalmente son de 4.7 KOhm. Usa un cristal de cuarzo estándar de 32.768Hz. La alimentación es normalmente 5 voltios DC y utiliza una pila de litio de 3V para el respaldo de la alimentación

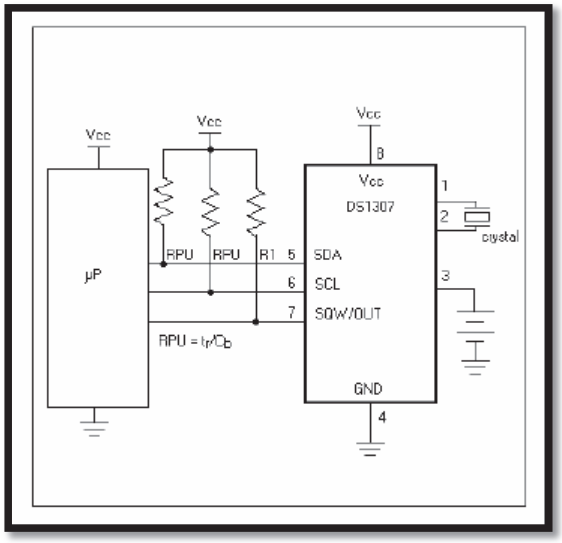


Figura 1. 2 Esquema de operación típico

1.2.1.5 Mapa de Direcciones

En la figura 1.3, se presenta el mapa de direcciones para los registros del Reloj de Tiempo Real y la RAM del DS1307. Los registros se localizan en las direcciones 0x00 hasta 0x07, mientras que la RAM va desde 0x08 hasta 0x3F.

El puntero de direcciones es de 6 bits, de tal manera que cuando se realiza accesos múltiples y se alcanza la dirección 0x3F, ó fin del espacio de la RAM, el puntero regresa a 0 con el siguiente acceso.

---

[2] Autor: Ing. Carlos Narváez -Universidad de Oriente

http://www.bolivar.udo.edu.ve/microinternet/articulos/Reloj%20de%20Tiempo%20Real%20DS1307.p

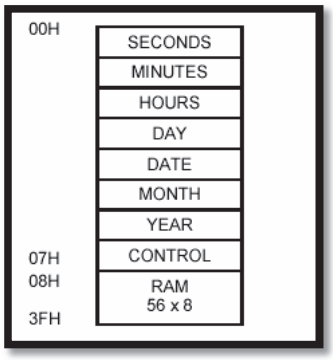


Figura 1. 3 Mapa de direcciones

1.2.1.6 Reloj Calendario

La información del tiempo y el calendario está en formato BCD, y es obtenida leyendo los registros apropiados los cuales se muestran en la figura. Los bits marcados con 0 en esta figura son intrascendentes para el funcionamiento del circuito. El bit 7 del registro 0 es el bit CH (Clock Halt). Cuando este bit se establece a 1, el oscilador se deshabilita, deteniendo el funcionamiento del reloj. Cuando se establece a 0, el oscilador es habilitado, y el circuito funciona normalmente.

El RTC, como se mencionó puede trabajar en formato de 12 horas ó de 24 horas siendo el bit 6 del registro 2 el encargado de determinar esta función. Cuando se establece a 1, funciona en modo 12 horas, y el bit 5 es el indicador AM/PM, siendo 1 cuando es PM. En formato 24 horas, el bit 5 de este registro es la parte más significativa de las decenas de horas.

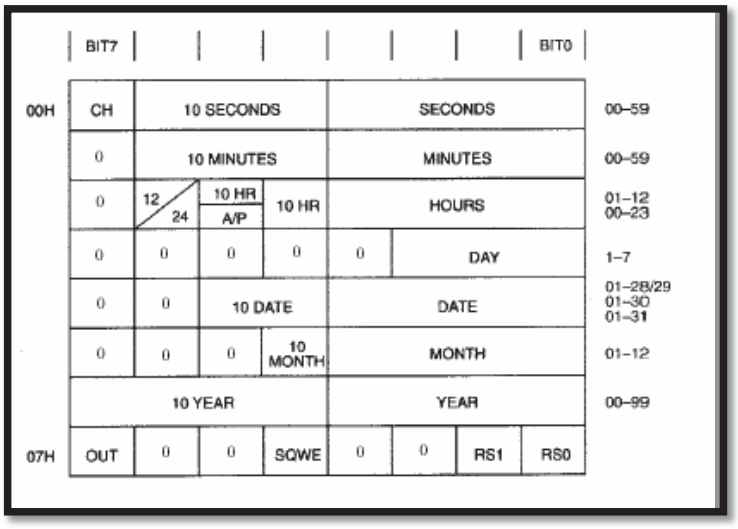


Figura 1. 4 Registros DS1307

El registro 7 es el registro de control. Este registro es utilizado para determinar el funcionamiento del pin de salida SQW/OUT. El bit 7 ó OUT determina el nivel de salida cuando la salida de onda cuadrada esta deshabilitada (SQWE = 0). Será alto si el bit OUT = 1 y bajo si OUT = 0. El bit 4 ó SQWE, habilita la salida del oscilador de onda cuadrada, cuya frecuencia dependerá de los bits RS1 y RS0. La figura1.5, muestra las frecuencias que se pueden obtener según la configuración de los bits RS1 y RS0.

RS1	RS0	Frecuencia de salida
0	0	1Hz
0	1	4KHz
1	0	8KHz
1	1	32KHz

Figura 1. 5 Frecuencia de Salida pin SQW/OUT

1.2.2 BUS I2C[3]

El bus I2C (Inter-Integrated Circuit), fue desarrollado por Phillips Semiconductors con el propósito de comunicar elementos que se encuentren en una misma tarjeta o circuito. Utiliza un protocolo serial sincrónico que solamente requiere de dos líneas, SDA (Serial Data Line) y SCL (Serial Clock Line) las cuales se comportan bidireccionalmente. Estas líneas se conectan al positivo de la fuente de alimentación a través de resistencias pull-up. Cuando ambas líneas están libres permanecen en un estado lógico alto. Para el modo I2C del módulo MSSP estas líneas son típicamente RC3 y RC4 respectivamente. Estas son las mismas líneas usadas por el modo SPI del MSSP. La figura 1.6. Muestra un ejemplo de configuración del bus I2C en un sistema.

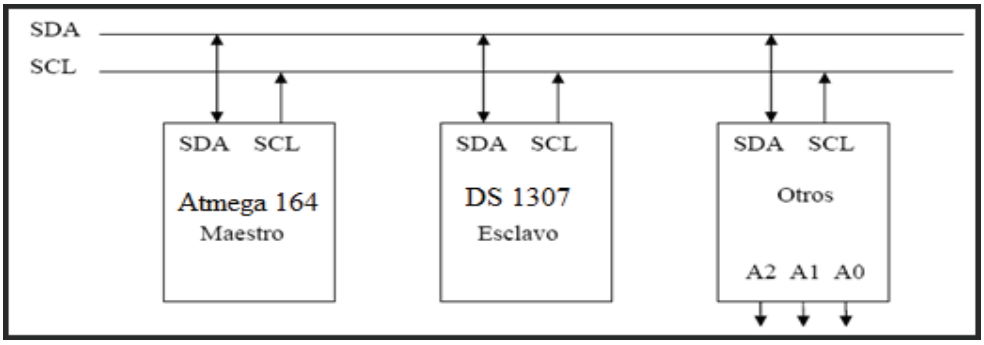


Figura 1. 6 Esquema de configuración I2C

Usar sólo dos líneas para la comunicación, requiere de una aproximación diferente para el flujo de datos. En SPI, los datos fluyen del maestro al esclavo y simultáneamente del esclavo hacia el maestro. En el modo I2C se usa un sistema secuencial en el cual el flujo de datos ocurre del maestro hacia el esclavo sobre la línea SDA y luego, si es necesario (en caso de que el maestro necesite leer del esclavo), los datos fluyen del esclavo hacia el maestro sobre la misma línea SDA. En cualquiera de los casos el maestro siempre suministra la señal de reloj necesaria para la comunicación Maestro/Esclavo a través de la línea SCL.

Cada dispositivo que se conecta al bus tiene asignada una dirección compuesta de 7 bits. El DS1307 tiene establecida internamente la dirección en b1101000.

[3] http://robots-argentina.com.ar/Comunicacion_busI2C.htm

Para facilitar la programación el bit siguiente, correspondiente a lectura/escritura, se incorpora a los siete bits anteriores, a fin de completar un byte. En el caso de escritura la dirección del DS1307 es entonces 0xD0 y en el caso de lectura 0xD1.

La figura 1.7 y la figura 1.8, muestran la actividad del maestro y del esclavo cuando el maestro realiza una operación de escritura sobre el esclavo y cuando el maestro realiza una operación de lectura sobre el esclavo. En ambas operaciones, la parte sombreada corresponde a la actividad del maestro sobre el bus y la no sombreada a la actividad del esclavo sobre el bus. El bit de acuse de recibo (ACK) siempre lo genera el dispositivo que recibe los datos, excepto cuando el maestro se comporta como receptor, éste no debe generar el último acuse de recibo (ACK), en su lugar debe enviar un no acuse de recibo (NACK)

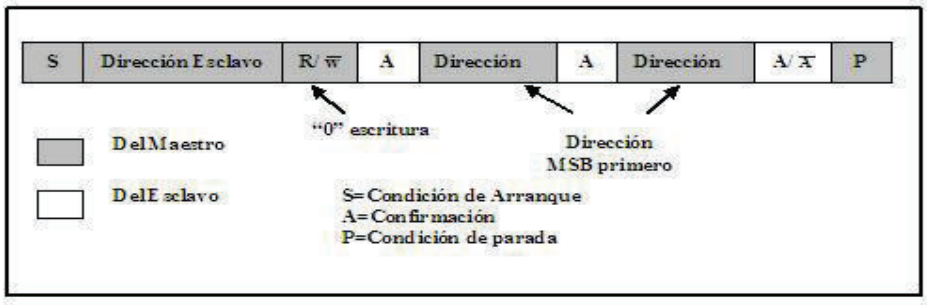


Figura 1. 7 Operación de escritura

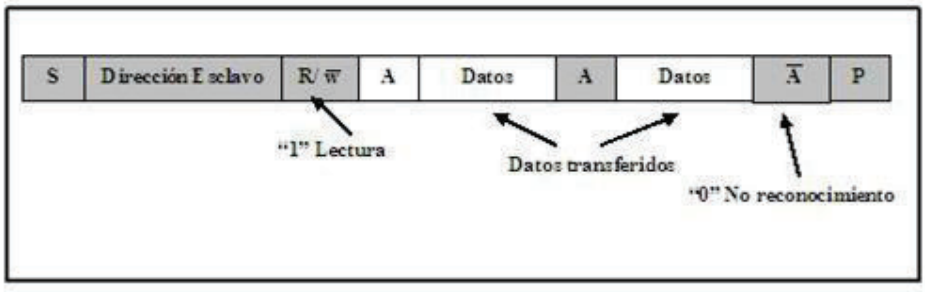


Figura 1. 8 Operación de lectura

1.2.3 MICROCONTROLADOR[4]

1.2.3.1 Introducción

Los microcontroladores están conquistando el mundo. Están presentes en nuestro trabajo, en nuestra casa y en nuestra vida, en general. Se pueden encontrar controlando el funcionamiento de los ratones y teclados de los computadores, en los teléfonos, en los hornos microondas, en los televisores de nuestro hogar.

Hace tres décadas, los controladores se construían exclusivamente con componentes de lógica discreta, posteriormente se emplearon los microprocesadores, que se rodeaban con chips de memoria y E/S sobre una tarjeta de circuito impreso. En la actualidad, todos los elementos del controlador se han podido incluir en un chip, el cual recibe el nombre de microcontrolador.

Realmente consiste en un sencillo pero completo computador contenido en el corazón (chip) de un circuito integrado.

1.2.3.2 Diferencia entre microprocesador y microcontrolador

El microprocesador es un circuito integrado que contiene la Unidad Central de Proceso (UCP), también llamada procesador de un computador. La UCP está formada por la unidad de control que interpreta las instrucciones, y el camino de datos, que las ejecuta, siendo el camino de datos sistema digitales utilizados con frecuencia para la manipulación de datos y cálculos numéricos complejos.

Los pines de un microprocesador sacan al exterior las líneas de sus buses de direcciones, datos y control, para permitir conectarle con la Memoria y los Módulos de E/S y configurar un computador implementado por varios circuitos integrados. Se dice que un microprocesador es un sistema abierto porque su configuración es variable de acuerdo con la aplicación a la que se destine.

[4] http://www.monografias.com/trabajos12/micrcont/micrcont.shtml

En la figura 1.9 se muestra la estructura de un sistema abierto basado en un microprocesador, y la figura 1.10 muestra la estructura de un microcontrolador.

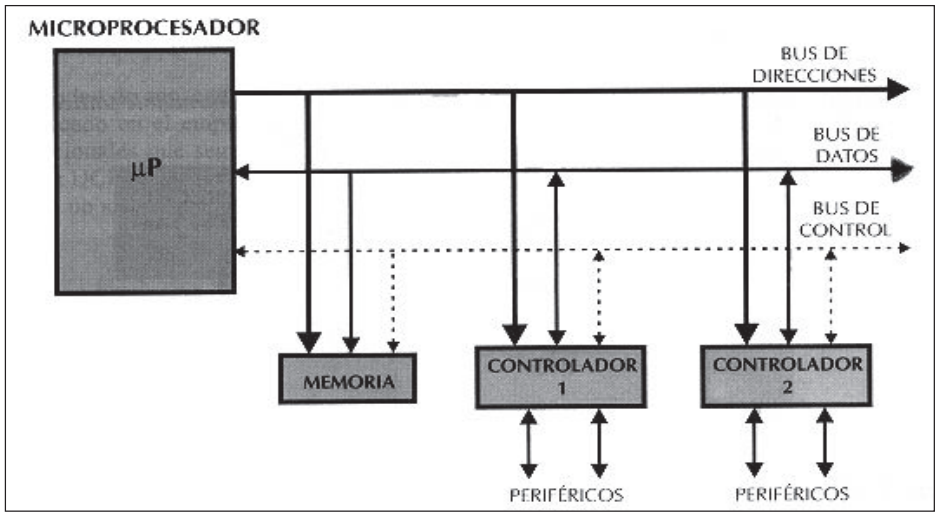


Figura 1. 9 Estructura de un sistema abierto basado en un microprocesador.

Si sólo se dispusiese de un modelo de microcontrolador, éste debería tener muy potenciados todos sus recursos para poderse adaptar a las exigencias de las diferentes aplicaciones. Esta potenciación supondría en muchos casos un despilfarro. En la práctica cada fabricante de microcontroladores oferta un elevado número de modelos diferentes, desde los más sencillos hasta los más poderosos. Es posible seleccionar la capacidad de las memorias, el número de líneas de E/S, la cantidad y potencia de los elementos auxiliares, la velocidad de funcionamiento, etc. Por todo ello, un aspecto muy destacado del diseño es la selección del microcontrolador a utilizar.

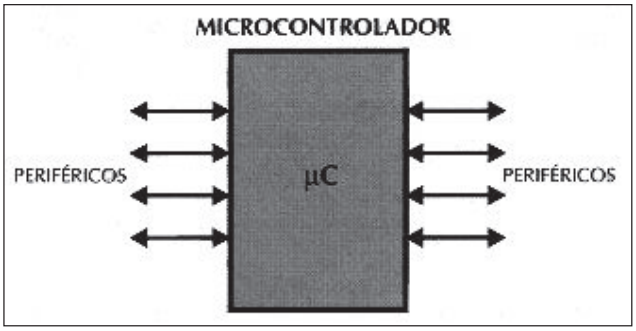


Figura 1. 10 Estructura de un microcontrolador (perifericos)

El microcontrolador es un sistema cerrado. Todas las partes del computador están contenidas en su interior y sólo salen al exterior las líneas que gobiernan los periféricos.

1.2.3.3 Recursos comunes de todos los microcontroladores

La figura 1.11 muestra las partes de un microcontrolador, al estar todos los microcontroladores integrados en un chip, su estructura fundamental y sus características básicas son muy parecidas. Todos deben disponer de los bloques esenciales. Procesador, memoria de datos y de instrucciones, líneas de E/S, oscilador de reloj y módulos controladores de periféricos.

Sin embargo, cada fabricante intenta enfatizar los recursos más idóneos para las aplicaciones a las que se destinan preferentemente.

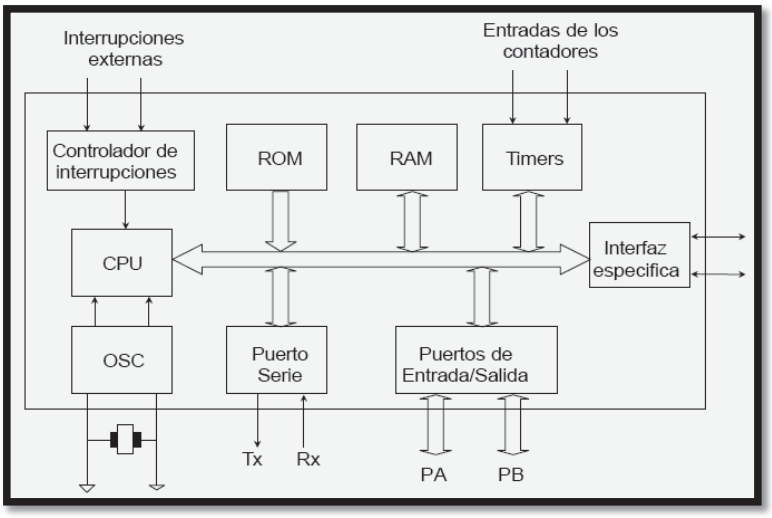


Figura 1. 11 Partes de un Microcontrolador

1.2.3.4 Arquitectura

En la figura 1.12 se muestra la arquitectura Harvard, aunque inicialmente todos los microcontroladores adoptaron la arquitectura clásica de von Neumann, en el momento presente se impone la arquitectura Harvard. La arquitectura de von Neumann se caracteriza por disponer de una sola memoria principal donde se almacenan datos e instrucciones de forma indistinta. A dicha memoria se accede a través de un sistema de buses único (direcciones, datos y control).

La arquitectura Harvard dispone de dos memorias independientes una, que contiene sólo instrucciones y otra, sólo datos. Ambas disponen de sus respectivos sistemas de buses de acceso y es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias.

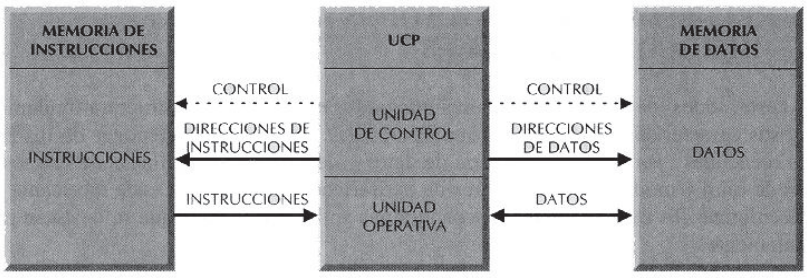


Figura 1. 12 Arquitectura Harvard

La arquitectura Harvard dispone de dos memorias independientes para datos y para instrucciones, permitiendo accesos simultáneos.

1.2.3.5 El procesador o CPU

Es el elemento más importante del microcontrolador y determina sus principales características, tanto a nivel hardware como software.

Se encarga de direccionar la memoria de instrucciones, recibir el código OP de la instrucción en curso, su decodificación y la ejecución de la operación que implica la instrucción, así como la búsqueda de los operandos y el almacenamiento del resultado.

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

a) **CISC**: Un gran número de procesadores usados en los microcontroladores están basados en la filosofía CISC (Computadores de Juego de Instrucciones Complejo).

Disponen de más de 80 instrucciones máquina en su repertorio, algunas de las cuales son muy sofisticadas y potentes, requiriendo muchos ciclos para su ejecución.

Una ventaja de los procesadores CISC es que ofrecen al programador instrucciones complejas que actúan como macros.

b) **RISC**: Tanto la industria de los computadores comerciales como la de los microcontroladores están decantándose hacia la filosofía RISC (Computadores de Juego de Instrucciones Reducido). En estos procesadores el repertorio de instrucciones máquina es muy reducido y las instrucciones son simples y, generalmente, se ejecutan en un ciclo.

La sencillez y rapidez de las instrucciones permiten optimizar el hardware y el software del procesador.

c) **SISC**: En los microcontroladores destinados a aplicaciones muy concretas, el juego de instrucciones, además de ser reducido, es "específico", o sea, las instrucciones se adaptan a las necesidades de la aplicación prevista. Esta filosofía se ha bautizado con el nombre de SISC (Computadores de Juego de Instrucciones Específico).

1.2.3.6 Memoria

En los microcontroladores la memoria de instrucciones y datos está integrada en el propio chip. Una parte debe ser no volátil, tipo ROM, y se destina a contener el programa de instrucciones que gobierna la aplicación. Otra parte de memoria será tipo RAM, volátil, y se destina a guardar las variables y los datos.

a) ROM con máscara

Es una memoria no volátil de sólo lectura cuyo contenido se graba durante la fabricación del chip. El elevado coste del diseño de la máscara sólo hace aconsejable el empleo de los microcontroladores con este tipo de memoria cuando se precisan cantidades superiores a varios miles de unidades.

b) OTP

El microcontrolador contiene una memoria no volátil de sólo lectura "programable una sola vez" por el usuario. OTP (One Time Programmable). Es el usuario quien puede escribir el programa en el chip mediante un sencillo grabador controlado por un programa desde un PC.

La versión OTP es recomendable cuando es muy corto el ciclo de diseño del producto, o bien, en la construcción de prototipos y series muy pequeñas.

Tanto en este tipo de memoria como en la EPROM, se suele usar la encriptación mediante fusibles para proteger el código contenido.

c) EPROM

Los microcontroladores que disponen de memoria EPROM (Erasable Programmable Read Only Memory) pueden borrarse y grabarse muchas veces. La grabación se realiza, como en el caso de los OTP, con un grabador gobernado desde un PC. Si, posteriormente, se desea borrar el contenido, disponen de una ventana de cristal en su superficie por la que se somete a la EPROM a rayos ultravioleta durante varios minutos. Las cápsulas son de material cerámico y son más caros que los microcontroladores con memoria OTP que están hechos con material plástico.

d) EEPROM

Se trata de memorias de sólo lectura, programables y borrables eléctricamente EEPROM (Electrical Erasable Programmable Read Only Memory). Tanto la programación como el borrado, se realizan eléctricamente desde el propio grabador y bajo el control programado de un PC. Es muy cómoda y rápida la operación de grabado y la de borrado. No disponen de ventana de cristal en la superficie.

Los microcontroladores dotados de memoria EEPROM una vez instalados en el circuito, pueden grabarse y borrarse cuantas veces se quiera sin ser retirados de dicho circuito. Para ello se usan "grabadores en circuito" que confieren una gran flexibilidad y rapidez a la hora de realizar modificaciones en el programa de trabajo.

e) FLASH

Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. Funciona como una ROM y una RAM pero consume menos y es más pequeña.

A diferencia de la ROM, la memoria FLASH es programable en el circuito. Es más rápida y de mayor densidad que la EEPROM.

La alternativa FLASH está recomendada frente a la EEPROM cuando se precisa gran cantidad de memoria de programa no volátil. Es más veloz y tolera más ciclos de escritura/borrado.

Las memorias EEPROM y FLASH son muy útiles al permitir que los microcontroladores que las incorporan puedan ser reprogramados "en circuito", es decir, sin tener que sacar el circuito integrado de la tarjeta. Así, un dispositivo con este tipo de memoria incorporado al control del motor de un automóvil permite que pueda modificarse el programa durante la rutina de mantenimiento periódico, compensando los desgastes y otros factores tales como la compresión, la instalación de nuevas piezas, etc. La reprogramación del microcontrolador puede convertirse en una labor rutinaria dentro de la puesta a punto.

1.2.3.7 Puertas de entrada y salida

La principal utilidad de los pines que posee la cápsula que contiene un microcontrolador es soportar las líneas de E/S que comunican al computador interno con los periféricos exteriores.

Según los controladores de periféricos que posea cada modelo de microcontrolador, las líneas de E/S se destinan a proporcionar el soporte a las señales de entrada, salida y control.

1.2.3.8 Reloj

Todos los microcontroladores disponen de un circuito oscilador que genera una onda cuadrada de alta frecuencia, que configura los impulsos de reloj usados en la sincronización de todas las operaciones del sistema.

Generalmente, el circuito de reloj está incorporado en el microcontrolador y sólo se necesitan unos pocos componentes exteriores para seleccionar y estabilizar la frecuencia de trabajo. Dichos componentes suelen consistir en un cristal de cuarzo junto a elementos pasivos o bien un resonador cerámico o una red R-C.

Aumentar la frecuencia de reloj supone disminuir el tiempo en que se ejecutan las instrucciones pero lleva aparejado un incremento del consumo de energía.

1.2.3.9 Recursos Especiales

Cada fabricante oferta numerosas versiones de una arquitectura básica de microcontrolador. En algunas amplía las capacidades de las memorias, en otras incorpora nuevos recursos, en otras reduce las prestaciones al mínimo para aplicaciones muy simples, etc. La labor del diseñador es encontrar el modelo mínimo que satisfaga todos los requerimientos de su aplicación. De esta forma, minimizará el coste, el hardware y el software.

Los principales recursos específicos que incorporan los microcontroladores son:

• Temporizadores o "Timers".

• Perro guardián o "Watchdog".

• Protección ante fallo de alimentación o "Brownout".

• Estado de reposo o de bajo consumo.

• Conversor A/D.

• Conversor D/A.

• Comparador analógico.

• Modulador de anchura de impulsos o PWM.

• Puertas de E/S digitales.

• Puertas de comunicación.

a) Temporizadores o "Timers"

Se emplean para controlar periodos de tiempo (temporizadores) y para llevar la cuenta de acontecimientos que suceden en el exterior (contadores).

Para la medida de tiempos se carga un registro con el valor adecuado y a continuación dicho valor se va incrementando o decrementando al ritmo de los impulsos de reloj o algún múltiplo hasta que se desborde y llegue a 0, momento en el que se produce un aviso.

Cuando se desean contar acontecimientos que se materializan por cambios de nivel o flancos en alguna de las patitas del microcontrolador, el mencionado registro se va incrementando o decrementando al ritmo de dichos impulsos.

b) Perro guardián o "Watchdog"

Cuando el computador personal se bloquea por un fallo del software u otra causa, se pulsa el botón del reset y se reinicializa el sistema. Pero un microcontrolador funciona sin el control de un supervisor y de forma continuada las 24 horas del día. El Perro guardián consiste en un temporizador que, cuando se desborda y pasa por 0, provoca un reset automáticamente en el sistema.

Se debe diseñar el programa de trabajo que controla la tarea de forma que refresque o inicialice al Perro guardián antes de que provoque el reset. Si falla el programa o se bloquea, no se refrescará al Perro guardián y, al completar su temporización, "ladrará y ladrará" hasta provocar el reset.

c) Protección ante fallo de alimentación o "Brownout"

Se trata de un circuito que resetea al microcontrolador cuando el voltaje de alimentación (VDD) es inferior a un voltaje mínimo ("brownout"). Mientras el voltaje de alimentación sea inferior al de brownout el dispositivo se mantiene reseteado, comenzando a funcionar normalmente cuando sobrepasa dicho valor.

d) Estado de reposo ó de bajo consumo

Son abundantes las situaciones reales de trabajo en que el microcontrolador debe esperar, sin hacer nada, a que se produzca algún acontecimiento externo que le ponga de nuevo en funcionamiento.

Para ahorrar energía, (factor clave en los aparatos portátiles), los microcontroladores disponen de una instrucción especial (SLEEP en los PIC), que les pasa al estado de reposo o de bajo consumo, en el cual los requerimientos de potencia son mínimos.

En dicho estado se detiene el reloj principal y se "congelan" sus circuitos asociados, quedando sumido en un profundo "sueño" el microcontrolador. Al activarse una interrupción ocasionada por el acontecimiento esperado, el microcontrolador se despierta y reanuda su trabajo.

e) Conversor A/D (CAD)

Los microcontroladores que incorporan un Conversor A/D (Analógico/Digital) pueden procesar señales analógicas, tan abundantes en las aplicaciones. Suelen disponer de un multiplexor que permite aplicar a la entrada del CAD diversas señales analógicas desde las patitas del circuito integrado.

f) Conversor D/A (CDA)

Transforma los datos digitales obtenidos del procesamiento del computador en su correspondiente señal analógica que saca al exterior por una de las patitas de la cápsula. Existen muchos efectores que trabajan con señales analógicas.

g) Comparador analógico

Algunos modelos de microcontroladores disponen internamente de un amplificador operacional que actúa como comparador entre una señal fija de referencia y otra variable que se aplica por una de las patitas de la cápsula. La salida del comparador proporciona un nivel lógico 1 ó 0 según una señal sea mayor o menor que la otra.

También hay modelos de microcontroladores con un módulo de tensión de referencia que proporciona diversas tensiones de referencia que se pueden aplicar en los comparadores.

h) Modulador de anchura de impulsos o PWM

Son circuitos que proporcionan en su salida impulsos de anchura variable, que se ofrecen al exterior a través de las patitas del encapsulado.

i) Puertos de E/S digitales

Todos los microcontroladores destinan algunas de sus patitas a soportar líneas de E/S digitales. Por lo general, estas líneas se agrupan de ocho en ocho formando Puertos.

Las líneas digitales de los Puertos pueden configurarse como Entrada o como Salida cargando un 1 ó un 0 en el bit correspondiente de un registro destinado a su configuración.

j) Puertos de comunicación

Con objeto de dotar al microcontrolador de la posibilidad de comunicarse con otros dispositivos externos, otros buses de microprocesadores, buses de sistemas, buses de redes y poder adaptarlos con otros elementos bajo otras normas y protocolos. Algunos modelos disponen de recursos que permiten directamente esta tarea, entre los que destacan:

UART, adaptador de comunicación serie asíncrona.

USART, adaptador de comunicación serie síncrona y asíncrona

Puerta paralela esclava para poder conectarse con los buses de otros microprocesadores.

USB (Universal Serial Bus), que es un moderno bus serie para los PC.

Bus I2C, que es un interfaz serie de dos hilos desarrollado por Philips.

CAN (Controller Area Network), para permitir la adaptación con redes de conexionado multiplexado desarrollado conjuntamente por Bosch e Intel para el cableado de dispositivos en automóviles. En EE.UU. se usa el J185O.

1.2.4 EL MICROCONTROLADOR ATMEGA 164P[5]

El ATmega164P es un microcontrolador CMOS de 8 bits de bajo consumo basado como se muestra en la figura 1.13, en la arquitectura RISC mejorada. Sus instrucciones se ejecutan en un ciclo de máquina, el ATmega164P consigue transferencia de información alrededor de 1 MIPS por MHz admitido por el sistema, permitiendo al diseñador del sistema optimizar el consumo de energía versus la velocidad de procesamiento.

[5] http://atmega164p_spa_ol.pdf

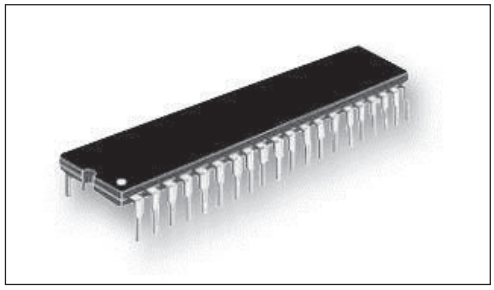


Figura 1. 13 Microcontrolador Atmega 164P

1.2.4.1 Arquitectura Avanzada RISC

- 131 instrucciones.
- La mayoría de un solo ciclo de reloj de ejecución.
- 32 registros de trabajo de 8 bits para propósito general.
- Funcionamiento estático total.
- Capacidad de procesamiento de unos 20 MIPS a 20 MHz.

1.2.4.2 Memorias de programa y de datos no volátiles de alta duración

- 16 K bytes de FLASH.
- 512bytes de EEPROM.
- 1K bytes de SRAM Interna.
- Ciclos de escritura/borrado: 10.000 en Flash / 100.000 en EEPROM.
- Retención de Datos: 20 años a 85ºC / 100 años a 25ºC.
- Sección opcional de código Boot con bits de bloqueo independientes.
- Operación de lectura durante la escritura.
- Bloqueo programable para la seguridad del software.

1.2.4.3 Interface JTAG

- Capacidades de Boundary Scan de acuerdo con el estándar JTAG.
- Soporte Extendido Debug dentro del chip.
- Programación de FLASH, EEPROM, fusibles y bits de bloqueo a través de la interface JTAG.

1.2.4.4 Características de los periféricos

- Dos Timer/Contadores de 8 bits con pre escalamiento separado y modo comparación.
- Un Timer/Contador de 16 bits con preescalamiento separado, modo comparación y modo de captura.
- Contador en Tiempo Real con Oscilador separado.
- 6 Canales para PWM.
- ADC de 10 bits y 8 canales.
- Interface serie de dos hilos con byte orientado.
- Dos puertos Seriales USART Programables.
- Interfaz Serial SPI maestro-esclavo.
- Watchdog Timer programable con oscilador independiente.
- Comparador Analógico dentro del mismo Chip.
- Interrupt and Wake-up on Pin Change.

1.2.4.5 Características especiales

- Power-on Reset (en el encendido) y detección de Brown-out (pérdida de polarización) programable.
- Oscilador RC interno calibrado.
- Fuentes de interrupción externas e internas.
- 6 modos de descanso: Idle, Reducción de Ruido ADC, Power-save, Power-down, Standby y Standby extendido.

1.2.4.1.6 Diagrama de pines y Funciones

La figura 1.14 muestra la distribución de pines del microcontrolador ATmega 164P

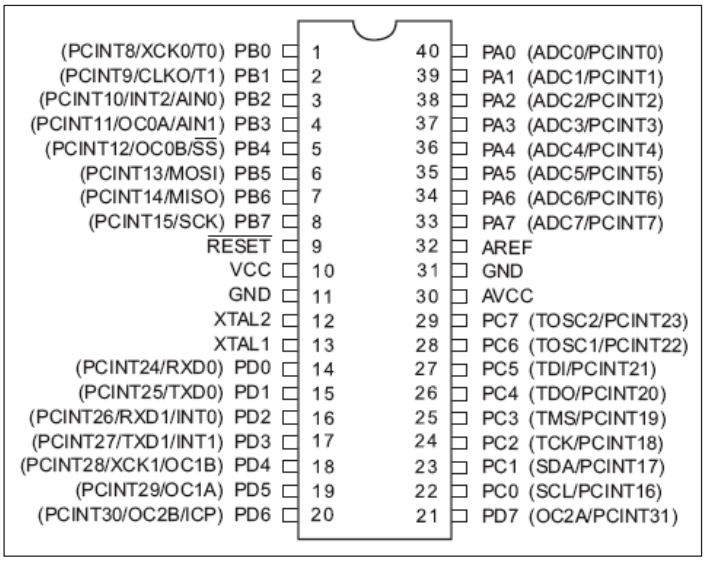


Figura 1. 14 Distribución de pines del Microcontrolador ATmega 164P

a) Puerto A (PA7:PA0)

El puerto A sirve como entradas analógicas para el conversor Análogo Digital.
El puerto A también sirve como un puerto bidireccional de 8 bits con resistencias internas de pull up (seleccionables para cada bit). Los buffers de salida del puerto A tienen características simétricas controladas con fuentes de alta capacidad.

Los pines del puerto A están en tri-estado cuando las condiciones de reset están activadas o cuando el reloj no esté corriendo. El puerto A también sirve para varias funciones especiales del ATmega164P como la Conversión Análoga Digital.

b) Port B (PB7:PB0)

El puerto B es un puerto bidireccional de 8 bits de E/S con resistencias internas de pull up.

Las salidas de los buffers del puerto B tienen características simétricas controladas con fuentes de alta capacidad.
Los pines del puesto B están en tri-estado cuando las condiciones de reset están activadas o cuando el reloj no esté corriendo. El puerto B también sirve para varias funciones especiales del ATmega164P.

c) Port C (PC7:PC0)

El puerto C es un puerto bidireccional de 8 bits de E/S con resistencias internas de pull up (seleccionadas por cada bit). Las salidas de los buffers del puerto C tienen características simétricas controladas con fuentes de alta capacidad.

Los pines del puerto C están en tri-estado cuando las condiciones de reset están activadas siempre y cuando el reloj no esté corriendo. El puerto C también sirve para las funciones de Interfaz del JTAG, con funciones especiales del ATmega164P.

d) Port D (PD7:PD0)

El Puerto D es un puerto bidireccional de entradas y salidas con resistencias internas de pull up (seleccionadas por cada bit). Las salidas de los buffers del puerto D tienen características simétricas controladas con sumideros de fuentes de alta capacidad.
Los pines del Puerto D están en tri-estado cuando llega una condición de reset activa, siempre y cuando el reloj no esté corriendo**.**
El puerto D también sirve para varias funciones especiales del ATmega164P.

e) Reset

Entrada del Reset. Un pulso de nivel bajo en este pin por períodos de pulso mínimo genera un reset, siempre y cuando el reloj no esté corriendo. Pulsos cortos no son garantizados para generar un reset.

f) XTAL1

Entrada para el amplificador del oscilador invertido y entrada para el circuito de operación del reloj interno.

g) XTAL2

Salida del Oscilador amplificador de salida**.**

h) AVCC

AVCC es la alimentación de voltaje para el pin del Puerto F y el Conversor Análogo a Digital. Este debe ser conectado externamente a VCC, siempre y cuando el ADC no sea usado. Si el ADC es usado, este deberá ser conectado a VCC a través de un filtro paso bajo.

i) AREF

Esta es la referencia para el pin de la conversión Análoga a Digital.

j) VCC

Alimentación de Voltaje Digital

k) GND

Tierra

1.2.5 DISPLAY[6]

1.2.5.1 Introducción

La figura 1.15 muestra un ejemplo de display, dentro de la familia de semiconductores hay uno que tiene la particular característica de emitir luz. La existencia de este tipo de dispositivos ha abierto un amplio campo de investigación. Este nuevo campo de investigación es la *Optoelectrónica*.

La optoelectrónica es el nexo de unión entre los sistemas ópticos y los sistemas electrónicos. En esta área juega un papel importante el LED. Que está cada vez más de moda. Hoy en día parece imposible mirar cualquier aparato electrónico y no ver un panel lleno de luces o de dígitos más, o menos espectaculares. Por

[6] http://es.wikipedia.org/wiki/Visualizador

ejemplo, la mayoría de los *walkmanes* disponen de un piloto rojo que nos avisa que las pilas ya han agotado y que deben cambiarse.

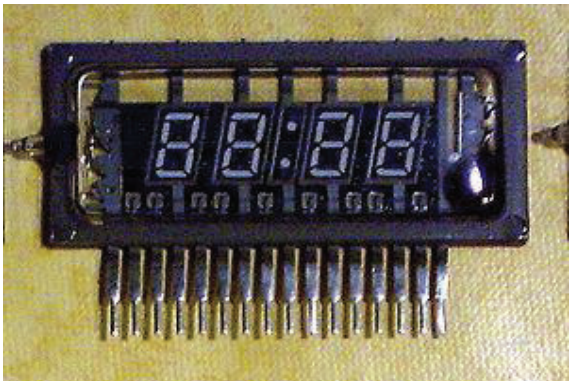


Figura 1. 15 Arreglo de displays

1.2.5.2 Visualizador de segmentos

Existe una gran variedad de formas, intensidades luminosas, dimensiones, colores, etc. Hay diversas empresas que ofrecen dispositivos que mejoran la eficiencia en la utilización de los LED, creando un soporte externo a éste que en la mayoría de casos es más bien de tipo mecánico. Por ejemplo, una de ellas, además de los LED's con encapsulado SMD, los intermitentes que incorporan un circuito integrado en su interior para generar intermitencias de 3 Hz., y las matrices de LED's miniatura, se dedica a fabricar principalmente reflectores, monturas, soportes, LED's con cablecillos etc.

El display de siete segmentos es una forma de representar números en equipos electrónicos.

Está compuesto de siete segmentos que se pueden encender o apagar individualmente. Cada segmento tiene la forma de una pequeña línea.

El display de 7 segmentos o visualizador de 7 segmentos como se muestra en la figura 1.16 es un componente que se utiliza para la representación de números en muchos dispositivos electrónicos debido en gran medida a su simplicidad.

Aunque externamente su forma difiere considerablemente de un diodo LED (diodos emisores de luz) típico, internamente están constituidos por una serie de diodos LED con unas determinadas conexiones internas, estratégicamente ubicados de tal forma que forme un número 8.

A cada uno de los segmentos que forman el display se les denomina a, b, c, d, e, f y g y están ensamblados de forma que se permita activar cada segmento por separado consiguiendo formar cualquier dígito numérico.

Los diodos led trabajan a baja tensión y con pequeña potencia, por tanto, podrán excitarse directamente con puertas lógicas.

Normalmente se utiliza un codificador (en nuestro caso decimal/BCD) que activando un solo pin de la entrada del codificador, activa las salidas correspondientes mostrando el número deseado.

Figura 1. 16 Display de 7 segmentos

Los hay de dos tipos: **ánodo común y cátodo común**.

En los de tipo de ánodo común, todos los ánodos de los leds o segmentos están unidos internamente a una patilla común que debe ser conectada a potencial

positivo (nivel “1”). El encendido de cada segmento individual se realiza aplicando potencial negativo (nivel “0”) por la patilla correspondiente a través de una resistencia que limite el paso de la corriente.

En los de tipo de cátodo común, todos los cátodos de los leds o segmentos están unidos internamente a una patilla común que debe ser conectada a potencial negativo (nivel “0”). El encendido de cada segmento individual se realiza aplicando potencial positivo (nivel “1”) por la patilla correspondiente a través de una resistencia que limite el paso de la corriente.

1.2.6 RELÉ[7]

La figura 1.17 muestra ejemplo de relés.

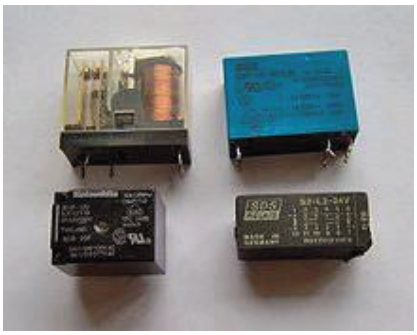


Figura 1. 17 Relé

1.2.6.1 Características

El relé o relevador es un dispositivo electromecánico. Funciona como un interruptor controlado por un circuito eléctrico en el que, por medio de una bobina y un electroimán, se acciona un juego de uno o varios contactos que permiten abrir o cerrar otros circuitos eléctricos independientes

Dado que el relé es capaz de controlar un circuito de salida de mayor potencia que el de entrada, puede considerarse, en un amplio sentido, como un amplificador eléctrico.

[7] http://es.wikipedia.org/wiki/Rel%C3%A9

Como tal se emplearon en telegrafía, haciendo la función de repetidores que generaban una nueva señal con corriente procedente de pilas locales a partir de la señal débil recibida por la línea. Se les llamaba "relevadores" De ahí "relé".

Se denominan contactos de trabajo aquellos que se cierran cuando la bobina del relé es alimentada y contactos de reposo a los cerrados en ausencia de alimentación de la misma. De este modo, los contactos de un relé pueden ser normalmente abiertos, NA o NO, *Normally Open* por sus siglas en inglés, normalmente cerrados, NC, *Normally Closed*, o de conmutación. La lámina central se denomina lámina inversora o de contactos inversores o de conmutación que son los contactos móviles que transmiten la corriente a los contactos fijos.

Los contactos normalmente abiertos conectan el circuito cuando el relé es activado; el circuito se desconecta cuando el relé está inactivo. Este tipo de contactos es ideal para aplicaciones en las que se requiere conmutar fuentes de poder de alta intensidad para dispositivos remotos.

Los contactos normalmente cerrados desconectan el circuito cuando el relé es activado; el circuito se conecta cuando el relé está inactivo. Estos contactos se utilizan para aplicaciones en las que se requiere que el circuito permanezca cerrado hasta que el relé sea activado.

Los contactos de conmutación controlan dos circuitos: un contacto NA y uno NC con una terminal común.

1.2.6.2 Ventajas del uso de relés

La gran ventaja de los relés electromagnéticos es la completa separación eléctrica entre la corriente de accionamiento, la que circula por la bobina del electroimán, y los circuitos controlados por los contactos, lo que hace que se puedan manejar altos voltajes o elevadas potencias con pequeñas tensiones de control.

También ofrecen la posibilidad de control de un dispositivo a distancia mediante el uso de pequeñas señales de control.

En el caso presentado podemos ver un grupo de relés en bases interface que son controlado por módulos digitales programables que permiten crear funciones de temporización y contador como si de un miniPLC se tratase. Con estos modernos sistemas los relés pueden actuar de forma programada e independiente lo que supone grandes ventajas en su aplicación aumentando su uso en aplicaciones sin necesidad de utilizar controles como PLC's u otros medios para comandarlos.

1.2.7 SIRENA[8]

Una sirena es un instrumento acústico como muestra la figura 1.18, capaz de generar sonidos mediante las interrupciones periódicas de una corriente de aire o vapor, por uno o más discos con agujeros situados formando un círculo. La sirena emite un sonido de frecuencia igual al producto del número de orificios por el de revoluciones.

La sirena es un equipo sonoro de alerta en caso de un accidente, un robo, etc.

Figura 1. 18 Sirena de 12 v

[8] http://es.wikipedia.org/wiki/Sirena_(instrumento_ac%C3%BAstico)

CAPÍTULO 2

CONSTRUCCIÓN DEL HARDWARE Y DESARROLLO DEL SOFTWARE DEL SISTEMA

2.1 INTRODUCCIÓN

En el capítulo anterior se dio una descripción de los elementos electrónicos principales que se va a utilizar en el presente proyecto, en este capítulo se detalla la construcción del hardware y software del proyecto siendo el objetivo el diseño e implementación de un sistema automático de control de la fecha y hora para el cambio de horario de clases y visualización de un mensaje de bienvenida para la escuela “Vencedores”

2.2 ANÁLISIS DE LA SITUACIÓN ACTUAL

La escuela Vencedores para realizar el cambio de hora de clases, lo realiza de manera manual. Por medio de un alumno semanero el cual se encarga de hacer sonar la sirena por medio de un pulsador en el cambio de cada hora esto hace que no siempre se timbre a la misma hora y con el mismo lapso de tiempo la sirena

Por lo que el objetivo de la presente tesis es que se lo realice de manera automática.

2.3 CONSTRUCCIÓN DEL HARDWARE DEL SISTEMA

Al presente proyecto se lo puede dividir por etapas como se muestra en la figura 2.1, de esta manera se facilita la explicación de todos los circuitos contenidos en cada una de las etapas.

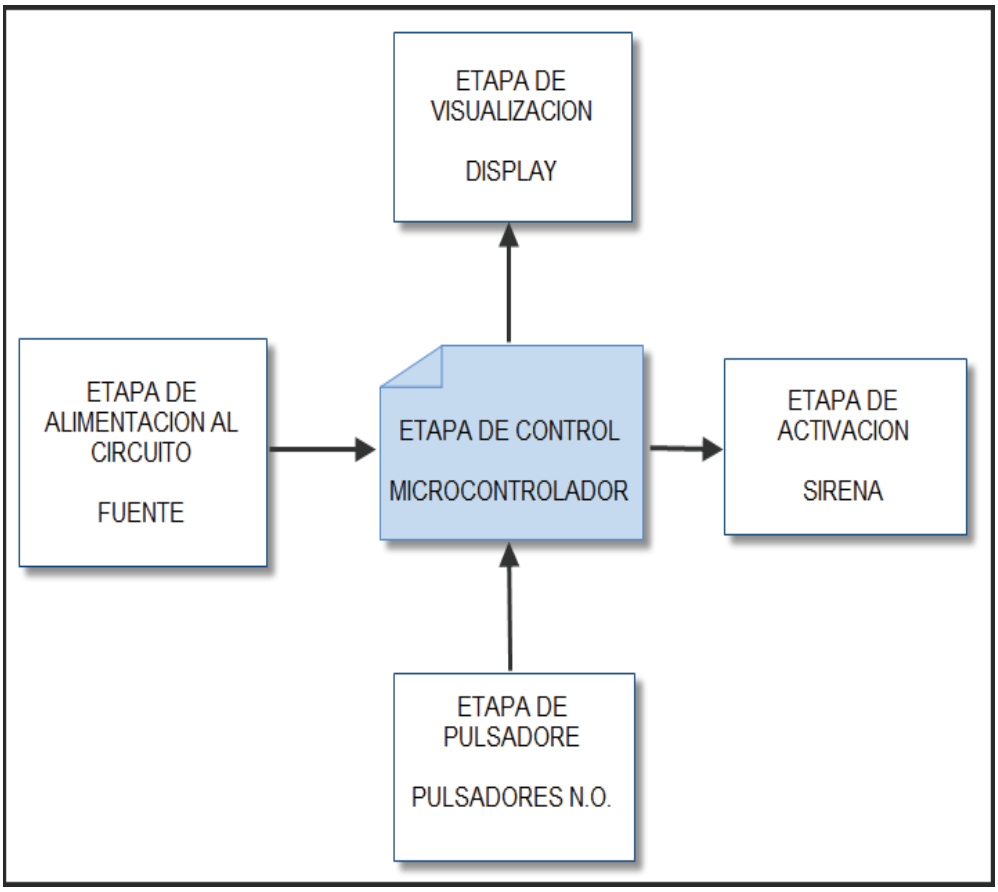


Figura 2. 1 Diagrama en bloques del proyecto

2.3.1 ETAPA DE CONTROL

2.3.1.1 Introducción

La etapa de control es una de las más relevantes del presente proyecto, ya que es el cerebro del sistema, todo gira alrededor de ésta, es el paso por el cual todas las demás etapas deben pasar obligatoriamente para logren cumplir su respectiva y correcta función en donde el elemento principal es el microcontrolador ATmega164, el cual se encarga de tomar las decisiones a la hora de activar la sirena, cuando se detecte el cambio de hora.

2.3.1.2 Diagrama de bloques de las etapas de control

Se detalla de manera general las etapas que forman el control principal del proyecto, como muestran las figuras 2.2 y 2.3.

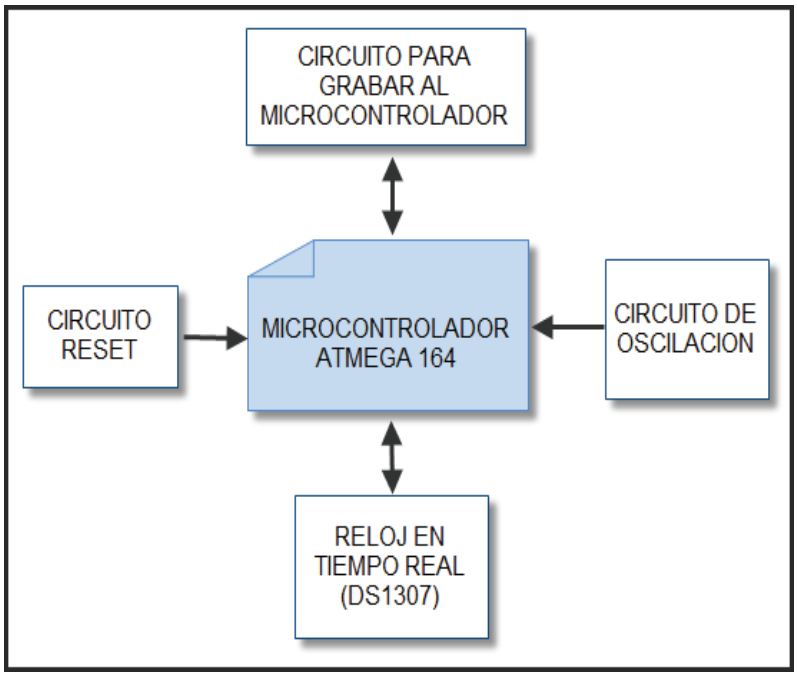


Figura 2. 2 Diagrama de bloques de la etapa de control

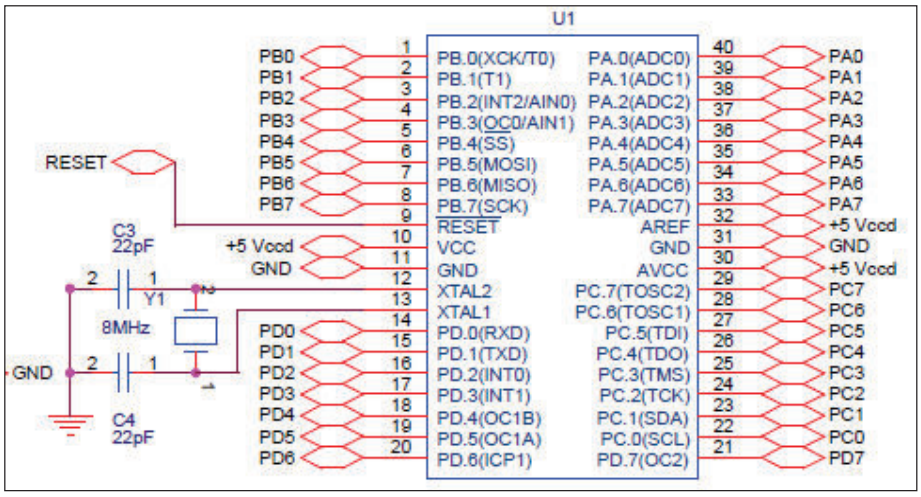


Figura 2. 3 Etapa de control (Microcontrolador Atmega164)

El microcontrolador cumple las siguientes funciones:

- Leer los datos del DS1307 por medio del bus I^2C, decodificar e imprimir en los diplays.

- Realizar el barrido de los displays para que se pueda prender un display a la vez y se pueda visualizar los datos.
- Realizar la comparación de la hora y fecha actual para sonar la sirena.
- Permite que en la hora indicada suene la sirena por 8 segundos.
- Imprimir un mensaje en los displays, el cual se programa por medio del software.

2.3.1.3 Pines del microcontrolador usados en las diferentes etapas

En la tabla 2.1 se muestra una descripción rápida de los pines del microcontrolador destinados para cada una de las etapas.

Tabla 2. 1 Distribución de pines para cada etapa con sus respectivas funciones

ETAPA	PINES	NOMBRE	FUNCIÓN
ALIMENTACIÓN	10 11 30 31 32	Vccd GND Vccd GND Vccd	FUENTE
MANIPULACIÓN	36 35 34 33 19	PA4 PA5 PA6 PA7 PD5	PULSADORES
VISUALIZACIÓN	1 2 3 4 5 6 7 8	PB0 PB1 PB2 PB3 PB4 PB5 PB6 PB7	DISPLAYS
DETECCIÓN	22 23	PC0(SCL) PC1(SDA)	RTC
ACTIVACIÓN	37	PA3	SIRENA

2.3.1.4 Circuito de Reset

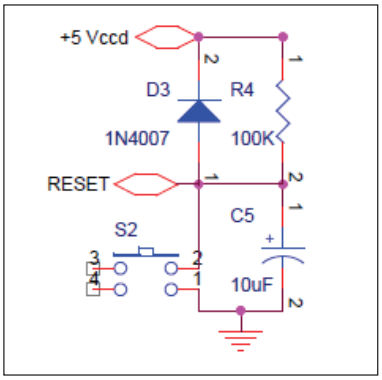


Figura 2. 4 Diagrama del circuito de reset

La figura 2.4 muestra el diagrama del circuito de reset, el reset necesita un 0L para cumplir su función. La resistencia (R4) limita la corriente y lo que entraría es un 1L directo al PIN 9. Siempre va a estar en ese estado hasta el momento en que se presiona el pulsador donde el capacitor se descarga, cuando éste se está descargando el PIN 9 baja a cero y cuando se deja de presionar el capacitor (C5) se vuelve a cargar y comienza otra vez a subir para permanecer en su estado normal, es así como el microcontrolador se resetea. El diodo (D3) es de protección para descargas. La figura 2.4 muestra el diagrama del circuito de reset.

La entrada RESET tiene la función de reiniciar el estado del microcontrolador, teniendo como consecuencia dos acciones importantes:

- Se carga un 0 en el contador de programa, de forma que después de un Reset siempre se ejecuta la instrucción que está en la posición 0 de la memoria de programa.
- Los registros de estado y control toman un estado conocido y determinado.

2.3.1.5 Circuito para conectar el oscilador externo

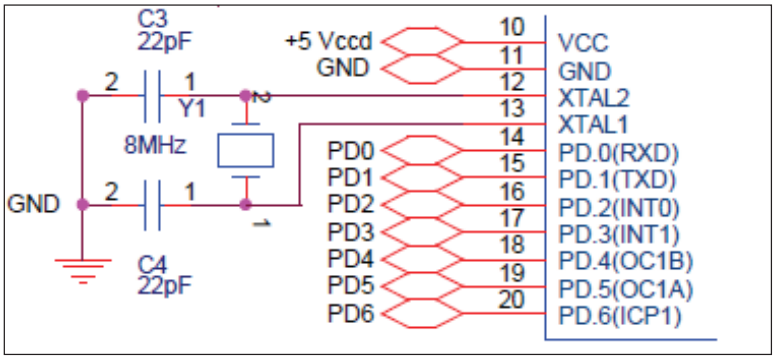


Figura 2. 5 Diagrama del circuito para conectar el oscilador externo

Es necesario conectar un oscilador externo como se indica en la figura 2.5, que garantiza mayor precisión y buen arranque del microcontrolador, ya que la frecuencia que genera éste viene dada por el oscilador.

El circuito, posee un Oscilador (Y1) de 8 MHz, basado en un cristal de cuarzo conectado a dos capacitores (C3 y C4) de 22µf puestos a tierra, es un circuito oscilador tipo XT (Cristal/Resonador cerámico externo, media frecuencia), el oscilador está conectado a los PINES

12: XTAL2 (Salida del Oscilador amplificador de salida)

13: XTAL1 (Entrada para el amplificador del oscilador invertido y entrada para el circuito de operación del reloj interno).

2.3.1.6 Circuito para grabar el Microcontrolador

Para grabar el microcontrolador ATMEGA 164P, se necesita de un grabador, que en este caso es el progisp167 con comunicación USB, el cual consta de los siguientes pines GND, VCC, RESET, SCK, MISO y MOSI, los cuales irán conectado a las bornera (JP3) de la placa, y los tres últimos pines irán conectadas a los pines del microcontrolador PB7, PB6 Y PB5 respectivamente como muestra la figura 2.6, lo que permite que el microcontrolador sea grabado dentro de su propio circuito.

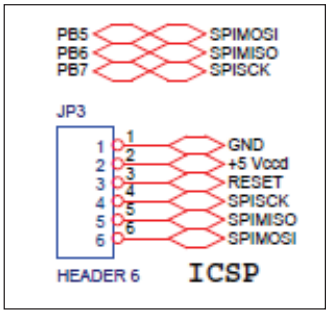


Figura 2. 6 Bornera para conectar los pines del grabador progisp167

2.3.1.7 Circuito del Reloj en Tiempo Real

En la figura 2.7 muestra el diagrama de conexión del DS1307.

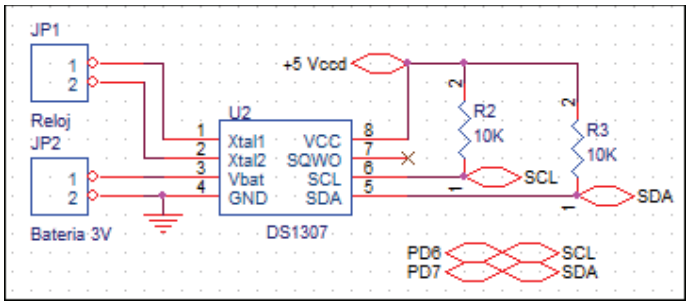


Figura 2. 7 Conexión del DS1307

Como habíamos dicho en el capítulo 1 el bus I2C, permite la comunicación con algunos dispositivos como las memorias 24CXX, los procesadores de señal, codificadores de video, sensores de temperatura en este caso el RTC (reloj de tiempo real) se realizara la lectura y escritura del RTC, para qué comience a trabajar el DS1307 necesita.

- Un cristal de cuarzo de 32.768KHz para que el oscilador interno genere la temporización adecuada.
- Una batería (pila) de 3 voltios para mantener el reloj funcionando cuando se quita la alimentación.
- Dispone de 56 bytes de memoria RAM interna no volátil (mantenida con la batería).
- La configuración, lectura y escritura mediante conexión serie I2C

2.3.1.7.1 Lógica de funcionamiento del DS1307

1. El maestro (ATmega 164) aplica la condición de start al bus. Esto alerta a los esclavos que algo está por venir y entonces, estos sincronizan el flujo de datos.

2. El maestro (ATmega 164) entonces envía un byte con la dirección del esclavo que va a ser activado. Todos los esclavos reciben este byte, pero sólo el esclavo cuya dirección se iguale entra en actividad, los otros ignoran la comunicación hasta ver una nueva condición de start en el bus. Los 7 bits más significativos de este byte es la dirección del esclavo, el bit menos significativo controla la dirección de la próxima transferencia (uno o más bytes). Un "0" indica una transferencia de maestro a esclavo, y un "1" indica una transferencia de esclavo a maestro. Este bit normalmente se denomina R/W.

3. Los datos entonces son transferidos según el bit R/W.

4. Los datos pueden ser transferidos en ambas direcciones durante un intercambio reenviando el pulso de start y cambiando el bit de R/W en el byte de dirección.

5. Finalmente, el maestro (ATmega 164) debe enviar la condición de stop al bus para concluir la transferencia. El compilador BASIC de BASCOM- AVR, incorpora funciones de alto nivel que nos permiten el manejo del bus I2C.

El bus I2C puede implementarse usando el hardware del microcontrolador.

Para realizar el reloj en tiempo real se utiliza el circuito integrado DS1307 que es un chip que tiene un reloj calendario en el cual sus datos son en BCD para los números de las horas minutos y segundos, mientras que para los días es un numero por ejemplo el lunes.

El dato que me entrega el DS1307 es un código BCD.

El microcontrolador ATMEGA 164 con el DS1307 se conecta por medio de un bus I^2C, el cual es de lectura y los datos que me entrega el DS1307 son decodificados por medio del microcontrolador ATMEGA 164 el cual por un lado se encarga de

convertir de BCD a los siete segmentos de los displays y también como en el proyecto se quiere que suene una sirena cuando se la hora programada se realiza con la comparación del dato que me entrega el DS1307 y el dato de constante del microcontrolador por ejemplo para que suene el día lunes será comparado con la constante.

Y para que suene a la hora indicada también se realiza la comparación con el dato que se encuentra en el microcontrolador.

Tiene un oscilador el cual me da la frecuencia de trabajo y una pila para que no se desiguale.

2.3.2 ETAPA DE VISUALIZACIÓN

2.3.2.1 Introducción

La etapa de visualización es la encargada de presentar por medio de ocho display's de siete segmentos la hora actual, el día actual, y un mensaje de bienvenida. Esto se logra por medio del programa almacenado en el microcontrolador ATmega 164, como también sus etapas de manejo de los segmentos de cada display y el barrido de cada display.

2.3.2.2 Diagrama de bloques de la etapa de visualización

Se detalla de manera general las etapas que forman la etapa de visualización del proyecto como muestra la figura 2.8.

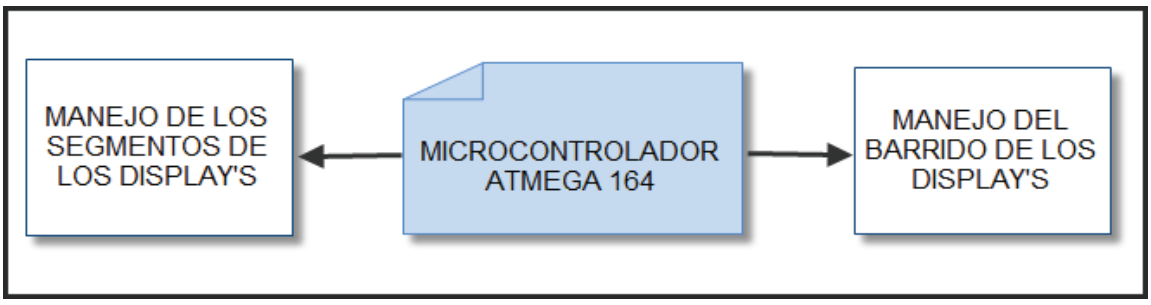


Figura 2. 8 Diagrama de bloques de la etapa de visualización

2.3.2.3 Manejo de los segmentos de cada display

En la figura 2.9 se muestra el circuito de aislamiento entre la etapa de control y visualización.

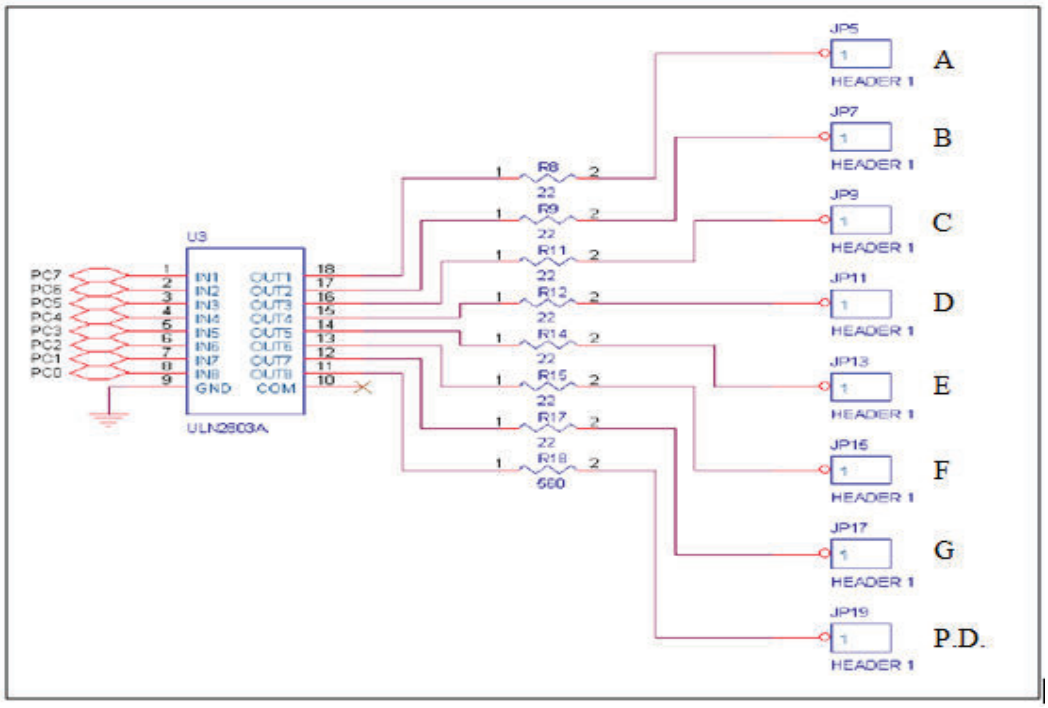


Figura 2. 9 Circuito de aislamiento entre la etapa de control y visualización de cada segmento

Debido a la pequeña corriente que suministra el microcontroladores ATmega164, de 25mA por pin, sólo se puede hacer funcionar directamente pocos leds en serie. Por este motivo es necesario amplificar a las salidas del microcontrolador ATmega164 en función a los segmentos de los display's que vamos a controlar. Un método sencillo y económico es emplear el integrado ULN2003A, que es un conjunto de Darlington (darlington array) montados en un solo chip con el que podemos controlar cargas de hasta medio amperio. El chip lleva diodos de protección contra las sobretensiones producidas por cargas inductivas.

Las ocho entradas del ULN2803 se conectan a las salidas del microcontrolador ATmega164; las cuales son las encargadas de manejar los segmentos individualmente, a la salida del ULN2803 se conecta a los segmentos de los display's, los cuales están cortocircuitados entre sí de manera ordenada.

Para formar un letra el microcontrolador se lo configura como salida, los pines correspondientes a la letra que se desea formar, por ejemplo si se desea formar la letra a se activan los segmentos correspondientes a dicha letra, es decir los segmentos a,b,c,e,f,g. Los display's que se utiliza son de ánodo común por lo que los segmentos se van a activar con 0L o GND. Ya que el común del display va conectado a VCC, como muestra la figura 2.10.

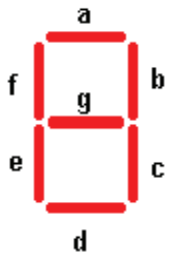


Figura 2. 10 Distribución de los segmentos de un display

2.3.2.4 Manejo del barrido de los display's

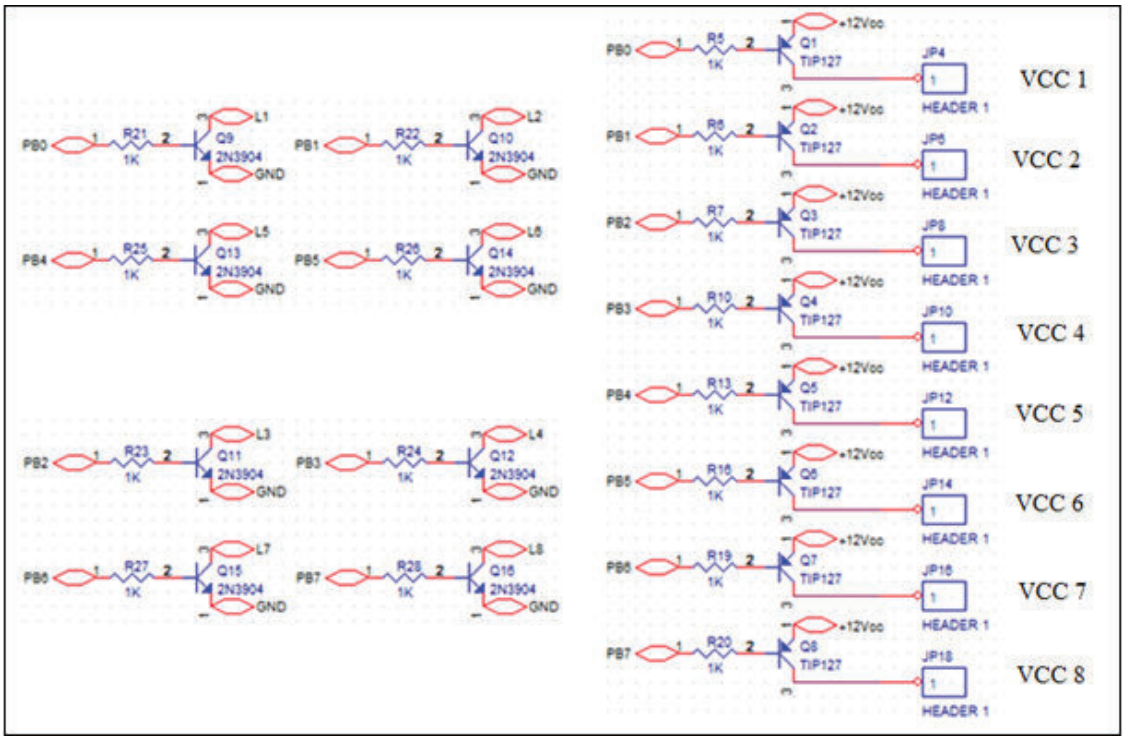


Figura 2. 11 Diagrama para conexión del barrido de los display's

Para visualizar los datos se usa ocho displays de ánodo común, esto quiere decir que sus ánodos están conectados entre sí. Para activar un segmento del display se conecta a tierra su respectivo cátodo.

Para activar un display a la vez se realiza polarizando el ánodo común de cada display, con esto se consigue realizar un barrido y que las letras se desplacen de derecha a izquierda.

Los display que se utilizan en el presente proyecto funcionan con 12 VCC y 35mA por lo cual toca realizar una etapa de acoplamiento de voltaje, ya que el microcontrolador funciona con 5 VCC y entrega por cada pin 25mA

Para realizar el acoplamiento de voltaje a los display se utiliza dos transistores. Un transistor 2n3904 para realizar la etapa de control del TIP127. El transistor 2n3904 es de tipo NPN, por lo que para ser saturado necesita de un voltaje positivo a la base, al entrar en saturación dicho transistor de colector a emisor permite el paso de la corriente, al permitir el paso de la corriente hace que en la base del TIP127 entre un voltaje negativo, al ser de este transistor del tipo PNP hace que se sature permitiendo el paso de corriente de emisor a colector y con eso que se active en correspondiente ánodo de los display’s.

En la figura 2.11 se observa la conexión de los dos transistores en la cual de PB0, PB1, PB2, PB3, PB4, PB5, PB6, PB7 son salidas del microcontrolador y VCC1, VCC2, VCC3, VCC4, VCC5, VCC6, VCC7, VCC8 son los ánodos de cada display respectivamente

2.3.3 ETAPA DE ALIMENTACIÓN

En la figura 2.12 se observa la parte de regulación de la fuente de alimentación.

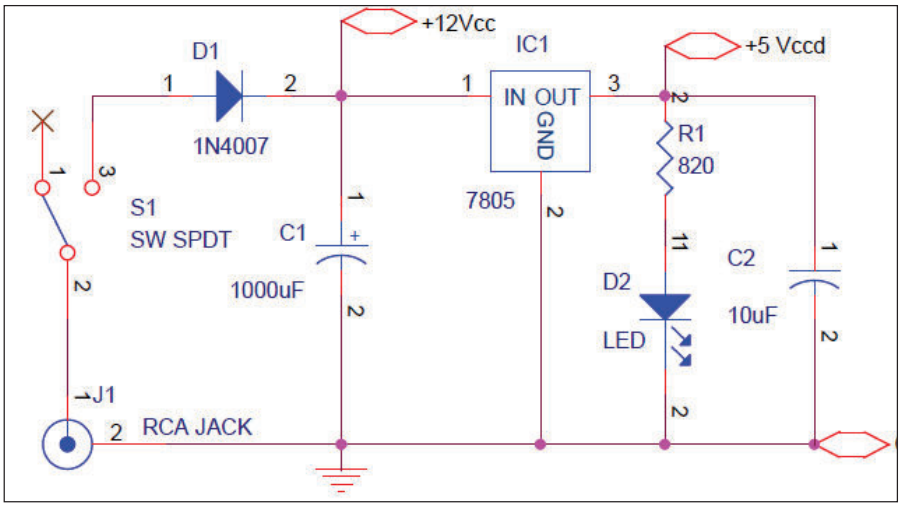


Figura 2. 12 Diagrama de la fuente de alimentación del circuito

En la entrada se utiliza una fuente conmutada de 12 VDC, se coloca a la entrada un diodo (1N4007) para proteger al circuito de una conexión opuesta de polos de la fuente de 12 Voltios.

El capacitor C1 (1000uF) permite estabilizar el voltaje a la entrada, cuando la sirena se enciende, permitiendo al condensador mantener estabilidad de voltaje. El integrado de regulación de voltaje IC1 (7805) permite regular a 5 voltios de corriente continua, este integrado soporta voltajes de entrada de 7 a 36 Voltios.

En led D2 indica la existencia de voltaje a 5 Voltios, mientras que el capacitor C2 (10uF) mantiene el voltaje, y evita variaciones del mismo.

Para alimentar a los display se conecta directo a la entrada de 12 VDC.

2.3.4 ETAPA DE MANIPULACIÓN

2.3.4.1 Introducción

La etapa de manipulación es la encargada de ingresar datos al microcontrolador ATmega164 para poder realizar la igualación de la hora y fecha del reloj en tiempo real, y la activación manual del relé.

2.3.4.2 Diagrama de bloques de la etapa de manipulación

Se detalla de manera general las etapas que forman la etapa de manipulación del proyecto, como se muestra en la figura 2.13.

Figura 2. 13 Diagrama de bloques de la etapa de manipulación

2.3.4.3 Manipulación del pulsador manual para activar la sirena

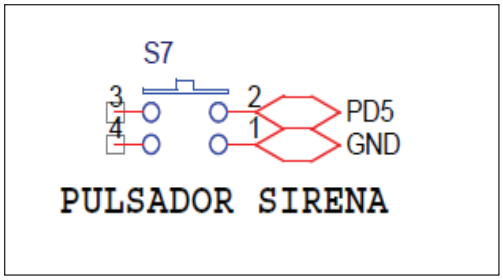


Figura 2. 14 Pulsador de sirena

Para la activación de la sirena manualmente, se realiza por medio de un pulsador N.O. (normalmente abierto) que se encuentra conectado en el pin PD5 del microcontrolador ATmega164, como indica la figura 2.14.

Internamente se activa la resistencia de pull-up que tiene el microcontrolador y así por medio de programa en el momento que se activa el pulsador se da un dato de 0L al microcontrolador ATmega164 para que se active la sirena

Para aprovechar la activación manual y realizar una sirena de evacuación si se produce un desastre natural se la configura que en el momento que se la presione el pulsador va a sonar la sirena cinco segundos y si se vuelve a presionar otros c segundos haciendo que se sume cada vez que se presione. Los tonos de la sirena quedarían como a continuación.

Para el cambión de hora la sirena suena 8 segundos (se activa automáticamente por medio del programa que se encuentra en el microcontrolador ATmega164)

Ante la presencia de un desastre natural la sirena suena 10 segundos (se activa al pulsar dos veces seguidas)

Para la reunión de profesores la sirena suena 5 segundos (se activa al presionar una solo vez el pulsador)

2.3.4.4 Manipulación de pulsadores de ingreso de datos para igualar la hora y fecha

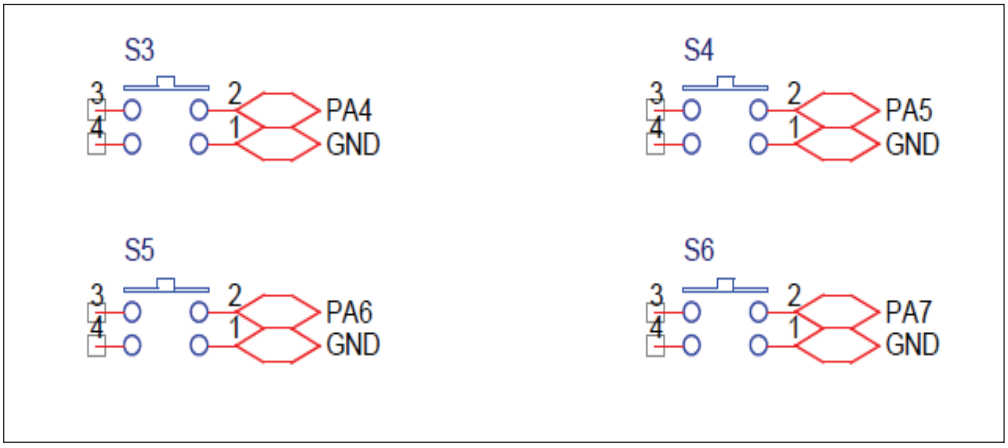


Figura 2. 15 Diagrama de pulsadores (hora y fecha)

Estos cuatro pulsadores nos permiten igualar la hora y la fecha en el caso de que se hubiese desigualado el reloj como indica la figura 2.15, y son los siguientes:

1.- Enter, le corresponde el ping PA7 del microcontrolador ATmega164

2.- Subir, le corresponde el ping PA4 del microcontrolador ATmega164

3.- Bajar, le corresponde el ping PA5 del microcontrolador ATmega164

4.- Mover, le corresponde el ping PA6 del microcontrolador ATmega164

Se configura todos estos pines como entradas de datos al microcontrolador y se activa sus respectivas resistencias de pull-up para que en el momento de presionar cambie de estado y por medio de programa realice una función específica.

Primero para proceder a igualar la hora o fecha, se pulsa el botón enter. Se prende el punto decimal de los displays indicando si es hora, minuto o segundo.

Si se pulsa el botón subir me permite incrementar en un digito el dato.

Si se pulsa el botón bajar me permite decrementar en un digito el dato.

Si se pulsa el botón mover me permite realizar el desplazamiento y también permite cambiar de horas a minutos para poder igualar el reloj.

Y si sigo aplastando el botón de mover me dirijo a igualar la el día y la fecha.

2.3.5 ETAPA DE ACTIVACIÓN

2.3.5.1 Introducción

La etapa de activación es la encargada de accionar la sirena, en las horas programadas en el microcontrolador ATmega164 y se va a activar un tiempo determinando, luego del cual se va a desactivar automáticamente.

2.3.5.2 Diagrama de bloques de la etapa de activación

Se detalla de manera general las etapas que forman la etapa de activación del proyecto, como se indica en la figura 2.16.

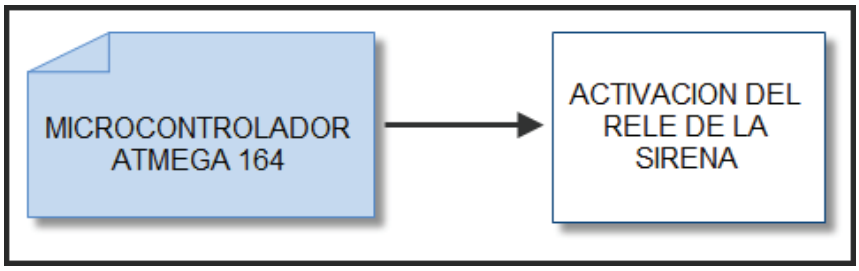


Figura 2. 16 Diagrama de bloques de la etapa de activación

Esta etapa cumple la función de aviso de cambio de hora, como se muestra en la figura 2.17

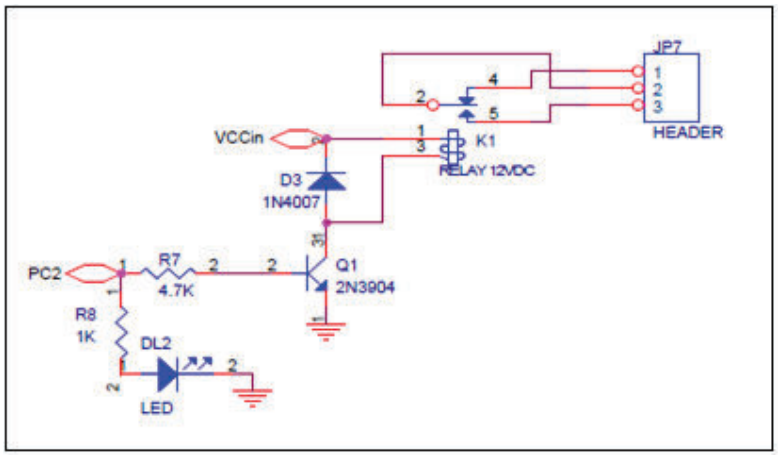


Figura 2. 17 Diagrama para conexión de sirena

Para realizar la activación de sirena se lo realiza por medio del cierre de los contactos de un relé, el cual está configurado como se muestra en la figura 2.17.

Consta de las siguientes partes:

Un led indicador, el cual se va a prender en el momento de que se active la sirena

Para realizar la activación y desactivación del relé se realiza por medio de un transistor NPN 2N3904 el cual trabaja en dos estados que son corte y saturación.

En corte no permite el paso de corriente entre sus terminales de emisor y colector, mientras que en saturación permite el paso de corriente entre colector y emisor.

En el momento que el microcontrolador envía un cambio de estado es decir un 1L (5 VDC) a la base del transistor este entra en saturación haciendo que se permita la circulación de corriente entre colector y emisor.

Cuando el transistor entra en saturación hace que se energice la bobina del relé y se produzca el cierre de su contacto normalmente abierto, se utiliza un diodo conectado en polarización inversa entre el transistor y el relé para proteger al transistor de descargas de la bobina del relé.

2.4 DISENO DEL CIRCUITO ESQUEMÁTICO Y DE PISTAS

Al realizar todas las pruebas necesarias al circuito, se procede a la fabricación del circuito impreso (PCB).

Primero se tiene que realizar el circuito esquemático, esto consiste en dibujar el circuito, utilizando los símbolos electrónicos en la computadora, para la realización del presente proyecto se utiliza el programa OrCAD Capture para Windows en el cual se desarrolla todos los diagramas esquemáticos y circuitales, como se puede ver en la figura 2.18.

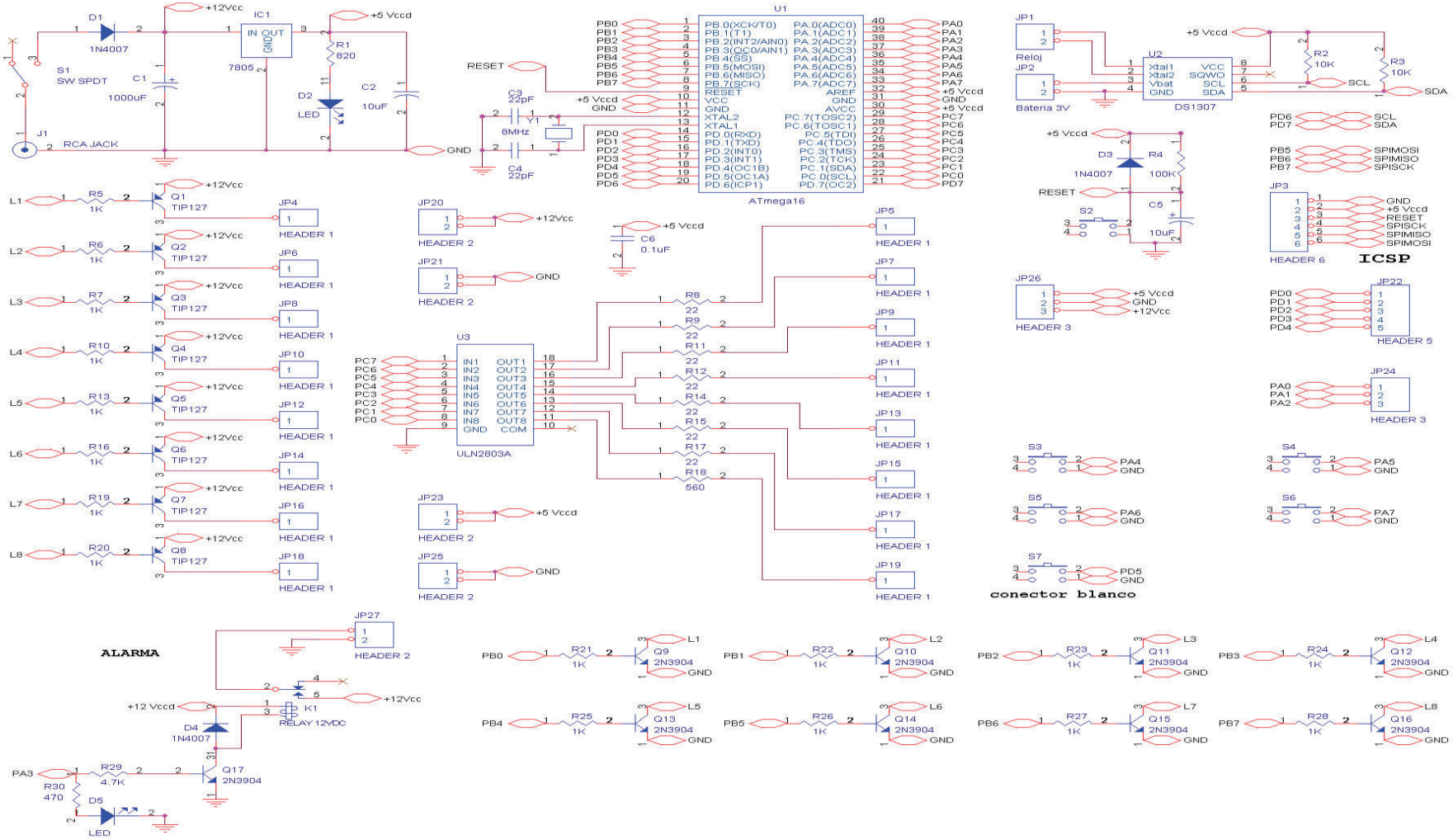


Figura 2. 18 Diagrama Esquemático del proyecto

Después de realizar el diagrama esquemático, se procede a realizar el trazado de las pistas para ello se utiliza el programa de OrCAD layout**,** en la pantalla de este programa, una vez que ya témenos todos los elementos se procede a ubicarlos de acuerdo a nuestra necesidad.

Una vez que esté bien colocado los elementos, se procede a rutear y como resultado final se tiene el diagrama de pistas, como se indica en la figura 2.19

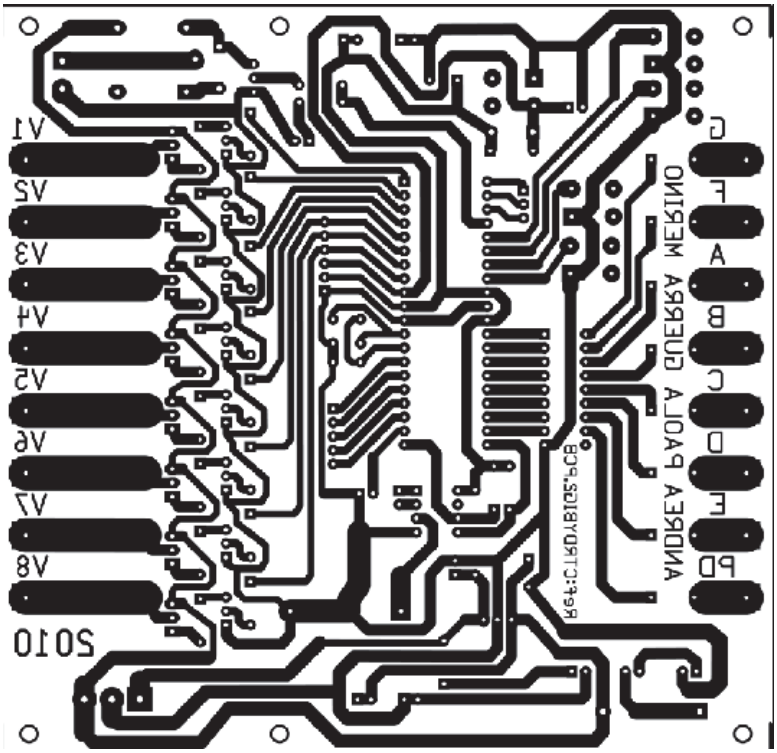


Figura 2. 19 Diagrama del Circuito Impreso

Para imprimir el screen de elementos, se tomará en cuenta que debe estar en efecto espejo y sin las pistas, es decir seleccionado Top Silk y Mirror, como se indica en la figura 2.20

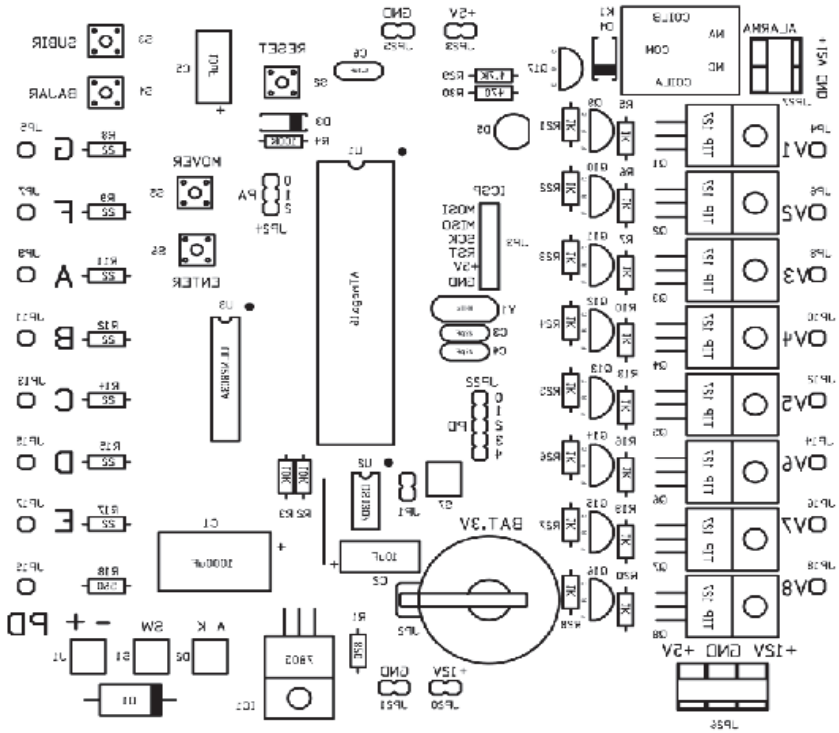


Figura 2. 20 Screen de los Elemento

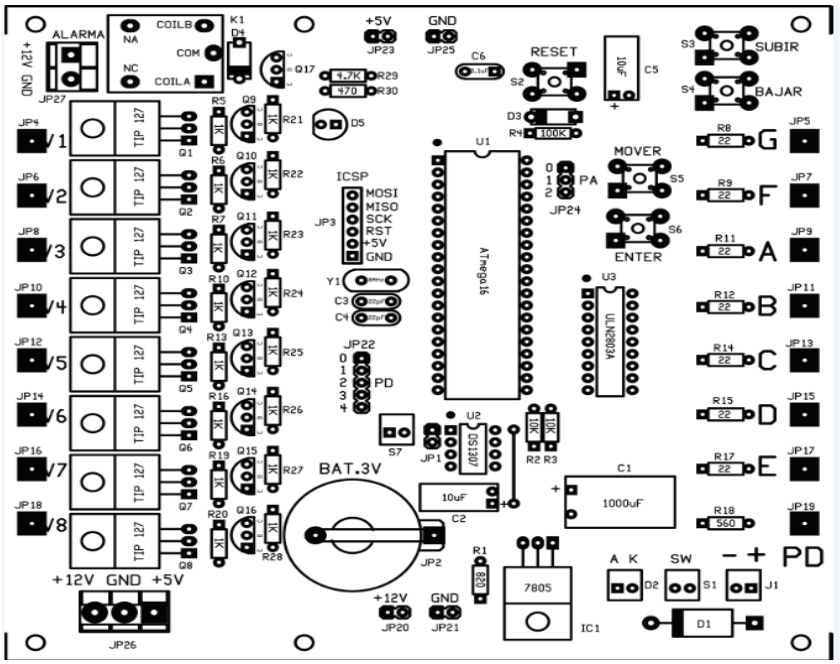


Figura 2. 21 Diagrama posicional de los elementos

2.5 SOFTWARE

Una parte primordial en el desarrollo de cualquier proyecto es la programación de los microcontroladores, después de haber realizado todo el hardware se procede a realizar la parte de programación que va a ser el alma del proyecto.

Para la programación de los microcontroladores antiguamente se lo realizaba por medio de lenguaje de ensamblador, lenguaje de bajo nivel, con instrucciones básicas y de alta complejidad para entender, actualmente existen nuevas herramientas de programación con un lenguaje mucho más entendible y fácil de utilizar

2.5.1.1 Lenguaje de programación para microcontroladores AVR

Hay diferentes tipos de lenguaje de programación, así como también compiladores para microcontroladores, los cuales tienen sus ventajas y desventajas para este trabajo se utiliza el compilador BASCOM AVR, de leguaje de programación en BASIC, se eligió este compilador porqué dispone de recursos y funciones integradas propias, para llevar con éxito el presente proyecto

2.5.1.1.1 Lenguaje de programación Bascom AVR

El software BASCOM- AVR es desarrollado por la empresa MCS Electronics, sirve para realizar programas de alto nivel para microcontroladores AVR. Ofrece una completa solución para editar, compilar, simular y programar. Posee un compilador y un ensamblador que traduce las instrucciones estructuradas en lenguaje de máquina.

Ventajas

- BASIC estructurado con etiquetas
- De programación estructurado con if-then-else-END IF, DO-LOOP, MIENTRAS-WEND, de SELECT-CASE.
- Rápido código de máquina en lugar de código interpretado.
- Variables y las etiquetas pueden ser tan largo como 32 caracteres.
- Bit, Byte, Integer, Word, Long, único y de cadenas de variables.
- Amplio conjunto de trigonométricas funciones de punto flotante. Fecha y hora de cálculo funciones.

- Compilado programas de trabajo con todos los microprocesadores AVR que tienen memoria interna.
- Las declaraciones son altamente compatibles con Microsoft la VB / QB.
- Comandos especiales para pantallas LCD, chips I2C y 1WIRE chips, PC keyboad, matriz-keyboad, RC5 recepción, el software UART, SPI, con pantalla LCD gráfica, enviar IR RC5, RC6 o código de Sony. TCP / IP W3100A con chip.
- Variables locales, las funciones de usuario, apoyo de biblioteca.
- Integrado emulador de terminal con opción de descarga.
- Integrado simulador para la prueba.

2.5.2 DESARROLLO DEL SOFTWARE

En la figura 2.22 se indica el diagrama de pasos para programar un microcontrolador, para desarrollar cualquier proyecto con microcontroladores se debe seguir los siguientes pasos:

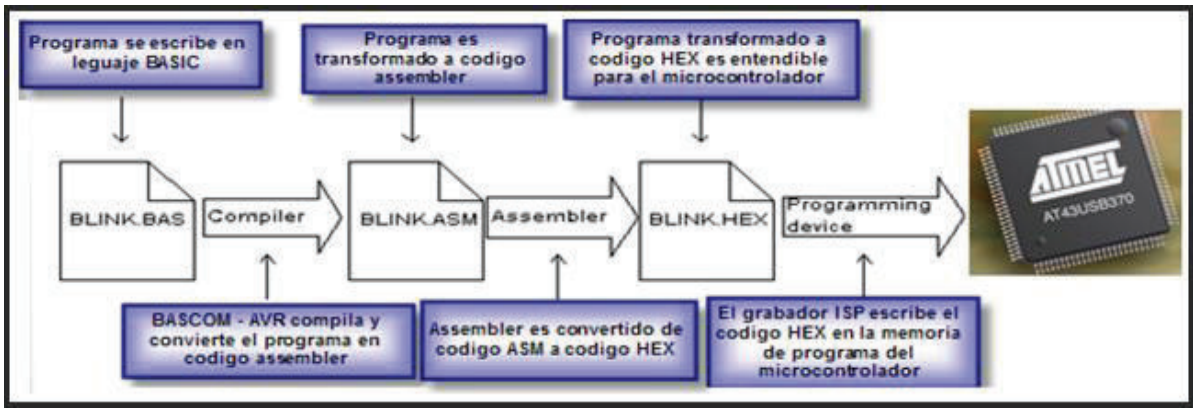


Figura 2. 22 Diagrama de bloques de pasos para programar un microcontrolador

Pasos para crear un programa:

I. Escribir el programa en BASIC, crea un archivo BAS.
II. Compilar el programa y ver si no contiene errores
III. Si no tiene errores se crea un archivo ASM que es un archivo en ensamblador

IV. El archivo de ensamblador es entendible para la maquina en donde se está programando pero para el microcontrolador por lo cual crea un archivo en Hexadecimal HEX

V. El archivo HEX es entendible para el microcontrolador, este archivo es el que se graba en la memoria de programa por medio de un grabador o programador ISP

2.5.3 GRABANDO EL PROGRAMA AL MICROCONTROLADOR

Al finalizar de escribir el programa se compila, creándose algunos archivos de los cuales el que nos sirve para grabar en el microcontrolador es el archivo hexadecimal .hex el cual posee todas las instrucciones que el microcontrolador necesita para que pueda funcionar, este archivo es guardado en la memoria de programa del microcontrolador utilizando el grabador o programador.

Al otro lado del grabador se conecta el microcontrolador con los pines específicos para poder realizar esta función que son: miso, mosi, sck, reset, vcc y gnd, como se muestra en la figura 2.23

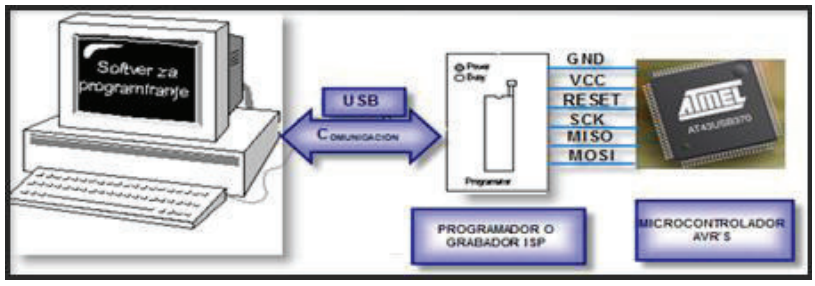


Figura 2. 23 Pines de conexión entre grabador y el microcontrolador

En el mercado se encuentra una diversidad de circuitos grabadores de AVR, los cuales nos muestran principalmente el tipo de microcontrolador, los fusibles y el archivo a cargar en el microcontrolador.

Por ejemplo dentro de la ayuda de BASCOM, se encuentra un circuito grabador, llamado STK 200-300 (ISP programmer), el cual utiliza el puerto paralelo (DB25) para grabar al microcontrolador

Para realizar la grabación del microcontrolador se ha utilizado el grabador PROGISP USB-ATMEL realizado por INE4C, como muestra la figura 2.24

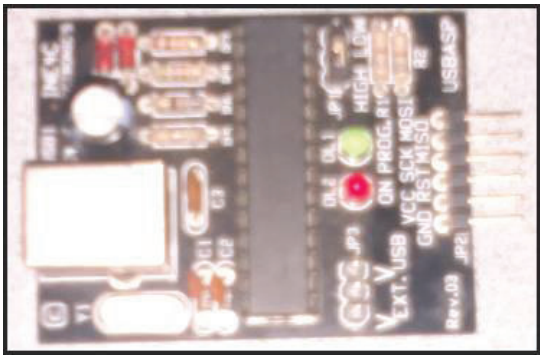


Figura 2. 24 Programador USB para el microcontrolador AVR

El grabador posee las siguientes características:

1. Se comunica con el computador mediante un puerto USB

2. Posee un jumper el cual permite alimentar al microcontrolador con el voltaje del computador o con una fuente externa.

3. Posee un jumper que selecciona la velocidad de grabación.

4. Para guardar el archivo hexadecimal el grabador posee 6 pines de conexión

El grabador, utiliza el programa progisp 1.6.7, que es un software muy amigable, que nos permite grabar los “fuse bits” y el archivo .HEX.

Dicho software tiene la siguiente pantalla, que se puede observar en la figura 2.25

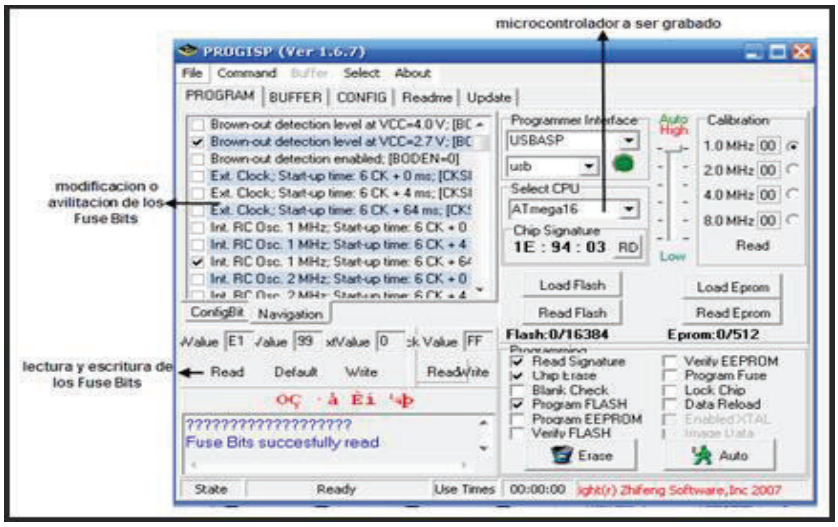


Figura 2. 25 Pantalla de grabación del programador USB PROGISP 1.6.7

Este software sirve para las siguientes opciones como también la configuración de los fuses

- Leer el contenido del microcontrolador
- Borrar el contenido del microcontrolador
- Verificar si la grabación se realizó correctamente
- Cargar automáticamente el archivo hexadecimal
- Proteger al archivo hexadecimal contra lectura

Los fuses son bits que determinan la funciones propias con que puede trabajar el microcontrolador

- Seleccionar el nivel de voltaje de funcionamiento
- El oscilador con el cual trabajará el microcontrolador
- Definir si el oscilador es interno o externo
- Tipo de comunicación
- Grabar algún archivo en la memoria de datos del microcontrolador

Los fuses bits, están divididos en fuses alto y fuses bajo. A continuación una breve descripción de estos bits:

En el registro de bits de fuse altos se configuran los siguientes:

- **OCDEN:** Habilita algunos osciladores a pesar de estar en modo sleep.
- **JTAGEN:** Habilita el JTAG, interfaz que cumple con el estándar 1149.1 de la IEEE.
- **SPIEN:** Habilita o deshabilita el uso del ISP.
- **CKOPT:** Su funcionalidad depende de los bits de CKSEL.
- **EESAVE:** Indica si se borra la memoria Eeprom o no durante el ciclo de borrado.
- **BOOTSZ1:** Configura el tamaño del arrancador.
- **BOOTSZ0:** Cargador.
- **BOOTRST:** Selecciona donde comienza el vector del reset.

En el registro de bits de fuses bajos se configuran los siguientes:

- **BODLEVEL:** Indica el nivel en el que se detecta el nivel de bajo voltaje.
- **BODEN:** Habilita el detector de nivel de bajo voltaje.
- **SUT1, SUT0:** Indica el tiempo que debe de esperar antes iniciar el programa dentro del microcontrolador.
- **CKSEL3, CKSEL2, CKSEL1, CKSEL0:** Se utiliza para seleccionar los tipos de reloj a utilizar desde el oscilador interno de 1 MHz hasta 8 MHz internos, o los osciladores externos que alcanzan hasta los 16MHz.

2.5.4 DESARROLLO DEL SOFTWARE DEL SISTEMA

Indica el desarrollo del software del proyecto, para lograr el correcto funcionamiento según los requerimientos propuestos en el proyecto. Las líneas de comando del programa se encuentra en el ANEXO A

CAPÍTULO 3

IMPLEMENTACIÓN Y PRUEBAS DE FUNCIONAMIENTO

3.1 PRUEBAS

3.1.1 INTRODUCCIÓN

En el presente capitulo se muestran las pruebas y los resultados que se obtuvieron al final de realizar el hardware y software, realizando la revisión final de los requerimientos, análisis y diseño para finalmente realizar la implementación.

El objetivo de las pruebas es encontrar fallas o errores para luego hacer una depuración del sistema y así asegurar que el proyecto ha sido desarrollado de acuerdo a los requerimientos y que todos los errores han sido detectados.

3.1.2 PRUEBA DE LOS DIFERENTES SISTEMAS

Se realizaron las pruebas a cada sistema, con el fin de lograr una depuración al detectar posibles errores.

Para llevar a cabo las pruebas de los diferentes sistemas se ha definido el siguiente formato:

Tabla 3. 1 Formato de pruebas de los diferentes sistemas

Número:		<Número de la prueba>	
Prueba:		<Nombre de la prueba>	
No.	**Tipo de prueba**	**Acción**	**SI/NO**
1	<Acción 1>	<Resultado obtenido 1>	S/N
2	<Acción 2>		S/N
…	…		S/N
n	<Acción n>		S/N
Conclusión:			

Para realizar las pruebas se ha dividido de manera individual a los sistemas que forma el proyecto que son:

- Sistema de reloj en tiempo real
- Sistema de visualización del mensaje
- Sistema de activación de la sirena

Las pruebas se van a realizar a cada sistema de manera separada y una prueba final a todo el sistema completo

3.1.2.1 Pruebas del Sistema de Reloj en Tiempo Real

En la siguiente tabla se muestran los resultados obtenidos al realizar las pruebas del reloj en tiempo real.

Tabla 3. 2 Cuadro de Pruebas del reloj en tiempo real

Número:	1		
Prueba:	Reloj en Tiempo Real		
No.	**Tipo de prueba**	**Acción**	**SI/NO**
1	Presenta la hora y día actual en los display	Se observa en los ocho display la hora y el día actual	SI
2	Se apaga la fuente de alimentación del circuito, que sucede con el reloj	En el momento de apagar y volver a prender la fuente de alimentación del circuito el reloj no se desiguala	SI
3	Prueba de los pulsadores	Por medio de los pulsadores puedo acceder al menú para igualar la hora y el día	SI
Conclusión:	El reloj de tiempo real (DS1307) funciona correctamente		

3.1.2.2 Pruebas del Sistema de Visualización del Mensaje

En la siguiente tabla se muestran los resultados obtenidos al realizar las pruebas de visualización del mensaje.

Tabla 3. 3 Cuadro de Pruebas de visualización del mensaje

Número:	1		
Prueba:	Visualización del mensaje		
No.	**Tipo de prueba**	**Acción**	**SI/NO**
1	Se realiza el barrido de los display	Se observa que se desplaza el mensaje de derecha hacia izquierda	SI
2	Presenta el mensaje	Presenta un mensaje de bienvenida, y después imprime la hora y día actual	SI
3	Se apaga la fuente de alimentación del circuito, que sucede con el mensaje	En el momento de apagar y volver a prender la fuente de alimentación del circuito el mensaje permanece, es decir no se borra	SI
Conclusión:	Todos los display, el barrido de cada una y la activación de cada segmento funciona correctamente		

3.1.2.3 Pruebas del Sistema de Activación de la Sirena

En la siguiente tabla se muestran los resultados obtenidos al realizar las pruebas activación de la sirena.

Tabla 3. 4 Cuadro de Pruebas de activación de sirena

Número:	1		
Prueba:	Activación de la Sirena		
No.	**Tipo de prueba**	**Acción**	**SI/NO**
1	Se activa la sirena en la horas programadas	Si activa la sirena en las horas que se encuentran programas en el microcontrolador, la sirena suena 8 segundos	SI
2	Se activa el pulsador manual una solo vez	La sirena suena cinco segundos	SI
3	Se activa el pulsador manual dos veces	La sirena suena diez segundos	SI
Conclusión:	La activación de sirena, tanto manual como automática funcionan correctamente		

3.2 IMPLEMENTACIÓN DEL PROYECTO FINAL

Culminado ya el diseño del hardware y del software del sistema se da comienzo con la implementación de todos los elementos y dispositivos que conforman el circuito del sistema del control de hora y el mensaje, tratando de ubicar en un lugar adecuado tanto para la visión como para la audición.

3.2.1 MODELO FINAL DEL PROYECTO

Al circuito y los displays se les ubico en una caja de madera como se muestra en la figura 3.1.

Figura 3. 1 Caja donde se ubicara el circuito

Esta caja de madera posee una medida de 1m de largo la cual estará dividida en tres partes para que pueda instalarse en su interior el circuito central, los displays y la fuente.

En la figura 3.2 se muestra la distribución de los circuitos, la caja queda dividida de la siguiente manera:

Para el circuito central una medida de 14cm de largo por 15cm de ancho y 9cm de profundidad.

Para los displays una medida de 73cm de largo por 15cm de ancho y 9cm de profundidad.

Para la fuente una medida de 13cm de largo por 15cm de ancho y 9cm de profundidad.

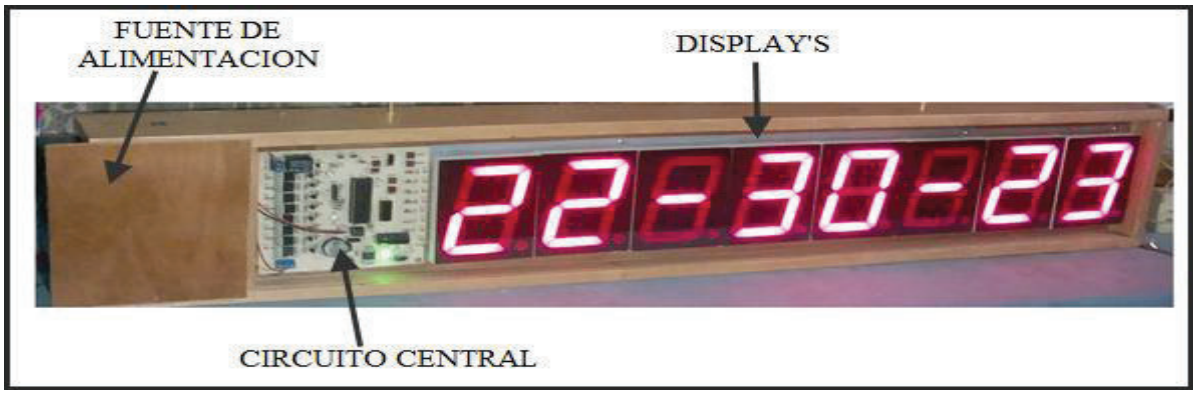


Figura 3. 2 Distribución de los circuitos del proyecto

3.2.1.1 Unidad Central del Sistema.

La unidad central del sistema está formada por los siguientes elementos como se muestra en la figura 3.3

- Un microcontrolador ATmega164
- Un reloj en tiempo real (DS1307)
- Una batería de 3.3 V para el reloj
- Ocho transistores 2N3904
- Ocho transistores TIP127
- Un ULN2803A
- Un regulador de voltaje 7805
- Cinco pulsadores
- Un relé para activar la sirena

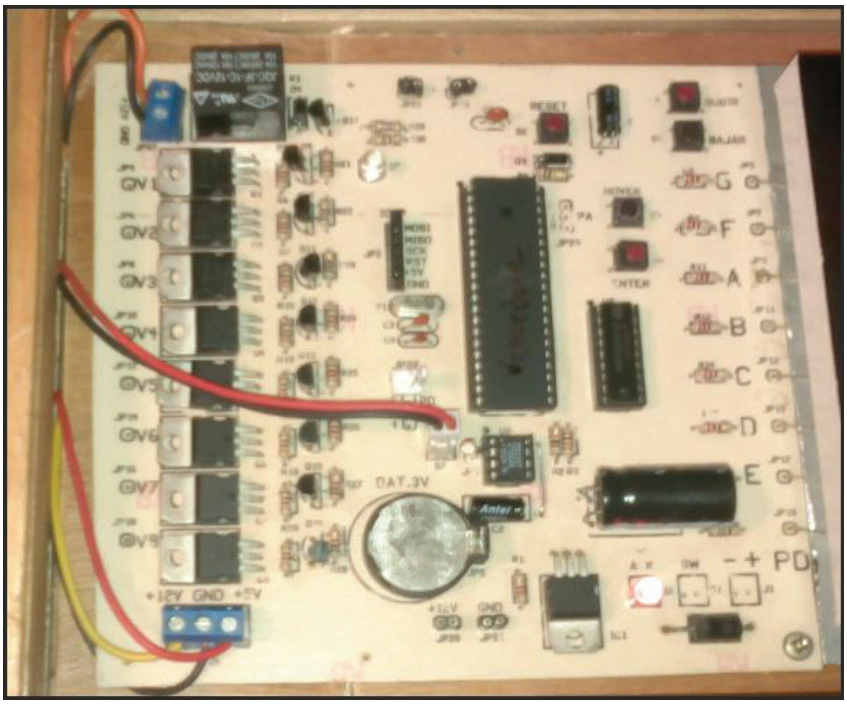


Figura 3. 3 Unidad central del sistema

3.2.1.2 Unidad de Visualización de los display's.

La de visualización de los display's está formada por los siguientes elementos como se muestra en la figura 3.4

- Ocho display de ánodo común

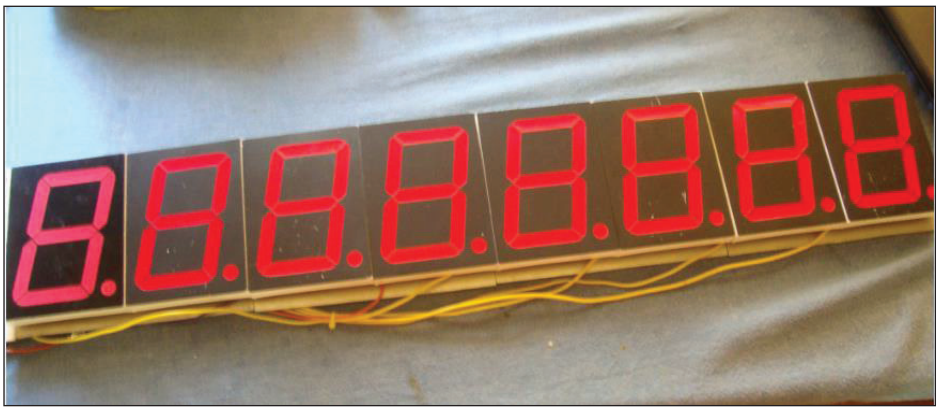


Figura 3. 4 Displays del sistema

3.2.1.3 Unidad de la fuente de alimentación

La unidad de la fuente de alimentación está formada por los siguientes elementos como se muestra en la figura 3.5

Una fuente conmutada de 12V, 3A

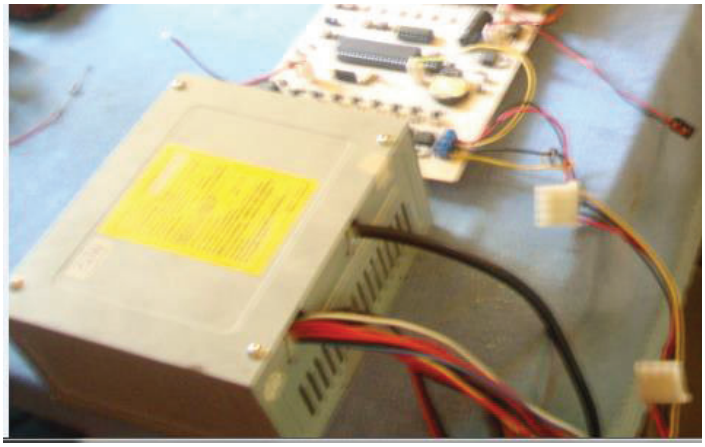


Figura 3. 5 Fuente del sistema

3.2.2 INSTALACIÓN DE SIRENA

Se encuentra ubicada en el patio principal del establecimiento, se conecta la fuente de la sirena para que se active por medio de un relé que es comandado por el sistema de la presente tesis, en la figura 3.6 se muestra la sirena.

Figura 3.6 Ubicación de la Sirena

3.2.3 INSTALACION DEL EQUIPO FINAL

El equipo se instaló en la parte lateral de la oficina de dirección. Está ubicado a la entrada al plantel y frente al patio principal, por lo que pueden observar con facilidad el público en general, como muestra la figura 3.7.

Figura 3. 7 Sitio de instalación del proyecto

3.3 PRUEBAS DE FUNCIONAMIENTO COMPLETO DEL SISTEMA

Al finalizar la implementación e instalación del sistema de seguridad, se realizó una serie de pruebas, para verificar el correcto funcionamiento del presente proyecto.

Para lo cual con la representante de la institución se procedió a realizar la prueba final, en la presento resultados favorables que se detallan a continuación.

- La sirena se activó automáticamente a la hora correcta que está programada
- La sirena se activó el lapso de 8 segundos (se activa automáticamente por medio del programa que se encuentra en el microcontrolador ATmega164)

- Al pulsar dos veces el pulsador la sirena se activó 10 segundos (ante la presencia de un desastre natural y la pronta evacuación de los estudiantes al patio)
- Al pulsar una solo vez la sirena se activó 5 segundos (para avisar que se va a realizar una reunión en la sala de profesores)
- El letrero presenta el mensaje de bienvenida (ESCUELA VENCEDORES BIENVENIDOS) de manera rotativa y al finalizar muestra la hora y el día actual como se muestra en la figura 3.7

Figura 3.8 Presentación de hora en el letrero

3.4 MANUAL DE USUARIO

EL sistema consta de las siguientes etapas como se muestra en la figura 3.9

- Relé de activación de la sirena
- Unidad de control
- Reset
- Pulsadores para igualar la hora y fecha

- Pulsador manual
- Batería de reloj
- Reloj
- Display

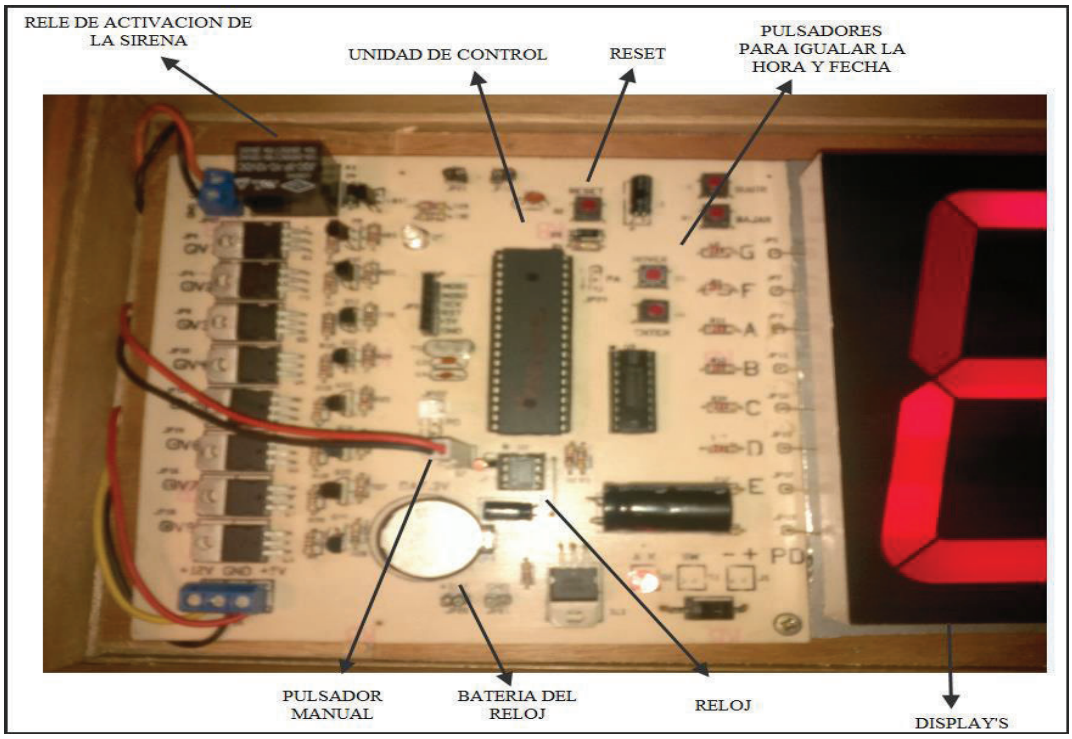


Figura 3.9 Etapas del sistema

3.4.1 RELE DE ACTIVACION DE LA SIRENA

Este relé me permite activar la sirena ya sea por medio del pulsador o automáticamente en las horas programas

3.4.2 UNIDAD DE CONTROL

Es el cerebro del sistema es el que toma todas las decisiones

3.4.3 RESET

Este pulsador me permite en el momento de ser presionado regresar el microcontrolador a su estado inicial, se lo pulsa en el caso de que el microcontrolador no realice los procesos adecuadamente.

3.4.4 PULSADORES PARA IGUALAR FECHA Y HORA

Está formado por cuatro pulsadores los que me permiten igualar la hora y fecha, en el caso que el reloj se haya desigualado, como se indica en la figura 3.10

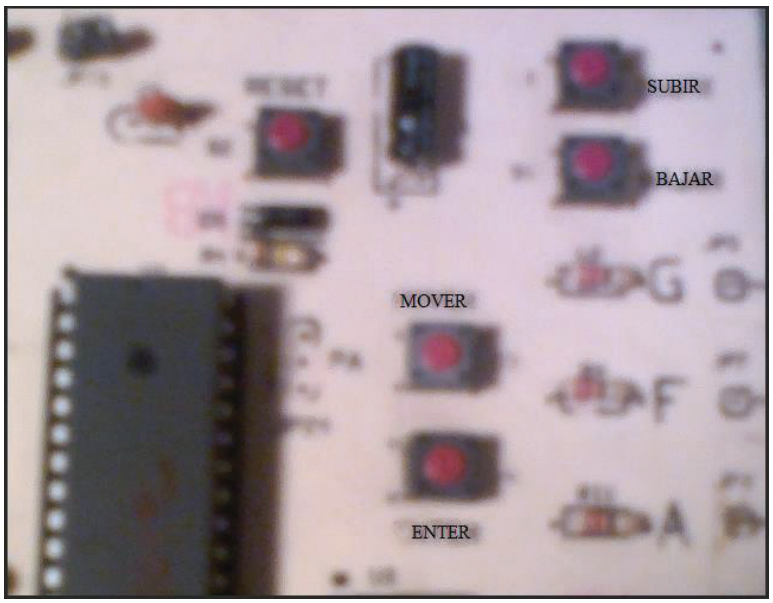


Figura 3.10 Distribución de los pulsadores

Cada pulsador cumple una función específica.

- **ENTER**

Me permite ingresar al menú principal si se lo presiona una vez y si se lo presiona por segunda vez me guarda los datos cambiados y sale de la sub-rutina de igualación, como se muestra en la figura 3.11

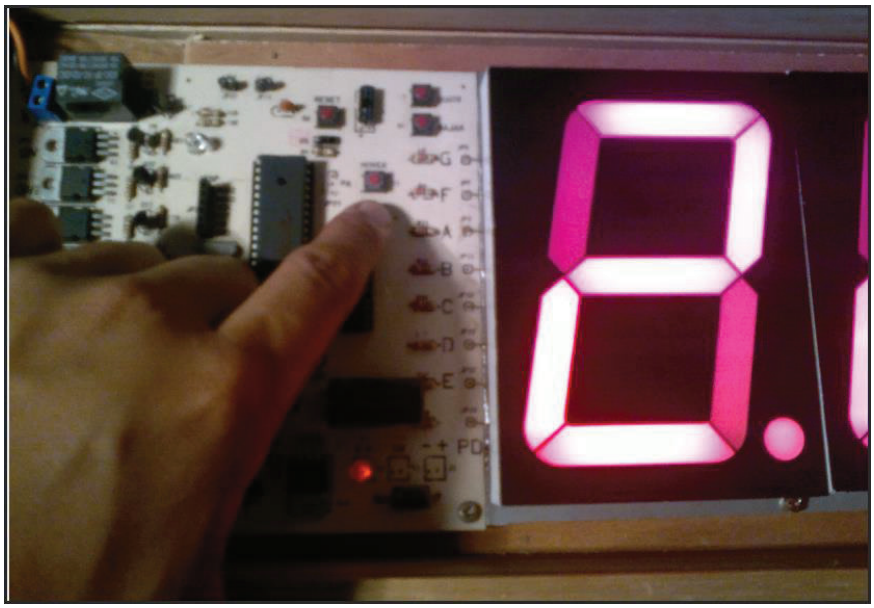


Figura 3.11 Ingreso al menú para igualar la hora y fecha

- **MOVER**

Me permite desplazar para poder igualar la hora, los minutos, los segundos, el día el mes el año e indicar que día es. Para igualar la hora los puntos decimales de los dos primeros display se encienden para indicar que se puede cambiar de dato, como indica la figura 3.12

Figura 3.12 Cambio de hora

Si se presiona nuevamente el botón de mover se desplazan los puntos decimales a los cuarto y quinto display para indicarme que se puede igualar los minutos, como en la figura 3.13

Figura 3.13 Cambio de los minutos

Si se presiona nuevamente el botón de mover se desplazan los puntos decimales a los séptimo y octavo display para indicarme que se puede igualar los segundos, como en la figura 3.14

Figura 3.14 Cambio de los segundos

Si se presiona nuevamente el botón de mover se desplazan los puntos decimales al primero y segundo display para indicarme que se puede igualar el día de la fecha como en la figura 3.15

Figura 3.15 Cambio de día

Si se presiona nuevamente el botón de mover se desplazan los puntos decimales a los al cuarto y quinto display para indicarme que se puede igualar el mes de la fecha, como en la figura 3.16

Figura 3.16 Cambio de mes

Si se presiona nuevamente el botón de mover se desplazan los puntos decimales a los al séptimo y octavo display para indicarme que se puede igualar el año de la fecha como en la figura 3.17

Figura 3.17 Cambio del año

Si se presiona nuevamente el botón de mover nos indica los días en el display, pudiendo igual el día correspondiente a fecha actual, como indica la figura 3.18

Figura 3.18 Cambio de día

Para poder incrementar o disminuir los datos para poder igualar se lo realiza con los pulsadores de subir y bajar.

- **SUBIR**

Me permite incrementar en una unidad el dato que se desea cambiar ya sea hora, minuto, segundo, día, mes y año

- **BAJAR**

Me permite disminuir en una unidad el dato que se desea cambiar ya sea hora, minuto, segundo, día, mes y año

3.4.5 PULSADOR MANUAL

Por medio de este pulsador podemos activar la sirena y va hacer que la sirena se active de acuerdo al número de veces que se lo aplaste.

- Al pulsar dos veces el pulsador la sirena se activó 10 segundos (ante la presencia de un desastre natural y la pronta evacuación de los estudiantes al patio)
- Al pulsar una solo vez la sirena se activó 5 segundos (para avisar que se va a realizar una reunión en la sala de profesores)
- Cada vez que el pulsador se aplaste se va a ir sumando cinco segundos de activación de la sirena

3.4.6 BATERIA DEL RELOJ

Es la batería que permite que el reloj no se desiguale en el momento que se quita la energía del circuito de control. Si el reloj se desiguala constantemente o en el momento de igualar no se guarda correctamente el dato, es porque la batería necesita ser remplazada

3.4.7 RELOJ

Es el C.I. DS1307 que es un reloj en tiempo real, si se presenta en el display el número 88 en los segundos significa que el C.I. se encuentra dañado por lo que es necesario su remplazo

3.4.8 DISPLAY'S

Está formado por ocho display de ánodo común. En los que se va a visualizar un mensaje de bienvenida a la escuela Vencedores, como también la hora y día actual.

El mensaje que se encuentra presentando solo se lo puede cambiar en el programa principal del microcontrolador. Es decir que si se quiere cambiar el mensaje se tiene

que hacer en el software BASCOM-AVR, después compilar y grabar el archivo .HEX en la memoria de programa del microcontrolador ATmega164

En las figuras 3.19 y 3.20 se observa la fecha y el mensaje del letrero.

Figura 3.19 Presentación de la hora del letrero

Figura 3.20 Presentación del mensaje del letrero

CAPÍTULO 4

CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

El objetivo de este proyecto, es la construcción e implementación de un sistema de control de hora y mensajería para la escuela Vencedores, donde se puede concluir lo siguiente:

- El sistema de seguridad cumple correctamente con las tareas señaladas descritas anteriormente como: visualización del mensaje de bienvenida y visualización de la hora exacta así como también la activación de la sirena cuando detecta el cambio de hora.

- El fácil manejo hace posible que el usuario, sea capaz de operar los distintos dispositivos con toda confianza.

- La elaboración del proyecto hace posible que podamos darnos cuenta que se puede realizar en el país proyectos serviciales, buscando siempre las necesidades propias y de los demás, tratando siempre de cuidar la economía.

- El paquete computacional BASCOM AVR, permite programar toda clase de microcontroladores AVR, de manera rápida y estructurada de alto nivel (lenguaje Basic), el cual se ha utilizado para realizar el proyecto construido.

4.2 RECOMENDACIONES

- No instalar la unidad de control en lugares sometidos a polvo excesivo, vibraciones mecánicas ni golpes, sino más bien siempre se la debe colocar en lugares seguros y sobre todo privados, donde los alumnos y personas extrañas no tengan acceso a manipular el sistema evitando así daños que puedan afectar el correcto funcionamiento del mismo.

- Para mayor seguridad se recomienda mantener dos bocinas de alarma (sirena), en caso de que la una falle entrará en funcionamiento la segunda.

- En caso de tener problemas como por ejemplo datos erróneos en la visualización de los display se recomienda hacer un RESET al Microcontrolador.

BIBLIOGRAFIA

[1] "Reloj en tiempo real"
http://es.wikipedia.org/wiki/Reloj_en_tiempo_real

[2] "Características del microcontrolador"
http://www.monografias.com/trabajos12/micrcont/micrcont.shtml

[3] "Características del microcontrolador"
http://usuarios.multimania.es/sfriswolker/pic/uno.htm

[4] "Microprocesador ATMEGA164"
http://atmega164p_spa_ol.pdf

[5] "Funcionamiento del display"
http://es.wikipedia.org/wiki/Visualizador

[6] "Funcionamiento del relé"
http://es.wikipedia.org/wiki/Rel%C3%A9

[7] "La sirena"
http://es.wikipedia.org/wiki/Sirena_(instrumento_ac%C3%BAstico)

[8] "Fuente de poder y funcionamiento"
http://www.informaticamoderna.com/Fuente_AT.htm

[9] "BASCOM AVR"
VALENCIA Ramiro, Aplicaciones Electrónicas con Microcontroladores

ANEXOS

ANEXO A

PROGRAMA DEL PROYECTO ESCRITO EN BASCOM

```
$regfile = "m164pdef.dat"
$crystal = 8000000                          'Reloj Interno
$baud = 9600

'*****************************************************************
Ddrb = 255                                  'Pórtico B como salida
Portb = 255                                 'Estado inicial del pórtico B=255
Ddrc = 255                                  'Pórtico C como salida
Portc = 0                                   'Estado inicial del pórtico C=0
Datos Alias Portc
Habilita Alias Portb
'************* PULSADORES *******************************
Ddra.4 = 0
Porta.4 = 1
Ddra.5 = 0
Porta.5 = 1
Ddra.6 = 0
Porta.6 = 1
Ddra.7 = 0
Porta.7 = 1
Tsubir Alias Pina.4
Tbajar Alias Pina.5
Tmover Alias Pina.6
Tenter Alias Pina.7
'************* SALIDAS ***************************************
Ddrd.2 = 1                                  'Pórtico D.2 como salida
Portd.2 = 0                                 'Estado inicial del pin D.2=0
Ddrd.3 = 1                                  'Pórtico D.3 como salida
Portd.3 = 0                                 'Estado inicial del pin D.3=0
Ddrd.4 = 1                                  'Pórtico D.4 como salida
Portd.4 = 0                                 'Estado inicial del pin D.4=0
Ddra.2 = 1                                  'Pórtico A.2 como salida
Porta.2 = 0                                 'Estado inicial del pin A.2=0

Out2 Alias Portd.2
Out3 Alias Portd.3
Out4 Alias Portd.4
Out5 Alias Porta.2
'************* PULSADOR SIRENA ****************************
Ddrd.5 = 0
```

```
Portd.5 = 1
Bsirena Alias Pind.5
'************* SIRENA ************************************************************
Ddra.3 = 1
Porta.3 = 0
Sirena Alias Porta.3

'******************* INTERRUPCION Timer0 ************************************
Config Timer0 = Timer , Prescale = 256
On Timer0 Refrescar
'******************* INTERRUPCION Timer0 ************************************
Config Timer1 = Timer , Prescale = 1
On Timer1 Revisaruart
'***************** DS1307 RTC *********************************************
Config Sda = Portd.7                                'Configura I2C pin Datos PORTD.7
Config Scl = Portd.6                                'Configura I2C pin Reloj PORTD.6

'Direccion del DS1307
Const Ds1307w = &HD0                                 'Constante escritura DS1307
Const Ds1307r = &HD1                                 'Constante lectura DS1307

'Declara variables necesarias para el DS1307
Dim Dias As Byte
Dim Segu As Byte
Dim Minu As Byte
Dim Hora As Byte
Dim Diam As Byte
Dim Mes As Byte
Dim Anio As Byte
Dim Tempo As Byte
Dim Segu1 As Byte
Dim Minu1 As Byte
Dim Hora1 As Byte
Dim Dias1 As Byte
Dim D1 As String * 3
Dim Dd1 As String * 3
Dim A1 As Word
Dim A2 As Word
Dim Control As Byte
Dim Totalcom As Word
Dim Totalcom1 As Word

Dim Mensaje As String * 255
Dim Diasemana As String * 15

Dim Digito(10) As Byte
Dim Ram(10) As Byte
Dim Valor As String * 1

Dim A As Word
```

```
Dim B As Byte
Dim C As Byte
Dim D As Word
Dim F As Byte
Dim H As Byte
Dim I As Byte
Dim V As Byte
Dim W As Word
Dim X As Byte
Dim X1 As Byte
Dim X2 As Byte

Dim Aa As Word
Dim Cc As Byte
Dim Ccc As Byte
Dim Dd As Byte
Dim Velocidad As Word
Dim Rotar As Byte

Dim A3 As Word
Dim Flag1 As Bit
Dim Flag2 As Bit
Dim Flag3 As Bit
Dim Bandera0 As Byte

Dim Pos As Byte                         'variable para posición del cursor
Dim Tempod As Byte
Dim Tempou As Byte

'Velocidad = 50

Dim Aux As Byte
Dim Aux1 As Byte
Const Delay1 = 8                        'Retardo 8 para rebote
Const Tiempounseg = 122                     'Para TIMER0, interrupción de 1 Seg.
Const Tiempoalarma = 8                      'Tiempo que suena la alarma en Segs(Horas
programadas)
Const Tiempoalarma1 = 5                     'Tiempo que suena la alarma en
Segs(Pulsador)

Const Horaalarma01 = 7
Const Minualarma01 = 30
Const Segualarma01 = 0

Const Horaalarma02 = 8
Const Minualarma02 = 15
Const Segualarma02 = 0

Const Horaalarma03 = 9
Const Minualarma03 = 0
```

```
Const Segualarma03 = 0

Const Horaalarma04 = 9
Const Minualarma04 = 45
Const Segualarma04 = 0

Const Horaalarma05 = 10
Const Minualarma05 = 15
Const Segualarma05 = 0

Const Horaalarma06 = 11
Const Minualarma06 = 0
Const Segualarma06 = 0

Const Horaalarma07 = 11
Const Minualarma07 = 45
Const Segualarma07 = 0

Const Horaalarma08 = 12
Const Minualarma08 = 30
Const Segualarma08 = 0

Const Horaalarma09 = 13
Const Minualarma09 = 0
Const Segualarma09 = 0

Const Horaalarma10 = 13
Const Minualarma10 = 40
Const Segualarma10 = 0

Const Horaalarma11 = 14
Const Minualarma11 = 20
Const Segualarma11 = 0

Const Horaalarma12 = 15
Const Minualarma12 = 0
Const Segualarma12 = 0

Const Horaalarma13 = 15
Const Minualarma13 = 40
Const Segualarma13 = 0

Const Horaalarma14 = 16
Const Minualarma14 = 0
Const Segualarma14 = 0

Const Horaalarma15 = 16
Const Minualarma15 = 40
Const Segualarma15 = 0
```

```
Const Horaalarma16 = 17
Const Minualarma16 = 20
Const Segualarma16 = 0

Const Horaalarma17 = 18
Const Minualarma17 = 0
Const Segualarma17 = 0

'Dias = 7                                   'día de la semana(D=1,L=2,M=3,M=4,J=5,V=6,S=7)
'Diam = 18
'Mes = 7
'Anio = 9
'Gosub Setdate
'Segu = 0
'Minu = 39
'Hora = 15
'Gosub Settime
'Do
'Loop

Readeeprom Totalcom1 , 510
Readeeprom Velocidad , 508

If Totalcom1 > 500 Or Totalcom1 = 0 Then
  Totalcom1 = 7
  Writeeeprom Totalcom1 , 510
  Waitms 10
  Readeeprom Totalcom1 , 510
  Valor = "A"
    Writeeeprom Valor , 1
    Waitms 10
  Valor = "L"
    Writeeeprom Valor , 2
    Waitms 10
  Valor = "L"
    Writeeeprom Valor , 3
    Waitms 10
  Valor = "_"
    Writeeeprom Valor , 4
    Waitms 10
  Valor = "u"
    Writeeeprom Valor , 5
    Waitms 10
  Valor = "C"
    Writeeeprom Valor , 6
    Waitms 10
End If
If Velocidad > 100 Or Velocidad = 0 Then
  Velocidad = 30
  Writeeeprom Velocidad , 508
```

```
  Waitms 10
  Readeeprom Velocidad , 508
End If

'Mensaje = "ESCUELA VENCEDORES BIENVENIDOS"

Gosub Cleardig
Aux = 0
Aux1 = 0
Flag2 = 0
Flag3 = 0
Bandera0 = 0
Timer0 = 0
Disable Timer0                              'Habilita Interrupción Timer0
Enable Interrupts
Enable Timer1
Timer1 = 63000
Start Timer1
Do
   If Flag3 = 1 Then
     Disable Interrupts
     Disable Timer1
     A1 = 1
     Flag3 = 0
     Control = Ccc
     Select Case Control
       Case 240
         Input D1 Noecho
         Print D1
         Totalcom = Val(d1)
         Totalcom1 = Totalcom + 1
         Writeeeprom Totalcom1 , 510
         Readeeprom Totalcom1 , 510
         Do
           Input D1 Noecho
           Writeeeprom D1 , A1
           Readeeprom D1 , A1
           Print D1
           A1 = A1 + 1
         Loop Until A1 > Totalcom
           Input D1 Noecho
           Print D1
           Velocidad = Val(d1)
           Writeeeprom Velocidad , 508
           Readeeprom Velocidad , 508
       Case 170
         Input D1 Noecho
         Print D1
         Segu = Val(d1)
         Shift Segu , Left , 4
```

```
Input D1 Noecho
Print D1
Tempo = Val(d1)
Segu = Segu + Tempo
Input D1 Noecho
Print D1
Minu = Val(d1)
Shift Minu , Left , 4
Input D1 Noecho
Print D1
Tempo = Val(d1)
Minu = Minu + Tempo
Input D1 Noecho
Print D1
Hora = Val(d1)
Shift Hora , Left , 4
Input D1 Noecho
Print D1
Tempo = Val(d1)
Hora = Hora + Tempo
Input D1 Noecho
Print D1
Dias = Val(d1)
Shift Dias , Left , 4
Input D1 Noecho
Print D1
Tempo = Val(d1)
Dias = Dias + Tempo
Input D1 Noecho
Print D1
Diam = Val(d1)
Shift Diam , Left , 4
Input D1 Noecho
Print D1
Tempo = Val(d1)
Diam = Diam + Tempo
Input D1 Noecho
Print D1
Mes = Val(d1)
Shift Mes , Left , 4
Input D1 Noecho
Print D1
Tempo = Val(d1)
Mes = Mes + Tempo
Input D1 Noecho
Print D1
Anio = Val(d1)
Shift Anio , Left , 4
Input D1 Noecho
Print D1
```

```
            Tempo = Val(d1)
            Anio = Anio + Tempo
            Gosub Setdate
            Gosub Settime
        End Select
        A2 = 1
        Gosub Cleardig
        Timer1 = 63000
        Enable Interrupts
        Enable Timer1
        Start Timer1
    End If

    Gosub Accionpulsadores
    Readeeprom B , A2
    Gosub Buscar
    Gosub Llenardig
    A2 = A2 + 1
    If A2 >= Totalcom1 Then
        A2 = 1
        B = &H20
        Gosub Buscar
        Gosub Llenardig
        Gosub Getdatetime2
        For V = 1 To Len(diasemana)
            Valor = Mid(diasemana , V , 1)
            B = Valor
            Gosub Buscar
            Digito(9) = Ram(1)
            Gosub Llenardig
        Next V
        Gosub Getdatetime
        Gosub Llenardig1
        For Aa = 0 To 1100                          '1100 = 10 seg
            Gosub Getdatetime1
            Gosub Display1
            Gosub Accionpulsadores
        Next Aa
        Gosub Cleardig
    End If
Loop

'************************* SUBRUTINAS **********************************
Accionpulsadores:
    Select Case Bandera0
        Case 1
            Disable Interrupts
        Case 2
            Disable Interrupts
            Gosub Rebote3
```

```
      While Tenter = 0
        Gosub Display3
      Wend
      Gosub Rebote3
      Gosub Teclas
      Gosub Cleardig
      A2 = 0
      Enable Interrupts
    Case 3
      Disable Timer0                    'Habilita Interrupción Timer0
      Gosub Rebote3
      While Bsirena = 0
        Gosub Display3
      Wend
      Gosub Rebote3
      Sirena = 1
      Aux = 0
      Aux1 = 0
      Flag2 = 1
      Timer0 = 0
      Enable Timer0                     'Habilita Interrupción Timer0
    Case 4
      Gosub Rebote3
      While Tmover = 0
        Gosub Display3
      Wend
      Gosub Rebote3
      Toggle Out2
  End Select
  Bandera0 = 0
Return
'*************************************************************************
Revisarpulsadores:
  If Flag3 = 1 Then
    X = 255
    Aa = 65535
    A2 = 1
    A = 255
    D = 255
    Gosub Cleardig
    Bandera0 = 1
  Elseif Tenter = 0 Then
    X = 255
    Aa = 65535
    A2 = 1
    A = 255
    D = 255
    Gosub Cleardig
    Bandera0 = 2
  Elseif Bsirena = 0 Then
```

```
      Sirena = 1
      Aux = 0
      Aux1 = 0
      Flag2 = 1
      Timer0 = 0
      Enable Timer0                          'Habilita Interrupción Timer0
  Elseif Tmover = 0 Then
      Gosub Display3
      Gosub Display3
      Gosub Display3
      Gosub Display3
      Toggle Out2
      Gosub Display3
      Gosub Display3
      Gosub Display3
      Gosub Display3
  End If
Return
'*****************************************************************************
Display1:
  Rotar = 1
  Habilita = 0
  For X = 1 To 8
    Datos = Digito(x)
    Habilita = Rotar
    Waitms 1
    Habilita = 0
    Datos = 0
    Waitus 1
    Shift Rotar , Left , 1
    Gosub Revisarpulsadores
    Gosub Getdatetime2
    Gosub Horasalarma
  Next X
Return
'*****************************************************************************
Display2:
  Rotar = 1
  Habilita = 0
  For X1 = 1 To 8
    Datos = Digito(x1)
    Habilita = Rotar
    Waitms 1
    Habilita = 0
    Datos = 0
    Waitus 1
    Shift Rotar , Left , 1
  Next X1
Return
'*****************************************************************************
```

```
Display3:
  Rotar = 1
  Habilita = 0
  For X2 = 1 To 8
    Datos = Digito(x2)
    Habilita = Rotar
    Waitms 1
    Habilita = 0
    Datos = 0
    Waitus 1
    Shift Rotar , Left , 1
    Gosub Getdatetime2
    Gosub Horasalarma
  Next X2
Return
'*************************************************************************
Llenardig:
  Digito(9) = Ram(1)
  For D = 1 To Velocidad
    Gosub Display1
    Gosub Accionpulsadores
    If Flag3 = 0 Then
      Gosub Getdatetime2
      Gosub Horasalarma
    End If
  Next D
  Gosub Moverl
Return
'*************************************************************************
Llenardig1:
  A = 1
  While A < 7                                   '9
    Digito(9) = Ram(a)
    Gosub Moverl
    A = A + 1
    For D = 1 To Velocidad
      Gosub Display1
      Gosub Accionpulsadores
      If Flag3 = 0 Then
        Gosub Getdatetime2
        Gosub Horasalarma
      End If
    Next D
  Wend
  Gosub Getdatetime3
  While A < 9
    Digito(9) = Ram(a)
    Gosub Moverl
    A = A + 1
    For D = 1 To Velocidad
```

```
        Gosub Display1
        Gosub Accionpulsadores
        If Flag3 = 0 Then
          Gosub Getdatetime2
          Gosub Horasalarma
        End If
      Next D
    Wend
Return
'*************************************************************************
Horasalarma:
  If Dias1 = 1 Or Dias1 = 7 Then
  Else
    If Hora1 = Horaalarma01 Then
      If Minu1 = Minualarma01 Then
        If Segu1 = Segualarma01 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
          Sirena = 1
          Aux = 0
          Aux1 = 0
        End If
      End If
    End If
  '-------------------------------------------------------------------
    If Hora1 = Horaalarma02 Then
      If Minu1 = Minualarma02 Then
        If Segu1 = Segualarma02 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
          Sirena = 1
          Aux = 0
          Aux1 = 0
        End If
      End If
    End If
  '-------------------------------------------------------------------

    If Hora1 = Horaalarma03 Then
      If Minu1 = Minualarma03 Then
        If Segu1 = Segualarma03 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
          Sirena = 1
          Aux = 0
          Aux1 = 0
        End If
```

```
    End If
  End If
'--------------------------------------------------------------------
  If Hora1 = Horaalarma04 Then
    If Minu1 = Minualarma04 Then
      If Segu1 = Segualarma04 Then
        Timer0 = 0
        Enable Timer0                    'Habilita Interrupción Timer0
        Enable Interrupts
        Sirena = 1
        Aux = 0
        Aux1 = 0
      End If
    End If
  End If
'--------------------------------------------------------------------
  If Hora1 = Horaalarma05 Then
    If Minu1 = Minualarma05 Then
      If Segu1 = Segualarma05 Then
        Timer0 = 0
        Enable Timer0                    'Habilita Interrupción Timer0
        Enable Interrupts
        Sirena = 1
        Aux = 0
        Aux1 = 0
      End If
    End If
  End If
'--------------------------------------------------------------------
  If Hora1 = Horaalarma06 Then
    If Minu1 = Minualarma06 Then
      If Segu1 = Segualarma06 Then
        Timer0 = 0
        Enable Timer0                    'Habilita Interrupción Timer0
        Enable Interrupts
        Sirena = 1
        Aux = 0
        Aux1 = 0
      End If
    End If
  End If
'--------------------------------------------------------------------
  If Hora1 = Horaalarma07 Then
    If Minu1 = Minualarma07 Then
      If Segu1 = Segualarma07 Then
        Timer0 = 0
        Enable Timer0                    'Habilita Interrupción Timer0
        Enable Interrupts
        Sirena = 1
        Aux = 0
```

```
          Aux1 = 0
        End If
      End If
    End If
'--------------------------------------------------------------------
    If Hora1 = Horaalarma08 Then
      If Minu1 = Minualarma08 Then
        If Segu1 = Segualarma08 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
          Sirena = 1
          Aux = 0
          Aux1 = 0
        End If
      End If
    End If
'--------------------------------------------------------------------
    If Hora1 = Horaalarma09 Then
      If Minu1 = Minualarma09 Then
        If Segu1 = Segualarma09 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
          Sirena = 1
          Aux = 0
          Aux1 = 0
        End If
      End If
    End If
'--------------------------------------------------------------------
    If Hora1 = Horaalarma10 Then
      If Minu1 = Minualarma10 Then
        If Segu1 = Segualarma10 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
          Sirena = 1
          Aux = 0
          Aux1 = 0
        End If
      End If
    End If
'--------------------------------------------------------------------
    If Hora1 = Horaalarma11 Then
      If Minu1 = Minualarma11 Then
        If Segu1 = Segualarma11 Then
          Timer0 = 0
          Enable Timer0                    'Habilita Interrupción Timer0
          Enable Interrupts
```

```
            Sirena = 1
            Aux = 0
            Aux1 = 0
          End If
        End If
      End If
    '-------------------------------------------------------------------------
      If Hora1 = Horaalarma12 Then
        If Minu1 = Minualarma12 Then
          If Segu1 = Segualarma12 Then
            Timer0 = 0
            Enable Timer0                   'Habilita Interrupción Timer0
            Enable Interrupts
            Sirena = 1
            Aux = 0
            Aux1 = 0
          End If
        End If
      End If
    '-------------------------------------------------------------------------
      If Hora1 = Horaalarma13 Then
        If Minu1 = Minualarma13 Then
          If Segu1 = Segualarma13 Then
            Timer0 = 0
            Enable Timer0                   'Habilita Interrupción Timer0
            Enable Interrupts
            Sirena = 1
            Aux = 0
            Aux1 = 0
          End If
        End If
      End If
    '-------------------------------------------------------------------------
    End If
  Return
  '*****************************************************************************
  Cleardig:
    For A3 = 1 To 10
      Digito(a3) = 0
    Next A3
  Return
  '*****************************************************************************
  Setdig:
    For A3 = 1 To 10
      Digito(a3) = 255
    Next A3
  Return
  '*****************************************************************************
  Moverl:
      For A3 = 1 To 10
```

```
        Digito(a3) = Digito(a3 + 1)
      Next A3
Return
'*************************************************************************
Moverd:
      For A3 = 10 To 2 Step -1
        Digito(a3) = Digito(a3 - 1)
      Next A3
Return
'*************************************************************************
Rebote:
   For W = 0 To Delay1
      Gosub Display1
   Next W
Return
'*************************************************************************
Rebote1:
   For W = 0 To Delay1
      Gosub Display2
   Next W
Return
'*************************************************************************
Rebote3:
   For W = 0 To Delay1
      Gosub Display3
   Next W
Return
'*************************************************************************
Getdatetime:
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307w                          'Envía Constante escritura del DS1307
  I2cwbyte 0                                'Dirección Inicial de memoria del DS1307
  I2cstart                                 'Genera inicio de I2C
  I2cwbyte Ds1307r                         'Envía Constante lectura del DS1307
  I2crbyte Segu , Ack                      'Lee los segundos
  I2crbyte Minu , Ack                       'Lee los minutos
  I2crbyte Hora , Ack                       'Lee la hora
  I2crbyte Dias , Ack                        'Lee el día de la
semana(D=1,L=2,M=3,M=4,J=5,V=6,S=7)
  I2crbyte Diam , Ack                         'Lee día del mes
  I2crbyte Mes , Ack                          'Lee el mes del año
  I2crbyte Anio , Nack                         'Lee el año
  I2cstop
  Cc = Hora And &B11110000
  Cc = Cc / 16
  Dd = Lookup(cc , Dta1)
  Ram(1) = Dd
  Cc = Hora And &B00001111
  Dd = Lookup(cc , Dta1)
  Ram(2) = Dd
```

```
  Ram(3) = 128
  Cc = Minu And &B11110000
  Cc = Cc / 16
  Dd = Lookup(cc , Dta1)
  Ram(4) = Dd
  Cc = Minu And &B00001111
  Dd = Lookup(cc , Dta1)
  Ram(5) = Dd
  Ram(6) = 128
  Cc = Segu And &B11110000
  Cc = Cc / 16
  Dd = Lookup(cc , Dta1)
  Ram(7) = Dd
  Cc = Segu And &B00001111
  Dd = Lookup(cc , Dta1)
  Ram(8) = Dd
Return
'*************************************************************************
Getdatetime1:
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307w                        'Envía Constante escritura del DS1307
  I2cwbyte 0                              'Dirección Inicial de memoria del DS1307
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307r                        'Envía Constante lectura del DS1307
  I2crbyte Segu , Ack                     'Lee los segundos
  I2crbyte Minu , Ack                     'Lee los minutos
  I2crbyte Hora , Ack                     'Lee la hora
  I2crbyte Dias , Ack                     'Lee el día de la
semana(D=1,L=2,M=3,M=4,J=5,V=6,S=7)
  I2crbyte Diam , Ack                       'Lee día del mes
  I2crbyte Mes , Ack                       'Lee el mes del año
  I2crbyte Anio , Nack                      'Lee el año
  I2cstop
  Cc = Hora And &B11110000
  Cc = Cc / 16
  Dd = Lookup(cc , Dta1)
  Digito(1) = Dd
  Cc = Hora And &B00001111
  Dd = Lookup(cc , Dta1)
  Digito(2) = Dd
  Digito(3) = 128
  Cc = Minu And &B11110000
  Cc = Cc / 16
  Dd = Lookup(cc , Dta1)
  Digito(4) = Dd
  Cc = Minu And &B00001111
  Dd = Lookup(cc , Dta1)
  Digito(5) = Dd
  Digito(6) = 128
  Cc = Segu And &B11110000
```

```
  Cc = Cc / 16
  Dd = Lookup(cc , Dta1)
  Digito(7) = Dd
  Cc = Segu And &B00001111
  Dd = Lookup(cc , Dta1)
  Digito(8) = Dd
Return
'************************************************************************
Getdatetime1a:
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307w                        'Envía Constante escritura del DS1307
  I2cwbyte 0                              'Dirección Inicial de memoria del DS1307
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307r                        'Envía Constante lectura del DS1307
  I2crbyte Segu , Ack                     'Lee los segundos
  I2crbyte Minu , Ack                     'Lee los minutos
  I2crbyte Hora , Ack                     'Lee la hora
  I2crbyte Dias , Ack                     'Lee el día de la
semana(D=1,L=2,M=3,M=4,J=5,V=6,S=7)
  I2crbyte Diam , Ack                        'Lee día del mes
  I2crbyte Mes , Ack                         'Lee el mes del año
  I2crbyte Anio , Nack                       'Lee el año
  I2cstop
Return
'************************************************************
Getdatetime2:
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307w                        'Envía Constante escritura del DS1307
  I2cwbyte 0                              'Dirección Inicial de memoria del DS1307
  I2cstart                                'Genera inicio de I2C
  I2cwbyte Ds1307r                        'Envía Constante lectura del DS1307
  I2crbyte Segu , Ack                     'Lee los segundos
  I2crbyte Minu , Ack                     'Lee los minutos
  I2crbyte Hora , Ack                     'Lee la hora
  I2crbyte Dias , Ack                     'Lee el día de la
semana(D=1,L=2,M=3,M=4,J=5,V=6,S=7)
  I2crbyte Diam , Ack                        'Lee día del mes
  I2crbyte Mes , Ack                         'Lee el mes del año
  I2crbyte Anio , Nack                       'Lee el año
  I2cstop
  Hora1 = Makedec(hora)
  Minu1 = Makedec(minu)
  Segu1 = Makedec(segu)
  Dias1 = Makedec(dias)
  Select Case Dias1
   Case 1
     Diasemana = " DOMINGO "
   Case 2
     Diasemana = " LUNES "
   Case 3
```

```
    Diasemana = " MARTES "
  Case 4
    Diasemana = " MIERCOLES "
  Case 5
    Diasemana = " JUEVES "
  Case 6
    Diasemana = " VIERNES "
  Case 7
    Diasemana = " SABADO "
 End Select
Return
'*********************************************************************
Getdatetime3:
 I2cstart                          'Genera inicio de I2C
 I2cwbyte Ds1307w                  'Envía Constante escritura del DS1307
 I2cwbyte 0                        'Dirección Inicial de memoria del DS1307
 I2cstart                          'Genera inicio de I2C
 I2cwbyte Ds1307r                  'Envía Constante lectura del DS1307
 I2crbyte Segu , Nack              'Lee los segundos
 I2cstop
 Cc = Segu And &B11110000
 Cc = Cc / 16
 Dd = Lookup(cc , Dta1)
 Ram(7) = Dd
 Cc = Segu And &B00001111
 Dd = Lookup(cc , Dta1)
 Ram(8) = Dd
Return
'*************************************************************
Setdate:
 I2cstart                          'Genera inicio de I2C
 I2cwbyte Ds1307w                  'Envía Constante escritura del DS1307
 I2cwbyte 3                        'Dirección del día del mes del DS1307
 I2cwbyte Dias                     'Nuevo día de la semana
 I2cwbyte Diam                     'Nuevo día del mes
 I2cwbyte Mes                      'Nuevo mes del año
 I2cwbyte Anio                     'Nuevo Año
 I2cstop
Return
'*************************************************************
Settime:
 I2cstart                          'Genera inicio de I2C
 I2cwbyte Ds1307w                  'Envía Constante escritura del DS1307
 I2cwbyte 0                        'Dirección de segundos del DS1307
 I2cwbyte Segu                     'Nuevo Segundo
 I2cwbyte Minu                     'Nuevo Minuto
 I2cwbyte Hora                     'Nueva Hora
 I2cstop
Return
'*************************************************************
```

```
Buscar:
  If B < &H20 Or B > &H7F Then
    Ram(1) = 0
  Elseif B >= &H20 And B <= &H7F Then
       B = B - &H20
       Ram(1) = Lookup(b , Datoascii)
       B = 0
  End If
Return
'*****************************************************************
Presentar:
   If Flag3 = 0 Then
      Cc = Hora And &B11110000
      Cc = Cc / 16
      Dd = Lookup(cc , Dta1)
      Digito(1) = Dd
      Cc = Hora And &B00001111
      Dd = Lookup(cc , Dta1)
      Digito(2) = Dd
      Digito(3) = 128
      Cc = Minu And &B11110000
      Cc = Cc / 16
      Dd = Lookup(cc , Dta1)
      Digito(4) = Dd
      Cc = Minu And &B00001111
      Dd = Lookup(cc , Dta1)
      Digito(5) = Dd
      Digito(6) = 128
      Cc = Segu And &B11110000
      Cc = Cc / 16
      Dd = Lookup(cc , Dta1)
      Digito(7) = Dd
      Cc = Segu And &B00001111
      Dd = Lookup(cc , Dta1)
      Digito(8) = Dd
   Else
      Cc = Diam And &B11110000
      Cc = Cc / 16
      Dd = Lookup(cc , Dta1)
      Digito(1) = Dd
      Cc = Diam And &B00001111
      Dd = Lookup(cc , Dta1)
      Digito(2) = Dd
      Digito(3) = 4                          '128
      Cc = Mes And &B11110000
      Cc = Cc / 16
      Dd = Lookup(cc , Dta1)
      Digito(4) = Dd
      Cc = Mes And &B00001111
      Dd = Lookup(cc , Dta1)
```

```
      Digito(5) = Dd
      Digito(6) = 4                              '128
      Cc = Anio And &B11110000
      Cc = Cc / 16
      Dd = Lookup(cc , Dta1)
      Digito(7) = Dd
      Cc = Anio And &B00001111
      Dd = Lookup(cc , Dta1)
      Digito(8) = Dd
   End If
Return
'************** Subrutina Leer Teclas ***************************
Teclas:
   Gosub Getdatetime1a
   Gosub Presentar
   Flag1 = 1
   Pos = 0
   While Flag1 = 1
     Select Case Pos
       Case 0
         Flag3 = 0
         Gosub Presentar
         Digito(1) = Digito(1) Or &B00000001
         Digito(2) = Digito(2) Or &B00000001
         Digito(4) = Digito(4) And &B11111110
         Digito(5) = Digito(5) And &B11111110
         Digito(7) = Digito(7) And &B11111110
         Digito(8) = Digito(8) And &B11111110
       Case 1
         Digito(1) = Digito(1) And &B11111110
         Digito(2) = Digito(2) And &B11111110
         Digito(4) = Digito(4) Or &B00000001
         Digito(5) = Digito(5) Or &B00000001
         Digito(7) = Digito(7) And &B11111110
         Digito(8) = Digito(8) And &B11111110
       Case 2
         Digito(1) = Digito(1) And &B11111110
         Digito(2) = Digito(2) And &B11111110
         Digito(4) = Digito(4) And &B11111110
         Digito(5) = Digito(5) And &B11111110
         Digito(7) = Digito(7) Or &B00000001
         Digito(8) = Digito(8) Or &B00000001
       Case 3
         Flag3 = 1
         Gosub Presentar
         Digito(1) = Digito(1) Or &B00000001
         Digito(2) = Digito(2) Or &B00000001
         Digito(4) = Digito(4) And &B11111110
         Digito(5) = Digito(5) And &B11111110
         Digito(7) = Digito(7) And &B11111110
```

```
    Digito(8) = Digito(8) And &B11111110
  Case 4
    Digito(1) = Digito(1) And &B11111110
    Digito(2) = Digito(2) And &B11111110
    Digito(4) = Digito(4) Or &B00000001
    Digito(5) = Digito(5) Or &B00000001
    Digito(7) = Digito(7) And &B11111110
    Digito(8) = Digito(8) And &B11111110
  Case 5
    Digito(1) = Digito(1) And &B11111110
    Digito(2) = Digito(2) And &B11111110
    Digito(4) = Digito(4) And &B11111110
    Digito(5) = Digito(5) And &B11111110
    Digito(7) = Digito(7) Or &B00000001
    Digito(8) = Digito(8) Or &B00000001
  Case 6
    Dias1 = Makedec(dias)
      Select Case Dias1
        Case 1
          Diasemana = "DOMINGO "
        Case 2
          Diasemana = "LUNES   "
        Case 3
          Diasemana = "MARTES  "
        Case 4
          Diasemana = "MIERCOLE"
        Case 5
          Diasemana = "JUEVES  "
        Case 6
          Diasemana = "VIERNES "
        Case 7
          Diasemana = "SABADO  "
      End Select

      For V = 1 To 8
        Valor = Mid(diasemana , V , 1)
        B = Valor
        Gosub Buscar
        Digito(v) = Ram(1)
      Next V
      Digito(1) = Digito(1) Or &B00000001
      Digito(2) = Digito(2) Or &B00000001
      Digito(3) = Digito(3) Or &B00000001
      Digito(4) = Digito(4) Or &B00000001
      Digito(5) = Digito(5) Or &B00000001
      Digito(6) = Digito(6) Or &B00000001
      Digito(7) = Digito(7) Or &B00000001
      Digito(8) = Digito(8) Or &B00000001
End Select
Gosub Display2
```

```
If Tsubir = 0 Then
   Gosub Rebote1
   While Tsubir = 0
     Gosub Display2
   Wend
   Gosub Rebote1
   Select Case Pos
      Case 0
         Tempo = Hora
         Gosub Bcdabin
         Tempo = Tempo + 1
         If Tempo = 24 Then
            Tempo = 0
         End If
         Gosub Binabcd
         Hora = Tempo
         Cc = Hora And &B11110000
         Cc = Cc / 16
         Dd = Lookup(cc , Dta1)
         Digito(1) = Dd Or &B00000001
         Cc = Hora And &B00001111
         Dd = Lookup(cc , Dta1)
         Digito(2) = Dd Or &B00000001
      Case 1
         Tempo = Minu
         Gosub Bcdabin
         Tempo = Tempo + 1
         If Tempo = 60 Then
            Tempo = 0
         End If
         Gosub Binabcd
         Minu = Tempo
         Cc = Minu And &B11110000
         Cc = Cc / 16
         Dd = Lookup(cc , Dta1)
         Digito(4) = Dd Or &B00000001
         Cc = Minu And &B00001111
         Dd = Lookup(cc , Dta1)
         Digito(5) = Dd Or &B00000001
      Case 2
         Tempo = Segu
         Gosub Bcdabin
         Tempo = Tempo + 1
         If Tempo = 60 Then
            Tempo = 0
         End If
         Gosub Binabcd
         Segu = Tempo
         Cc = Segu And &B11110000
```

```
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(7) = Dd Or &B00000001
    Cc = Segu And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(8) = Dd Or &B00000001
Case 3
    Tempo = Diam
    Gosub Bcdabin
    Tempo = Tempo + 1
    If Tempo = 32 Then
       Tempo = 1
    End If
    Gosub Binabcd
    Diam = Tempo
    Cc = Diam And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(1) = Dd Or &B00000001
    Cc = Diam And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(2) = Dd Or &B00000001
Case 4
    Tempo = Mes
    Gosub Bcdabin
    Tempo = Tempo + 1
    If Tempo = 13 Then
       Tempo = 1
    End If
    Gosub Binabcd
    Mes = Tempo
    Cc = Mes And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(4) = Dd Or &B00000001
    Cc = Mes And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(5) = Dd Or &B00000001
Case 5
    Tempo = Anio
    Gosub Bcdabin
    Tempo = Tempo + 1
    If Tempo = 100 Then
       Tempo = 0
    End If
    Gosub Binabcd
    Anio = Tempo
    Cc = Anio And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
```

```
            Digito(7) = Dd Or &B00000001
            Cc = Anio And &B00001111
            Dd = Lookup(cc , Dta1)
            Digito(8) = Dd Or &B00000001
         Case 6
            Tempo = Dias
            Gosub Bcdabin
            Tempo = Tempo + 1
            If Tempo = 8 Then
               Tempo = 1
            End If
            Select Case Dias
               Case 1
                  Diasemana = "DOMINGO "
               Case 2
                  Diasemana = "LUNES   "
               Case 3
                  Diasemana = "MARTES  "
               Case 4
                  Diasemana = "MIERCOLE"
               Case 5
                  Diasemana = "JUEVES  "
               Case 6
                  Diasemana = "VIERNES "
               Case 7
                  Diasemana = "SABADO  "
            End Select
            Gosub Binabcd
            Dias = Tempo
            For V = 1 To 8
               Valor = Mid(diasemana , V , 1)
               B = Valor
               Gosub Buscar
               Digito(v) = Ram(1)
            Next V
      End Select
   End If
'-------------------------------------------------
   If Tbajar = 0 Then
      Gosub Rebote1
      While Tbajar = 0
        Gosub Display2
      Wend
      Gosub Rebote1
      Select Case Pos
         Case 0
            Tempo = Hora
            Gosub Bcdabin
            Tempo = Tempo - 1
            If Tempo = 255 Then
```

```
      Tempo = 23
    End If
    Gosub Binabcd
    Hora = Tempo
    Cc = Hora And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(1) = Dd Or &B00000001
    Cc = Hora And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(2) = Dd Or &B00000001
Case 1
    Tempo = Minu
    Gosub Bcdabin
    Tempo = Tempo - 1
    If Tempo = 255 Then
      Tempo = 59
    End If
    Gosub Binabcd
    Minu = Tempo
    Cc = Minu And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(4) = Dd Or &B00000001
    Cc = Minu And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(5) = Dd Or &B00000001
Case 2
    Tempo = Segu
    Gosub Bcdabin
    Tempo = Tempo - 1
    If Tempo = 255 Then
      Tempo = 59
    End If
    Gosub Binabcd
    Segu = Tempo
    Cc = Segu And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(7) = Dd Or &B00000001
    Cc = Segu And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(8) = Dd Or &B00000001
Case 3
    Tempo = Diam
    Gosub Bcdabin
    Tempo = Tempo - 1
    If Tempo = 0 Then
      Tempo = 31
    End If
```

```
    Gosub Binabcd
    Diam = Tempo
    Cc = Diam And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(1) = Dd Or &B00000001
    Cc = Diam And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(2) = Dd Or &B00000001
Case 4
    Tempo = Mes
    Gosub Bcdabin
    Tempo = Tempo - 1
    If Tempo = 0 Then
       Tempo = 12
    End If
    Gosub Binabcd
    Mes = Tempo
    Cc = Mes And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(4) = Dd Or &B00000001
    Cc = Mes And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(5) = Dd Or &B00000001
Case 5
    Tempo = Anio
    Gosub Bcdabin
    Tempo = Tempo - 1
    If Tempo = 255 Then
       Tempo = 99
    End If
    Gosub Binabcd
    Anio = Tempo
    Cc = Anio And &B11110000
    Cc = Cc / 16
    Dd = Lookup(cc , Dta1)
    Digito(7) = Dd Or &B00000001
    Cc = Anio And &B00001111
    Dd = Lookup(cc , Dta1)
    Digito(8) = Dd Or &B00000001
Case 6
    Tempo = Dias
    Gosub Bcdabin
    Tempo = Tempo - 1
    If Tempo = 0 Then
       Tempo = 7
    End If
    Select Case Dias
       Case 1
```

```
                    Diasemana = "DOMINGO "
                  Case 2
                    Diasemana = "LUNES   "
                  Case 3
                    Diasemana = "MARTES  "
                  Case 4
                    Diasemana = "MIERCOLE"
                  Case 5
                    Diasemana = "JUEVES  "
                  Case 6
                    Diasemana = "VIERNES "
                  Case 7
                    Diasemana = "SABADO  "
               End Select
               Gosub Binabcd
               Dias = Tempo
               For V = 1 To 8
                  Valor = Mid(diasemana , V , 1)
                  B = Valor
                  Gosub Buscar
                  Digito(v) = Ram(1)
               Next V
            End Select
         End If
'-------------------------------------------------
         If Tmover = 0 Then
            Gosub Rebote1
            While Tmover = 0
              Gosub Display2
            Wend
            Gosub Rebote1
            Pos = Pos + 1
            If Pos = 7 Then
               Pos = 0
            End If
            Select Case Pos
               Case 0
                  Digito(1) = Digito(1) Or &B00000001
                  Digito(2) = Digito(2) Or &B00000001
                  Digito(4) = Digito(4) And &B11111110
                  Digito(5) = Digito(5) And &B11111110
                  Digito(7) = Digito(7) And &B11111110
                  Digito(8) = Digito(8) And &B11111110
               Case 1
                  Digito(1) = Digito(1) And &B11111110
                  Digito(2) = Digito(2) And &B11111110
                  Digito(4) = Digito(4) Or &B00000001
                  Digito(5) = Digito(5) Or &B00000001
                  Digito(7) = Digito(7) And &B11111110
                  Digito(8) = Digito(8) And &B11111110
```

```
            Case 2
               Digito(1) = Digito(1) And &B11111110
               Digito(2) = Digito(2) And &B11111110
               Digito(4) = Digito(4) And &B11111110
               Digito(5) = Digito(5) And &B11111110
               Digito(7) = Digito(7) Or &B00000001
               Digito(8) = Digito(8) Or &B00000001
            Case 3
               Digito(1) = Digito(1) Or &B00000001
               Digito(2) = Digito(2) Or &B00000001
               Digito(4) = Digito(4) And &B11111110
               Digito(5) = Digito(5) And &B11111110
               Digito(7) = Digito(7) And &B11111110
               Digito(8) = Digito(8) And &B11111110
            Case 4
               Digito(1) = Digito(1) And &B11111110
               Digito(2) = Digito(2) And &B11111110
               Digito(4) = Digito(4) Or &B00000001
               Digito(5) = Digito(5) Or &B00000001
               Digito(7) = Digito(7) And &B11111110
               Digito(8) = Digito(8) And &B11111110
            Case 5
               Digito(1) = Digito(1) And &B11111110
               Digito(2) = Digito(2) And &B11111110
               Digito(4) = Digito(4) And &B11111110
               Digito(5) = Digito(5) And &B11111110
               Digito(7) = Digito(7) Or &B00000001
               Digito(8) = Digito(8) Or &B00000001
            Case 6
               Digito(1) = Digito(1) Or &B00000001
               Digito(2) = Digito(2) Or &B00000001
               Digito(4) = Digito(4) And &B11111110
               Digito(5) = Digito(5) And &B11111110
               Digito(7) = Digito(7) And &B11111110
               Digito(8) = Digito(8) And &B11111110
         End Select
      End If
'-------------------------------------------------
      If Tenter = 0 Then
         Gosub Rebote1
         While Tenter = 0
           Gosub Display2
         Wend
         Gosub Rebote1
         Gosub Settime
         Gosub Setdate
         Flag1 = 0
      End If
'-------------------------------------------------
   Wend
```

```
Return
'************** Subrutina BCD a Binario ***************************
Bcdabin:
   Tempou = Tempo And $0f
   Shift Tempo , Right , 4
   Tempod = Tempo * 10
   Tempo = Tempod + Tempou
Return
'************** Subrutina Binario a BCD ***************************
Binabcd:
   Tempod = Tempo \ 10
   Shift Tempod , Left , 4
   Tempou = Tempo Mod 10
   Tempo = Tempod + Tempou
Return
'*****************************************************************
Revisaruart:
     Ccc = Inkey()
     If Ccc > 0 Then
       Flag3 = 1
       Disable Timer1
       Disable Timer0
       Stop Timer1
       Timer1 = 63000
       Dd1 = Chr(ccc)
       Print Dd1
     End If
     Disable Interrupts
Return
'*****************************************************************
Refrescar:
  Incr Aux
  If Aux >= Tiempounseg Then
     Aux = 0
     Incr Aux1
     If Flag2 = 0 Then
       If Aux1 >= Tiempoalarma Then
          Disable Timer0                    'Habilita Interrupción Timer0
          Disable Interrupts
          Timer0 = 0
          Sirena = 0
          Aux = 0
          Aux1 = 0
          Flag2 = 0
       End If
     Else
       If Aux1 >= Tiempoalarma1 Then
          Disable Timer0                    'Habilita Interrupción Timer0
          Disable Interrupts
          Timer0 = 0
```

```
          Sirena = 0
          Aux = 0
          Aux1 = 0
          Flag2 = 0
        End If
      End If
    End If
Return
'*************************************************************
End

Dta1:
Data 126 , 24 , 182 , 188 , 216 , 236 , 238 , 56 , 254 , 252

Datoascii:
Data 0
Data 0                                  ' !
Data 0 ' "
Data 0                                  ' #
Data 0                                  ' $
Data 0                                  ' %
Data 0                                  ' &
Data 16                                  ''
Data 120                                 ' (
Data 60                                  ' )
Data 0                                  ' *
Data 0                                  ' +
Data 0                                  ' ,
Data 128                                 ' -
Data 0                                  '.
Data 0                                  ' /
Data 126                                 ' 0
Data 24                                  ' 1
Data 182                                 ' 2
Data 188                                 ' 3
Data 216                                 ' 4
Data 236                                 ' 5
Data 238                                 ' 6
Data 56                                  ' 7
Data 254                                 ' 8
Data 252                                 ' 9
Data 0                                  ' :
Data 0                                  ' ;
Data 0                                  ' <
Data 132                                 ' =
Data 0                                  ' >
Data 178                                 ' ?
Data 0                                  ' @
Data 250                                 ' A
Data 206                                 ' B
```

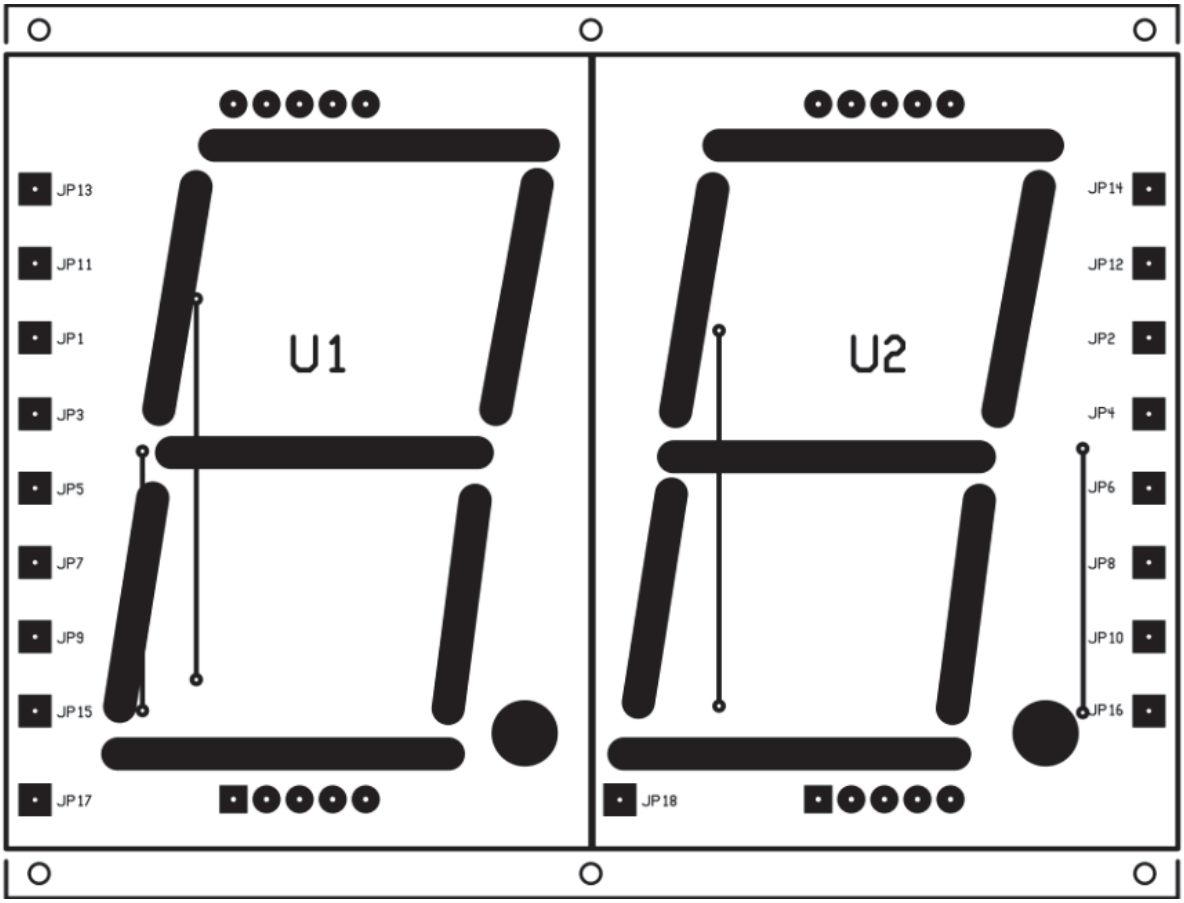
```
Data 102                    'C
Data 158                    'D
Data 230                    'E
Data 226                    'F
Data 252                    'G
Data 218                    'H
Data 66                     'I
Data 30                     'J
Data 0                      'K
Data 70                     'L
Data 122                    'M
Data 202                    'N
Data 126                    'O
Data 242                    'P
Data 248                    'Q
Data 194                    'R
Data 236                    'S
Data 198                    'T
Data 94                     'U
Data 14                     'V
Data 0                      'W
Data 0                      'X
Data 216                    'Y
Data 182                    'Z
Data 120                    '[
Data 0                      '\
Data 60                     ']
Data 0                      '^
Data 4                      '_
Data 0                      '`
Data 190                    'a
Data 206                    'b
Data 134                    'c
Data 158                    'd
Data 246                    'e
Data 226                    'f
Data 252                    'g
Data 218                    'h
Data 66                     'i
Data 30                     'j
Data 0                      'k
Data 24                     'l
Data 122                    'm
Data 202                    'n
Data 142                    'o
Data 242                    'p
Data 248                    'q
Data 194                    'r
Data 236                    's
Data 198                    't
```

```
Data 94                         ' u
Data 14                         ' v
Data 0                          ' w
Data 0                          ' x
Data 216                        ' y
Data 182                        ' z
Data 120                        ' {
Data 24                         ' |
Data 60                         ' }
Data 0                          ' ~
Data 0
```

UBICACIÓN DE LOS ELEMENTOS DEL CIRCUITO EN LA PLACA

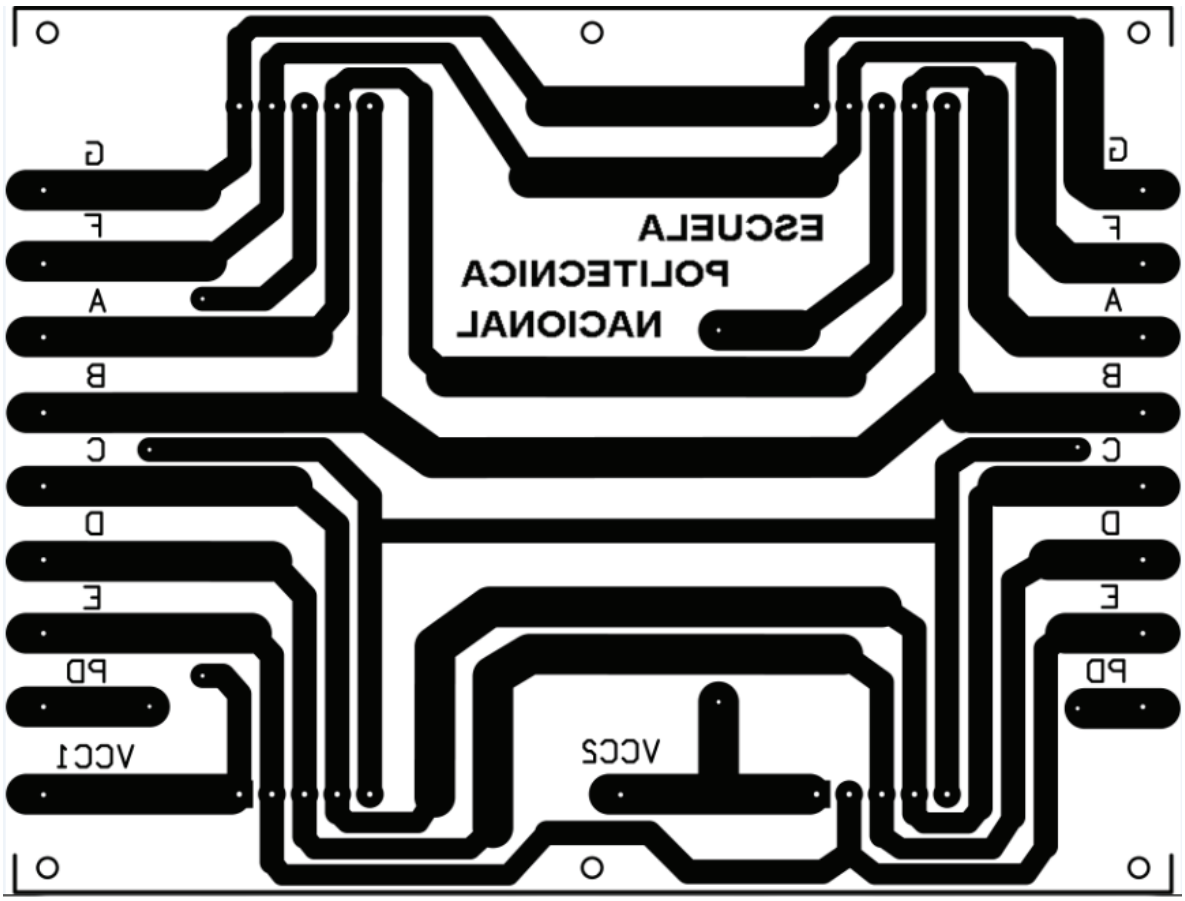
ANEXO B

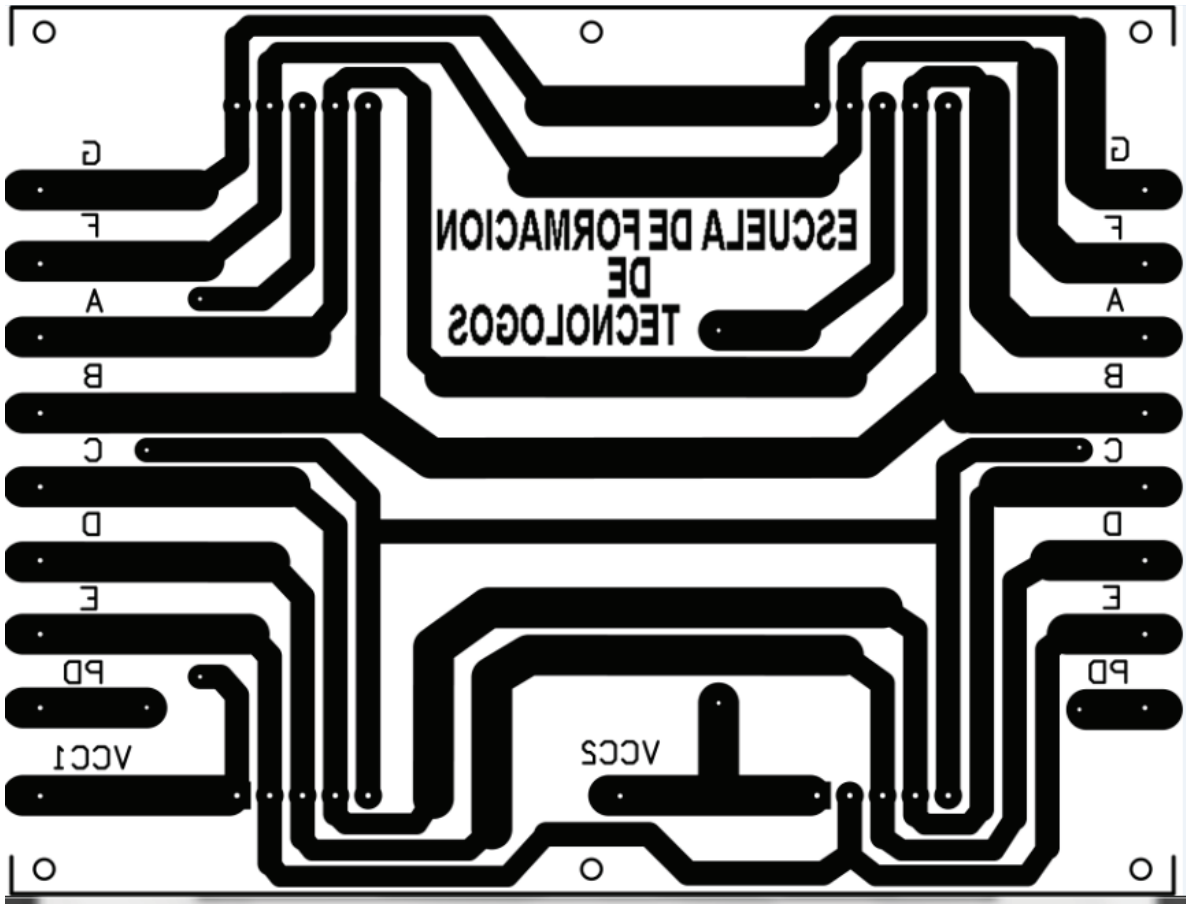
UBICACIÓN DE LOS DISPLAYS EN LA PLACA



ANEXO C

DIAGRAMA DE PISTAS DEL CIRCUITO




ESCUELA DE FORMACION
DE
TECNOLOGOS
G
F
A
B
C
D
E
PD
VCC1
VCC2

ANEXO D

DESCRIPCIÓN DEL MICROCONTROLADOR ATMEGA164

Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
  - 16/32/64K Bytes of In-System Self-Programmable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 512B/1K/2K Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 1/2/4K Bytes Internal SRAM
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel, 10-bit ADC
    Differential mode with selectable gain at 1x, 10x or 200x
  - Byte-oriented Two-wire Serial Interface
  - One/Two Programmable Serial USART (ATmega644, ATmega164/324)
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
- Operating Voltages
  - 1.8 - 5.5V for ATmega164/324/644V
  - 2.7 - 5.5V for ATmega164/324/644
- Speed Grades
  - ATmega164/324/644V: 0 - 4MHz @ 1.8 - 5.5V, 0 - 10MHz @ 2.7 - 5.5V
  - ATmega164/324/644: 0 - 10MHz @ 2.7 - 5.5V, 0 - 20MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 3V, 25°C for ATmega644
  - Active: 240 µA @ 1.8V, 1MHz
  - Power-down Mode: 0.1 µA @ 1.8V

8-bit AVR® Microcontroller with 16/32/64K Bytes In-System Programmable Flash

ATmega164/V
ATmega324/V
ATmega644/V

Advance Information Summary

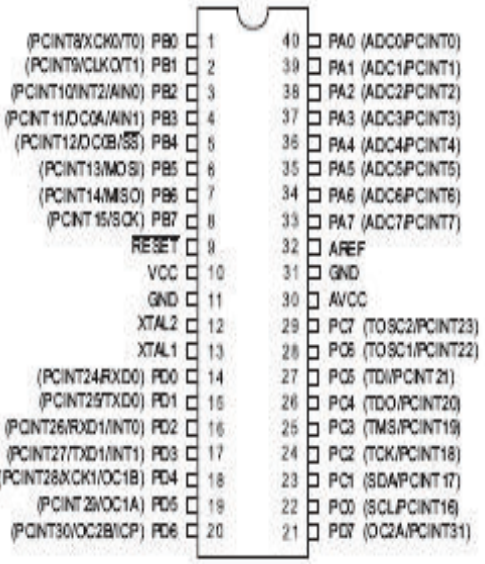
2593AS–AVR–05/05

Note: This is a summary document. A complete document is available on our Web site at www.atmel.com.

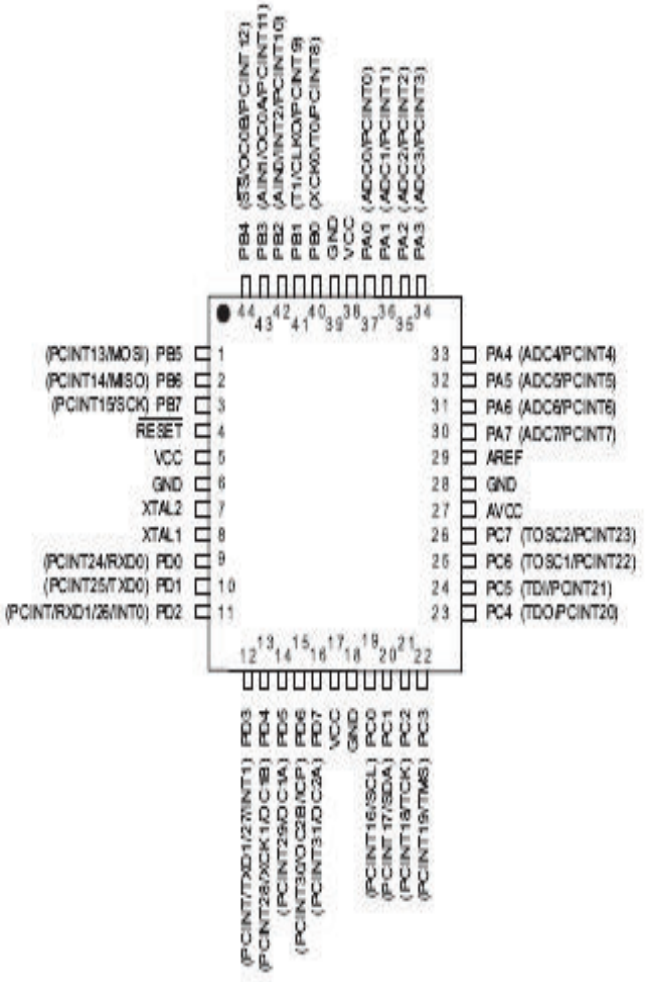
Pin Configurations

Figure 1. Pinout ATmega164/324

PDIP



TQFP/QFN/MLF



Disclaimer

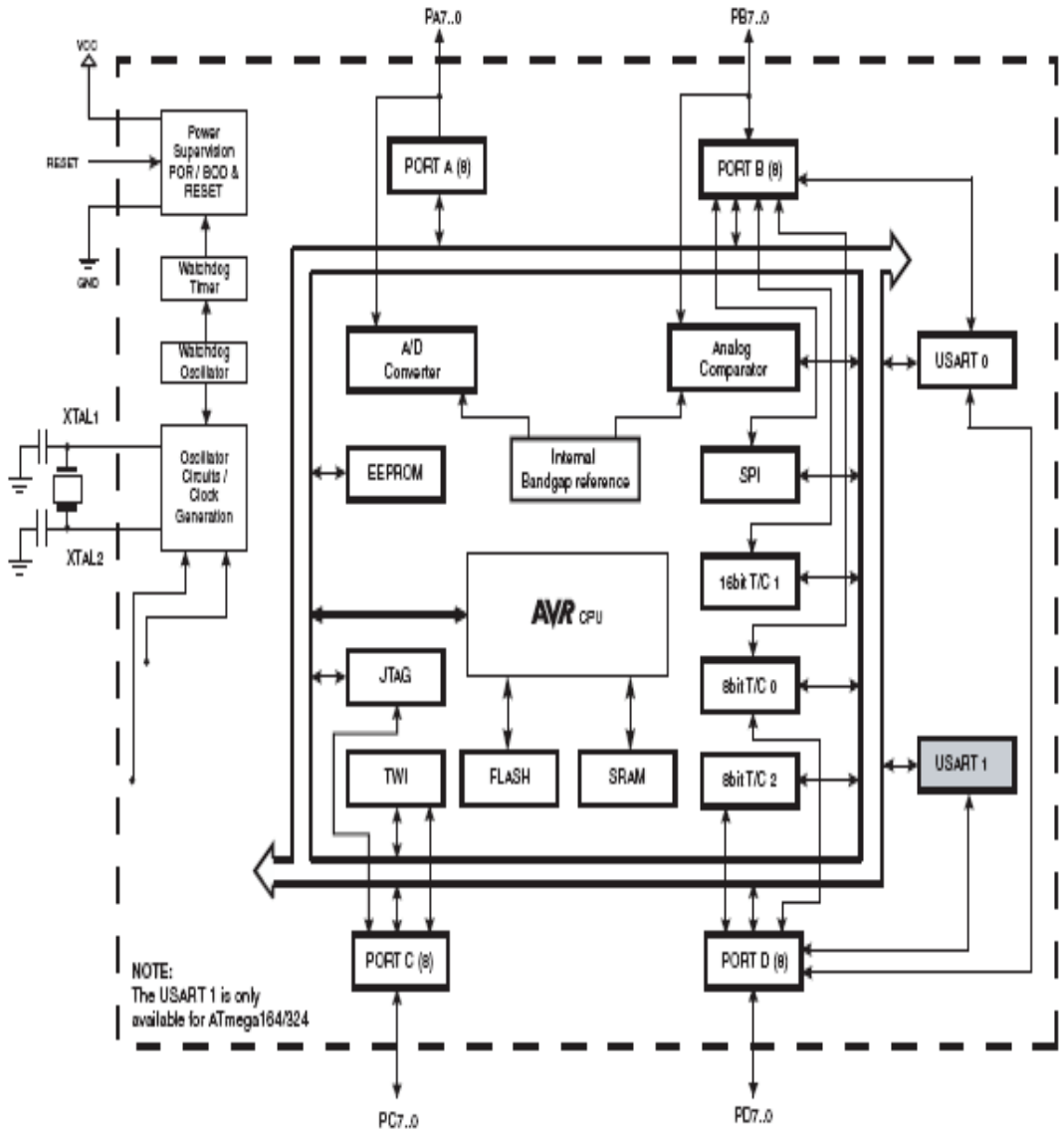
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Overview

The ATmega164/324/644 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega164/324/644 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 3. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega164/324/644 provides the following features: 16/32/64K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512B/1K/2K bytes EEPROM, 1/2/4K bytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible Timer/Counters with compare modes and PWM, 2 USARTs, a byte oriented 2-wire Serial Interface, a 8-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega164/324/644 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega164/324/644 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

Comparison Between ATmega164, ATmega324 and ATmega644

Table 1. Differences between ATmega164 and ATmega644

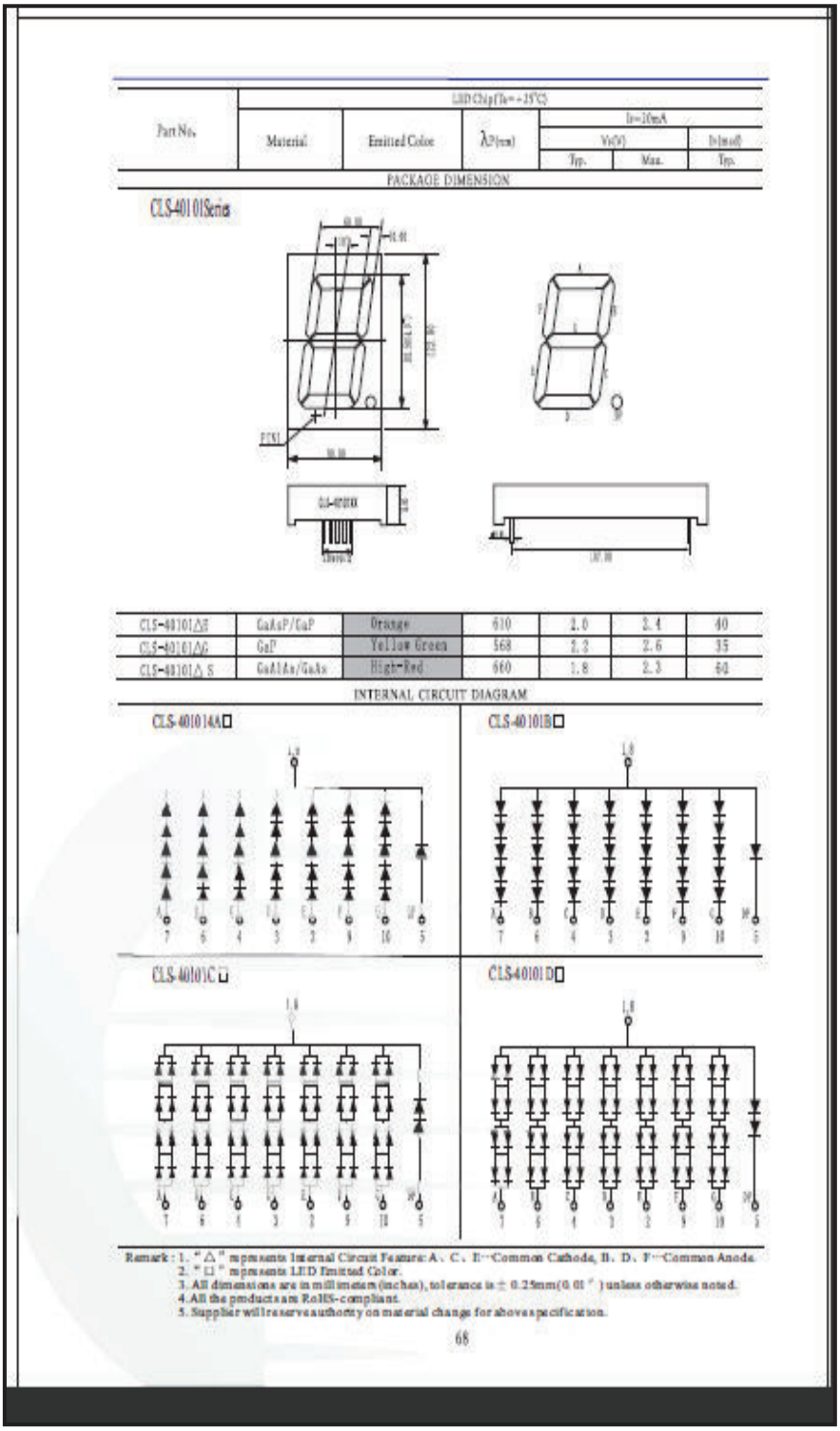
Device	Flash	EEPROM	RAM
ATmega164	16 Kbyte	512 Bytes	1 Kbyte
ATmega324	32 Kbyte	1 Kbyte	2 Kbyte
ATmega644	64 Kbyte	2 Kbyte	4 Kbyte

Pin Descriptions

VCC	Digital supply voltage.
GND	Ground.
Port A (PA7..PA0)	Port A serves as analog inputs to the Analog-to-digital Converter. Port A also serves as an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port A also serves the functions of various special features of the ATmega164/324/644 as listed on page 71.
Port B (PB7..PB0)	Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B also serves the functions of various special features of the ATmega164/324/644 as listed on page 73.
Port C (PC7..PC0)	Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port C also serves the functions of the JTAG interface, along with special features of the ATmega164/324/644 as listed on page 76.
Port D (PD7..PD0)	Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the ATmega164/324/644 as listed on page 78.
$\overline{\text{RESET}}$	Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 20 on page 44. Shorter pulses are not guaranteed to generate a reset.
XTAL1	Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.
XTAL2	Output from the inverting Oscillator amplifier.
AVCC	AVCC is the supply voltage pin for Port F and the Analog-to-digital Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.
AREF	This is the analog reference pin for the Analog-to-digital Converter.

ANEXO E

DIAGRAMA DEL DISPLAY



Part No.	LED Chip(Ta=+25°C)					
	Material	Emitted Color	λP(nm)	If=20mA		
				Vf(V)		Iv(mcd)
				Typ.	Max.	Typ.

PACKAGE DIMENSION

CLS-40101Series

CLS-40101△E	GaAsP/GaP	Orange	610	2.0	2.4	40
CLS-40101△G	GaP	Yellow Green	568	2.2	2.6	35
CLS-40101△S	GaAlAs/GaAs	High-Red	660	1.8	2.2	60

INTERNAL CIRCUIT DIAGRAM

CLS-40101A□

CLS-40101B□

CLS-40101C□

CLS-40101D□

Remark : 1. "△" represents Internal Circuit Feature: A、C、E:--Common Cathode, B、D、F:--Common Anode.
2. "□" represents LED Emitted Color.
3. All dimensions are in millimeters (inches), tolerance is ± 0.25mm(0.01") unless otherwise noted.
4. All the products are RoHS-compliant.
5. Supplier will reserve authority on material change for above specification.

68

ANEXO F

DESCRIPCIÓN DE PINES DEL REGULADOR DE VOLTAJE LM7805

National Semiconductor

February 1995

LM78XX Series Voltage Regulators

General Description

The LM78XX series of three terminal regulators is available with several fixed output voltages making them useful in a wide range of applications. One of these is local on card regulation, eliminating the distribution problems associated with single point regulation. The voltages available allow these regulators to be used in logic systems, instrumentation, HiFi, and other solid state electronic equipment. Although designed primarily as fixed voltage regulators these devices can be used with external components to obtain adjustable voltages and currents.

The LM78XX series is available in an aluminum TO-3 package which will allow over 1.0A load current if adequate heat sinking is provided. Current limiting is included to limit the peak output current to a safe value. Safe area protection for the output transistor is provided to limit internal power dissipation. If internal power dissipation becomes too high for the heat sinking provided, the thermal shutdown circuit takes over preventing the IC from overheating.

Considerable effort was expanded to make the LM78XX series of regulators easy to use and mininize the number of external components. It is not necessary to bypass the output, although this does improve transient response. Input bypassing is needed only if the regulator is located far from the filter capacitor of the power supply.

For output voltage other than 5V, 12V and 15V the LM117 series provides an output voltage range from 1.2V to 57V.

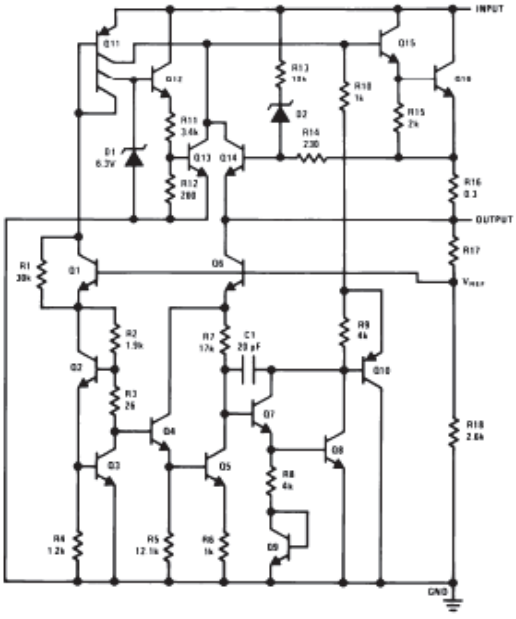
Features

- Output current in excess of 1A
- Internal thermal overload protection
- No external components required
- Output transistor safe area protection
- Internal short circuit current limit
- Available in the aluminum TO-3 package

Voltage Range

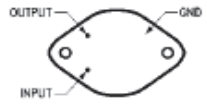
LM7805C	5V
LM7812C	12V
LM7815C	15V

Schematic and Connection Diagrams



TL/H/7746–1

Metal Can Package
TO-3 (K)
Aluminum

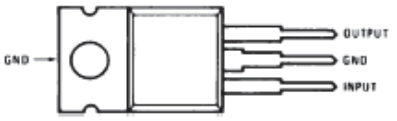


TL/H/7746–2

Bottom View

Order Number LM7805CK,
LM7812CK or LM7815CK
See NS Package Number KC02A

Plastic Package
TO-220 (T)



TL/H/7746–3

Top View

Order Number LM7805CT,
LM7812CT or LM7815CT
See NS Package Number T03B

RRD-B30M115/Printed in U. S. A.

ANEXO G

DESCRIPCIÓN DE PINES DEL TRANSISTOR TIP122

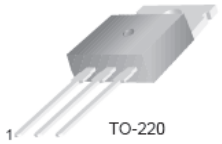
FAIRCHILD
SEMICONDUCTOR®

TIP120/121/122

TIP120/121/122

Medium Power Linear Switching Applications

- Complementary to TIP125/126/127



TO-220

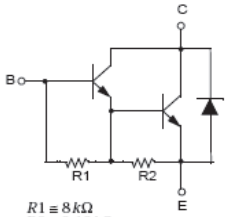
1.Base 2.Collector 3.Emitter

NPN Epitaxial Darlington Transistor

Absolute Maximum Ratings T_C=25°C unless otherwise noted

Symbol	Parameter		Value	Units
V_{CBO}	Collector-Base Voltage	: TIP120	60	V
		: TIP121	80	V
		: TIP122	100	V
V_{CEO}	Collector-Emitter Voltage	: TIP120	60	V
		: TIP121	80	V
		: TIP122	100	V
V_{EBO}	Emitter-Base Voltage		5	V
I_C	Collector Current (DC)		5	A
I_{CP}	Collector Current (Pulse)		8	A
I_B	Base Current (DC)		120	mA
P_C	Collector Dissipation (T_a=25°C)		2	W
	Collector Dissipation (T_C=25°C)		65	W
T_J	Junction Temperature		150	°C
T_{STG}	Storage Temperature		- 65 ~ 150	°C

Equivalent Circuit



$R1 \cong 8k\Omega$
$R2 \cong 0.12k\Omega$

Electrical Characteristics T_C=25°C unless otherwise noted

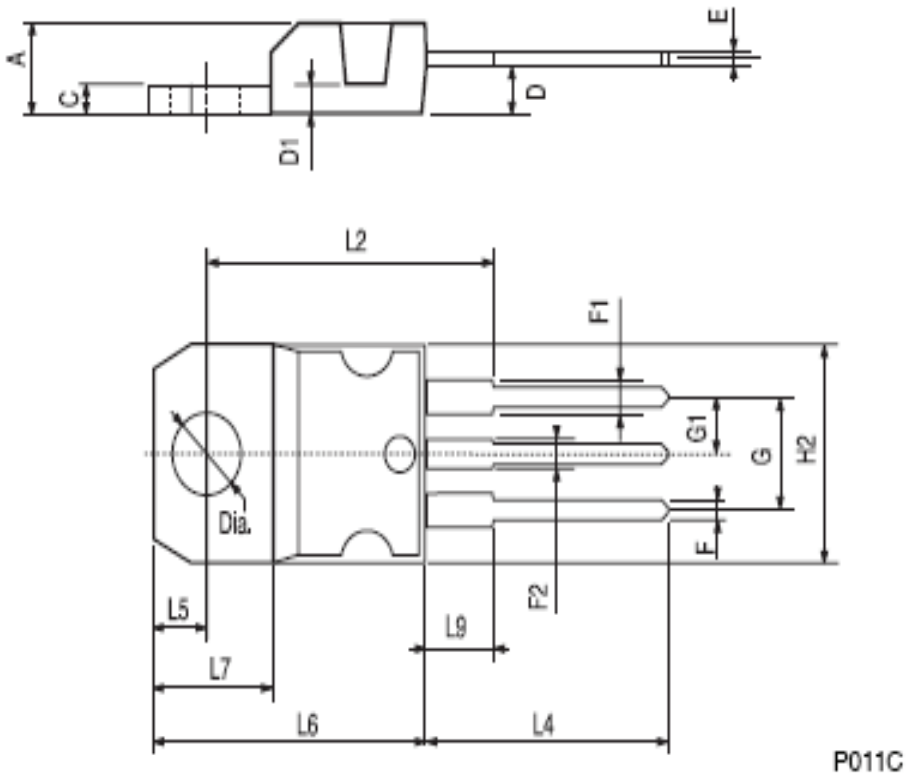
Symbol	Parameter	Test Condition	Min.	Max.	Units
V_{CEO}(sus)	Collector-Emitter Sustaining Voltage				
	: TIP120	I_C = 100mA, I_B = 0	60		V
	: TIP121		80		V
	: TIP122		100		V
I_{CEO}	Collector Cut-off Current				
	: TIP120	V_{CE} = 30V, I_B = 0		0.5	mA
	: TIP121	V_{CE} = 40V, I_B = 0		0.5	mA
	: TIP122	V_{CE} = 50V, I_B = 0		0.5	mA
I_{CBO}	Collector Cut-off Current				
	: TIP120	V_{CB} = 60V, I_E = 0		0.2	mA
	: TIP121	V_{CB} = 80V, I_E = 0		0.2	mA
	: TIP122	V_{CB} = 100V, I_E = 0		0.2	mA
I_{EBO}	Emitter Cut-off Current	V_{BE} = 5V, I_C = 0		2	mA
h_{FE}	* DC Current Gain	V_{CE} = 3V, I_C = 0.5A	1000		
		V_{CE} = 3V, I_C = 3A	1000		
V_{CE}(sat)	* Collector-Emitter Saturation Voltage	I_C = 3A, I_B = 12mA		2.0	V
		I_C = 5A, I_B = 20mA		4.0	V
V_{BE}(on)	* Base-Emitter ON Voltage	V_{CE} = 3V, I_C = 3A		2.5	V
C_{ob}	Output Capacitance	V_{CB} = 10V, I_E = 0, f = 0.1MHz		200	pF

* Pulse Test : PW≤300μs, Duty cycle ≤2%

Rev. A1, June 2001

TO-220 MECHANICAL DATA

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A	4.40		4.60	0.173		0.181
C	1.23		1.32	0.048		0.051
D	2.40		2.72	0.094		0.107
D1		1.27			0.050	
E	0.49		0.70	0.019		0.027
F	0.61		0.88	0.024		0.034
F1	1.14		1.70	0.044		0.067
F2	1.14		1.70	0.044		0.067
G	4.95		5.15	0.194		0.203
G1	2.4		2.7	0.094		0.106
H2	10.0		10.40	0.393		0.409
L2		16.4			0.645	
L4	13.0		14.0	0.511		0.551
L5	2.65		2.95	0.104		0.116
L6	15.25		15.75	0.600		0.620
L7	6.2		6.6	0.244		0.260
L9	3.5		3.93	0.137		0.154
DIA.	3.75		3.85	0.147		0.151



ANEXO H

DESCRIPCION DEL ULN2803

Order this document by ULN2803/D

MOTOROLA

Octal High Voltage, High Current Darlington Transistor Arrays

ULN2803
ULN2804

OCTAL PERIPHERAL DRIVER ARRAYS

SEMICONDUCTOR TECHNICAL DATA

The eight NPN Darlington connected transistors in this family of arrays are ideally suited for interfacing between low logic level digital circuitry (such as TTL, CMOS or PMOS/NMOS) and the higher current/voltage requirements of lamps, relays, printer hammers or other similar loads for a broad range of computer, industrial, and consumer applications. All devices feature open–collector outputs and free wheeling clamp diodes for transient suppression.

The ULN2803 is designed to be compatible with standard TTL families while the ULN2804 is optimized for 6 to 15 volt high level CMOS or PMOS.

A SUFFIX
PLASTIC PACKAGE
CASE 707

MAXIMUM RATINGS (T_A = 25°C and rating apply to any one device in the package, unless otherwise noted.)

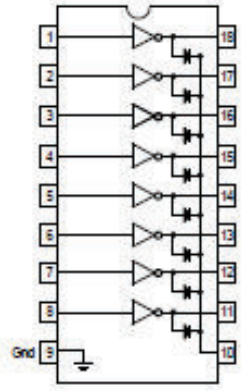
Rating	Symbol	Value	Unit
Output Voltage	V_O	50	V
Input Voltage (Except ULN2801)	V_I	30	V
Collector Current – Continuous	I_C	500	mA
Base Current – Continuous	I_B	25	mA
Operating Ambient Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature	T_J	125	°C

$R_{\theta JA}$ = 55°C/W
Do not exceed maximum current limit per driver.

ORDERING INFORMATION

Device	Characteristics		
	Input Compatibility	V_{CE}(Max)/I_C(Max)	Operating Temperature Range
ULN2803A ULN2804A	TTL, 5.0 V CMOS 6 to 15 V CMOS, PMOS	50 V/500 mA	T_A = 0 to +70°C

PIN CONNECTIONS



Rev 1

ULN2803 ULN2804

ELECTRICAL CHARACTERISTICS (T_A = 25°C, unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
Output Leakage Current (Figure 1)		I_{CEX}				μA
(V_O = 50 V, T_A = +70°C)	All Types		–	–	100	
(V_O = 50 V, T_A = +25°C)	All Types		–	–	50	
(V_O = 50 V, T_A = +70°C, V_I = 6.0 V)	ULN2802		–	–	500	
(V_O = 50 V, T_A = +70°C, V_I = 1.0 V)	ULN2804		–	–	500	
Collector–Emitter Saturation Voltage (Figure 2)		$V_{CE(sat)}$				V
(I_C = 350 mA, I_B = 500 μA)	All Types		–	1.1	1.6	
(I_C = 200 mA, I_B = 350 μA)	All Types		–	0.95	1.3	
(I_C = 100 mA, I_B = 250 μA)	All Types		–	0.85	1.1	
Input Current – On Condition (Figure 4)		$I_{I(on)}$				mA
(V_I = 17 V)	ULN2802		–	0.82	1.25	
(V_I = 3.85 V)	ULN2803		–	0.93	1.35	
(V_I = 5.0 V)	ULN2804		–	0.35	0.5	
(V_I = 12 V)	ULN2804		–	1.0	1.45	
Input Voltage – On Condition (Figure 5)		$V_{I(on)}$				V
(V_{CE} = 2.0 V, I_C = 300 mA)	ULN2802		–	–	13	
(V_{CE} = 2.0 V, I_C = 200 mA)	ULN2803		–	–	2.4	
(V_{CE} = 2.0 V, I_C = 250 mA)	ULN2803		–	–	2.7	
(V_{CE} = 2.0 V, I_C = 300 mA)	ULN2803		–	–	3.0	
(V_{CE} = 2.0 V, I_C = 125 mA)	ULN2804		–	–	5.0	
(V_{CE} = 2.0 V, I_C = 200 mA)	ULN2804		–	–	6.0	
(V_{CE} = 2.0 V, I_C = 275 mA)	ULN2804		–	–	7.0	
(V_{CE} = 2.0 V, I_C = 350 mA)	ULN2804		–	–	8.0	
Input Current – Off Condition (Figure 3) (I_C = 500 μA, T_A = +70°C)	All Types	$I_{I(off)}$	50	100	–	μA
DC Current Gain (Figure 2) (V_{CE} = 2.0 V, I_C = 350 mA)	ULN2801	h_{FE}	1000	–	–	–
Input Capacitance		C_I	–	15	25	pF
Turn–On Delay Time (50% E_I to 50% E_O)		t_{on}	–	0.25	1.0	μs
Turn–Off Delay Time (50% E_I to 50% E_O)		t_{off}	–	0.25	1.0	μs
Clamp Diode Leakage Current (Figure 6) (V_R = 50 V)	T_A = +25°C T_A = +70°C	I_R	–	–	50 100	μA
Clamp Diode Forward Voltage (Figure 7) (I_F = 350 mA)		V_F	–	1.5	2.0	V

ANEXO I

CARACTERISTICAS DEL RTC DS1307

DALLAS SEMICONDUCTOR

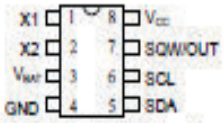
DS1307
64 x 8 Serial Real-Time Clock

www.maxim-ic.com

FEATURES

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- 56-byte, battery-backed, nonvolatile (NV) RAM for data storage
- Two-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500nA in battery backup mode with oscillator running
- Optional industrial temperature range: -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Underwriters Laboratory (UL) recognized

PIN ASSIGNMENT



DS1307 8-Pin DIP (300-mil)

DS1307 8-Pin SOIC (150-mil)

ORDERING INFORMATION

DS1307	8-Pin DIP (300-mil)
DS1307Z	8-Pin SOIC (150-mil)
DS1307N	8-Pin DIP (Industrial)
DS1307ZN	8-Pin SOIC (Industrial)

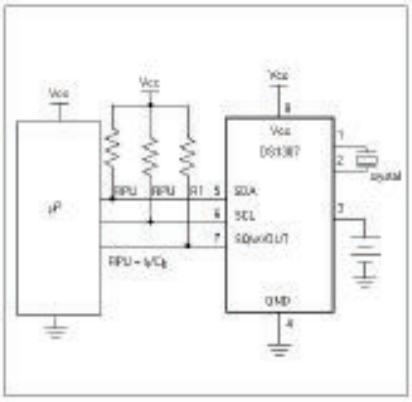
PIN DESCRIPTION

V_{CC}	- Primary Power Supply
X1, X2	- 32.768kHz Crystal Connection
V_{BAT}	- +3V Battery Input
GND	- Ground
SDA	- Serial Data
SCL	- Serial Clock
SQW/OUT	- Square Wave/Output Driver

DESCRIPTION

The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power sense circuit that detects power failures and automatically switches to the battery supply.

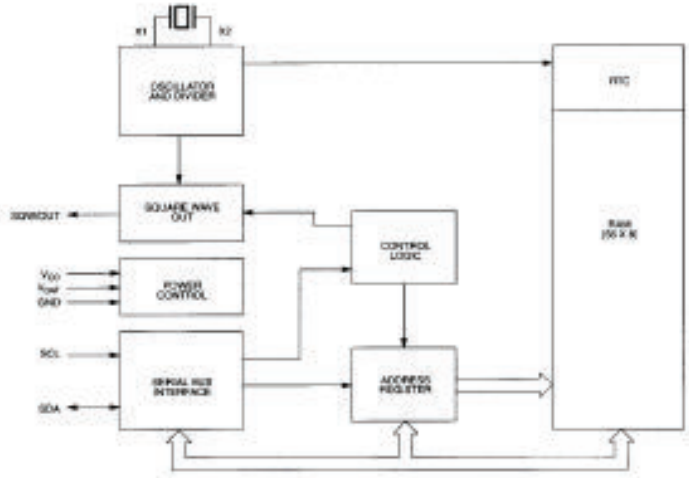
TYPICAL OPERATING CIRCUIT



OPERATION

The DS1307 operates as a slave device on the serial bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When V_{CC} falls below 1.25 x V_{BAT} the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out of tolerance system. When V_{CC} falls below V_{BAT} the device switches into a low-current battery backup mode. Upon power-up, the device switches from battery to V_{CC} when V_{CC} is greater than V_{BAT} + 0.2V and recognizes inputs when V_{CC} is greater than 1.25 x V_{BAT}. The block diagram in Figure 1 shows the main elements of the serial RTC.

DS1307 BLOCK DIAGRAM Figure 1



SIGNAL DESCRIPTIONS

V_{CC}, GND – DC power is provided to the device on these pins. V_{CC} is the +5V input. When 5V is applied within normal limits, the device is fully accessible and data can be written and read. When a 3V battery is connected to the device and V_{CC} is below 1.25 x V_{BAT}, reads and writes are inhibited. However, the timekeeping function continues unaffected by the lower input voltage. As V_{CC} falls below V_{BAT} the RAM and timekeeper are switched over to the external power supply (nominal 3.0V DC) at V_{BAT}.

V_{BAT} – Battery input for any standard 3V lithium cell or other energy source. Battery voltage must be held between 2.0V and 3.5V for proper operation. The nominal write protect trip point voltage at which access to the RTC and user RAM is denied is set by the internal circuitry as 1.25 x V_{BAT} nominal. A lithium battery with 48mAhr or greater will back up the DS1307 for more than 10 years in the absence of power at 25°C. UL recognized to ensure against reverse charging current when used in conjunction with a lithium battery.

See "Conditions of Acceptability" at http://www.maxim-ic.com/TechSupport/QA/ntrl.htm.

SCL (Serial Clock Input) – SCL is used to synchronize data movement on the serial interface.

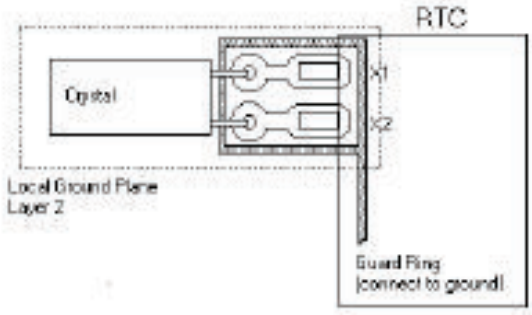
SDA (Serial Data Input/Output) – SDA is the input/output pin for the 2-wire serial interface. The SDA pin is open drain which requires an external pullup resistor.

SQW/OUT (Square Wave/Output Driver) – When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square wave frequencies (1Hz, 4kHz, 8kHz, 32kHz). The SQW/OUT pin is open drain and requires an external pull-up resistor. SQW/OUT will operate with either Vcc or Vbat applied.

X1, X2 – Connections for a standard 32.768kHz quartz crystal. The internal oscillator circuitry is designed for operation with a crystal having a specified load capacitance (CL) of 12.5pF.

For more information on crystal selection and crystal layout considerations, please consult Application Note 58, "Crystal Considerations with Dallas Real-Time Clocks." The DS1307 can also be driven by an external 32.768kHz oscillator. In this configuration, the X1 pin is connected to the external oscillator signal and the X2 pin is floated.

RECOMMENDED LAYOUT FOR CRYSTAL



CLOCK ACCURACY

The accuracy of the clock is dependent upon the accuracy of the crystal and the accuracy of the match between the capacitive load of the oscillator circuit and the capacitive load for which the crystal was trimmed. Additional error will be added by crystal frequency drift caused by temperature shifts. External circuit noise coupled into the oscillator circuit may result in the clock running fast. See Application Note 58, "Crystal Considerations with Dallas Real-Time Clocks" for detailed information.

Please review Application Note 95, "Interfacing the DS1307 with a 8051-Compatible Microcontroller" for additional information.

RTC AND RAM ADDRESS MAP

The address map for the RTC and RAM registers of the DS1307 is shown in Figure 2. The RTC registers are located in address locations 00h to 07h. The RAM registers are located in address locations 08h to 3Fh. During a multi-byte access, when the address pointer reaches 3Fh, the end of RAM space, it wraps around to location 00h, the beginning of the clock space.

DS1307 ADDRESS MAP Figure 2

CLOCK AND CALENDAR

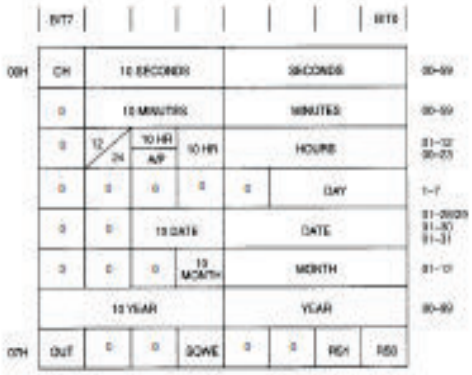
The time and calendar information is obtained by reading the appropriate register bytes. The RTC registers are illustrated in Figure 3. The time and calendar are set or initialized by writing the appropriate register bytes. The contents of the time and calendar registers are in the BCD format. Bit 7 of register 0 is the clock halt (CH) bit. When this bit is set to a 1, the oscillator is disabled. When cleared to a 0, the oscillator is enabled.

Please note that the initial power-on state of all registers is not defined. Therefore, it is important to enable the oscillator (CH bit = 0) during initial configuration.

The DS1307 can be run in either 12-hour or 24-hour mode. Bit 6 of the hours register is defined as the 12- or 24-hour mode select bit. When high, the 12-hour mode is selected. In the 12-hour mode, bit 5 is the AM/PM bit with logic high being PM. In the 24-hour mode, bit 5 is the second 10 hour bit (20-23 hours).

On a 2-wire START, the current time is transferred to a second set of registers. The time information is read from these secondary registers, while the clock may continue to run. This eliminates the need to re-read the registers in case of an update of the main registers during a read.

DS1307 TIMEKEEPER REGISTERS Figure 3



CONTROL REGISTER

The DS1307 control register is used to control the operation of the SQW/OUT pin.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OUT	0	0	SQWE	0	0	RS1	RS0

OUT (Output control): This bit controls the output level of the SQW/OUT pin when the square wave output is disabled. If SQWE = 0, the logic level on the SQW/OUT pin is 1 if OUT = 1 and is 0 if OUT = 0.

SQWE (Square Wave Enable): This bit, when set to a logic 1, will enable the oscillator output. The frequency of the square wave output depends upon the value of the RS0 and RS1 bits. With the square wave output set to 1Hz, the clock registers update on the falling edge of the square wave.

RS (Rate Select): These bits control the frequency of the square wave output when the square wave output has been enabled. Table 1 lists the square wave frequencies that can be selected with the RS bits.

SQUAREWAVE OUTPUT FREQUENCY Table 1

RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1Hz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

DS1307 TIMEKEEPER REGISTERS Figure 3

CONTROL REGISTER

The DS1307 control register is used to control the operation of the SQW/OUT pin.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OUT	0	0	SQWE	0	0	RS1	RS0

OUT (Output control): This bit controls the output level of the SQW/OUT pin when the square wave output is disabled. If SQWE = 0, the logic level on the SQW/OUT pin is 1 if OUT = 1 and is 0 if OUT = 0.

SQWE (Square Wave Enable): This bit, when set to a logic 1, will enable the oscillator output. The frequency of the square wave output depends upon the value of the RS0 and RS1 bits. With the square wave output set to 1Hz, the clock registers update on the falling edge of the square wave.

RS (Rate Select): These bits control the frequency of the square wave output when the square wave output has been enabled. Table 1 lists the square wave frequencies that can be selected with the RS bits.

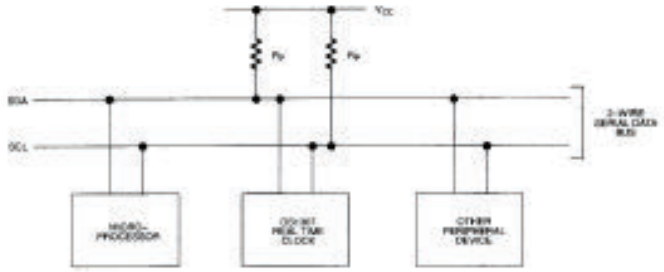
SQUAREWAVE OUTPUT FREQUENCY Table 1

RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1Hz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

2-WIRE SERIAL DATA BUS

The DS1307 supports a bi-directional, 2-wire bus and data transmission protocol. A device that sends data onto the bus is defined as a transmitter and a device receiving data as a receiver. The device that controls the message is called a master. The devices that are controlled by the master are referred to as slaves. The bus must be controlled by a master device that generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions. The DS1307 operates as a slave on the 2-wire bus. A typical bus configuration using this 2-wire protocol is show in Figure 4.

TYPICAL 2-WIRE BUS CONFIGURATION Figure 4



Figures 5, 6, and 7 detail how data is transferred on the 2-wire bus.

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is high will be interpreted as control signals.

Accordingly, the following bus conditions have been defined:

Bus not busy: Both data and clock lines remain HIGH.

Start data transfer: A change in the state of the data line, from HIGH to LOW, while the clock is HIGH, defines a START condition.

Stop data transfer: A change in the state of the data line, from LOW to HIGH, while the clock line is HIGH, defines the STOP condition.

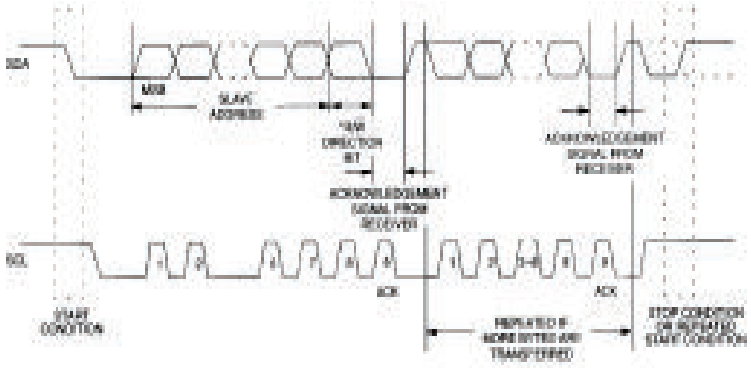
Data valid: The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal. The data on the line must be changed during the LOW period of the clock signal. There is one clock pulse per bit of data.

Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of data bytes transferred between START and STOP conditions is not limited, and is determined by the master device. The information is transferred byte-wise and each receiver acknowledges with a ninth bit. Within the 2-wire bus specifications a regular mode (100kHz clock rate) and a fast mode (400kHz clock rate) are defined. The DS1307 operates in the regular mode (100kHz) only.

Acknowledge: Each receiving device, when addressed, is obliged to generate an acknowledge after the reception of each byte. The master device must generate an extra clock pulse which is associated with this acknowledge bit.

A device that acknowledges must pull down the SDA line during the acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Of course, setup and hold times must be taken into account. A master must signal an end of data to the slave by not generating an acknowledge bit on the last byte that has been clocked out of the slave. In this case, the slave must leave the data line HIGH to enable the master to generate the STOP condition.

DATA TRANSFER ON 2-WIRE SERIAL BUS Figure 5



Depending upon the state of the R/$\overline{W}$ bit, two types of data transfer are possible:

1. **Data transfer from a master transmitter to a slave receiver.** The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte. Data is transferred with the most significant bit (MSB) first.

2. **Data transfer from a slave transmitter to a master receiver.** The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. This is followed by the slave transmitting a number of data bytes. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a "not acknowledge" is returned.

The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the bus will not be released. Data is transferred with the most significant bit (MSB) first.

The DS1307 may operate in the following two modes:

1. **Slave receiver mode (DS1307 write mode):** Serial data and clock are received through SDA and SCL. After each byte is received an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and *direction bit (See Figure 6). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7 bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a write, is a 0. After receiving and decoding the address byte the device outputs an acknowledge on the SDA line. After the DS1307 acknowledges the slave address + write bit, the master transmits a register address to the DS1307 This will set the register pointer on the DS1307. The master will then begin transmitting each byte of data with the DS1307 acknowledging each byte received. The master will generate a stop condition to terminate the data write.

DATA WRITE – SLAVE RECEIVER MODE Figure 6

2. **Slave transmitter mode (DS1307 read mode):** The first byte is received and handled as in the slave receiver mode. However, in this mode, the *direction bit will indicate that the transfer direction is reversed. Serial data is transmitted on SDA by the DS1307 while the serial clock is input on SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer (See Figure 7). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7-bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a read, is a 1. After receiving and decoding the address byte the device inputs an acknowledge on the SDA line. The DS1307 then begins to transmit data starting with the register address pointed to by the register pointer. If the register pointer is not written to before the initiation of a read mode the first address that is read is the last one stored in the register pointer. The DS1307 must receive a "not acknowledge" to end a read.

DATA READ – SLAVE TRANSMITTER MODE Figure 7