ESCUELA POLITÉCNICA DEL EJÉRCITO

FACULTAD DE INGENIERÍA ELECTRÓNICA

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO EN INGENIERÍA ELECTRÓNICA

"DESARROLLO DE UN SISTEMA REPRODUCTOR DE ARCHIVOS MPEG I/II LAYER 3 (MP3) BASADO EN EL MICROCONTROLADOR ATMEL AT89C51SND1C"

GUILLERMO ALFONSO CISNEROS VALLEJO

SANTIAGO PAÚL PADILLA GALARZA

QUITO – ECUADOR

OCTUBRE 2005

CERTIFICACIÓN

Certifico que el presente Proyecto de Grado “DESARROLLO DE UN SISTEMA REPRODUCTOR DE ARCHIVOS MPEG I/II LAYER 3 (MP3) BASADO EN EL MICROCONTROLADOR ATMEL AT89C51SND1C” fue realizado en su totalidad por los señores Guillermo Alfonso Cisneros Vallejo y Santiago Paúl Padilla Galarza, como requerimiento parcial a la obtención del titulo de Ingeniero Electrónico con especialidad en Automatización y Control, bajo nuestra dirección.

Atentamente,

Ing. Byron Navas	Ing. Víctor Proaño
Director	Codirector

AGRADECIMIENTOS

A nuestros padres, por todo el apoyo y la fuerza que nos han brindado a lo largo de nuestra carrera universitaria, por el cariño desinteresado, y por el aliento y paciencia que nos han tenido durante cada instante de nuestras vidas.

A nuestros mejores amigos, que en los momentos difíciles supieron brindarnos su apoyo en la forma en que solo ellos saben hacerlo, por brindarnos sus sabios consejos cuando los necesitábamos.

A nuestro Director, Ing. Byron Navas ya que gracias a su investigación previa fue posible la realización de este proyecto. A nuestro Codirector, Ing. Víctor Proaño, que con su apoyo nos ayudo a cumplir con esta meta.

A la Escuela Politécnica del Ejército, por darnos cabida en su seno durante el periodo de nuestra formación superior.

Al Colegio San Gabriel, por habernos dado bases sólidas que nos permitieron un buen desenvolvimiento a lo largo de nuestra vida universitaria.

Al Ing. Alfonso Torres B., por la ayuda prestada en la fase final del desarrollo del presente trabajo.

GGO & GUASHO

DEDICATORIA

A mi mis padres que a lo largo de mi vida siempre han me han brindado su apoyo incondicional en todos los proyectos e ideas que me he propuesto.

A mi hermano por estar presente siempre que lo necesité.

A todos mis amigos que supieron acompañarme a lo largo de mi vida….

Guillermo

DEDICATORIA

A mis padres por todo su apoyo, paciencia y comprensión, gracias por toda la ayuda que me han dado ya que sin ustedes no hubiese podido culminar esta etapa de la vida.

A mis hermanos que de una u otra manera han estado ahí apoyándome y brindándome su ayuda cuando la necesitaba y a mis amigos gracias por su apoyo.

Santiago

PRÓLOGO

En el presente proyecto titulado se implementó un prototipo que decodifica y reproduce archivos MPEG I/II Capa 3 (MP3), que se encuentran almacenados en un dispositivo de memoria externa.

En el primer capítulo se realizará una introducción a las características principales del microcontrolador, es decir, se explicará la arquitectura interna y descripción de pines.

En el segundo capítulo se presenta información básica sobre los formatos de compresión de audio MPEG I/II capa 3 necesaria para la correcta comprensión de la codificación y compresión de los archivos de audio.

En el tercer capítulo se estudiará la forma en que el microcontrolador graba y reproduce los archivos MP3.

En el cuarto capítulo se presentará un manual de usuario y tutoriales para la correcta utilización de todos los programas necesarios para el manejo del microcontrolador.

En el quinto capítulo se presentará el diseño del prototipo y la programación efectuada en el microcontrolador que permita la reproducción de archivos MP3.

En el sexto capítulo se encuentran las conclusiones y recomendaciones obtenidas en el desarrollo del proyecto.

ÍNDICE GENERAL

CAPITULO I
ARQUITECTURA DEL MICROCONTROLADOR AT89C51SND1C 1

1.1 DESCRIPCIÓN DE PINES 4

1.1.1 Descripción de las señales de los puertos 4

1.1.2 Descripción de las señales de reloj 5

1.1.3 Descripción de las señales del Timer 0 y Timer 1 6

1.1.4 Descripción de las señales de la interfaz de audio 6

1.1.5 Descripción de las señales del controlador USB 7

1.1.6 Descripción de las señales de la interfaz multimedia 7

1.1.7 Descripción de las señales del UART 8

1.1.8 Descripción de las señales del controlador SPI 8

1.1.9 Descripción de las señales del controlador TWI 9

1.1.10 Descripción de las señales de Conversor A/D 9

1.1.11. Descripción de las señales de la interfaz de Teclado (Keypad) 9

1.1.12. Descripción de las señales de Acceso Externo 10

1.1.13. Descripción de las señales del sistema 10

1.1.14. Descripción de las señales de energía 11

1.2 CONTROLADOR DE RELOJ 12

1.2.1 Oscilador 12

1.2.2 Descripción del PLL 13

1.2.2.1 Programación del PLL. 15

1.2.3 Registros de control 16

1.2.3.1 Registros de control del módulo de reloj (CKCON). 16

1.2.3.2 Registros de control del módulo PLL (PLLCON). 17

1.2.3.3 Registros del divisor N del PLL 18

(PLLNDIV).
1.2.3.4 Registros del divisor R de la señal PLL (PLLRDIV). 18
1.3 MEMORIA DE PROGRAMA/CODIGO 18
1.3.1 Arquitectura de la Memoria Flash 19
1.3.2 Ejecución de la memoria de inicialización (Boot Memory) 20
1.3.2.1 Diagrama de flujo del proceso de inicialización. 21
1.3.3 Registros de memoria 22
1.3.3.1 Registro auxiliar 1 (AUXR1) 22
1.4 MEMORIA DE DATOS 22
1.4.1 RAM inferior de 128 Bytes 23
1.4.2 RAM superior de 128 Bytes 24
1.4.3 RAM expandida 24
1.4.4 Espacio externo 25
1.4.4.1 Interfaz de memoria. 25
1.4.5 Registro de control auxiliar (AUXR) 26
1.5 REGISTROS DE FUNCIONES ESPECIALES 27
1.5.1 Registros principales 27
1.5.2 Manejo del sistema 27
1.5.3 Registros especiales de PLL 27
1.5.4 Registros especiales de memoria flash 27
1.5.5 Registros especiales del decodificador MP3 28
1.5.6 Registros especiales de la interfaz de audio 28
1.5.7 Registro de interfaz IDE 28
1.5.8 Registro de interfaz de teclado 28
1.6 MANEJO DE ENERGÍA 28
1.6.1 Registros de control 29
1.6.1.1 Registros de configuración de energía (PCON). 29
1.6.2 Reset 30
1.6.2.1 Condiciones de los pines en los modos 30

de operación especial.
1.6.2.2 Reseteo en frío 30
1.6.2.3 Reseteo en caliente. 31
1.7 INTERFAZ DE TECLADO 31
1.7.1 Descripción 32
1.7.2 Modo de reducción de energía 33
1.7.3 Registros de control 33
1.7.3.1 Registro de control de teclado KBCON 33
1.7.3.2 Registro de estado y control de teclado KBSTA 33

CAPITULO II
CARACTERÍSTICAS DEL FORMATO MPEG I/II CAPA 3 35
2.1 INTRODUCCIÓN 35
2.2 COMPRESIÓN DE AUDIO 36
2.2.1 Digitalización 36
2.2.2 Codificación y Compresión 37
2.2.3 Codificación sub-banda (SBC) 39
2.3 MPEG AUDIO 40
2.3.1 El estándar MPEG audio 40
2.3.2 Introducción al sistema MPEG-1 40

CAPITULO III
FUNCIONAMIENTO DEL MÓDULO MP3 44
3.1 MÓDULO MP3 44
3.1.1 Descripción 44
3.1.2 Funcionamiento 45
3.1.2.1 Datos MP3. 45
3.1.2.2 Reloj MP3. 46
3.1.3 Controles de audio 47
3.1.3.1 Control de volumen. 47
3.1.3.2 Control de ecualización. 47
3.1.3.3 Efectos especiales. 48

3.1.4 Errores de decodificación 48
3.1.5 Información de la trama 48
3.2 INTERFASE DE AUDIO 49
3.2.1 Descripción 49
3.2.2 Generador de reloj 49
3.2.3 Conversor de datos 50
3.2.4 Buffer MP3 51
3.2.5 Reproducción canción MP3 52
3.3 REGISTROS 52

CAPITULO IV
SOFTWARE Y PROGRAMACIÓN 59
4.1 KEIL µVision2 IDE 59
4.1.1 Introducción 59
4.1.2 Instalación 63
4.1.3 Entorno de Keil µVision2, Descripción general 64
4.1.4 Menú de comandos, Barras de herramientas y Accesos Directos 70
4.1.5 Creación de Proyectos 75
4.1.6 Depuración y Simulación 81
4.1.7 Ejemplo de Programa 86
4.2. ATMEL Flip 2.4.2 88
4.2.1 Introducción 88
4.2.2 Instalación del controlador USB 89
4.2.3 Entorno de FLIP 2.4.2 93
4.2.4 Programación del dispositivo 98

CAPITULO V
DESARROLLO DEL REPRODUCTOR MP3 100
5.1 MANEJO DE MEMORIA EXTERNA 100
5.2 MANEJO DE TECLADO 102
5.3 DECODIFICADOR MP3 105
5.4 INTERFASE DE AUDIO 110

5.5 MANEJO DE LCD 113

CAPÍTULO VI
CONCLUSIONES Y RECOMENDACIONES 117
6.1 CONCLUSIONES 117
6.2 RECOMENDACIONES 118

BIBLIOGRAFÍA 120

ANEXO 1
Código del Programa
ANEXO 2
Diagrama Esquemático
ANEXO 3
Listado de Elementos
ANEXO 4
Fotografías
ANEXO 5
Hoja de Datos Técnicos de los Componentes

INDICE DE FIGURAS

INDICE DE TABLAS

GLOSARIO

CAPITULO I

ARQUITECTURA DEL MICROCONTROLADOR AT89C51SND1C

Un microcontrolador es un circuito integrado que posee todas las características de un computador. Este puede ser programado para realizar cualquier tarea a bajo costo. Las extensas áreas de aplicación de estos microcontroladores exigen un gran trabajo de diseño y fabricación.

La compañía Atmel se dedica al desarrollo de microcontroladores, dispositivos lógicos, entre otras cosas. Atmel posee dentro de su amplia gama de productos un integrado capaz de reproducir archivos MP3 con un mínimo de circuitería externa y la posibilidad de interconectarse la mayoría de dispositivos de almacenamiento.

El microcontrolador AT89C51SND1C, se basa en la estructura del microcontrolador 8051, pero ha sido repotenciado al incluir memoria FLASH de 64K bytes y varios módulos dentro de un mismo circuito integrado. El 8051 es un microcontrolador desarrollado por Intel en 1980 para el uso en productos integrados y todavía sigue siendo uno de los microcontroladores más populares. Los núcleos de los CPU del 8051 y del 8031 son usados en más de 100 dispositivos de más de 20 fabricantes independientes como Dallas, Philips y Atmel.

El integrado posee un decodificador de MPEG I/II Capa 3 y el microcontrolador C51 a 20 Mhz con 4K de memoria para inicialización (Boot) y 64K de memoria Flash para el código encargado de manejar los dispositivos de entrada y salida (teclados, LCD, etc.) y la transferencia de los datos del archivo MP3 al decodificador. Para la salida del archivo MP3 decodificado el microcontrolador

tiene a disposición 2 tipos de formatos, un PCM y otro I2S, que servirán para obtener el audio luego de su conversión y amplificación.

Con el microcontrolador AT89C51SND1C es posible la interfase directa sin circuitos adicionales con dispositivos IDE, Compact Flash, SmartMedia y MMC, la conexión es casi directa necesitándose en algunos casos solo unos cuantos componentes pasivos.

Adicionalmente cuenta con un puerto USB y una UART para comunicación o programación In-System (En - Línea). Para la programación In-System se emplea los 4K de memoria Boot para almacenar el código encargado de leer los datos de una de las interfase seriales y escribirlos en la memoria Flash.

El decodificador MPEG maneja archivos codificados desde 32kbps hasta 320kbps codificados a 48, 44.1, 32, 24, 22.05 o 16 kHz. Este microcontrolador puede decodificar en tiempo real archivos codificados en MP3 a datos de audio PCM. Además posee herramientas para controlar volumen, bajos, medio y agudos.

Este microcontrolador se encuentra disponible en 3 tipos de encapsulados: PLCC84 (tarjeta de desarrollo), TQFP80 y BGA81. La diferencia entre estos encapsulados radica fundamentalmente en el tamaño de los mismos. Adicionalmente se debe recalcar que para el desarrollo del proyecto se utilizó el encapsulado PLCC84 y un adaptador, el mismo que permitió trabajar con el microcontrolador en un Proto Board.

Entre las principales aplicaciones en donde se puede utilizar el microcontrolador AT89C51SND1C, están:

- Reproductor MP3
- Teléfonos Móviles
- PDA
- Cámaras

- Reproductor multimedia para autos

A continuación se presentan todos los módulos incorporados en el microcontrolador en forma de diagrama de bloques:

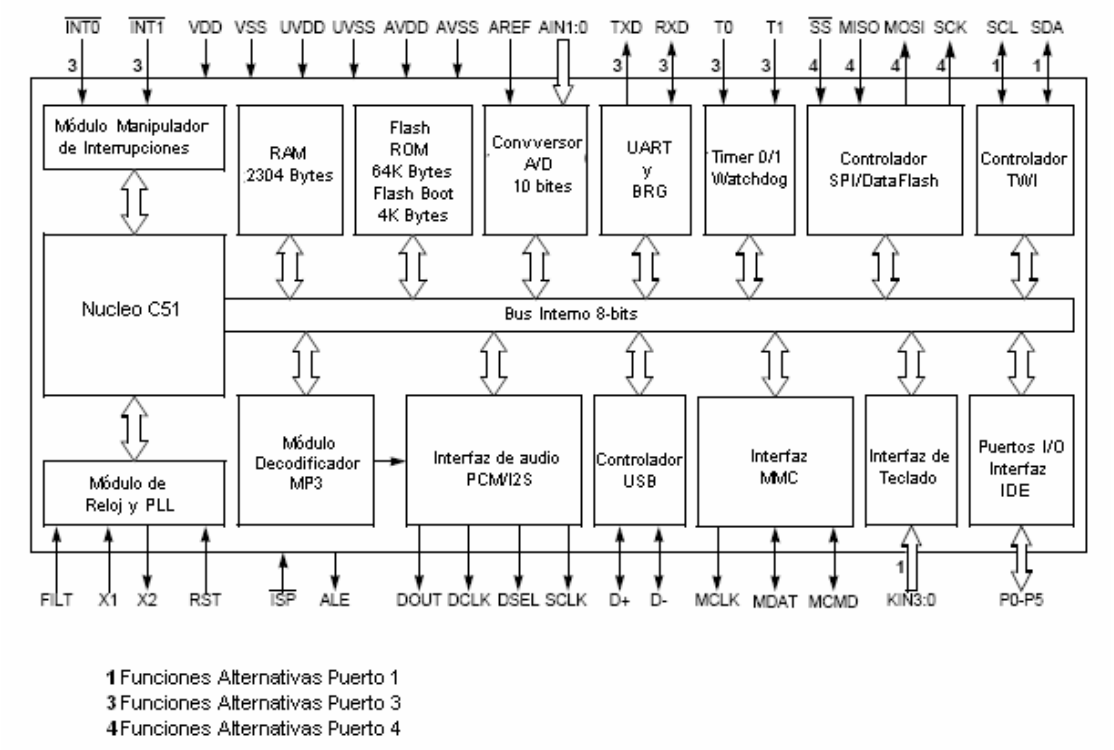


Figura. 1.1. Diagrama de Bloques del Microcontrolador AT89C51SND1C

1.1 DESCRIPCIÓN DE PINES

Empaquetamiento de 84 pines PLCC

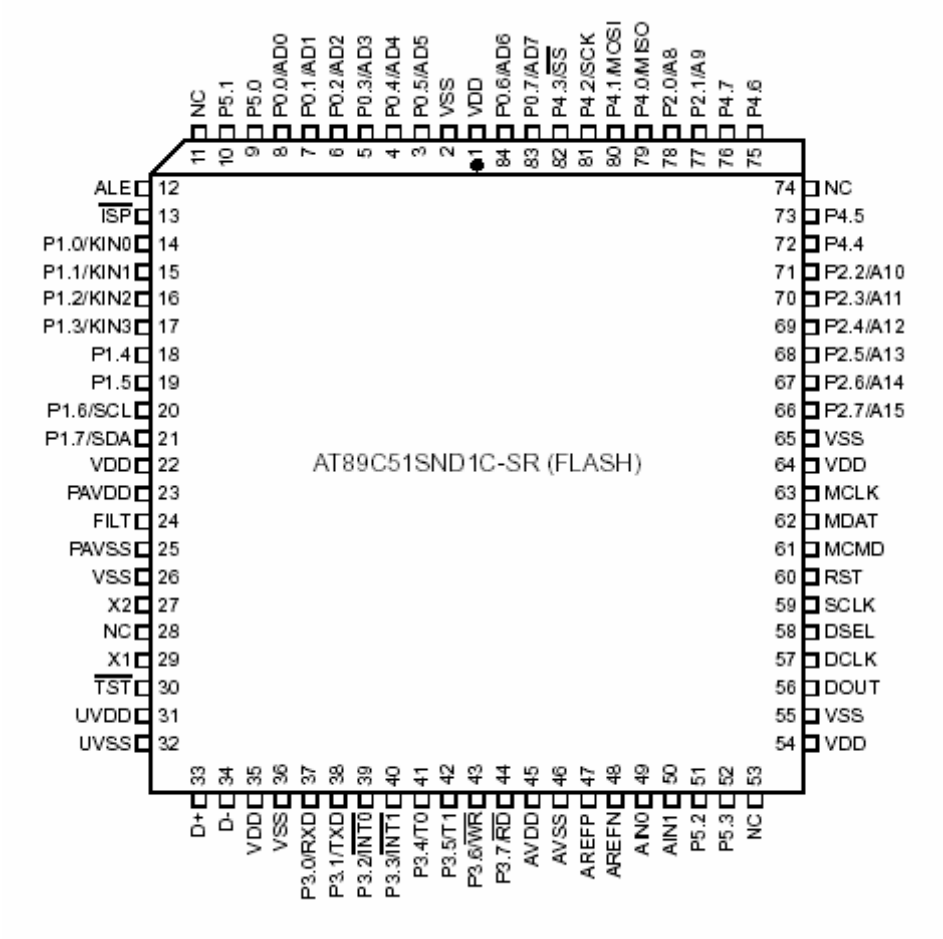


Figura. 1.2. Microcontrolador AT89C51SND1C

Todas las señales del microcontrolador AT89C51SND1C son detalladas por su función a continuación:

1.1.1 Descripción de las señales de los puertos

Puertos	Tipo	Descripción	Función alterna
P0.7:0	I/O	**Puerto 0:** El Puerto 0 es un puerto bidireccional open-drain (drenaje abierto) de 8 bits, los pines del puerto 0 pueden ser utilizados como entradas de alta impedancia. Para evitar cualquier tipo de corriente parásita, las entradas flotantes del puerto 0 deben ser polarizadas a V_{SS} o V_{DD}	AD7:0

P1.7:0	I/O	**Puerto 1:** El Puerto 1 es un puerto bidireccional de 8 bits con pull-ups internos	KIN3:0 SCL SDA
P2.7:0	I/O	**Puerto 2:** El Puerto 2 es un puerto bidireccional de 8 bits con pull-ups internos	A15:8
P3.7:0	I/O	**Puerto 3:** El Puerto 3 es un puerto bidireccional de 8 bits con pull-ups internos	RXD TXD INT0 INT1 T0 T1 WR RD
P4.7:0	I/O	**Puerto 4:** El Puerto 4 es un puerto bidireccional de 8 bits con pull-ups internos	MISO MOSI SCK SS
P5.3:0	I/O	**Puerto 5:** El Puerto 5 es un puerto bidireccional de 4 bits con pull-ups internos	-

Tabla. 1.1. Descripción de puertos

1.1.2 Descripción de las señales de reloj

Nombre de la señal	Tipo	Descripción	Función alternativa
X1	I	**Entrada al oscilador amplificador inversor interno:** Para usar el oscilador interno, un circuito de cristal/resonador es conectado a este pin. Si un oscilador externo es usado, la salida de este es conectada a este pin. X1 es la fuente de reloj para sincronización interna.	-
X2	O	**Salida al oscilador amplificador inversor interno:** Para usar el oscilador interno, un circuito de cristal/resonador es conectado a este pin. Si un oscilador externo es usado, se debe dejar X2 desconectado	-
FILT	I	**Entrada del filtro pasa-bajos del PLL:** Al pin FILT se conecta una red RC del filtro pasa-bajos PLL.	-

Tabla. 1.2. Descripción de señales de reloj

1.1.3 Descripción de las señales del Timer 0 y Timer 1

Nombre de la Señal	Tipo	Descripción	Función alternativa
[1]INT0*	I	**Entrada de la puerta del Timer 0:** INT0* sirve como un control externo para arrancar el timer 0, cuando es seleccionado el bit GATE0 en el registro TCON. **Interrupción externa 0**: La entrada INT0* setea[2] IE0 en el registro TCON. Si el bit IT0 en este registro es seteado, el bit IE0 es activado por un flanco descendente en INT0*. Si el bit IT0 es puesto en cero, el bit IE0 es seteado por un nivel bajo en INT0*	P3.2
INT1*	I	**Entrada de la puerta del Timer 1:** INT1* sirve como un control externo para arrancar el timer 1, cuando es seleccionado el bit GATE1 en el registro TCON. **Interrupción externa 1**: La entrada INT1* setea IE1 en el registro TCON. Si el bit IT1 en este registro es seteado, el bit IE1 es activado por un flanco descendente en INT1*. Si el bit IT1 es puesto en cero, el bit IE1 es seteado por un nivel bajo en INT1*	P3.3
T0	I	**Entrada de reloj externa del Timer 0:** Cuando el Timer 0 opera como un contador, un flanco descendente en el pin T0 incrementa el contador	P3.4
T1	I	**Entrada de reloj externa del Timer 1:** Cuando el Timer 1 opera como un contador, un flanco descendente en el pin T1 incrementa el contador	P3.5

Tabla. 1.3. Descripción de las señales del Timer 0 y Timer 1

1.1.4 Descripción de las señales de la interfase de audio

Nombre de la Señal	Tipo	Descripción	Función alternativa
DCLK	O	**Reloj del bit de datos del DAC**	-
DOUT	O	**Datos de audio del DAC**	-

[1] El * luego del nombre de una señal indica que esta se activa con un bajo nivel
[2] Setea indica que el bit del que se habla se pone en 1

DSEL	O	**Señal de selección del canal del DAC** DSEL es la salida de la tasa de muestreo del reloj	-
SCLK	O	**Reloj de sistema del DAC** SCLK es un sobre-muestreo del reloj sincronizado a los datos digitales de audio (DOUT) y la señal de selección de canal (DSEL)	-

Tabla. 1.4. Descripción de señales de la interfase de audio

1.1.5 Descripción de las señales del controlador USB

Nombre de la Señal	Tipo	Descripción	Función alternativa
D+	I/O	**Datos positivos del puerto USB:** Este pin requiere un pull-up externo hacia V_{DD} utilizando una resistencia de 1.5 KΩ para una correcta operación a máxima velocidad.	-
D-	I/O	**Datos negativos del puerto USB**	-

Tabla. 1.5. Descripción de las señales del controlador USB

1.1.6 Descripción de las señales de la interfase multimedia

Nombre de la Señal	Tipo	Descripción	Función alternativa
MCLK	O	**MMC salida de reloj:** Reloj para transferencia de datos o comandos	-
MCMD	I/O	**MMC línea de comandos:** Canal de comandos bidireccional usado para la inicialización de la tarjeta y comandos para transferencia de datos. Para evitar cualquier consumo de corrientes parásitas, si no se utiliza la señal MCMD se debe polarizar a V_{SS} o V_{DD}	-
MDAT	I/O	**MMC línea de datos:** Canal de datos bidireccional. Para evitar cualquier consumo de corrientes parásitas, si no se utiliza la señal MDAT se debe polarizar a V_{SS} o V_{DD}	-

Tabla. 1.6. Descripción de la señal de interfase multimedia

1.1.7 Descripción de las señales del UART

Nombre de la Señal	Tipo	Descripción	Función alternativa
RXD	I/O	**Recepción de datos seriales:** RXD envía y recibe datos en el modo serial 0 y solo recibe datos en los modos seriales I/O 1, 2 y 3.	P3.0
TXD	O	**Transmisión de datos seriales:** TXD saca el reloj cambiado en el modo serial 0 y transmite datos en los modos seriales I/O 1, 2 y 3.	P3.1

Tabla. 1.7. Descripción de las señales del UART

1.1.8 Descripción de las señales del controlador SPI

Nombre de la Señal	Tipo	Descripción	Función alternativa
MISO	I/O	**Línea de datos SPI. Entrada maestra – salida esclava:** En el modo maestro, MISO recibe datos desde el esclavo periférico. Cuando está en modo esclavo, MISO envía datos hacia el controlador maestro.	P4.0
MOSI	I/O	**Línea de datos SPI. Salida maestra–entrada esclava:** En el modo maestro, MISO envía datos hacia el periférico esclavo. Cuando esta en modo esclavo, MISO recibe datos desde el controlador maestro.	P4.1
SCK	I/O	**Línea de reloj SPI:** Cuando está en modo maestro, SCK envía señales de reloj hacia el periférico esclavo. Cuando está en modo esclavo, SCK recibe señales de reloj desde el controlador maestro.	P4.2
SS*	I	**Línea selectora de esclavo SPI:** Cuando en el modo de esclavo controlado, SS* habilita el modo esclavo	P4.3

Tabla. 1.8. Descripción de las señales del controlador SPI

1.1.9 Descripción de las señales del controlador TWI

Nombre de la Señal	Tipo	Descripción	Función alternativa
SCL	I/O	**Reloj serial TWI:** Cuando el controlador TWI está en modo maestro, SCL envía la señal de reloj hacia el periférico esclavo. Cuando el controlador TWI está en modo esclavo, SCL recibe la señal de reloj del controlador maestro.	P1.6
SDA	I/O	**Datos seriales TWI:** SDA es una línea de datos bidireccional de dos hilos	P1.7

Tabla. 1.9. Descripción de las señales del controlador TWI

1.1.10 Descripción de las señales de Conversor A/D

Nombre de la Señal	Tipo	Descripción	Función alternativa
AIN1:0	I	**Entradas Analógicas del conversor A/D**	-
AREFP	I	**Entrada Analógica de Voltaje de Referencia Positivo**	-
AREFN	I	**Entrada Analógica de Voltaje de Referencia Negativo** Este pin esta conectado internamente con AVSS	-

Tabla. 1.10. Descripción de las señales del conversor A/D

1.1.11. Descripción de las señales de la interfase de Teclado (Keypad)

Nombre de la Señal	Tipo	Descripción	Función alternativa
KIN3:0	I	**Líneas de Entrada del Teclado** Manteniendo uno de estos pines en alto o bajo por periodos de 24 oscilaciones se dispara la interrupción del teclado	P1.3:0

Tabla. 1.11. Descripción de las señales de la interfase de teclado

1.1.12. Descripción de las señales de Acceso Externo

Nombre de la Señal	Tipo	Descripción	Función alternativa
A15:8	I/O	**Líneas de Dirección** Líneas de dirección más significativas para bus externo Direcciones más significativas y líneas de datos multiplexadas para la interfase IDE	P2.7:0
AD7:0	I/O	**Líneas de Dirección / Datos** Direcciones menos significativas y líneas de datos multiplexadas para memoria externa o interfase IDE	P0.7:0
ALE	O	**Salida habilitadora del Latch de direcciones** ALE indica el comienzo de un ciclo de bus externo e indica que esta disponible una dirección válida en las líneas A7:0. Un latch externo es usado para demultiplexar la dirección del bus de datos / direcciones.	-
ISP*	I/O	**Entrada habilitadora ISP** Esta señal debe mantenerse en GND mediante una resistencia de pull – down, en flanco descendente de reset para forzar la ejecución del bootloader	-
RD*	O	**Señal de Lectura** Señal de lectura activada cuando se ejecuta una operación de lectura en memoria externa	P3.7
WR*	O	**Señal de Escritura** Señal de escritura activada cuando se ejecuta una operación de escritura en memoria externa	P3.6

Tabla. 1.12. Descripción de las señales de acceso externo

1.1.13. Descripción de las señales del sistema

Nombre de la Señal	Tipo	Descripción	Función alternativa
RST	I	**Entrada de Reset** Manteniendo este pin en alto por 64 periodos de oscilación mientras el oscilador esta corriendo resetea el dispositivo. Los pines de los puertos vuelven a sus condiciones de reseteo cuando se aplica un voltaje menor que V_{IL}, esté o no	-

		corriendo el oscilador. Este pin tiene una resistencia interna de pull – down que permite al dispositivo ser reseteado mediante la conexión de una capacitor entre este pin y VDD. Mediante un reset se puede salir del modo IDLE o Power Down y regresar al modo normal de operación.	
TST*	I	**Entrada de Prueba** Señal de entrada al modo de prueba. Este pin debe esta conectado a VDD.	-

Tabla. 1.13. Descripción de las señales del sistema

1.1.14. Descripción de las señales de energía

Nombre de la Señal	Tipo	Descripción	Función alternativa
VDD	PWR	**Suministro de Voltaje Digital** Conecte estos pines a un suministro de voltaje de +3V	-
VSS	GND	**Tierra del Circuito** Conecte este pin a tierra	-
AVDD	PWR	**Suministro de Voltaje Analógico** Conecte estos pines a un suministro de voltaje de +3V	-
AVSS	GND	**Tierra Analógica** Conecte este pin a tierra	-
PVDD	PWR	**Suministro de Voltaje PLL** Conecte estos pines a un suministro de voltaje de +3V	-
PVSS	GND	**Tierra PLL** Conecte este pin a tierra	-
UVDD	PWR	**Suministro de Voltaje USB** Conecte estos pines a un suministro de voltaje de +3V	-
UVSS	GND	**Tierra USB** Conecte este pin a tierra	-

Tabla. 1.14. Descripción de las señales de energía

1.2 CONTROLADOR DE RELOJ

El controlador de reloj del AT89C51SND1C está basado en un oscilador interno que alimenta a un PLL (Phase Lock Loop) interno, todos los relojes internos que van hacia los periféricos y al núcleo del CPU son generados por este controlador.

1.2.1 Oscilador

El AT89C51SND1C posee los pines X1 y X2 que son la entrada y la salida de un inversor interno de fase simple, que puede ser configurado por componentes externos, como indica la Figura 1.3. Los valores de los capacitores y del cristal se presentan a continuación:

Símbolo	Parámetro	Min	Típico	Máx	Unidad
C_{X1}	Capacitancia interna		10		PF
C_{X2}	Capacitancia externa		10		PF
C_L	Capacitancia de carga equivalente		5		PF
DL	Drive Level			50	MW
F	Frecuencia del Cristal			20	MHz
RS	Resistencia en serie del cristal			40	Ω
CS	Capacitancia Shunt del Cristal			6	PF

Tabla. 1.15. Valores típicos

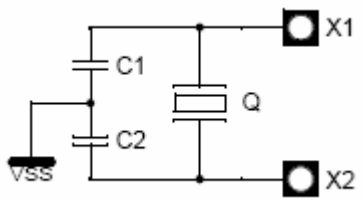


Figura. 1.3. Conexión del cristal

Para una operación normal del cristal ningún componente externo será añadido a este esquema. En casos especiales puede ser necesario aumentar capacitores externos de máximo 10 pF entre las entradas X1 y X2 a tierra. X1 y X2 no deben ser usados para manejar otros circuitos.

El oscilador genera tres señales diferentes: Señal de reloj para PLL, para CPU central y para los periféricos, tal como se nuestra en la figura 1.4.

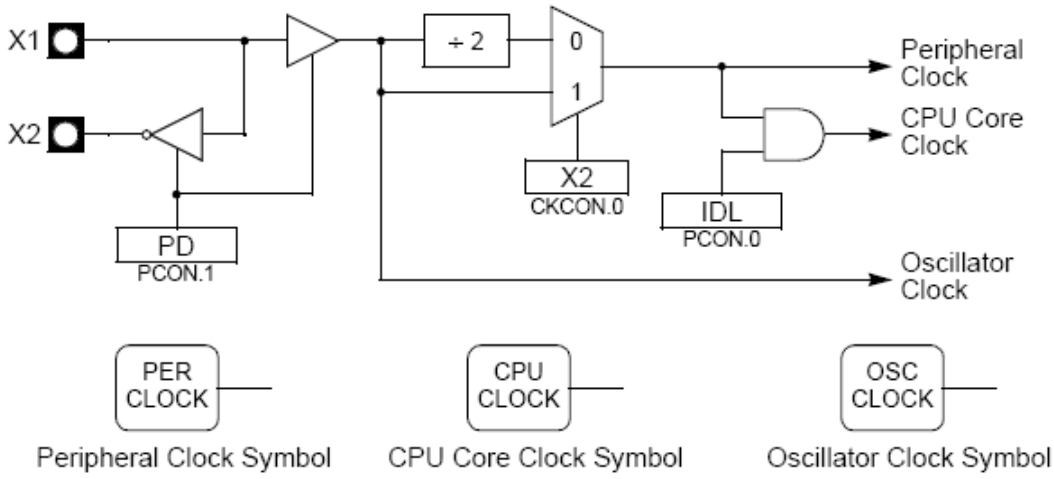


Figura. 1.4. Diagrama de bloques del oscilador

Estas señales pueden ser habilitadas o deshabilitadas dependiendo del modo de reducción de potencia. La señal de reloj de los periféricos es usada para generar: Timer 0, Timer 1, MMC, ADC, SPI y los tiempos de muestreo de los puertos.

1.2.2 Descripción del PLL

Es usado para generar un reloj interno de alta frecuencia (reloj PLL) sincronizado con un reloj externo de baja frecuencia (reloj de oscilador), el reloj PLL proporciona relojes a la interfase de audio, decodificador MP3 e interfase USB. En la siguiente figura se muestra la estructura interna del PLL.

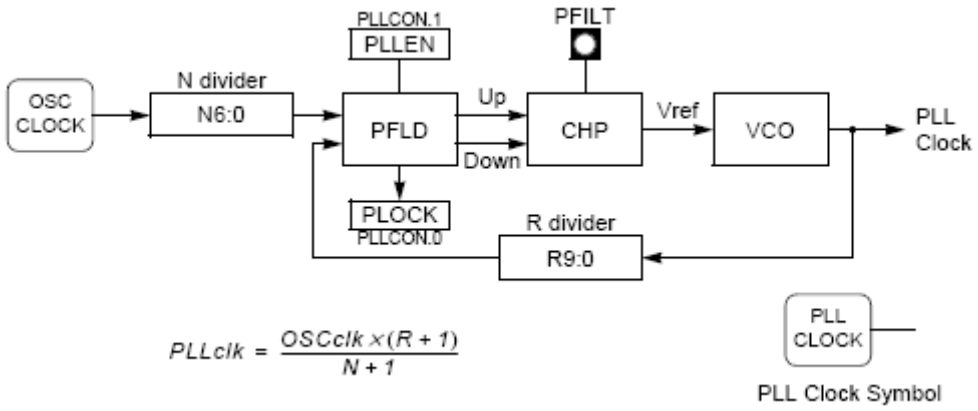


Figura. 1.5. Diagrama de bloques del PLL

El bloque PFLD (Phase Frequency Comparator and Lock Detector), hace la comparación entre el reloj de referencia proveniente del divisor N y el reloj invertido proveniente del divisor R y genera algunos pulsos altos o bajos dependiendo de la posición límite del reloj inverso. El bit PLLEN en el registro PLLCON es usado para habilitar el generador de reloj. Cuando el PLL es asegurado, el bit PLOCK se setea. El bloque CHP (Charge Pump), genera un voltaje de referencia para el bloque VCO este voltaje de referencia depende directamente del filtro que se conecta al pin PFILT, como se muestra en la figura.

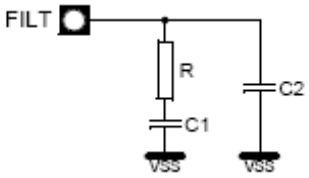


Figura. 1.6. Conexión del filtro PLL

Símbolo	Parámetro	Min	Típico	Máx	Unidad
R	Resistencia del filtro		100		Ω
C1	Capacitancia 1 del filtro		10		nF
C2	Capacitancia 2 del filtro		2.2		nF

Tabla. 1.16. Valores típicos

1.2.2.1 Programación del PLL.

Para la programación del PLL se necesita seguir una serie de instrucciones, primero se habilita la generación del reloj, luego se debe esperar hasta que el PLL se encuentre asegurado lo cual indica que la señal de reloj se ha estabilizado. La frecuencia de la señal del PLL va a depender de las frecuencias de los relojes del decodificador de MP3 además de la interfase de audio

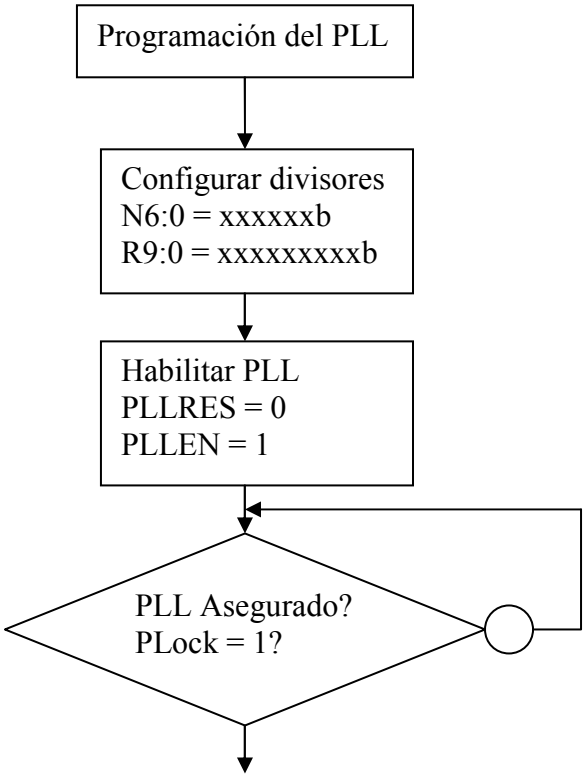


Figura. 1.7. Diagrama de flujo para la programación del PLL

1.2.3 Registros de control

1.2.3.1 Registros de control del módulo de reloj (CKCON).

7	6	5	4	3	2	1	0
-	WDX2	-	-	-	T1X2	T0X2	X2

# de bit	Mnemónico	Descripción
7	-	**Reservado** EL valor leído de este bit es indeterminado. No se debe setear este bit
6	WDX2	**Bit de Control del reloj de Watchdog**. Con un 1 selecciona la división del reloj por 2 como la entrada del reloj de watchdog. Con un 0 el reloj periférico es seleccionado como reloj de entrada
5 - 3	-	**Reservado** EL valor leído de estos bits es indeterminado. No se deben setear estos bits.
2	T1X2	**Bit de control de Timer 1**. Con un 1 selecciona el reloj del oscilador dividido por 2 como el reloj de entrada del timer 1 (X2 independiente). Con un 0 selecciona el reloj de periféricos como reloj de entrada del timer 1 (X2 dependiente).
1	T0X2	**Bit de control de Timer 0**. Con un 1 selecciona el reloj del oscilador dividido por 2 como el reloj de entrada del timer 0 (X2 independiente). Con un 0 selecciona el reloj periférico como reloj de entrada del timer 0 (X2 dependiente).
0	X2	**Bit de control del reloj del sistema.** Con un 0 selecciona 12 períodos del reloj por ciclo de máquina. Con un 1 selecciona 6 períodos del reloj por ciclo de máquina

Tabla. 1.17. Registros de control del la módulo de reloj

Valor del registro luego del reset 0000 000Xb

1.2.3.2 Registros de control del módulo PLL (PLLCON).

7	6	5	4	3	2	1	0
R1	R0	-	-	PLLRES	-	PLLEN	PLOCK

# de bit	Mnemónico	Descripción
7 - 6	R 1:0	**Bits menos significativos del divisor R del PLL**.- 2 LSB del divisor de 10 bits
5 - 4	-	**Reservado**.- EL valor leído de estos bits es siempre 0. No se deben setear estos bits.
3	PLLRES	**Bit de reset del PLL**.- Con un 1 en este bit reseteará el PLL. Con un 0 libera el PLL y permite su habilitación.
2	-	**Reservado**.- EL valor leído de este bit es siempre 0. No se debe setear este bit.
1	PLLEN	**Bit de habilitación del PLL**. Con un 1 habilita el PLL. Con un 0 deshabilita el PLL
0	PLOCK	**Indicador de bloqueo del PLL.** Se setea vía hardware cuando el PLL está bloqueado. Se encera por hardware cuando el PLL esta desbloqueado

Tabla. 1.18. Registros de control del módulo señal PLL

Valor del registro luego del reset 0000 1000b

1.2.3.3 Registros del divisor N del PLL (PLLNDIV).

7	6	5	4	3	2	1	0
-	N6	N5	N4	N3	N2	N1	N0

# de bit	Mnemónico	Descripción
7	-	**Reservado**.- El valor leído de este bit es siempre 0. No se debe setear este bit.
6 - 0	N 6:0	**Divisor N del PLL**. Valor de 7 bits asignado al divisor N

Tabla. 1.19. Registros del divisor n

Valor del registro luego del reset 0000 0000b

1.2.3.4 Registros del divisor R de la señal PLL (PLLRDIV).

7	6	5	4	3	2	1	0
R9	R8	R7	R6	R5	R4	R3	R2

# de bit	Mnemónico	Descripción
7 - 0	R 9:2	**Bits mas significativos del divisor R del PLL**.- 8 MSB del divisor R de 10 bits

Tabla. 1.20. Registros del divisor r

Valor del registro luego del reset 0000 0000b

1.3 MEMORIA DE PROGRAMA/CODIGO

El microcontrolador AT89C51SND1C posee un espacio de memoria de 64K Bytes, la cual tiene las características de una Memoria Flash, en esta se almacenará el código del programa. No es posible ejecutar código desde una memoria externa.

La funcionalidad que proporciona una Memoria Flash es que se puede realizar una programación directa sin necesidad de retirar el microcontrolador del circuito. El voltaje necesario para programar o borrar la memoria es abastecido por el mismo circuito (V_{DD}). El microcontrolador además posee un espacio de memoria de 4K Bytes en el cual se encuentra un programa interno de arranque (on-chip boot Flash Memory), este espacio de memoria posee un código de arranque programado en la fábrica que permite una programación IPS (In-System Programing).

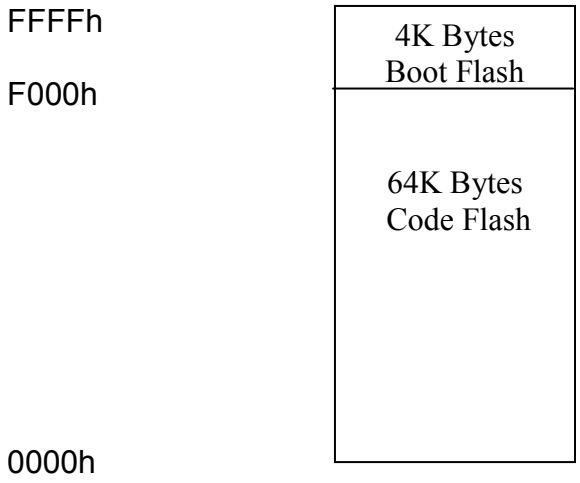


Figura. 1.8. Memoria de programa o código

1.3.1 Arquitectura de la Memoria Flash

Como se muestra en la figura 1.9 la memoria Flash del microcontrolador AT89C51SND1C está compuesta por cuatro espacios detallados a continuación:

Espacio de usuario.- Este espacio está compuesto por 64K Bytes de Memoria Flash organizada en 512 páginas de 128 Bytes. Que contiene el código de aplicación del usuario.

Espacio de inicialización.- Este espacio está compuesto por 64K Bytes de Memoria Flash. Contiene el programa de inicialización (boot - loader) para programación en-sistema (ISP) y las rutinas para programación en-aplicación. (IAP).

Espacio de seguridad de hardware.- Este espacio está compuesto por un Byte: el Hardware Security Byte (HSB) divido en 2 nibbles separados. El MSN contiene el bit X2 de modo de configuración y el bit de Boot Loader Jump y puede ser escrito por software mientras el LSN contiene el nivel de seguridad del sistema para proteger el contenido de la memoria en contra de piratería y puede ser solo escrito vía hardware.

Espacio de fila extra.- Este espacio está compuesto por dos Bytes:

• Software Boot Vector (SBV).

Este byte es usado por el software boot loader para crear la dirección de inicialización.

• Software Security Byte (SSB).

Este byte es usado para bloquear la ejecución de algún comando del boot loader.

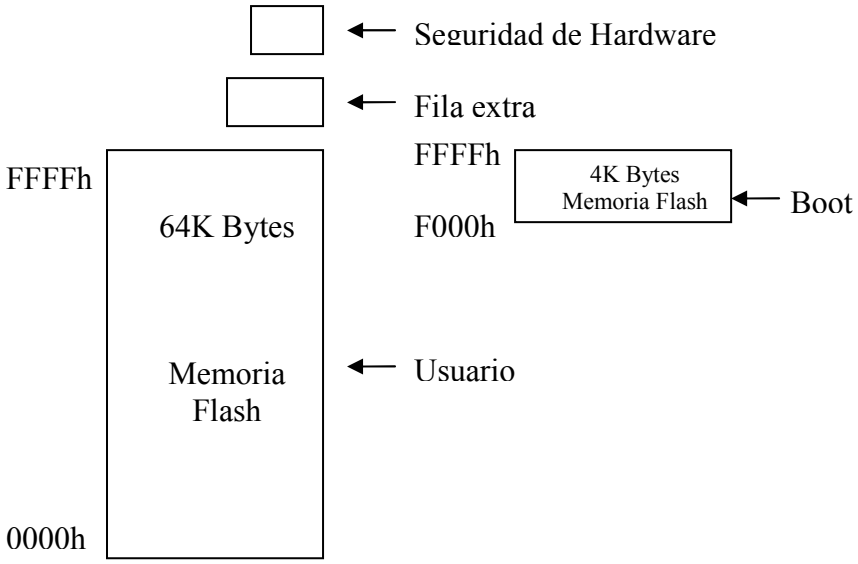


Figura. 1.9 Arquitectura memoria flash

1.3.2 Ejecución de la memoria de inicialización (Boot Memory)

Como ya se habló el espacio de memoria del microcontrolador C51 es de 64K Bytes, existen algunas vías para permitir que la memoria de inicialización sea corrida en el espacio de código asignado F000h a FFFFh. La memoria de inicialización es habilitada seteando el bit ENBOOT del registro auxiliar AUXR1. Existen tres maneras de setear este bit las cuales son detalladas a continuación:

- Por software. Consiste en escribir al registro AUXR1 desde el software de usuario. Esto habilita el boot loader o ejecuta rutinas API .

- Por hardware. Cuando el pin ISP es llevado a un bajo nivel el chip se resetea, y el bit ENBOOT se activa, haciendo que el vector del programa apunte a la dirección F000h en lugar de 0000h para que ejecute el software de inicialización.
- Por programación Esta basada en el Boot Loader Jump Bit (BLJB) del HSB. Cuando este bit es programado, el chip setea el bit ENBOOT y hace que el vector apunte a la dirección F000h en lugar de 0000h, para que ejecute el software de inicialización.

1.3.2.1 Diagrama de flujo del proceso de inicialización.

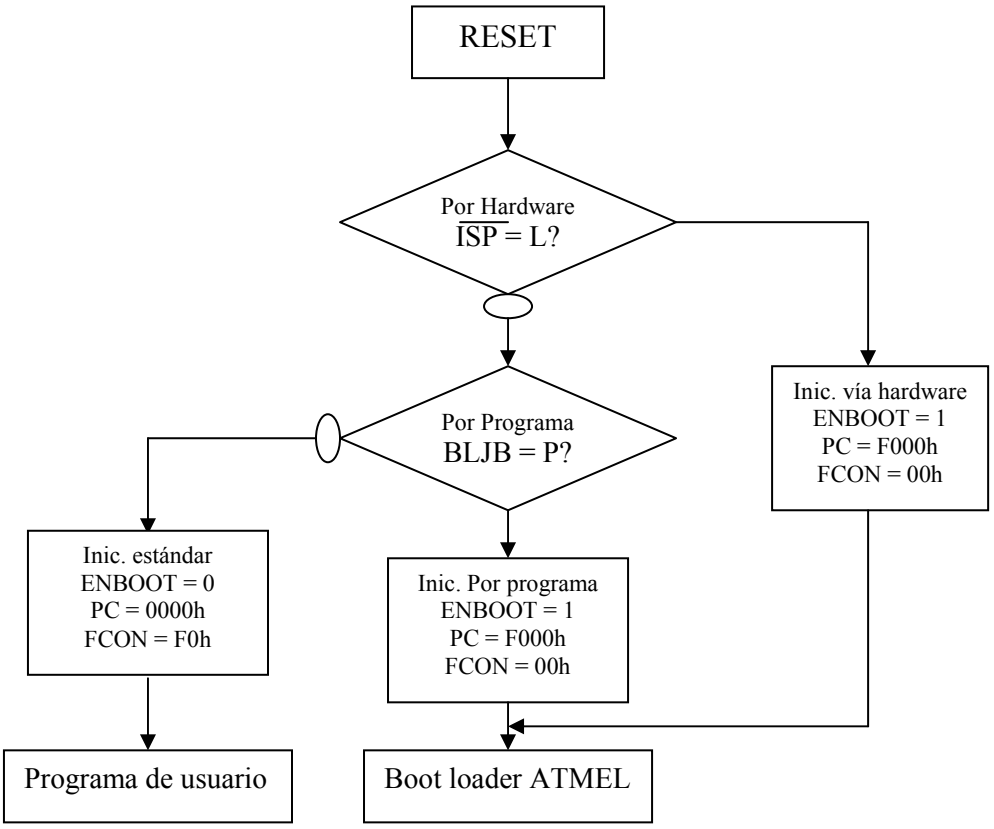


Figura. 1.10 Inicialización del microcontrolador

1.3.3 Registros de memoria

1.3.3.1 Registro auxiliar 1 (AUXR1)

7	6	5	4	3	2	1	0
-	-	ENBOOT	-	GF3	0	-	DPS

# de bit	Mnemónico	Descripción
7 – 6	-	**Reservado.** El valor de este bit es indeterminado. No se debe setear este bit.
5	ENBOOT	**Habilitador del Boot** Con un 1 este bit se direcciona el mapeado al espacio de memoria entre F000h y HF Con un 0 se deshabilita
4	-	**Reservado.** El valor de este bit es indeterminado. No se debe setear este bit.
3	GF3	**Bandera general** Es una bandera de propósito general
2	0	**Siempre nulo** Este bit siempre tiene cero para utilización propia de este registro
1	-	**Reservado.** El valor de este bit es indeterminado. No se debe setear este bit.
0	DPS	**Bit selector del puntero de datos** Con un 1 se selecciona el segundo puntero de datos DPTR1 Con un 0 se selecciona el primer puntero de datos DPTR0

Tabla. 1.21. Registro AUXR1

Valor del registro luego del reset XXXX 00X0b

1.4 MEMORIA DE DATOS

El microcontrolador AT89C51SND1C posee dos diferentes espacios de memoria de datos:

- Espacio interno mapeado en tres diferentes segmentos
  - Segmento inferior 128 Bytes RAM

- Segmento superior 128 Bytes RAM
- Segmento expandido 2048 Bytes RAM

- Espacio externo

En la siguiente figura se muestra la distribución del espacio interno y externo de memoria de datos.

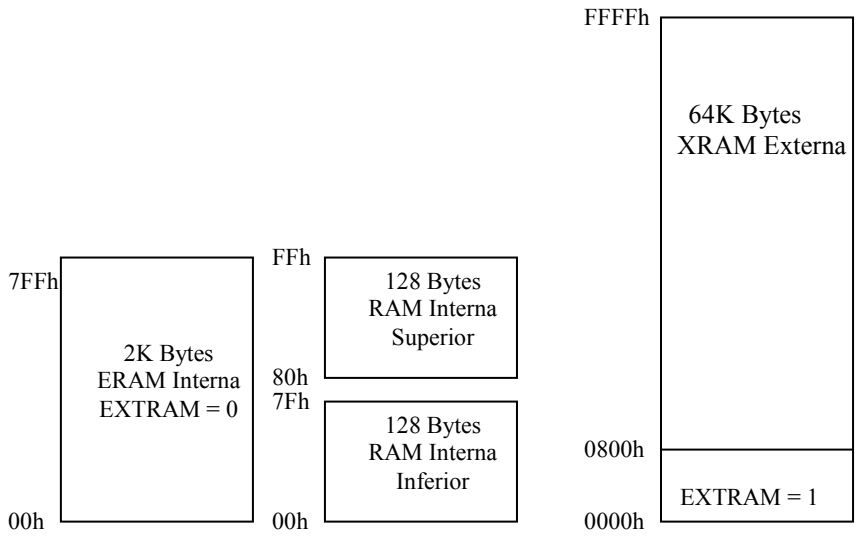


Figura. 1.11. Memoria de datos

1.4.1 RAM inferior de 128 Bytes

Este espacio de memoria está comprendido entre las direcciones 00h a 7Fh, utilizando modos de direccionamiento directo o indirecto. Los 32 Bytes inferiores están agrupados en cuatro bancos de 8 registros (R0 a R7). Estos cuatro bancos se seleccionan mediante combinaciones de 2 bits auxiliares RS0 y RS1 del registro PSW. A continuación se presenta una tabla con las respectivas combinaciones.

RS0	RS1	Descripción
0	0	Banco de registros 0 desde 00h a 07h
0	1	Banco de registros 1 desde 08h a 0Fh
1	0	Banco de registros 2 desde 10h a 17h
1	1	Banco de registros 3 desde 18h a 1Fh

Tabla. 1.22. Combinaciones para selección de bancos

1.4.2 RAM superior de 128 Bytes

La RAM superior de 128 Bytes va desde la dirección 80h a FFh usando solo modo de direccionamiento indirecto.

1.4.3 RAM expandida

Los 2K Bytes de memoria RAM expandida (ERAM) van desde las direcciones 0000h a 07FFh usando modo de direccionamiento indirecto a través de la instrucción MOVX. En este rango de direcciones, de bit EXTRAM del registro auxiliar AUXR es usado para seleccionar la ERAM (default) o la XRAM. Como se muestra en la figura 1.11 cuando EXTRAM = 0, se selecciona la ERAM y cuando EXTRAM = 1, se selecciona la XRAM.

La memoria ERAM puede ser redimensionada usando los bits XRS1:0 del registro AUXR. En la siguiente tabla se puede ver las diferentes combinaciones de los bits XRS0:1.

XRS0	XRS1	Tamaño ERAM	Direcciones
0	0	256 Bytes	0h a 00FFh
0	1	512 Bytes	0h a 01FFh
1	0	1k Byte	0h a 03FFh
1	1	2k Byte	0h a 07FFh

Tabla. 1.23. Selección de RAM

1.4.4 Espacio externo

1.4.4.1 Interfase de memoria.

La interfase de memoria externa comprende a un bus externo (puerto 0 y puerto 2) así como un bus con la señales de control (RD, WR, y ALE).

En la siguiente figura se muestra la estructura de un bus de direcciones externo. P0 transporta las direcciones A7:0 mientras P2 transporta las direcciones A15:8. Los datos D7:0 son multiplexados con A7:0 en P0.

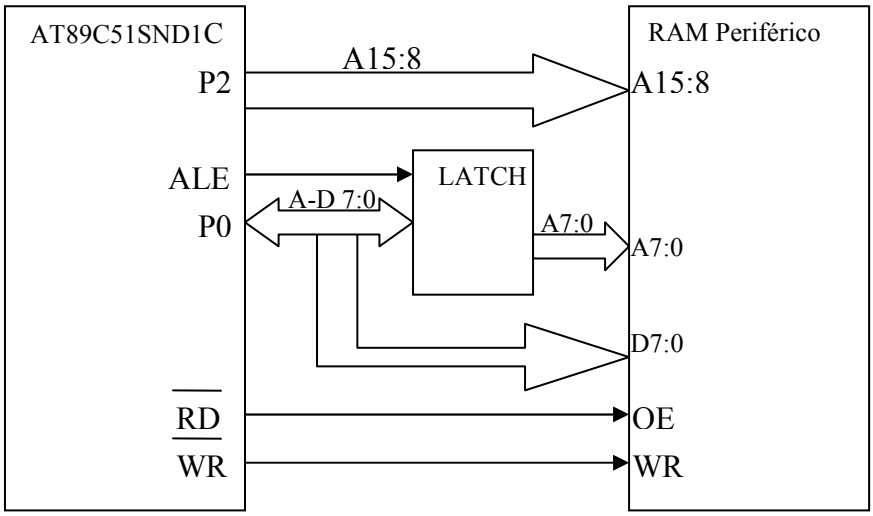


Figura. 1.12. Interfase de memoria

En la siguiente tabla se muestran las diferentes señales de la interfase de memoria externa

Nombre de la Señal	Tipo	Descripción	Función alternativa
A 15:8	O	**Líneas de direcciones.** Líneas superiores de direcciones del bus externo	P2.7:0
A-D 7:0	I/O	**Líneas de direcciones y datos.** Líneas inferiores de direcciones del bus externo,	P0.7:0

		multiplexadas y datos de la memoria externa	
ALE	O	**Habilitador del LATCH.** Esta señal indica que cierta información valida esta disponible en las líneas de direcciones A-D 7:0	-
RD	O	**Lectura** Lee la señal de salida del dispositivo externo de memoria	P3.7
WR	O	**Escritura** Escribe la señal de salida al dispositivo externo de memoria	P3.8

Tabla. 1.24. Señales de memoria externa

1.4.5 Registro de control auxiliar (AUXR)

7	6	5	4	3	2	1	0
-	EXT16	M0	DPHDIS	XRS1	XRS0	EXTRAM	AO

# de bit	Mnemónico	Descripción
7	-	**Reservado.** El valor de este bit es indeterminado
6	EXT16	**Bit de habilitación extena** Con un 1 habilita el acceso al modo 16-bit Con un 0 deshabilita el acceso al modo 16-bit y habilita el acceso al modo estándar de 8-bit
5	M0	**Bit de extensión de acceso a memoria externa.** Con un 1 se alarga la duración de las señales de reloj del RD y WR a 15 ciclos de reloj Con un 0 se mantiene en 3 ciclos
4	DPHDIS	**Bit para desactivar DPH** Con un 1 habilita el DPH Con un 0 deshabilita el DPH
3 - 2	XRS1:0	**Bits de expansión del tamaño de la RAM** Se comportan de acuerdo a la tabla mostrada anteriormente
1	EXTRAM	**Bit de habilitación de RAM externa** Con un 1 se selecciona la memoria XRAM Con un 0 Se trabaja normalmente
0	AO	**Bit habilitador de la salida ALE** Con un 1 la salida ALE solo se ejecuta durante las instrucciones MOVX Con un 0 la salida ALE se ejecuta normalmente.

Tabla. 1.25. Registro AUXR

Valor del registro luego del reset X000 1101b

1.5 REGISTROS DE FUNCIONES ESPECIALES

Los registros de funciones especiales (SFRs) del microcontrolador AT89C51SND1C están explicados en las diferentes tablas que se presentan a continuación.

1.5.1 Registros principales

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
ACC	E0h	Acumulador	-	-	-	-	-	-	-	-
B	F0h	Registro B	-	-	-	-	-	-	-	-
PSW	D0h	Estado de programa	CY	AC	F0	RS1	RS0	OV	F1	P
SP	81h	Puntero de Pila	-	-	-	-	-	-	-	-
DPL	82h	Puntero inferior de datos	-	-	-	-	-	-	-	-
DPH	83h	Puntero superior de datos	-	-	-	-	-	-	-	-

Tabla. 1.26. Registros principales

1.5.2 Manejo del sistema

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
PCON	87h	Control de Energía	SMOD1	SMOD0	-	-	GF1	GF0	PD	IDL
AUXR	8Eh	Registro Auxiliar 0	-	EXT16	M0	DPHDIS	XRS1	XRS0	EXTRAM	AO
AUXR1	A2h	Registro Auxiliar 1	-	-	ENBOOT	-	GF3	0	-	DPS
NVERS	FBh	Número de Versión	NV7	NV6	NV5	NV4	NV3	NV2	NV1	N0

Tabla. 1.27. Registro de sistema

1.5.3 Registros especiales de PLL

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
CKCON	8Fh	Control de reloj	-	-	-	-	-	-	-	X2
PLLCPN	E9h	Control PLL	R1	-	-	-	PLLRES		PLLEN	PLOCK
PLLNDIV	EEh	Divisor N PLL	-	N6	N5	N4	N3	N2	N1	N0
PLLRDIV	EFh	Divisor R PLL	R9	R8	R7	R6	R5	R4	R3	R2

Tabla 1.28. Registro PLL

1.5.4 Registros especiales de memoria flash

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
FCON	D1h	Control Flash	FPL3	FPL2	FPL1	FPL0	FPS	FMOD1	FMOD0	FBUSY

Tabla. 1.29. Registro memoria flash

1.5.5 Registros especiales del decodificador MP3

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
MP3CON	AAh	Control MP3	MPEN	MPBBST	CRCEN	MPKANC	MSKREQ	MSKLAY	MSKSYN	MSKCRC
MP3STA	C8h	Estado MP3	MPANC	MPREQ	ERRLAY	ERRSYN	ERRCRC	MPFS1	MPFS0	MPVER
MP3STA1	AFh	Estado 1 MP3	-	-	-	MPFREQ	MPBREQ	-	-	-
MP3DAT	ACh	Dato MP3	MPD7	MPD6	MPD5	MPD4	MPD3	MPD2	MPD1	MPD0
MP3ANC	ADh	Dato auxiliar MP3	AND7	AND6	AND5	AND4	AND3	AND2	AND1	AND0
MP3VOL	9Eh	Control de volomen izq	-	-	-	VOL4	VLO3	VOL2	VOL1	VOL0
MP3VOR	9Fh	Control de volumen der	-	-	-	VOR4	VOR3	VOR2	VOR1	VOR0
MP3BAS	B4h	Control de bajos	-	-	-	BAS4	BAS3	BAS2	BAS1	BAS0
MP3MED	B5h	Control de medios	-	-	-	MED4	MED3	MED2	MED1	MED0
MP3TRE	B6h	Control de agudos	-	-	-	TRE4	TRE3	TRE2	TRE1	TRE0
MP3CLK	EBh	Control de divisor clk	-	-	-	MPCD4	MPCD3	MPCD2	MPCD1	MPCD0

Tabla. 1.30. Registro decodificador MP3

1.5.6 Registros especiales de la interfase de audio

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
AUDCON0	9Ah	Control de audio 0	JUST4	JUST3	JUST2	JUST1	JUST0	POL	DSIZ	HLR
AUDCON1	9Bh	Control de audio 1	SRC	DRQEN	MSREQ	MUDRN	-	DUP1	DUP0	AUDEN
AUDSTA	9Ch	Estado del audio	SREQ	UDRN	AUBUSY	-	-	-	-	-
AUDDAT	9Dh	Datos de audio	AUD7	AUD6	AUD5	AUD4	AUD3	AUD2	AUD1	AUD0
AUDCLK	ECh	Divisor de reloj de aud	-	-	-	AUCD4	AUCD3	AUCD2	AUCD1	AUCD0

Tabla. 1.31. Registro interfase de audio

1.5.7 Registro de interfase IDE

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
DAT16H	F9h	Byte de orden alto	D15	D14	D13	D12	D11	D10	D9	D8

Tabla. 1.32. Registro interfase IDE

1.5.8 Registro de interfase de teclado

Mnemónico	Dirección	Nombre	7	6	5	4	3	2	1	0
KBCON	A3h	Control del teclado	KINL3	KINL2	KINL1	KINL0	KINM3	KINM2	KINM1	KINM0
KBSTA	A4h	Estado del teclado	KPDE	-	-	-	KINF3	KINF2	KINF1	KINF0

Tabla. 1.33. Registro interfase de teclado

1.6 MANEJO DE ENERGÍA

El micro controlador posee 2 modos de reducción de energía. El modo Idle y el modo Power-down. A estos modos se los puede activar o desactivar de acuerdo a

la conveniencia del programador o usuario mediante el seteo de bits en el registro PCON.

1.6.1 Registros de control

1.6.1.1 Registros de configuración de energía (PCON).

7	6	5	4	3	2	1	0
SMOD1	SMOD0	-	-	GF1	GF0	PD	IDL

# de bit	Mnemónico	Descripción
7	SMOD1	**Modo 1 del puerto serial**.- Con un 1 se selecciona doble tasa de baudio en los modos 1, 2 y 3
6	SMOD0	**Modo 0 del puerto serial**.- Con un 1 se selecciona el bit FE del registro SCON Con un 0 se selecciona el bit SM0 del registro SCON
5 - 4	-	**Reservado**.- EL valor leído de este bit es indeterminado
3	GF1	**Bandera de propósito general 1**.- Sirve para indicar si ha existido alguna interrupción durante la operación norma o en el modo Idle
2	GF0	**Bandera de propósito general 0**. Sirve para indicar si ha existido alguna interrupción durante la operación norma o en el modo Idle
1	PD	**Bit de modo Power - down.** Con un 1 se activa el modo Power – down Se pone en 0 cuando se ha peseteado el sistema o ha existido alguna interrupción
0	IDL	**Bit de modo Idle.-** Con un 1 se activa el modo Idle Se pone en 0 cuando se ha peseteado el sistema o ha existido alguna interrupción

Tabla. 1.34. Registro PCON

Valor del registro luego del reset 00XX 0000b

1.6.2 Reset

Para iniciar la operación del controlador o simplemente reinicar y dar un reset, tiene que ser aplicado un nivel alto en el pin RST. Un mal nivel aplicado puede dar ciertos errores en los registros. Un correcto reset inicializa el microcontrolador ubicando el putero en la dirección 0000h. La entrada RST tiene una resistencia de pull-down permitiendo que se obtenga un reset tan solo poniendo un capacitor a V_{DD}.

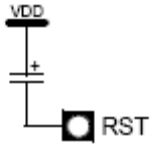


Figura. 1.13. Circuito típico de reset

1.6.2.1 Condiciones de los pines en los modos de operación especial.

Modo	Puerto 0	Puerto 1	Puerto 2	Puerto 3	Puerto 4	Puerto 5	MMC
Reset	Flotante	Alto	Alto	Alto	Alto	Alto	Flotante
Idle	Dato	Dato	Dato	Dato	Dato	Dato	Dato
Power- Down	Dato	Dato	Dato	Dato	Dato	Dato	Dato

Tabla. 1.35. Modos de operación especial

1.6.2.2 Reseteo en frio

Existen dos condiciones antes del arranque del CPU

- El voltaje VDD debe alcanzar el rango especificado
- El nivel de la entrada X1 debe estar afuera de las especificaciones de VIH y VIL

Si una de estas dos condiciones no se encuentran, el microcontrolador no va a arrancar correctamente y puede eliminar alguna instrucción del espacio de

programa. El pin RST debe estar activo el tiempo necesario para que estas dos condiciones se cumplan.

Para determinar el valor del capacitor se deben considerar dos parámetros, los cuales se presentan a continuación:

Oscilador Tiempo de inicio	**Tiempo de subida de V_{DD}**		
	1ms	**10ms**	**100ms**
5ms	820nf	1.2µf	12µf
20ms	2.7µf	3.9µf	12µf

Tabla. 1.36. Valores de capacitor

1.6.2.3 Reseteo en caliente.

Para obtener un reset válido, la señal del reset debe mantenerse activa por lo menos dos ciclos de máquina mientras el oscilador está corriendo, esto es 24 oscilaciones del periodo del reloj.

1.7 INTERFASE DE TECLADO

El microcontrolador AT89C51SND1C posee una interfase de teclado que permite la conexión de un teclado matricial de 4 x n. Está basado en cuatro entradas con capacidad de interrupciones programables tanto en alto como bajo nivel. Estas entradas están disponibles como función alterna de P1.3:0 además estas entradas permiten salir de los modos de control de energía Idle y Power – down.

1.7.1 Descripción

La interfase de teclado del microcontrolador utiliza dos registros de función especial: KBCON, el registro de control de teclado; y KBSTA, el registro de estado y control de teclado.

Las entradas de teclado son consideradas como interrupciones compartiendo el mismo vector de interrupciones. Un bit de interrupción habilitado (EKB en el registro IEN1) permite habilitar o deshabilitar las interrupciones del teclado. Cada entrada de teclado tiene la capacidad de detectar un nivel programable de acuerdo al valor de los bits KINL3:0 en el registro KBCON. La detección de nivel es luego reportada en las banderas de interrupción KINF3:0 del registro KBSTA.

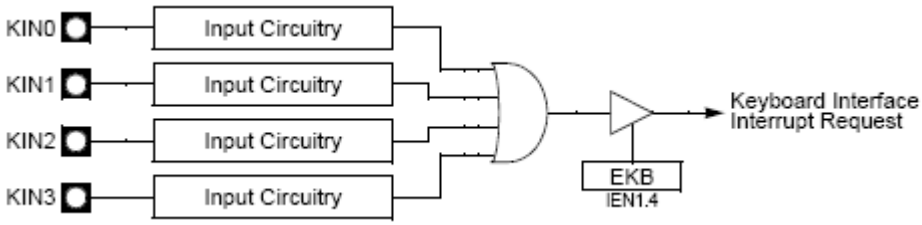


Figura. 1.13. Diagrama de bloques de la interfase de teclado

Una interrupción de teclado es solicitada cada vez que una o más banderas son activadas. Cada una de estas cuatro banderas pueden ser enmascaradas por software usando los bits KINM3:0 en el registro KBCON.

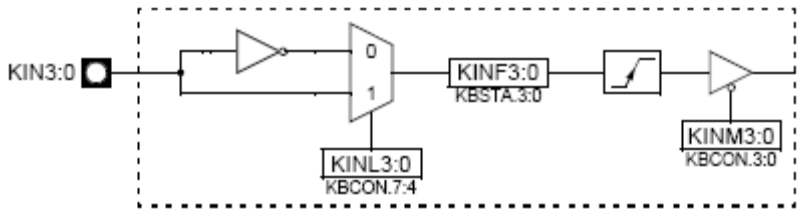


Figura. 1.14. Circuito de entrada de teclado

1.7.2 Modo de reducción de energía

La entradas KIN3:0 permiten salir de los modos idle y power-down como se detalló anteriormente. Para habilitar esta característica, el bit KPDE en el registro KBSTA debe estar activado con un 1 lógico.

1.7.3 Registros de control

1.7.3.1 Registro de control de teclado KBCON

7	6	5	4	3	2	1	0
KINL3	KINL2	KINL1	KINL0	KINM3	KINM2	KINM1	KINM0

# de bit	Mnemónico	Descripción
7 - 4	KINL3:0	**Nivel de entrada de teclado**.- Con un 1 habilita la detección de un nivel alto en las entradas de teclado. Con un 0 habilita la detección de un nivel bajo en las entradas de teclado
3 - 0	KINM3:0	**Mascará de entrada de teclado.-** Con un 1 se impide las respectivas banderas KINF3:0 generadas por las interrupciones de teclado. Con un 0 permita las respectivas banderas KINF3:0 generadas por las interrupciones de teclado.

Tabla. 1.37. Registro KBCON

Valor del registro luego del reset 0000 1111b

1.7.3.2 Registro de estado y control de teclado KBSTA

7	6	5	4	3	2	1	0
KPDE	-	-	-	KINF3	KINF2	KINF1	KINF0

# de bit	Mnemónico	Descripción
7	KPDE	**Bit habilitador de modo Power - down**.- Con un 1 habilita salir del modo Power – down con un a interrupción de teclado. Con un 0 deshabilita salir del modo Power – down con un a interrupción de teclado.
6 - 4	-	**Reservado**.-

		EL valor leído de estos bits son siempre 0
3 - 0	KINM3:0	**Bandera de interrupción de la entrada de teclado.-** Se activa con un 1 lógico vía hardware cuando las respectivas entradas KIN3:0 detectan un nivel programado.

Tabla. 1.38. Registro KBSTA

Valor del registro luego del reset 0000 0000b

CAPITULO II

CARACTERÍSTICAS DEL FORMATO MPEG I/II CAPA 3

2.1 INTRODUCCIÓN

MPEG proviene de las siglas en inglés “Moving Picture Expert Group”. MPEG es un grupo de personas que se encuentran dentro de la Organización Internacional de Estándares (ISO, por sus siglas en inglés) para generar estándares para video digital y compresión de audio, es decir, ellos definen una trama de bits comprimida, la cual implícitamente define un descompresor. Como cada empresa tiene sus propios algoritmos de compresión, es ahí donde recae la importancia de contar con un estándar internacional.

El formato MPEG (Moving Picture Experts Group) es un standard para compresión de video y de audio. Al ser creado se establecieron 4 tipos de MPEGs, MPEG-1, MPEG-2, MPEG-3 y MPEG-4. Cada uno de estos según su calidad y ancho de banda usado.

Las siglas MP3 responden a una abreviación de MPEG 1 capa 3, que es un esquema de codificación de audio general que debe su éxito a su gran capacidad de compresión sin pérdida aparente de calidad, superando una relación de 10 a 1, con un equipo medio y la posibilidad de descomprimir en tiempo real con carga de baja a media para el procesador.

Una capa representa un grupo de algoritmos de codificación. La capa 1 tiene poca complejidad; la capa 2 añade técnicas más avanzadas de localización de bits y requiere de una codificación y descodificación más compleja; la capa 3 añade un banco de filtros híbrido y una cuantificación no uniforme. Esas tres

capas pueden comprimir la transferencia continua de audio en cuatro, seis y doce veces, respectivamente.

2.2 COMPRESIÓN DE AUDIO

2.2.1 Digitalización

El sonido es una onda continua que puede propagarse a través de diferentes medios como el aire. Las ondas sonoras poseen las características propias de las ondas en general, tales como reflexión, refracción y difracción. Al tratarse de una onda continua, se requiere un proceso de digitalización para representarla como una serie de números. Hoy por hoy, la mayoría de las operaciones realizadas sobre señales de audio son digitales, dado que el almacenamiento como el procesado y transmisión de la señal en forma discreta proporciona grandes ventajas sobre los métodos analógicos o continuos, por ejemplo: menor sensibilidad al ruido en la transmisión, capacidad de incluir códigos de protección frente a errores, encriptación, etc.. La principal desventaja de la señal discreta es que requiere un ancho de banda mucho mayor que el de la señal analógica.

El proceso de digitalización se compone de dos fases: muestreo y cuantización. En el muestreo se divide el eje del tiempo en segmentos discretos: la frecuencia de muestreo será la inversa del tiempo que medie entre una medida y la siguiente. La cuantización, consiste simplemente en medir el valor de la señal en amplitud y guardarlo. El teorema de Nyquist garantiza que la frecuencia necesaria para muestrear una señal que tiene sus componentes más altas a una frecuencia dada f es como mínimo 2f. Por tanto, siendo el rango superior de la audición humana en torno a los 20 Khz, la frecuencia que garantiza un muestreo adecuado para cualquier sonido audible será de unos 40 Khz. Concretamente, para obtener sonido de alta calidad se utilizan frecuencias, por ejemplo, de 44.1 Khz en el caso del CD. Según la naturaleza de la aplicación, las frecuencias adecuadas pueden ser muy inferiores, de tal manera que el proceso de la voz acostumbra a realizarse a una frecuencia de entre 6 y 20 Khz. o incluso menos. En la

cuantización mientras más bits se utilicen para la división del eje de la amplitud, más exacta será la partición y por tanto menor el error al atribuir una amplitud concreta al sonido en cada instante. Por ejemplo, 8 bits ofrecen 256 niveles de cuantización. La división del eje se puede realizar a intervalos iguales o según una determinada función de densidad, buscando más resolución en ciertos tramos si la señal que se trata tiene más componentes en cierta zona de intensidad, como se mostrará en las técnicas de codificación. El proceso completo de muestreo y cuantización se denomina habitualmente PCM (Pulse Code Modulation).

2.2.2 Codificación y Compresión

Previo al estudio de los sistemas de codificación y compresión, se realizará un corto análisis de la percepción auditiva del ser humano, para comprender por qué se puede desechar una cantidad significativa de la información que proporciona el PCM.

El oído humano percibe un rango de frecuencias entre 20 Hz. y 20 Khz. En primer lugar, la sensibilidad es mayor en la zona alrededor de los 2 - 4 Khz., por lo tanto los sonidos más cercanos a los extremos de la escala resultan más difíciles de ser escuchados. En segundo lugar está el enmascaramiento, cuyas propiedades utilizan varios algoritmos, por ejemplo: cuando la componente a cierta frecuencia de una señal tiene una energía elevada, el oído no puede percibir componentes de menor energía en frecuencias cercanas, tanto inferiores como superiores. A una cierta distancia de la frecuencia enmascaradora, el efecto se reduce tanto que resulta despreciable; el rango de frecuencias en las que se produce el fenómeno se denomina banda crítica. Las componentes que pertenecen a la misma banda crítica se influyen mutuamente y no afectan ni se ven afectadas por las que aparecen fuera de ella. La amplitud de la banda crítica es diferente según la frecuencia en la que nos situemos y viene dada por unos determinados datos que demuestran que es mayor con la frecuencia. Todos estos datos se obtienen por experimentos psicoacústicos.

El fenómeno tratado anteriormente es el llamado enmascaramiento simultáneo o en frecuencia, y además existe el denominado enmascaramiento asimultáneo o en el tiempo. En conclusión, ciertas componentes en frecuencia de la señal admiten un mayor ruido del que generalmente consideraríamos tolerable y, por tanto, requieren menos bits para ser codificadas si se proporciona al codificador de los algoritmos adecuados para resolver máscaras.

La digitalización de la señal mediante PCM es la forma más simple de codificación de la señal. Al realizar la digitalización de una señal, se añade ruido indeseable. Para evitar que el ruido alcance un nivel excesivo hay que emplear un gran número de bits, de forma que a 44.1 Khz. y utilizando 16 bits para cuantizar la señal, uno de los dos canales de un CD produce más de 700 kilobits por segundo (kbps), gran parte de esta información es innecesaria y ocupa un ancho de banda que podría liberarse, aumentando así la complejidad del sistema decodificador e incurriendo en cierta pérdida de calidad.

Calidad	Muestreo	Bits/muestra	Modo	Tasa de bits	Frecuencia
Teléfono	8 Khz	8	Mono	64 Kbps	200-3400 Hz
Radio AM	11.025 Khz	8	Mono	88 Kbps	
Radio FM	22.050 Khz	16	Estéreo	705.6 Kbps	
CD	44.1 Khz	16	Estéreo	1411.2 Kbps	20 – 20000 Hz
DAT	48 Khz	16	Estéreo	1536 Kbps	20 – 20000 Hz

Tabla. 2.1. Comparación de formatos de calidad de audio

Para codificar la señal de mejor manera se utiliza un PCM no-lineal o cuantización logarítmica, que consiste en dividir el eje de la amplitud de tal forma que los escalones sean mayores cuanta más energía tiene la señal, con lo que se consigue una relación señal/ruido igual o mejor con menos bits. Con este método se puede reducir el canal de CD audio a 350 kbps, lo cual evidentemente es una mejora sustancial, aunque puede reducirse mucho más. Otros sistemas similares nos llevan a la cuantización adaptativa (APCM), diferencial (DPCM) y la mezcla de ambas, ADPCM.

2.2.3 Codificación sub-banda (SBC)

La codificación sub-banda o SBC (sub-band coding) codifica las señales de audio eficientemente. A diferencia de los métodos específicos para ciertas fuentes, el SBC puede codificar cualquier señal de audio sin importar su origen, ya sea voz, música o sonido de tipos variados. El estándar MPEG Audio es el ejemplo más popular de SBC.

El principio básico del SBC es la limitación del ancho de banda por descarte de información en frecuencias enmascaradas. El resultado simplemente no es el mismo que el original, pero si el proceso se realiza correctamente, el oído humano no percibe la diferencia.

La mayoría de los codificadores SBC utilizan el mismo esquema. Primero, uno o varios filtros, o algún otro mecanismo para descomponer la señal de entrada en varias subbandas. A continuación se determina los niveles de enmascaramiento en cada subbanda utilizando los datos psicoacústicos de que dispone. Considerando estos niveles de enmascaramiento se cuantizan y codifican las muestras de cada banda: si en una frecuencia dentro de la banda hay una componente por debajo de dicho nivel, se desecha. Si lo supera, se calculan los bits necesarios para cuantizarla y se codifica. Por último se agrupan los datos según el estándar correspondiente que estén utilizando el codificador y decodificador, de manera que el decodificador pueda descifrar los bits que le llegan del codificador y recomponer la señal.

La decodificación es mucho más sencilla, ya que no hay que aplicar ningún modelo psicoacústico. Simplemente se analizan los datos y se recomponen las bandas y sus muestras correspondientes.

2.3 MPEG AUDIO

2.3.1 El estándar MPEG audio

El estándar MPEG Audio contempla tres niveles diferentes de codificación-decodificación de la señal de audio, de los cuales sólo el primero está totalmente terminado. Los otros dos son aplicables, y de hecho se utilizan habitualmente, pero siguen abiertos a ampliaciones. Estos tres niveles son:

- MPEG-1: "Codificación de imágenes en movimiento y audio asociado para medios de almacenamiento digital hasta 1.5 Mbit/s"
- MPEG-2: "Codificación genérica de imágenes en movimiento e información de audio asociada". MPEG-3: la planificación original contemplaba su aplicación a sistemas HDTV; finalmente fue incluido dentro de MPEG-2.
- MPEG-4: "Codificación de objetos audiovisuales"

A su vez, MPEG describe tres capas de codificación de audio denominadas capa-1, capa-2 y capa-3. De la primera a la tercera aumentan tanto la complejidad del codificador como la calidad del sonido. Así, un decodificador capa-3 acepta los tres niveles de codificación, mientras el capa-2 sólo acepta el 1 y el 2.

MPEG define, para cada capa, el formato del flujo de bits (bitstream) y el decodificador. Debido a que es un sistema flexible no tiene un codificador definido. Hay que decir que tanto MPEG-1 como MPEG-2 emplean estas tres capas, pero esta última añade nuevas características como un modo especial de funcionamiento para tasas de bits especialmente bajas, utilizando frecuencias de muestreo hasta a la mitad que las habituales de 32, 44.1 y 48 Khz. y además una extensión multicanal.

2.3.2 Introducción al sistema MPEG-1

A continuación se presenta la descripción de la norma ISO (11172-3) en lo referente al sistema MPEG-1:

Codificación: el codificador procesa la señal de audio digital y produce el flujo de bits (bitstream) empaquetado para su almacenamiento y/o transmisión. El algoritmo de codificación no está determinado, y puede utilizar enmascaramiento, cuantización variable y escalado. Sin embargo, debe ajustarse a las especificaciones del decodificador.

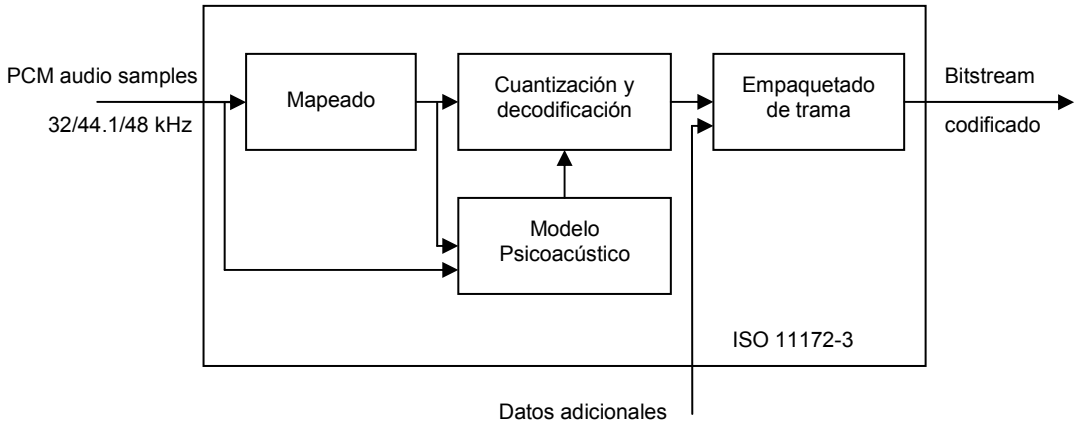


Figura. 2.1. Codificador según la norma ISO 11172-3

Las muestras se introducen en el codificador y a continuación el mapeador crea una representación filtrada y submuestreada de la señal de entrada. Las muestras mapeadas se denominan tanto muestras de subbanda (capas 1 y 2) como muestras de subbanda transformadas (capa 3). El modelo psicoacústico crea una serie de datos (dependiendo de la implementación del codificador) que sirven para controlar la cuantización y codificación. Este último bloque crea a su vez su propia serie de datos, de nuevo dependiendo de la implementación. Por último, el bloque de empaquetamiento de trama se encarga de agrupar como corresponde todos los datos, pudiendo añadir algunos más, llamados datos adicionales, como por ejemplo CRC o información del usuario.

Decodificación: el decodificador debe procesar el bitstream para reconstruir la señal de audio digital. La figura ilustra el esquema del decodificador.

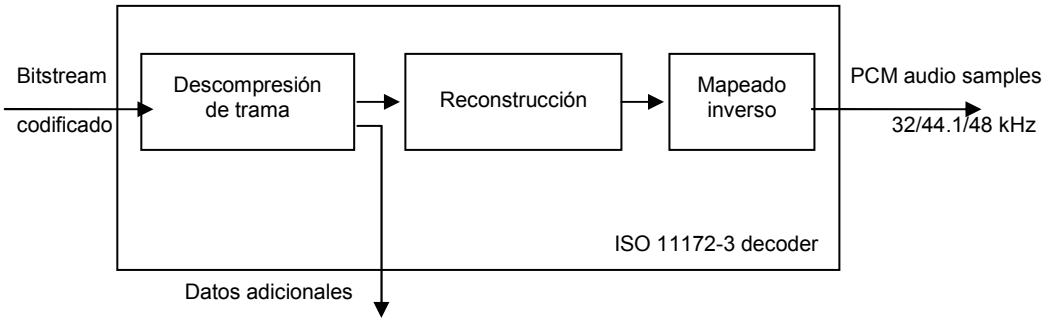


Figura. 2.2. Decodificador según la norma ISO 11172-3

Los datos del bitstream son desempaquetados para recuperar las diversas partes de la información. El bloque de reconstrucción recompone la versión cuantizada de la serie de muestras mapeadas. El mapeador inverso transforma estas muestras de nuevo a PCM.

A continuación se presenta una descripción de cada una de las Capas:

1. Incluye la división del mapeado básico de la señal de audio digital en 32 subbandas, segmentación para el formateo de los datos, modelo psicoacústico y cuantización fija. El retraso mínimo teórico es de 19 ms.

2. Incluye codificación adicional, factores de escala y diferente composición de trama. El retraso mínimo teórico es de 35 ms.

3. Incluye incremento de la resolución en frecuencia, basado en el uso de un banco de filtros híbrido. Cuantización no uniforme, segmentación adaptativa y codificación entrópica de los valores cuantizados. El retraso mínimo teórico es de 59 ms.

Capa	Tasa	Compresión	Calidad 64 kbps	Calidad128 kbps	Retardo
Capa-1	192 kbps	4 a 1			19 ms
Capa-2	128 kbps	6 a 1	2.1 a 2.6	Más de 4	35 ms
Capa-3	64 kbps	12 a 1	3.6 a 3.8	Más de 4	59 ms

Tabla. 2.2. Resumen de datos de las tres capas

La calidad viene dada del 1 al 5, siendo el 5 la superior. Hay que señalar que pese a los números de la norma ISO, el retraso típico acostumbra a ser tres veces mayor en la práctica.

Modos: hay cuatro modos de funcionamiento para cualquiera de estas tres capas y estos se enumeran a continuación:

- Single channel o canal único: una señal en un bitstream.
- Dual channel o canal doble: dos señales independientes en un mismo bitstream.
- Stereo: como el anterior, perteneciendo las señales al canal izquierdo y derecho de una señal estéreo original.
- Joint stereo: como el anterior, aprovechando ciertas características del estéreo como irrelevancia y redundancia de datos para reducir la tasa de bits.

CAPITULO III

FUNCIONAMIENTO DEL MÓDULO MP3

3.1 MÓDULO MP3

3.1.1 Descripción

El microcontrolador AT89C51SND1C posee un decodificador MPEG I/II capa 3. En el formato MPEG 1 (ISO 11172 – 3) se han estandarizado 3 capas de compresión para soportar 3 frecuencias de muestreo: 48, 44.1 y 32 Khz. De entre las tres capas, la capa 3 permite una alta tasa de compresión (en una relación de 12:1) sin perjudicar la calidad de sonido. Por ejemplo 3 minutos de audio en calidad CD, necesitaría alrededor de 32M bytes de memoria para su almacenamiento, esto puede ser codificado en solo 2.7M bytes en formato MP3. En el estándar MPEG II se adicionan tres frecuencias de muestreo: 24, 22.05, 16 Khz.

Este microcontrolador puede decodificar en tiempo real archivos codificados en MP3 a datos de audio PCM. Además posee herramientas para controlar volumen, bajos, medio, agudos y efecto reforzado de bajos (bass boost effect).

El decodificador MP3 se comunica con el microcontrolador con nueve registros de funciones especiales: MP3CON, Registro de Control de MP3; MP3STA, Registro de Estado del MP3; MP3DAT, Registro de Datos de MP3; MP3ANC, Registro de Ratos Auxiliares, MP3VOL y MP3VOR, Registros de Control de Volumen Izquierdo y Derecho; MP3BASS, MP3MED y MP3TRE, Registros de Control de Medios, Bajos y Agudos; y MP3CLK, Registro Divisor de Reloj del MP3.

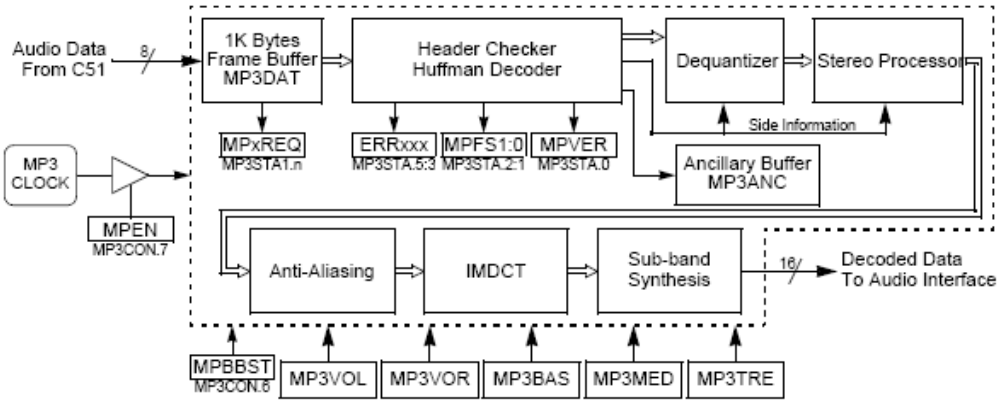


Figura. 3.1. Diagrama de bloques del decodificador MP3

3.1.2 Funcionamiento

3.1.2.1 Datos MP3.

El decodificador MP3 no empieza a decodificar ninguna trama de bits antes de tener una trama completa en el buffer de entrada (la primera petición de datos después de ser habilitado, consiste en 1024 Bytes de datos para llenar el buffer de entrada). Para el manejo de la carga de datos MP3 en el buffer de trama (frame buffer), se posee un protocolo de enlace de hardware (handshake), que consiste en una petición de datos y reconocimiento de datos.

Cada vez que el decodificador necesita datos MP3, éste activa las banderas MPREQ, MPFREQ y MPBREQ, en los registros MP3STA y MP3STA1 respectivamente. El CPU debe cargar la información en el buffer, a través de la escritura en el registro MP3DAT, después de reconocer la petición de datos. Como se muestra en la figura 3.2, la bandera MPFRQ se mantiene activa mientras los datos son requeridos por el decodificador. Esta es desactivada cuando no se requieren más datos y es vuelta a activar cuando nuevos datos son requeridos. La bandera MPBREQ se activa y desactiva cada vez que se escribe un byte.

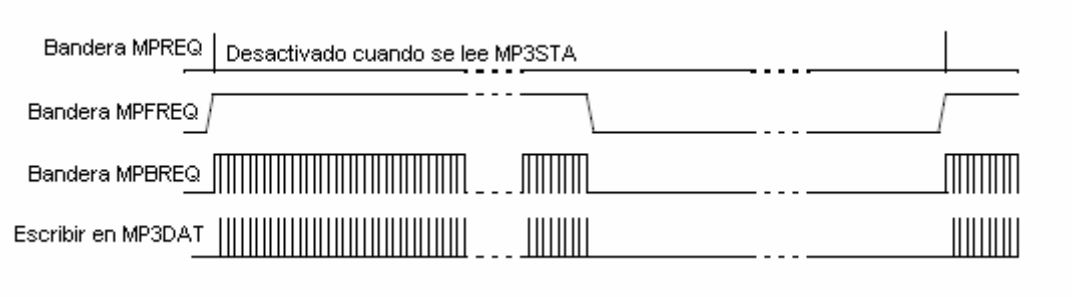


Figura. 3.2 Diagrama de sincronismo de datos

3.1.2.2 Reloj MP3.

El reloj del decodificador MP3 es generado por la división del reloj del PLL. El factor de división es dado por los bits MPCD 4:0, del registro MP3CLK. La frecuencia del reloj del decodificador MP3 solamente depende de la información de las tramas MP3 entrantes. La siguiente fórmula sirve para calcular el reloj del decodificador:

$$MP3CLK = \frac{PLLCLK}{MPCD4:0+1}$$

Tan pronto como la cabecera del archivo ha sido decodificada y extraída la versión MPEG, la frecuencia de entrada mínima del reloj MP3 debe ser programada de acuerdo a la siguiente tabla:

Versión MPEG	Reloj Mínimo de MP3 (MHz)
I	21
II	10.5

Tabla. 3.1. Frecuencia de reloj de MP3

3.1.3 Controles de audio

3.1.3.1 Control de volumen.

El decodificador MP3 implementa un controlador de volumen en ambos canales, derecho e izquierdo. Los registros MP3VOL y MP3VOR permiten un control de volumen de 32 pasos, de acuerdo a la siguiente tabla:

VOL4:0 o VOR4:0	Ganancia de volumen (dB)
00000	Silencio
00001	-33
00010	-27
11110	-1.5
11111	0 (volumen máximo)

Tabla. 3.2. Control de volumen

3.1.3.2 Control de ecualización.

El sonido puede ser ajustado usando tres bandas de ecualización: una banda de bajos hasta los 750Hz, una banda de medios entre los 750Hz y los 3300Hz y una banda de agudos sobre los 3300Hz. Los registros MP3BAS, MP3MED y MP3TRE, permiten un control de ganancia de 32 pasos en cada una de las bandas, de acuerdo a la siguiente tabla:

BAS4:0 , MED4:0 O TREB4:0	Ganancia (dB)
00000	-∞
00001	-14
00010	-10
11110	+1
11111	+1.5

Tabla. 3.3. Control de ecualización

3.1.3.3 Efectos especiales.

El bit MPBBST en el registro MP3CON permite habilitar un efecto de *Bass Boost*, con las siguientes características: incrementa la ganancia de 9 dB en las frecuencias bajo los 375Hz.

3.1.4 Errores de decodificación

Los tres diferentes errores que pueden aparecer durante el proceso de decodificación son detallados a continuación. Todos estos errores están en capacidad de activar una interrupción:

- Error de capa.- La bandera ERRSYN en el registro MP3STA se activa cuando se decodifica una capa que no puede ser soportada. Esta información puede ser encontrada en la cabecera del archivo que ha sido enviado al decodificador.
- Error de sincronización.- La bandera ERRSYN en el registro MP3STA se activa cuando no se encuentra un patrón de sincronización en los datos que tienen que ser enviados al decodificador.
- Error CRC.- Cuando el CRC de una trama no concuerda con la calculada, la bandera ERRCRC en el registro MP3STA es activada, en este caso, dependiendo del bit CRCEN en el registro MPCON la trama es reproducida o rechazada. En ambos casos aparecerá ruido en la salida de audio.

3.1.5 Información de la trama

La cabecera del archivo MP3, contiene información de los datos de audio. Esta información se hace disponible en el registro MP3STA para el conocimiento del programador. MPVER y los bits MPFS1:0, permiten la decodificación de la frecuencia de muestreo de acuerdo con la siguiente tabla. Adicionalmente el bit MPVER proporciona la versión de MPEG (I/II).

MPVER	MPFS1	MPFS0	F_S (kHz)
0	0	0	22.05 (MPEG II)
0	0	1	24 (MPEG II)
0	1	0	16 (MPEG II)
0	1	1	Reservado
1	0	0	44.1 (MPEG I)
1	0	1	48 (MPEG I)
1	1	0	32 (MPEG I)
1	1	1	Reservado

Tabla. 3.4. Frecuencia de muestreo de la trama MP3

3.2 INTERFASE DE AUDIO

3.2.1 Descripción

El microcontrolador At89c51snd1c contiene una interfase de salida de audio que permite una salida a modo de flujo de bits en varios formatos, estos son compatibles con los formatos PCM e I_2S y gracias al reloj interno PLL permite la conexión hacia la mayoría de conversores digital análogo (DAC) disponibles en el mercado.

El CPU se comunica con la interfase de audio mediante cinco registros de funciones especiales: AUDCON0 y AUDCON1, los registros de control de audio; AUDSTA, registro de estado de audio; AUDDAT, registro de datos de audio y AUDCLK, registro de divisor de reloj de audio.

3.2.2 Generador de reloj

El reloj de la interfase de audio es generado por la división de reloj de PLL, el factor de división es dado por los cuatro bits AUCD presentes en el registro CLKAUD, la frecuencia del reloj de la interfase de audio depende de las tramas provenientes del archivo MP3 y del conversor digital análogo utilizado, la siguiente fórmula es utilizada para calcular el reloj de la interfase de audio:

$$AUDCLK = \frac{PLLCLK}{AUCD4:0+1}$$

Tan pronto como es habilitada la interfase de audio, por la activación del bit AUDEN en el registro AUDCON1, el reloj maestro generado por el PLL es sacado por el pin SCLK, el cual va a ser el reloj del sistema del conversor análogo digital (en el caso de que se lo necesite). La salida de este reloj puede tener frecuencias de muestreo a 256 o 384 veces dependiendo de las capacidades del DAC. El bit HLR en el registro AUDCON0 debe ser activado de acuerdo a esta tasa para una generación adecuada del reloj para la sincronización de datos (en el pin DCLK) y para el reloj de selección (en el pin DSEL). Estos relojes no son generados cuando no hay datos disponibles en la entrada del conversor de datos.

Para la compatibilidad de DAC's, la frecuencia del reloj de sincronización puede ser programada para una salida de 16 o 32 bits por canal usando el bit DSIZ en el registro AUDCON0, y la señal de selección puede ser programada para una salida del canal izquierdo en nivel alto o bajo, de acuerdo al bit POL en el registro AUDCON0.

3.2.3 Conversor de datos

Este bloque convierte la entrada de flujo de datos de audio de un formato de 16 bits en paralelo a un formato serial. Para aceptar todos los formatos PCM e I_2S, se utilizan los bits JUST4:0 en el registro AUDCON0 para cambiar la forma de la salida de los datos. Como se muestra en la figura:

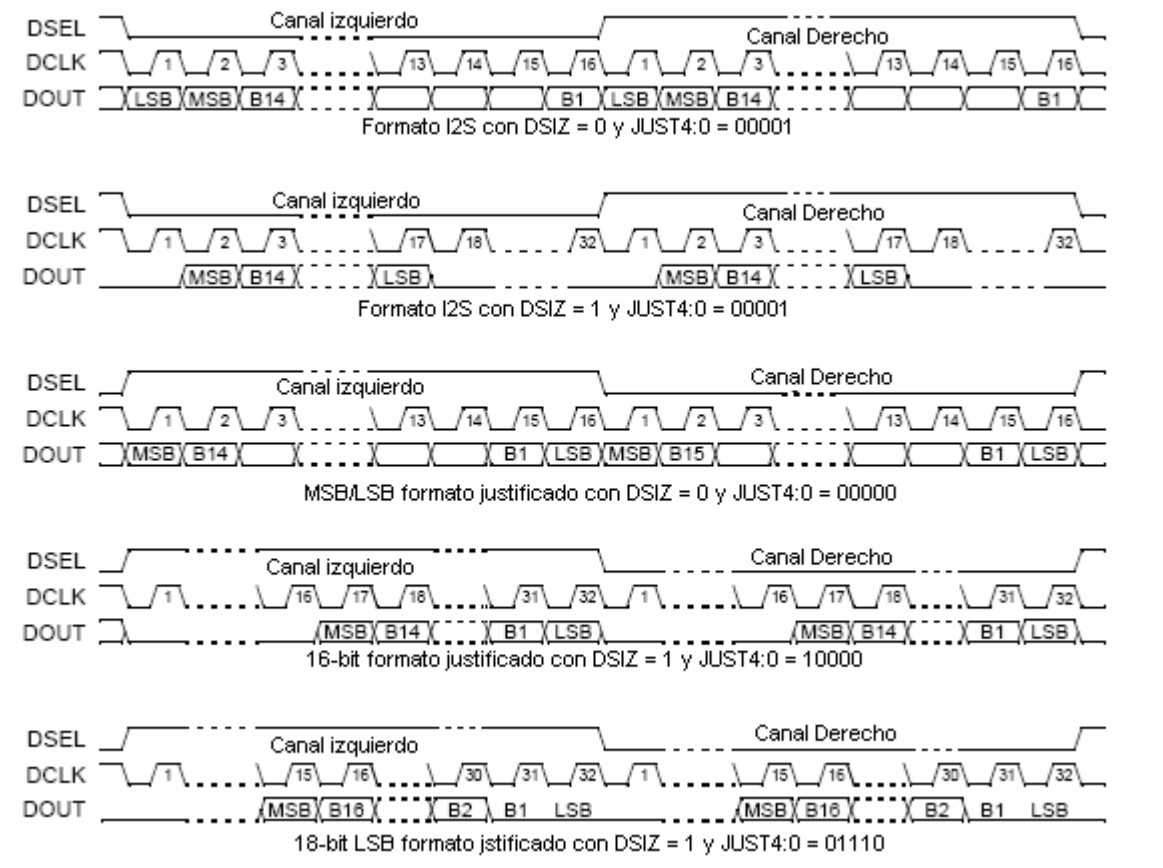


Figura. 3.3. Formato de salida de audio

El conversor de datos recibe el flujo de bits proveniente de dos fuentes que pueden ser seleccionadas por el bit SRC en el registro AUDCON1. Cuando este se encuentra desactivado el flujo de bits es proveniente del decodificador MP3 para reproducir una canción. Cuanto está activo, el flujo de bits es proveniente del CPU para reproducción de voz.

En cuanto el primer dato es ingresado al conversor, este habilita el generador de reloj obteniendo los relojes de sincronización y selección.

3.2.4 Buffer MP3

En el modo de reproducción de canción, el flujo de bits de audio es proveniente del decodificador de MP3 a través del buffer.

El buffer MP3 es usado para almacenar los datos decodificados. Está conectado al decodificador a través de una entrada de datos de 16 bits y una señal de petición de datos. Esta señal pide datos cuando el buffer tiene suficiente espacio para recibir nueva información. La petición de datos es condicionada por el bit DREQEN, en el registro AUDCON1. Cuando está activo, el buffer pide datos al decodificador MP3. Cuando está desactivado no se requieren más datos pero estos son sacados hasta que el buffer esté vacío. Este bit puede ser usado para suspender la generación de audio.

3.2.5 Reproducción canción MP3

En el modo de reproducción de canciones MP3, las operaciones a realizarse son la configuración del PLL y la interfase de audio acorde al DAC seleccionado. El reloj de audio es programado para generar una frecuencia de 256 x Fs o 384 x Fs, según la necesidad.

3.3 REGISTROS

Los registros que controlan la operación del módulo MP3 son:

7	6	5	4	3	2	1	0
MPEN	MPBBST	CRCEN	MSKANC	MSKREQ	MSKLAY	MSKSYN	MSKCRC

# de bit	Mnemónico	Descripción
7	MPEN	**Bit Habilitador Decodificador MP3** Con 1 se habilita el decodificador. Con 0 se deshabilita el decodificador
6	MPBBST	**Bit Bass Boost** Con 1 se habilita el efecto de sonido bass boost. Con 0 se deshabilita el efecto de sonido bass boost.
5	CRCEN	**CRC Check Enable Bit** Con 1 se habilita el procesamiento de tramas que contengan errores de CRC. La trama es reproducida sin importar el error. Con 0 se deshabilita el procesamiento de tramas que contengan errores de CRC. La trama es descartada.
4	MSKANC	**Bandera MPANC** Con 1 se impide a la bandera MPANC generar una interrupción de MP3. Con 0 se permite a la bandera MPANC generar un

		interrupción MP3.
3	MSKREQ	**Bandera MPREQ** Con 1 se impide a la bandera MPREQ generar una interrupción de MP3. Con 0 se permite a la bandera MPREQ generar un interrupción MP3.
2	MSKLAY	**Bandera ERRLAY** Con 1 se impide a la bandera ERRLAY generar una interrupción de MP3. Con 0 se permite a la bandera ERRLAY generar un interrupción MP3.
1	MSKSYN	**Bandera ERRSYN** Con 1 se impide a la bandera ERRSYN generar una interrupción de MP3. Con 0 se permite a la bandera ERRSYN generar un interrupción MP3.
0	MSKCRC	**Bandera ERRCRC** Con 1 se impide a la bandera ERRCRC generar una interrupción de MP3. Con 0 se permite a la bandera ERRCRC generar un interrupción MP3.

Tabla. 3.5. Registro MP3CON

7	6	5	4	3	2	1	0
MPANC	MPREQ	ERRLAY	ERRSYN	ERRCRC	MPFS1	MPFS0	MPVER

# de bit	Mnemónico	Descripción
7	MPANC	**Bandera de dato auxiliar disponible** Se activa vía hardware tan pronto como un dato auxiliar está disponible (el buffer no está vacío). Se desactiva vía hardware cuando no hay más datos auxiliares disponibles (el buffer está vacío).
6	MPREQ	**Bandera de petición de datos MP3** Se activa vía hardware cuando el decodificador MP3 pide datos Se desactiva cuando se lee el registro MP3STA.
5	ERRLAY	**Bandera de error de capa inválida** Se activa vía hardware cuando una capa inválida ha sido encontrada Se desactiva cuando se lee el registro MP3STA.
4	ERRSYN	**Bandera de error de sincronización de trama** Se activa vía hardware cuando no se encuentra un patrón de sincronización. Se desactiva cuando se lee el registro MP3STA.
3	ERRCRC	**Bandera de error CRC** Se activa vía hardware cuando una trama contiene un error CRC. Se desactiva cuando se lee el registro MP3STA.

1 - 2	MPFS1:0	**Bits de frecuencia de muestreo** Véase Tabla 3.4.
0	MPVER	**Bit de versión MPEG** Se activa por el decodificador MP3 cuando es cargada una trama MPEG I. Se desactiva por el decodificador MP3 cuando es cargada una trama MPEG II.

Tabla. 3.6. Registro MP3STA

7	6	5	4	3	2	1	0
MPD7	MPD6	MPD5	MPD4	MPD3	MPD2	MPD1	MPD0

# de bit	Mnemónico	Descripción
7 - 0	MPD7:0	**Buffer de entrada de flujo de datos** 8 Bits de datos MP3

Tabla. 3.7. Registro MP3DAT

7	6	5	4	3	2	1	0
-	-	-	MPFREQ	MPBREQ	-	-	-

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** EL valor leído de este bit es siempre 0
4	MPFREQ	**Bandera de petición de trama de datos MP3** Se activa vía hardware cuando el decodificador MP3 pide datos. Se desactiva cuando el decodificador MP3 deja de pedir datos.
3	MPBREQ	**Bandera de petición de Bytes de datos MP3** Se activa vía hardware cuando el decodificador MP3 pide datos. Se desactiva cuando el registro MP3DAT carga datos.
2 - 0	-	**Reservado** EL valor leído de este bit es siempre 0

Tabla. 3.8. Registro MP3STA1

7	6	5	4	3	2	1	0
AND7	AND6	AND5	AND4	AND3	AND2	AND1	AND0

# de bit	Mnemónico	Descripción
7 - 0	AND7:0	**Buffer de datos auxiliares** 8 Bits de datos auxiliares MP3

Tabla. 3.9. Registro MP3ANC

7	6	5	4	3	2	1	0
-	-	-	VOL4	VOL3	VOL2	VOL1	VOL0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** EL valor leído de este bit es siempre 0
4 - 3	VOL4:0	**Valor de volumen izquierdo** Véase Tabla 3.2. referido al control de volumen izquierdo

Tabla. 3.10. Registro MP3VOL

7	6	5	4	3	2	1	0
-	-	-	VOR4	VOR3	VOR2	VOR1	VOR0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** EL valor leído de este bit es siempre 0
4 - 3	VOR4:0	**Valor de volumen izquierdo** Véase Tabla 3.2. referido al control de volumen derecho

Tabla. 3.11. Registro MP3VOR

7	6	5	4	3	2	1	0
-	-	-	BAS4	BAS3	BAS2	BAS1	BAS0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** El valor leído de estos bits es siempre 0.
4 - 0	BAS4:0	**Valor de Ganancia de Bajos** Revisar la tabla 3.3 para la configuración de bajos.

Tabla. 3.12. Registro MP3BAS

7	6	5	4	3	2	1	0
-	-	-	MED4	MED3	MED2	MED1	MED0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** El valor leído de estos bits es siempre 0.
4 - 0	MED4:0	**Valor de Ganancia de Medios** Revisar la tabla 3.3 para la configuración de medios.

Tabla. 3.13. Registro MP3MED

7	6	5	4	3	2	1	0
-	-	-	TRE4	TRE3	TRE2	TRE1	TRE0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** El valor leído de estos bits es siempre 0.
4 - 0	TRE4:0	**Valor de Ganancia de Agudos** Revisar la tabla 3.3 para la configuración de agudos.

Tabla. 3.14. Registro MP3TRE

7	6	5	4	3	2	1	0
-	-	-	MPCD4	MPCD3	MPCD2	MPCD1	MPCD0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** El valor leído de estos bits es siempre 0.
4 - 0	MPCD4:0	**Divisor de Reloj del Decodificador de MP3** Divisor de 5 bits para la generación del reloj del decodificador de MP3.

Tabla. 3.15. Registro MP3CLK

7	6	5	4	3	2	1	0
JUST4	JUST3	JUST2	JUST1	JUST0	POL	DSIZ	HLR

# de bit	Mnemónico	Descripción
7 - 3	JUST4:0	**Justificación del Flujo de bits de Audio** Revisar la Figura 3.2 para configurar la salida de los bits según el DAC a utilizarse.
2	POL	**Polaridad de la Señal de Salida** Con 1 la señal del canal izquierdo se encuentra en nivel alto (Modo PCM). Con 0 se la señal del canal izquierdo se encuentra en nivel bajo (Modo I^2S).

1	DSIZ	**Tamaño de los Datos de Audio** Con 1 el formato de salida de los datos es 32 bits Con 0 el formato de salida de los datos es 16 bits
0	HLR	**Tasa de Bits** Con 1 desde software cuando la frecuencia del reloj de PLL es 384 · Fs Con 0 desde software cuando la frecuencia del reloj de PLL es 256 · Fs

Tabla. 3.16. Registro AUDCON0

7	6	5	4	3	2	1	0
SRC	DRQEN	MSREQ	MUDRN	-	DUP1	DUP0	AUDEN

# de bit	Mnemónico	Descripción
7	SRC	**Bit de Origen de Audio** Con 1 la fuente de audio es de voz o sonidos. Con 0 la fuente de audio es la salida del decodificador MP3 para reproducir canciones.
6	DRQEN	**Bit Habilitador de Petición de Datos MP3 Decodificados** Con 1 se habilita la petición de datos decodificados y empieza la reproducción de la canción. Con 0 se deshabilita la petición de datos.
5	MSREQ	**Mascara para la Bandera de Petición de Muestra de Audio** Con 1 se impide la activación de la bandera SREQ para la interrupción de audio. Con 0 se permite la activación de la bandera SREQ para la interrupción de audio.
4	MUDRN	**Máscara para la Bandera de Puesta en Marcha de Muestra de Audio** Con 1 se impide la activación de la bandera UDRN para la interrupción de audio. Con 0 se permite la activación de la bandera UDRN para la interrupción de audio.
3	-	**Reservado** El valor leído de estos bits es siempre 0.
2	DUP1:0	**Factor de Duplicación de Audio.** Se utilizan estos bits cuando el origen del audio proviene del microcontrolador, es decir, son archivos de voz.
0	AUDEN	**Bit Habilitador de la Interfase de Audio** Con 1 se habilita la interfase de audio. Con 0 se deshabilita la interfase de audio.

Tabla. 3.17. Registro AUDCON1

7	6	5	4	3	2	1	0
AUD7	AUD6	AUD5	AUD4	AUD3	AUD2	AUD1	AUD0

# de bit	Mnemónico	Descripción
7 - 0	AUD7:0	**Datos de Audio** Datos de 8 bits para reproducción de voz

Tabla. 3.18. Registro AUDDAT

7	6	5	4	3	2	1	0
-	-	-	AUCD4	AUCD3	AUCD2	AUCD1	AUCD0

# de bit	Mnemónico	Descripción
7 - 5	-	**Reservado** El valor leído de estos bits es siempre 0.
4 - 0	AUD7:0	**Divisor del Reloj de Audio** Divisor de 5 bits para la generación del reloj de Audio.

Tabla. 3.19. Registro AUDCLK

CAPITULO IV

SOFTWARE Y PROGRAMACIÓN

4.1 KEIL µVision2 IDE

4.1.1 Introducción

Existen actualmente una gran variedad de entornos de desarrollo de programas para la familia de microcontroladores 8051, y uno de los más utilizados por su facilidad es el *PK51 Professional Developer's Kit* de la empresa Keil Software. Este entorno integra programas que se emplean para compilar el código C, ensamblar los archivos fuentes de ensamblador, enlazar y colocar (link / locate) módulos y librerías, crear archivos hexadecimales (.HEX), y simular el programa compilado. Las herramientas que componen este entorno son las siguientes:

- µVision2 IDE
- A51 Macro Assembler
- C51 ANSI C Compiler
- BL51 Code Banking Linker/Locator
- OH51 Object-HEX Converter
- OC51 Banked Object Converter
- RTX51 Tiny Real-time Kernel
- µVision2 Debugger
- MON51 Target Monitor
- ISD51 In-System Debugger

Cuando se utiliza el entorno de Keil, el ciclo de desarrollo del proyecto es similar al utilizado para cualquier otro proyecto de desarrollo de software, donde los pasos a seguirse se muestran a continuación:

1.- Crear un proyecto, seleccionar el microcontrolador para el cual se va a desarrollar el programa y configurar las opciones adecuadas al sistema sobre el que se ejecutará el programa.
2.- Crear los archivos fuente en lenguaje C (*.C) o en lenguaje ensamblador (*.A51).
3.- Construir la aplicación, compilando los archivos *.C, ensamblando los archivos *.A51 y enlazado todos los archivos objetos en un único ejecutable.
4.- Corregir los errores en los archivos fuente.
5.- Depurar y comprobar el funcionamiento de la aplicación desarrollada.

En la figura siguiente se muestra el diagrama de bloques del proceso a seguir con las herramientas a utilizar en cada caso. Cabe destacar que todos los pasos necesarios se realizan dentro del entorno µVision2.

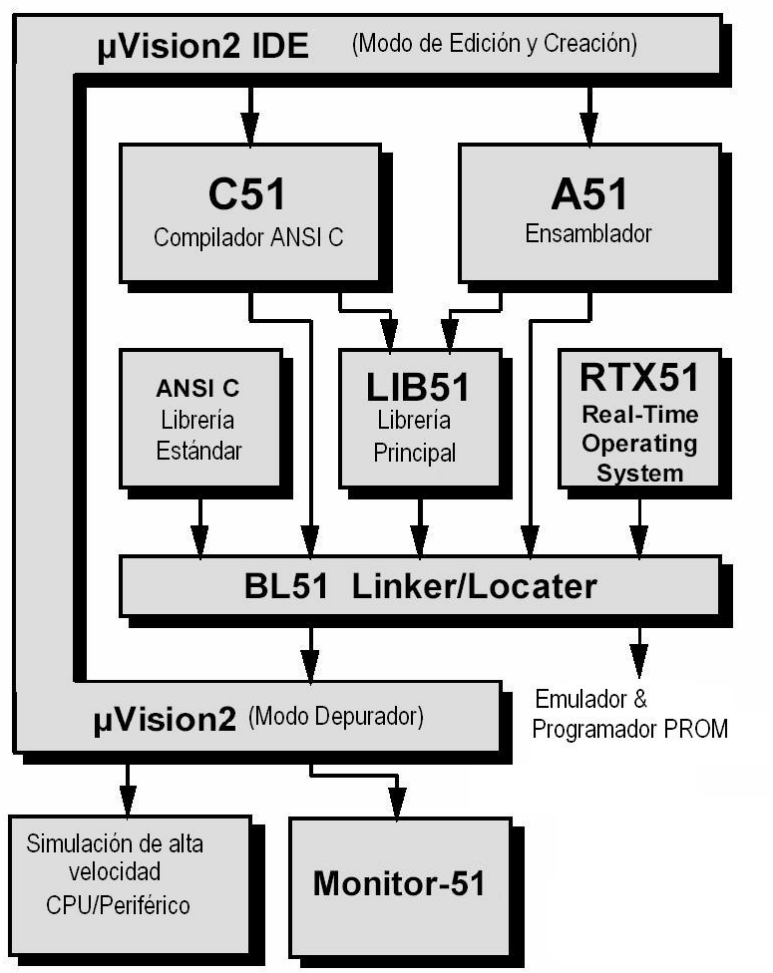


Figura. 4.1. Diagrama de Bloques del Ciclo de Desarrollo del Proyecto

Cada uno de los componentes del diagrama de bloques se describe a continuación:

µVision2 IDE (Entorno de Desarrollo Integrado): µVision2 sirve para la creación de archivos fuente y organizarlos en un proyecto que define la aplicación deseada. Este entorno automáticamente compila, ensambla, y enlaza esta aplicación. µVision2 combina herramientas para el manejo del proyecto (project management) y un importante editor con un corrector interactivo de errores, opciones de configuración y ayuda en línea.

Compilador C51 y Ensamblador A51: Los archivos son creados por µVision2 IDE y son pasados al compilador C51 o al ensamblador A51. El compilador y el ensamblador procesan los archivos fuente y crean archivos tipo objeto.

El compilador de Keil C51 es una implementación completa del lenguaje de programación C que soporta todas las características estándar de este lenguaje, adicionalmente se han agregado numerosas características para el apoyo directo de la arquitectura 8051.
El ensamblador A51 soporta el grupo completo de instrucciones para la familia 8051 y todos los derivados.

Librería Principal LIB51: Permite crear objetos tipo librería desde un archivo tipo objeto creado por el compilador o ensamblador. Las librerías son especialmente estructuradas y ordenadas con objetos módulo que pueden ser usados más adelante por el linker. Cuando el linker procesa una librería, solo utiliza los objetos módulo en la librería que son necesarios para crear el programa.

BL51 Linker/Locator: El BL51 Linker crea un objeto módulo absoluto usando los objetos módulo extraídos desde las librerías y los que son creados por el compilador y el ensamblador. Todo el código y los datos residen en direcciones fijas de memoria. Los archivos absolutos pueden ser usados:

- La programación de memorias EPROM u otros dispositivos de memoria.
- La simulación y depuración con el Depurador de µVision2
- La comprobación del programa con un emulador de un circuito

Depurador µVision2: El depurador incluye un simulador de alta velocidad que permite simular todo el entorno del sistema 8051 que incluye puertos y hardware externo. Los atributos de cada chip son automáticamente configurados cuando este es incluido desde la base de datos de dispositivos.
El depurador µVision2 presta varias formas de probar los programas para hardware real.

Monitor - 51: Monitor-51 es una herramienta adicional que para ser utilizada debe ser instalada adicionalmente. El programa monitor reside en la memoria del

hardware utilizado y se comunica con el simulador utilizando el puerto serial del 8051 y el puerto COM del PC.

RTX51 Real-Time Operating System: Esta herramienta solo se la puede utilizar cuando es instalado el paquete completo que proporciona el proveedor. Esta herramienta simplifica el diseño del sistema, la programación, y la depuración de aplicaciones complejas donde se requiere una rápida respuesta a situaciones críticas. El RTX51 esta íntimamente vinculado con el compilador C51.

4.1.2 Instalación

Los requerimientos mínimos tanto de hardware como software que se debe cumplir para que Keil µVision2 funcione adecuadamente son los siguientes:

- PC con Pentium, Pentium-II o un procesador compatible.
- Windows 95, Windows-98, Windows NT 4.0, o superior,
- 16 MB RAM mínimo,
- 20 MB de espacio libre en el disco.

Todos los productos Keil están provistos con un programa de instalación que permite una fácil instalación. Para la instalación se siguen los siguientes pasos:

- Insertar el CD-ROM de Keil Development Tools
- La PC debe automáticamente correr el CD Viewer cuando inserta el disco. Si es que no lo hizo debe correr el siguiente programa de su disco : \KEIL\SETUP\SETUP.EXE
- Seleccione *"Install Software"* (instalar software) del menú de instalación,
- Siga las instrucciones desplegadas por el programa de Configuración (Setup).

El programa de instalación copia todas las herramientas en sub – carpetas de la carpeta base, siendo esta C:\KEIL. A continuación se muestra una lista de las herramientas cuando se ha instalado el software completo:

C:\KEIL\C51\ASM	Archivos de definición del ensamblador y archivo fuente de plantilla.
C:\KEIL\C51\BIN	Archivos ejecutables de las herramientas para 8051
C:\KEIL\C51\EXAMPLES	Aplicaciones simples
C:\KEIL\C51\RTX51	RTX51 archivos completos
C:\KEIL\C51\RTX_TINY	RTX51 archivos muestra
C:\KEIL\C51\INC	Archivos “include” del compilador C
C:\KEIL\C51\LIB	Librerías, código de inicio y rutinas de entrada y salida.
C:\KEIL\C51\MONITOR	Archivos del Monitor y la configuración del Monitor para el hardware utilizado.
C:\KEIL\UV2	Archivos genéricos de µVision2

Tabla. 4.1. Herramientas de Keil uVision

4.1.3 Entorno de Keil µVision2, Descripción General

Keil µVision2 es una plataforma de desarrollo de software basado en Windows que combina un robusto editor de programas y herramientas para el manejo y administración del proyecto. µVision2 soporta todas las herramientas para la familia de microcontroladores 8051 anteriormente citadas y ayuda a acelerar el proceso de desarrollo de las aplicaciones.

La pantalla de µVision2 le proporciona una barra de menú para la entrada de comandos, una barra de herramientas donde se puede seleccionar rápidamente los botones de comando y ventanas de archivos fuente. µVision2 le permite abrir y ver simultáneamente múltiples archivos fuente.

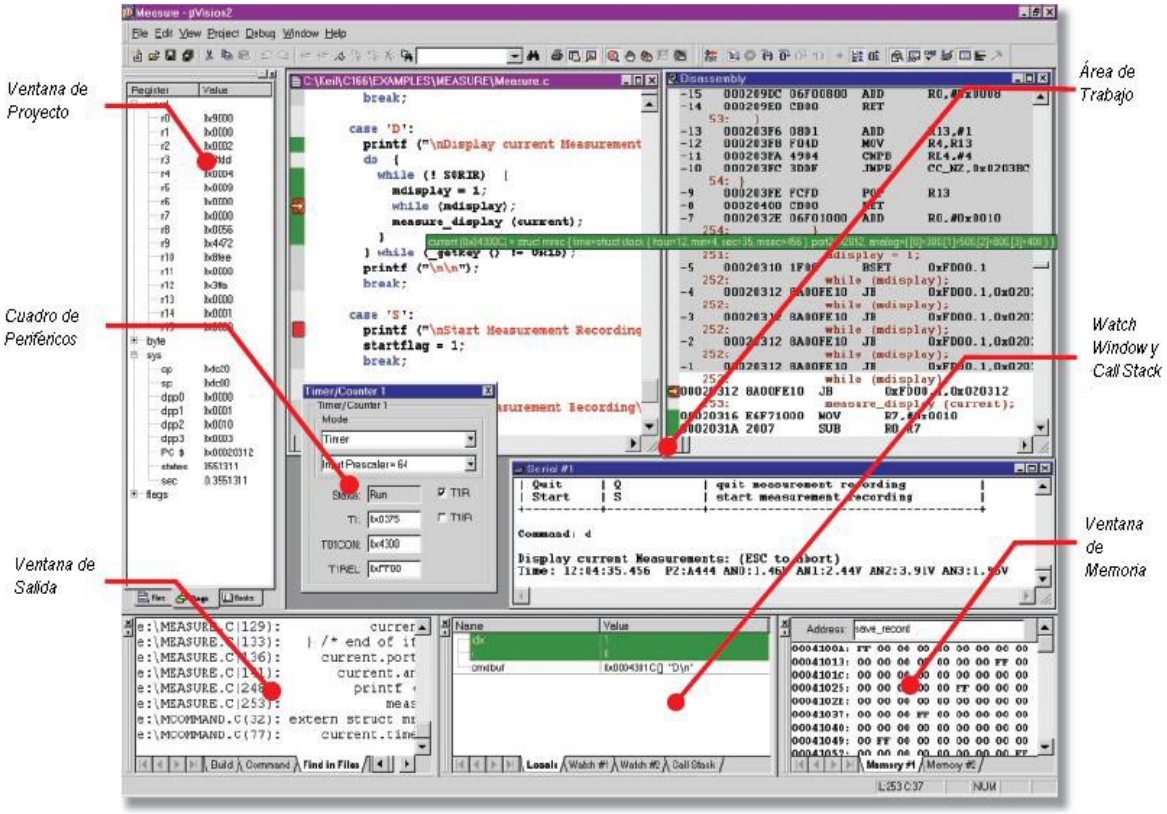


Figura. 4.2. Pantalla Principal de Keil µVision2

Para hacer sencillo el manejo y administración del proyecto, µVision2 incluye una gran variedad de características como por ejemplo:

- Grupos de Archivos que permiten asociar diferentes archivos en un solo proyecto. Esto es útil para agrupar archivos en los bloques funcionales o para identificar a los ingenieros en su equipo de software.
- Project Target permite crear varios programas de un solo proyecto. Se puede necesitar un target para realizar pruebas y otro target para descargar a la aplicación. Cada target permite el uso de herramientas individuales dentro del mismo proyecto.
- El menú Project proporciona acceso a las herramientas para administración del proyecto como :
    - New Project que permite crear un proyecto
    - Targets, Groups, Files que añaden componentes al proyecto. Los menús Locales en la ventana de proyecto (Project window) permite añadir archivos al proyecto. Un target se conoce como un programa

ejecutable generado en un proyecto y un grupo es un conjunto de archivos asociados en un solo proyecto.

- Open Project que permite abrir proyectos existentes.

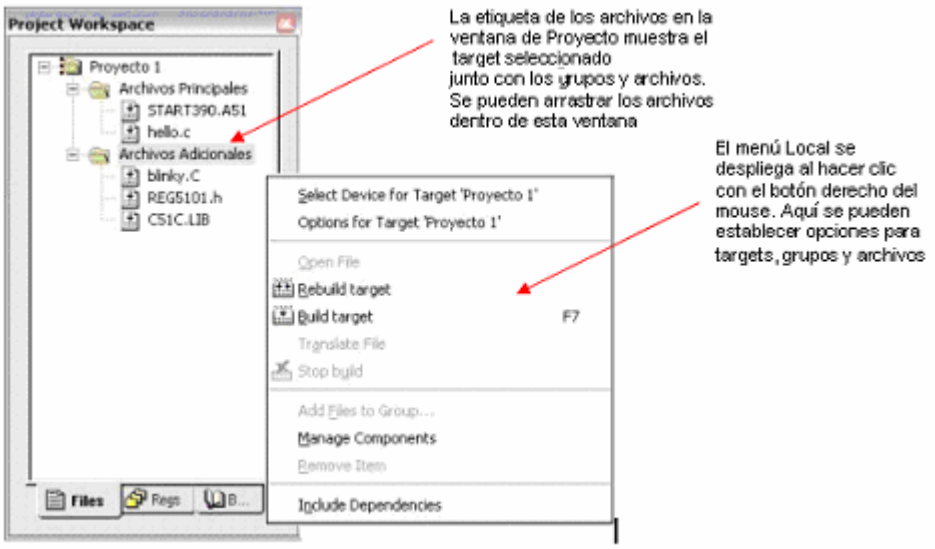


Figura. 4.3. Ventana de Proyecto

La base de datos de dispositivos (Device Database) facilita comenzar la escritura de programas para un CPU en particular. Simplemente hay que seleccionar el microcontrolador que se va a usar y µVision2 configura las opciones automáticamente. Si se requiere se pueden agregar nuevos dispositivos a la base de datos. El listado de dispositivos contiene una gran cantidad de fabricantes de microcontroladores que poseen un núcleo de la familia del 8051, tal como se muestra en la Figura 4.4.

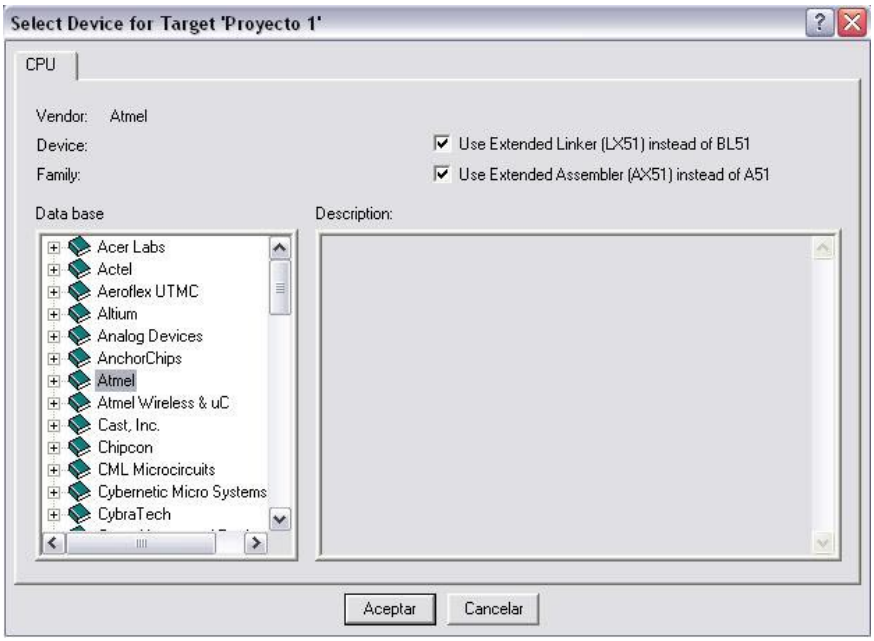


Figura. 4.4. Base de Datos de Dispositivos

µVision2 le permite poner las opciones para todos los archivos en un target, en un grupo o incluso en un solo archivo fuente. La ventana de Opciones se abre con el menú local en la ventana de proyecto o mediante el icono en la barra de construcción, tal como se muestra en la Figura 4.5. En la pestaña de Target, se especifican los parámetros del CPU y de memoria para el sistema. µVision2 usa esta información para configurar las opciones básicas de las herramientas incluyendo las opciones del linker/locater y del simulador.

Figura. 4.5. Barra de Construcción de Proyecto

La pestaña de Salida (Output) define los archivos de salida generados por el ensamblador, el compilador y el linker. La pestaña de Listing configura los archivos tipo *.lst. Las pestañas C51, AX51, LX51 permiten ingresar opciones específicas de la herramienta (como por ejemplo #defines) y muestra las opciones actuales. La pestaña de Depuración (Debug) permite configurar las opciones del depurador de µVision2.

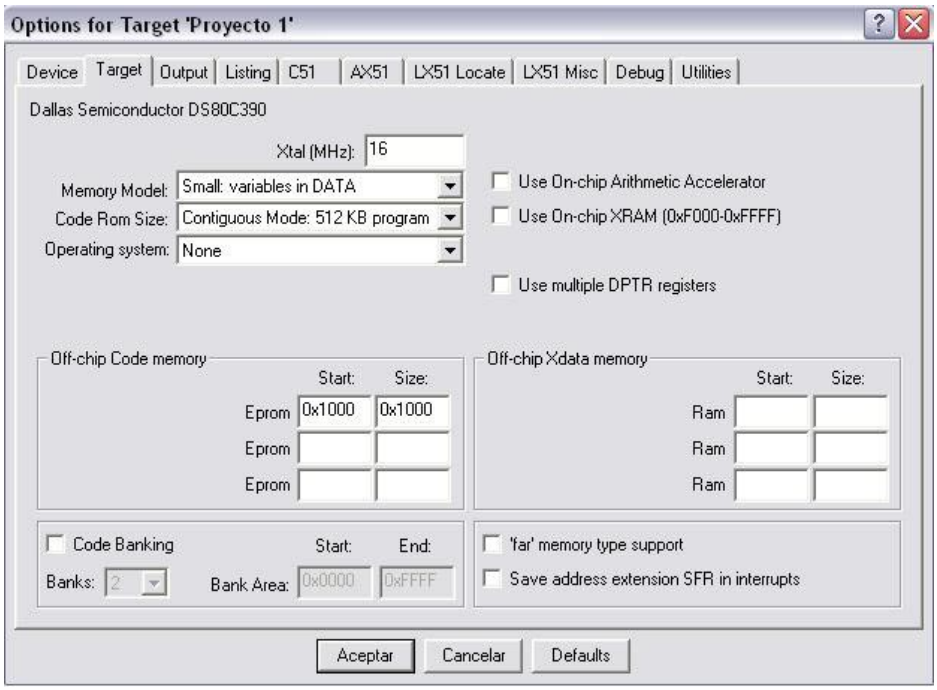


Figura. 4.6. Opciones del Target

Las opciones de edición están compuestas por todas las características que tienen la mayoría de entornos de desarrollos (IDEs). La sintaxis colorida y la sangría, resaltan el texto lo que facilita la revisión del código fuente en C.

Más funciones del editor pueden accederse rápidamente desde la barra de herramientas (Figura 4.7). El editor está disponible mientras se trabaja en modo de simulación. Esto da un ambiente natural de depuración que permite rápidamente corregir el código fuente.

Figura. 4.7. Barra de Edición

µVision2 marca el estado de la línea fuente en la columna de la ventana del editor. Esto proporciona una rápida apreciación global de los actuales puntos de ruptura. Haciendo doble clic en la página de Archivos (Files) de la Ventana del Proyecto se abre un archivo fuente seleccionado. Habilitando el Workbook Mode

en el menú View se puede seleccionar rápidamente cualquier documento del proyecto.

Es posible poner puntos de ruptura de programa (program breakpoints) mientras se este escribiendo el código. Simplemente se debe usar los botones de la barra de Edición para marcar los puntos de ruptura en las líneas deseadas. Cuando se corre el programa en el depurador, los breakpoints que se pusieron estarán activos. µVision2 marca el estado de la línea en la columna de Atributos de la ventana de edición. Esto proporciona una apreciación global rápida de los breakpoints actuales.

Un punto de ruptura situado en una línea de código se indica mediante un cuadradito a la izquierda de la ventana del fichero; dicho cuadrado es de color rojo cuando el punto de ruptura está habilitado y es de color blanco cuando está deshabilitado. Además de los puntos de ruptura, a la izquierda de cada ventana de código aparecen otros símbolos para señalar que la línea ha sido marcada (Bookmark), valor actual del contador de programa (Current PC, next statement), etc. Todo esto se puede apreciar en la Figura 4.8.

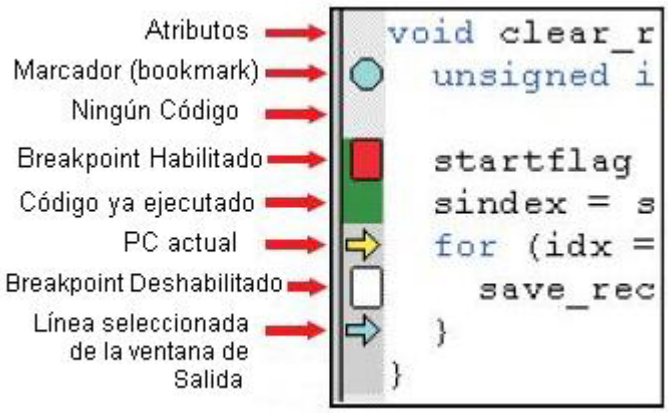


Figura. 4.8. Estado de la Línea en la ventana del editor

Adicionalmente de las herramientas estándares de edición, µVision2 presenta algunas otras herramientas poderosas, como por ejemplo “Find in Files” que

permite una búsqueda de un texto, función o variable en todos los archivos especificados. El resultado de la búsqueda es mostrado en la ventana de salida.

4.1.4 Menú de comandos, Barras de herramientas y Accesos Directos

A continuación se presenta un listado de todos los comandos que posee el Entorno de Desarrollo µVision2 con una breve descripción de los mismos.

Menú y Comandos de Archivo (File)

Menú de Archivo	Barra de Herramientas	Acceso Directo	Descripción
New		Ctrl. + N	Crea un Nuevo archivo fuente o de texto
Open		Ctrl. + O	Abre un archivo existente
Close			Cierra el archivo actual
Save		Ctrl. + S	Graba el archivo de fuente o texto actual
			Graba todos los archivos fuente o de texto
Save as…			Graba y renombra el archivo actual
Device Database			Guarda los dispositivos en la base de datos
Print Setup			Configuración de página para imprimir
Print		Ctrl. + P	Imprime el archivo actual
Print Preview			Muestra los archivos en presentación preliminar antes de imprimir
1 - 9			Abre los archivos fuente o de texto más actuales
Exit			Sale de µVision2 y sugiere guardar los archivos

Tabla. 4.2. Menú y Comandos de Archivo (File)

Menú y Comandos de Edición (Edit)

Menú de Edición	Barra de Herramientas	Acceso Directo	Descripción
		Inicio	Mueve el cursor al inicio de la línea
		Fin	Mueve el cursor al fin de la línea
		Ctrl. + Inicio	Mueve el cursor al inicio del archivo
		Ctrl. + Fin	Mueve el cursor al fin del archivo
Undo		Ctrl. + Z	Deshace la última operación
Redo		Ctrl + Shift + Z	Rehace la ultima operación de Undo
Cut		Ctrl. + X	Corta el texto seleccionado al portapapeles
		Ctrl. + Y	Corta el texto en la línea actual al portapapeles
Copy		Ctrl. + C	Copia el texto seleccionado al portapapeles
Paste		Ctrl. + V	Pega texto desde el portapapeles
Indent Selected Text			Asigna sangría al texto seleccionado hacia la derecha
Unindent Selected Text			Asigna sangría al texto seleccionado hacia la izquierda
Toggle		Ctrl. + F2	Habilita/Deshabilita un marcador a la línea actual

Bookmark			
Goto Next Bookmark		F2	Mueve el cursor al siguiente marcador (bookmark)
Goto Previous Bookmark		Shift + F2	Mueve el cursor al anterior marcador (bookmark)
Clear All Bookmarks			Elimina todos los marcadores del archivo actual
Find		Ctrl. + F	Busca un texto en el archivo actual
		F3	Repite la búsqueda del texto hacia adelante
		Shift + F3	Repite la búsqueda del texto hacia atrás
Replace		Ctrl. + H	Reemplaza un texto específico
Find in Files…			Busca un texto en todos los archivos del proyecto

Tabla. 4.3. Menú y Comandos de Edición (Edit)

Menú y Comandos de Vista (View)

Menú de Vista	Barra de Herramientas	Acceso Directo	Descripción
Status Bar			Muestra o oculta la barra de estado
File Toolbar			Muestra o oculta la barra de herramientas de Archivo
Build Toolbar			Muestra o oculta la barra de herramientas de Construcción de Proyecto
Debug Toolbar			Muestra o oculta la barra de herramientas de Depuración
Project Window			Muestra o oculta la ventana de Proyecto
Output Window			Muestra o oculta la ventana de Salida
Source Browser			Abre la ventana del buscador de origen
Disassembly Window			Muestra o oculta la ventana del desensamblador (código en ensamblador).
Watch & Call Stack Window			Muestra o oculta la ventana de Watch & Call Snack (Pila)
Memory Window			Muestra o oculta la ventana de Memoria
Code Coverage Window			Muestra o oculta la ventana de Respaldo de Código
Performance Analyzer Window			Muestra o oculta la ventana del Analizador de Desempeño
Symbol Window			Muestra o oculta la ventana de Símbolos
Serial Window #1			Muestra o oculta la ventana de ingreso o salida por Serial # 1
Serial Window #2			Muestra o oculta la ventana de ingreso o salida por Serial # 2
Toolbox			Muestra o oculta la caja de herramientas
Periodic Window Update			Actualiza la ventana de depuración periódicamente mientras corre el programa
Workbook			Muestra las ventanas en modo Woorkbook

Mode			
Options…			Cambia colores, fuentes, accesos directos, etc

Tabla. 4.4. Menú y Comandos de Vista (View)

Menú y Comandos de Proyecto (Project)

Menú de Proyecto	Barra de Herramientas	Acceso Directo	Descripción
New Project …			Crea un nuevo proyecto
Import µVision1 Project …			Convierte un proyecto de µVision1
Open Project …			Abre un proyecto existente
Close Project…			Cierra el proyecto actual
Target Environment			Define las rutas para archivos de librería e include
Targets, Groups, Files			Guarda los Targets, grupos y archivos de un proyecto
Select Device for Target			Selecciona un CPU desde la base de datos de dispositivos
Remove…			Quita un archivo o un grupo del proyecto
Options…		Alt + F7	Cambia las opciones de herramientas para el Target, grupo o archivo
			Cambia las opciones para el target actual
			Selección del Target o aplicación actual
Build Target		F7	Traduce los archivos modificados y crea la aplicación
Rebuild Target			Traduce todos los archivos y crea la aplicación
Translate…		Ctrl. + F7	Traduce el archivo actual
Stop Build			Detiene el proceso de construcción
1-9			Abre los proyectos últimamente usados.

Tabla. 4.5. Menú y Comandos de Proyecto (Project)

Menú y Comandos de Depuración (Debug)

Menú de Depuración	Barra de Herramientas	Acceso Directo	Descripción
Start/Stop Debugging		Ctrl. + F5	Empieza o detiene el modo de Depuración
Go		F5	Ejecuta hasta antes del próximo marcador
Step		F11	Ejecuta una instrucción dentro de la función
Step over		F10	Ejecuta una instrucción fuera de la función
Step out of current function		Ctrl. + F11	Ejecuta fuera de la función actual
Stop Running		ESC	Detiene la ejecución del programa

Breakpoints			Abre ventana de marcadores (breakpoints)
Insert / Remove Breakpoint			Inserta o remueve un breakpoint en la línea actual
Enable / Disable Breakpoint			Activa o desactiva el breakpoint de la línea actual
Disable All Breakpoints			Desactiva todos los breakpoints en el programa
Kill All Breakpoints			Remueve todos los breakpoints del programa
Show Next Statement			Muestra la siguiente instrucción a ejecutarse
Enable / Disable Trace Recording			Habilita el grabador de señales para revisión de instrucciones
View Trace Records			Revisa las instrucciones anteriormente ejecutadas
Memory Map…			Abre la ventana de mapa de memoria
Performance Analyzer…			Abre la ventana para configuración del Analizador de Desempeño
Inline Assembly…			Inserta una línea en código de ensamblador
Function Editor…			Edita funciones de depuración y archivos de depuración INI

Tabla. 4.6. Menú y Comandos de Depuración (Debug)

Menú de Periféricos (Peripherals)

Menú de Periféricos	Barra de Herramientas	Acceso Directo	Descripción
Reset CPU			Pone al CPU en estado de reset
Interrupt			Abre una ventana para configuración de periféricos del microcontrolador. Esta lista depende de del microcontrolador seleccionado en la base de datos de dispositivos.
I/O Ports			
Serail			
Timer			
A/D Converter			
D/A Converter			
I_2C Controller			
CAN Controller			
Watchdog			

Tabla. 4.7. Menú de Periféricos (Peripherals)

Menú de Herramientas (Tools)

Menú Tools	Barra de Herramientas	Acceso Directo	Descripción
Setup PC-Lint			Usando el menú de herramientas es posible correr programas externos, además permite configurar y correr Gimpel PC-Lint, Siemens Easy – Case, y personalizar programas. Para ejecutar los programas anteriormente citados, estos deben ser previamente instalados.
Siemens Easy - Case			
Customize Tools Menu			Añade programas del usuario al menú.

Tabla. 4.8. Menú de Herramientas (Tools)

Menú SVCS

En este menú es posible configurar y añadir comandos de Software Version Control System (SVCS).

Menú SVCS	Barra de Herramientas	Acceso Directo	Descripción
Configure Version Control…			Configura los comandos de SVCS

Tabla. 4.9. Menú de SVCS

Menú de Ventana (Window)

Menú de Ventana	Barra de Herramientas	Acceso Directo	Descripción
Cascade			Pone las ventanas superpuestas en cascada
Tile Horizontally			Pone las ventanas en forma horizontal para q no estén superpuestas
Tile Vertically			Pone las ventanas en forma vertical para q no estén superpuestas
Arrange Icons			Coloca los iconos al fondo de la pantalla
Split			Divide la ventana actual en 4
1-9			Activa la ventana seleccionada

Tabla. 4.10. Menú de Ventana (Window)

Menú de Ayuda

Menú de Ventana	Barra de Herramientas	Acceso Directo	Descripción
Help Topics			Abre la ayuda en línea
About µVision2			Despliega la versión del programa y información de la licencia.

Tabla. 4.11. Menú de Ayuda

4.1.5 Creación de Proyectos

Antes de escribir cualquier código en C, primero se necesita crear un proyecto asociado con el código. Esto se hace creando primero una nueva carpeta en el directorio de Keil en donde el proyecto será guardado. Luego la aplicación de Keil µVision2 puede ser abierta para la creación del proyecto, esto se realiza mediante los siguientes pasos:

- Cree una carpeta con el nombre del proyecto de preferencia en este directorio c:\keil\c51\examples \
- Corra la aplicación µVision2. Inicio→ Programas → Keil µVision2
- Cree un Proyecto nuevo. En la ventana Principal seleccione el menú *"Project"* y luego seleccione *"New project…"*. Se despliega una ventana como la que se muestra en la figura.

Figura. 4.9. Menú de Proyecto

- Seleccione la carpeta que usted ha creado previamente y en parte inferior de la ventana escriba el nombre de su nuevo proyecto, y presione *"Save"*.

Aparece la ventana de la base de datos de los dispositivos (Figura 4.10), en donde se va a seleccionar el microcontrolador que se va a utilizar en la aplicación, en este caso se va a seleccionar el microcontrolador de Atmel AT89C51SND1.

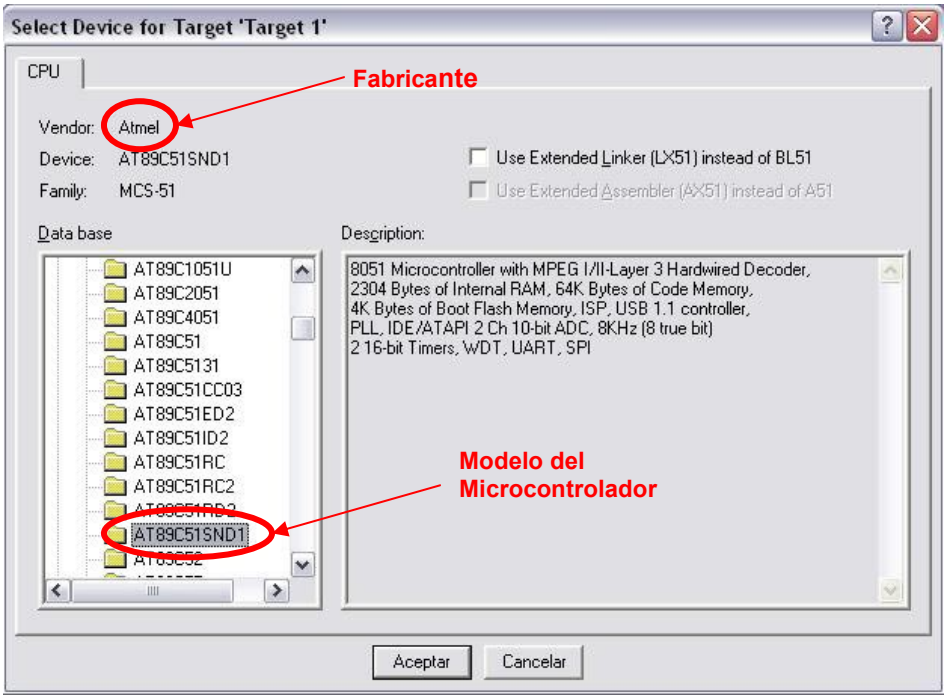


Figura. 4.10. Selección del Microcontrolador

El siguiente paso es configurar las opciones de salida para la aplicación. Se hace clic en el icono "options for target" que se encuentra localizado en la barra de construcción o se selecciona las opciones de configuración en el menú local (Figura 4.11).

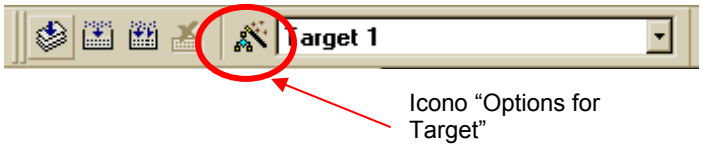


Figura. 4.11. Acceso a Opciones del Target

Se selecciona las opciones de salida donde se configura que tipo de archivos se va a obtener luego de la compilación (Figura 4.12). Se debe habilitar la opción *“Create Hex File”* con un clic en el check box, esto permitirá obtener un archivo hexadecimal de la aplicación que luego será grabada en el microcontrolador.

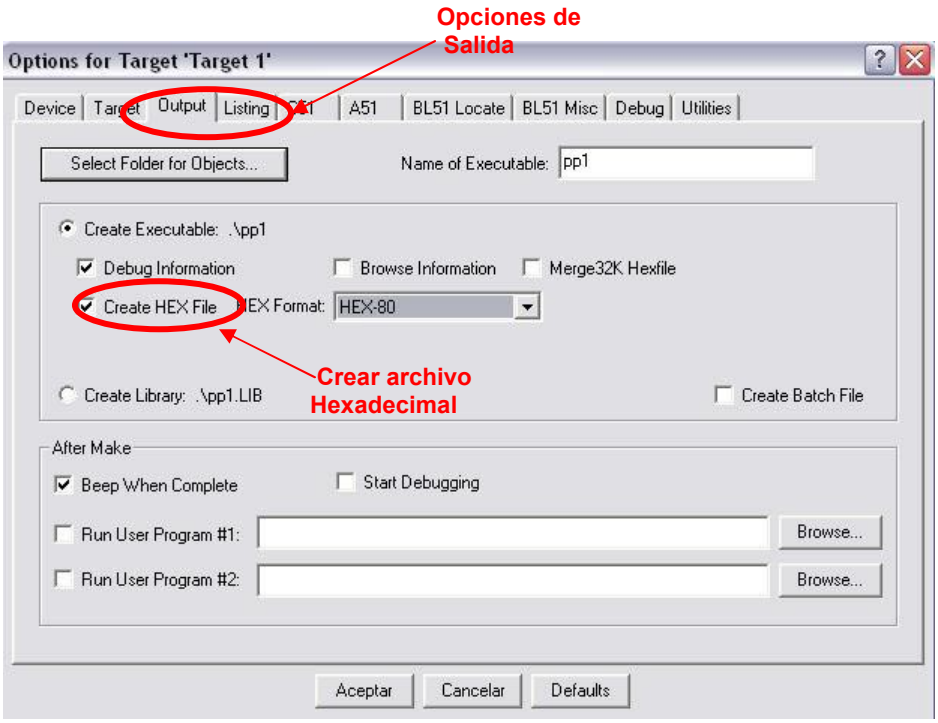


Figura. 4.12. Opciones del Target, Salida

El paso siguiente es la configuración de los parámetros de CPU y memoria, dentro de las opciones del target, en la pestaña *Target* (Figura 4.13) se especifican todas las particularidades del target para la cual se está realizando el código: mapas de memoria externa, bancos de memoria, frecuencia del oscilador, etc. En el resto de pestañas se especifican opciones del compilador, ensamblador, linker, etc. que se utilizarán al construir el ejecutable para ese target. Cabe recalcar que estas opciones sólo deben ser cambiadas si la aplicación lo necesita.

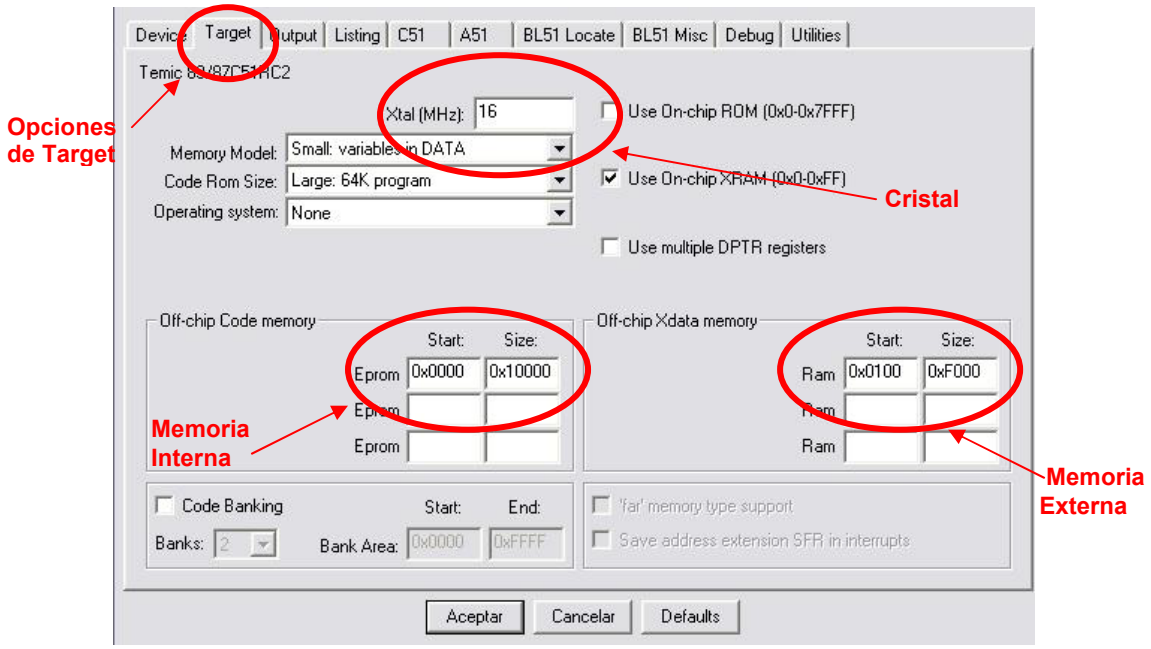


Figura. 4.13. Opciones del Target, Target

Ahora se puede empezar a escribir el programa en C. En la ventana principal, escoja el menú *"File"* (Archivo) y seleccione *"New"* para crear un documento nuevo ó hacer un clic en el icono para archivo nuevo (*"New file"*). Una nueva ventana de edición con el nombre de <text1> aparecerá en la pantalla, en donde se debe teclear el código fuente C.

Una vez que se ha escrito todo el código, se selecciona en el menú *"File"* la opción para guardar *"Save"*, entonces aparecerá una ventana para almacenar el archivo. Se debe seleccionar el directorio que se creó anteriormente para el proyecto y se nombra al programa con una extensión *.C. En el momento en que se almacena el archivo, automáticamente µVision2 reconoce el código y utiliza sintaxis colorida para facilitar la edición del código.

En esta fase, antes de compilar el programa en C, es necesario incluirlo en el proyecto que se está realizando. Para esto se debe hacer clic con el botón derecho del ratón en el grupo *"Source Group1"* y seleccionar agregar archivos a Source Group1 (Add Files to Group 'Source Group 1'). En la ventana que aparece, Figura 4.14 seleccionar el archivo anteriormente almacenado, presionar *"Add"* y luego "*Close*".

Dentro de cada proyecto se encuentran los archivos fuentes y de texto, organizados en grupos de archivos; en proyectos pequeños será suficiente con un grupo (*Source Group 1*), mientras que en proyectos grandes puede ser útil definir varios grupos para poder organizar de una mejor manera el código de la aplicación.

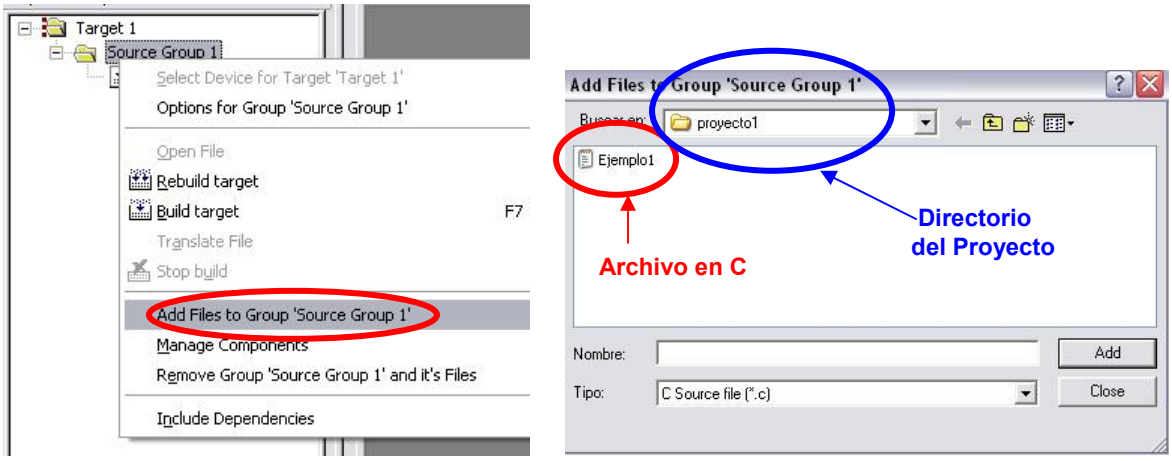


Figura. 4.14. Agregar Archivos a un Grupo

Pulsando el botón de construir proyecto *"Build Target"* o presionando la tecla "F7", se procesan todos los archivos asociados al proyecto; es decir, se compilan los archivos *.C, se ensamblan los archivos *.A51 y se enlazan todos los archivos objetos generados en un único ejecutable.

Si la compilación se completó exitosamente, se visualizará una ventana de salida similar a la de la Figura 4.15. Observe que, dicha ventana informa que no se han producido errores y además informa sobre el tamaño del programa, detallando los siguientes valores por ejemplo:

data=49.0	Bytes de memoria de datos interna utilizados para las variables
xdata=0	Bytes de memoria de datos externa utilizados para las variables
code=388	Bytes de memoria de código que ocupa el programa

Tabla. 4.12. Bytes utilizados de Memoria

En esta ventana, a veces si visualizarán advertencias (warnings) y errores (errors), lo cual indicará que no se ha completado con éxito el proceso de compilación, y es necesario revisar el código fuente.

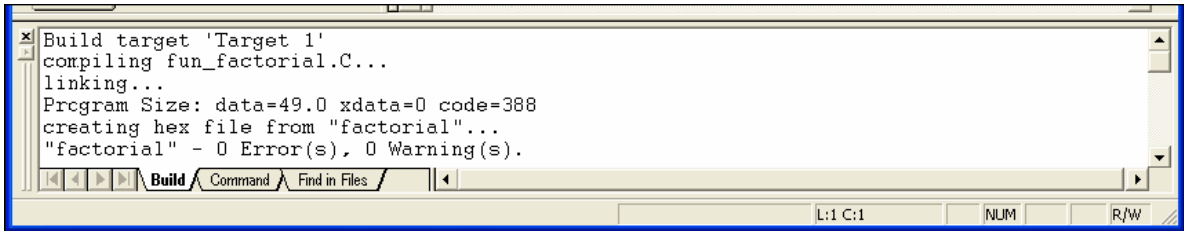


Figura. 4.15. Ventana de salida

En este momento, se han creado todos los archivos que se necesita en la carpeta que se creó para el proyecto. Estos archivos son:

- Ejemplo.hex: Archivo Intel-Standard, el archivo hexadecimal usado para transmitir el código al circuito vía el puerto serial o USB.
- Ejemplo: Este es el archivo de salida, se genera sin ninguna extensión y se usa en las sesiones de depuración.

Si el mensaje resultante de la compilación indica que había 1 o más errores, entonces los archivos de salida no se crearán. En este caso, el archivo se ha tecleado incorrectamente, y haciendo doble clic directamente en cualquier mensaje de error, obliga al software a resaltar la línea de código donde se ha producido el error.

4.1.6 Depuración y Simulación

Otra característica útil de µVision2 IDE es que permite ejecutar el código en un ambiente de simulación específico para el microcontrolador. Una vez obtenido el ejecutable sin errores, hay que comprobar el funcionamiento del programa; para ello, hay de pulsar el botón *Start/Stop debug sesion*, o se puede presionar <Ctrl+F7>, tal como se muestra en la Figura 4.16

Figura. 4.16. Botón de "Debug"

Se observa que el aspecto de µVision2 ha cambiado al Modo de Depuración, Figura 4.17. En la ventana de proyecto se visualizan los registros del modelo de programación de la CPU. En la ventana del archivo fuente aparece una flecha amarilla, a la izquierda del código, indicando la siguiente instrucción a ejecutar; en este caso será la primera sentencia de lenguaje C ejecutable en el programa principal. También aparece una nueva ventana en la parte inferior, es la ventana de *.Watch and call stack.*, en la que se visualizan de forma automática las variables locales a la función que se está ejecutando (en este caso la función main). Si esta ventana no aparece, actívela pulsando el botón *Watch window* (ver Tabla 4.4).

Se debe observar que los botones de la barra de construcción del ejecutable aparecen desactivados y aparece la barra de botones de depuración activada.

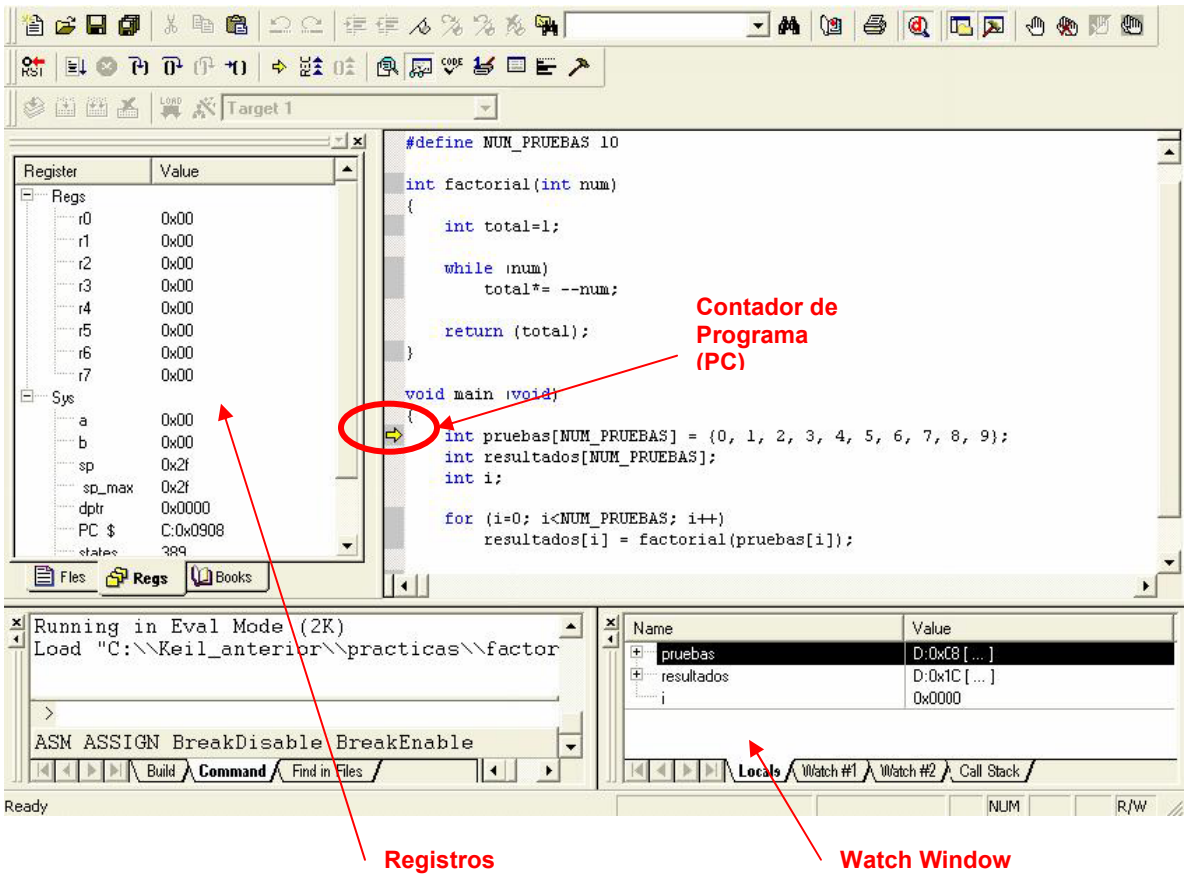


Figura. 4.17. Pantalla Principal - Modo de Depuración

Para comprobar el funcionamiento del programa se necesita visualizar cómodamente el valor de las variables. Para esto, se debe expandir (si está minimizada) la ventana de *Watch and call stack*, haciendo clic en el lado izquierdo de la misma, posteriormente se da doble clic en el nombre de las variables para desplegar sus valores y con el botón derecho se selecciona como sistema de numeración el decimal ó el apropiado.

En la ventana de *Watch*, además de visualizar las variables locales, se pueden definir expresiones que se evaluarán a cada paso que se dé en la ejecución de los programas; para introducir una expresión hay que seleccionar una de las pestañas *Watch #1* o *Watch #2*, hacer un clic en el texto y pulsar la tecla F2 y escribir la expresión. Ejemplos de expresiones que se podrían utilizar con las variables del programa principal son: i+1, pruebas[i], etc.

Se comprueba el funcionamiento del programa ejecutando paso a paso las sentencias de lenguaje C y observando la evolución de las variables. Se observa que al entrar en cualquier función se visualizan las variables de la misma en la ventana de *Watch* en la pestaña *Locals*.

Una herramienta importante es el *Performance Analyzer* (Figura 4.18), la misma que despliega el tiempo de ejecución grabado para las funciones y rangos de dirección que se especifique. Se activa la ventana *Performance Analyzer* pulsando el botón correspondiente de la barra de *Debug*, luego de lo cual aparecerá una nueva ventana similar a la que se muestra en la Figura 4.18.

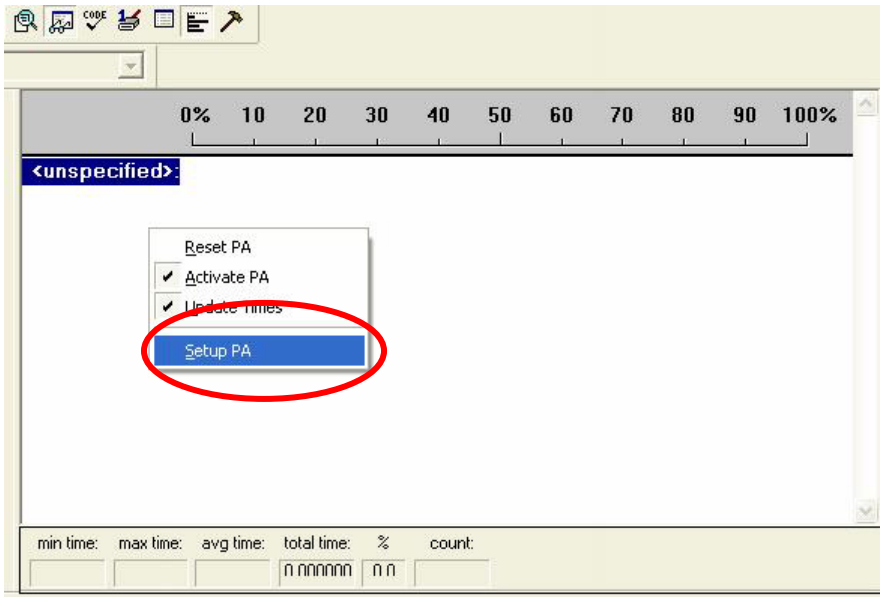


Figura. 4.18. Performance Analyzer

Sobre esta ventana (Figura 4.18) se debe pulsar con el botón de la derecha del ratón y seleccionar la opción *Setup PA*. En la nueva ventana que aparece incluir las funciones que existen en el proyecto seleccionándolas de la lista *Function Symbols* de la derecha y pulsando el botón *Define* (Figura 4.19).

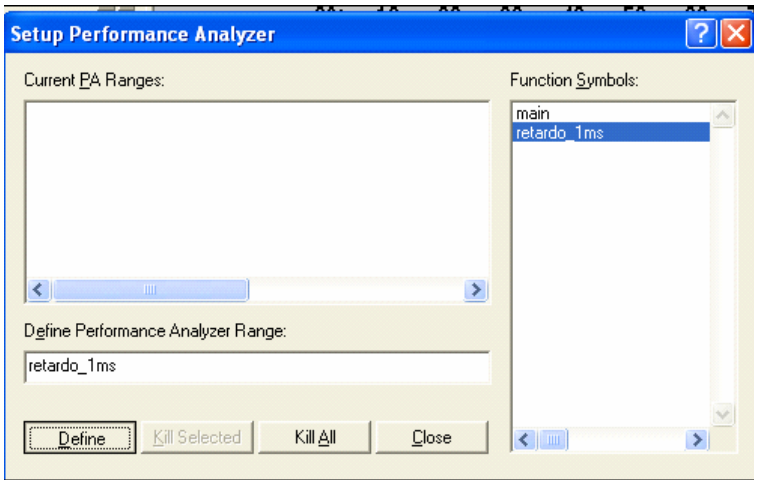


Figura. 4.19. Configuración de PA (Performance Analyzer)

Se ejecuta el programa y se observa que el *Performance Analyzer* indica en todo momento, qué porcentaje de tiempo dedica la CPU a cada una de las funciones definidas, y qué porcentaje de tiempo dedica al resto del código (*unspecified*). Al finalizar o parar la ejecución se puede seleccionar cada una de las funciones definidas y en la parte inferior se informa de los tiempos de ejecución de dichas funciones (Figura 4.20).

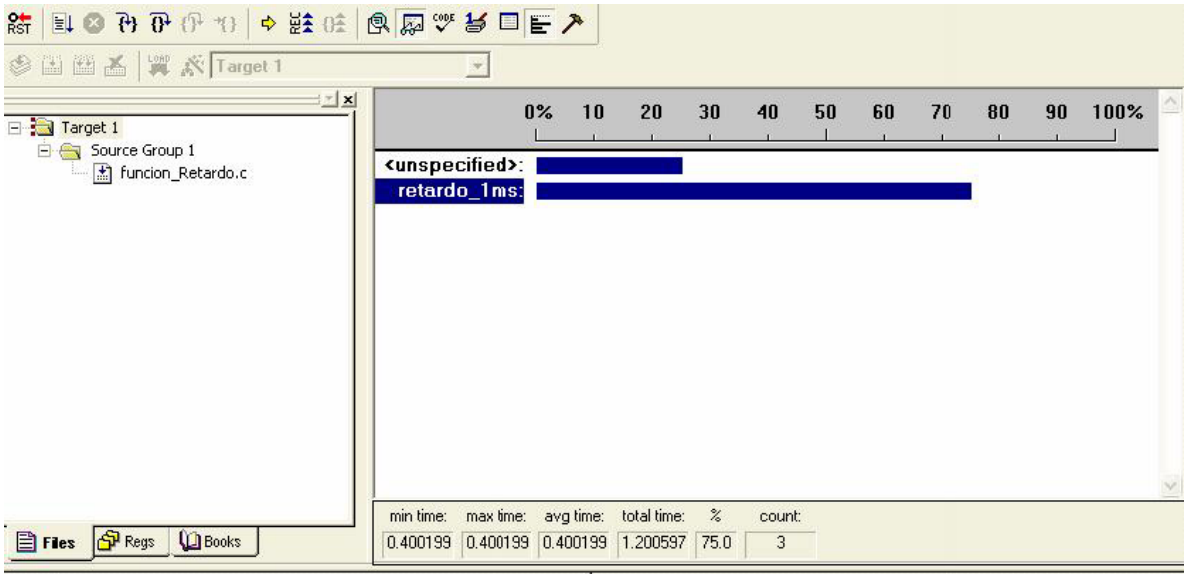


Figura. 4.20. Resultados del Performance Analyzer

Adicionalmente µVision2 IDE posee una ventana en donde se muestra el estado de todos los puertos de Entrada/Salida (Figura 4.21). Para acceder a esta

opción, en el menú de Periféricos (*"Peripheral"*) seleccione "*I/O Ports*" y posteriormente escoja el puerto deseado. En la ventana desplegada se puede visualizar el estado del puerto. Además, se puede con esta herramienta simular una interrupción externa simplemente desmarcando cualquier bit del puerto.

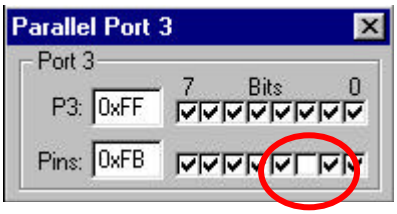


Figura. 4.21. Visualización del Estado del Puerto

4.1.7 Ejemplo de Programa

El siguiente programa mostrado a continuación, muestra la utilización de puertos, almacenamiento de datos en memoria externa y extracción de los mismos datos por otro puerto.

1. Crear un nuevo proyecto y guardarlo en una carpeta con el mismo nombre. (Ver Sección 4.1.5 Creación de Proyectos).
2. Seleccionar el microcontrolador AT89C51SND1C en la lista de dispositivos (Ver Figura 4.10)
3. Crear un archivo nuevo con extensión *.C (Ver Sección 4.1.5 Creación de Proyectos).
4. Ingresar el siguiente código:

```
#include<at8xc51snd1.h>
#include<absacc.h>

unsigned long i;
unsigned char a, b;

void main()
{
AUXR = 0X02;

b= P2;
XBYTE [0x01] = b;
```

```
while (1){
a = XBYTE [0x01];
P4 = ~a;
}
}
```

5. Compilar el proyecto, pulsando el botón de construir proyecto *"Build Target"* (Ver Tabla 4.5) o presionando la tecla "F7"
6. Pasar a modo de depuración o simulación (Ver Figura 4.16)
7. Abrir la ventana de memoria haciendo clic en el icono (Ver tabla 4.4), situándose en la posición 0x00 y donde se visualizará la siguiente figura (Figura 4.22) :

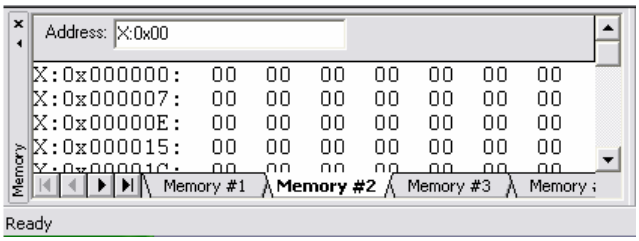


Figura. 4.22. Ventana de memoria externa

8. Abrir la ventana para la visualización de estado de Puerto 2 (Port 2) en el menú de Periféricos, opción I/O Ports (Ver Tabla 4.7) e ingresar un valor en dicho puerto (Ver Figura 4.23).

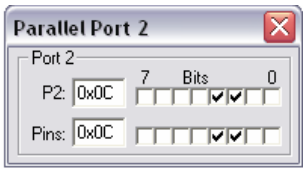


Figura. 4.23. Estado del puerto 2

9. Abrir la ventana de visualización e incluir la variable *a* y el *Puerto 4 (P4)*, tal como se explicó en la sección 4.16, se visualizará la Figura 4.24

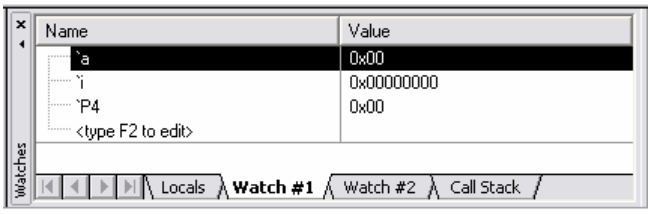


Figura. 4.24. Visualización de variables

10. Ejecutar la simulación presionando el botón de Step (Ver Tabla 4.6) y observar los resultados en las ventanas de memoria, puertos y en la de visualización.

4.2. ATMEL Flip 2.4.2

4.2.1 Introducción

Flip es una aplicación flexible para PC, que permite al usuario la programación y configuración en sistema de microcontroladores Atmel. Por lo tanto sus siglas en inglés son *"Flexible In-system Programmer"*. Flip proporciona algunas características como:

- Ejecuta programación en sistema ISP (In-System Programming) mediante las interfases RS232, USB o CAN.
- Se puede ejecutar bajo los sistemas operativos Windows 9x / Me / NT / 2000 / XP.
- Soporta objetos tipo Intel MCS-86 Hexadecimal y archivos con formato Code 88 para cargar y grabar los archivos.
- Las capacidades de edición del buffer son: llenado (fill), búsqueda (search), copiar (copy), reset, modificar (modify), ir a la dirección (goto address).
- El control de memoria del dispositivo consta de: borrado (erase), verificación de borrado (blank check), programación (program), verificación (verify), lectura (read) y niveles de seguridad.
- Las condiciones del hardware para ISP pueden ser configuradas por software.

4.2.2 Instalación del controlador USB

La instalación del controlador USB resulta sumamente sencilla desde la versión de FLIP 2.4.1. En esta versión del software o versiones superiores, se conecta el dispositivo USB al PC. Para realizar la instalación del controlador, el hardware debe tener las conexiones básicas tal como se muestra en la Figura 4.25. El microcontrolador posee un espacio de memoria en donde está almacenado un código de inicialización propio de la empresa ATMEL, el mismo permite al PC detectar la presencia de un dispositivo conectado al puerto USB, para así proceder a la instalación de su controlador. Hay que señalar que el proveedor del microcontrolador no proporciona los pasos a seguir para la instalación del controlador, la única información existente es un circuito básico, que en la mayoría de los casos no trabaja correctamente. La información proporcionada a continuación y el circuito mostrado en la Figura 4.25, son resultados de la investigación realizada previa al desarrollo del prototipo.

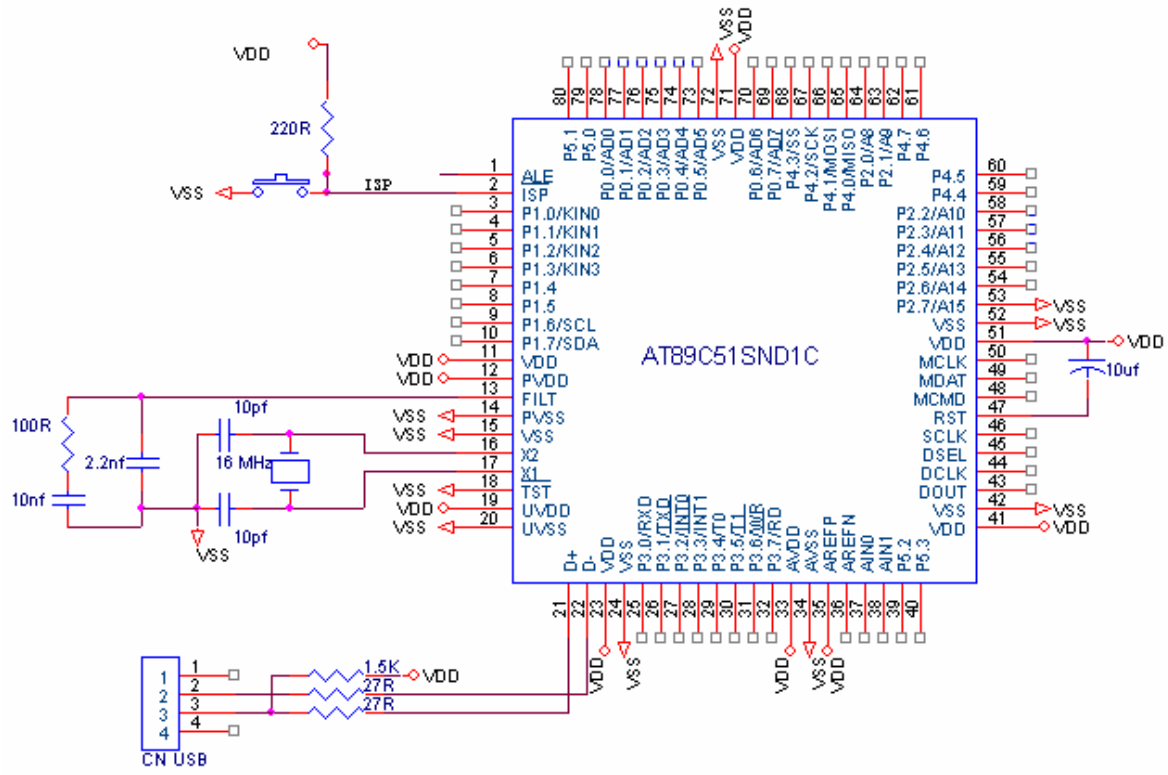


Figura. 4.25. Conexiones Básicas de Hardware.

Seguidamente se polariza el microcontrolador, luego de esto el sistema operativo del PC detectará que existe un nuevo hardware conectado en el puerto USB, desplegándose el asistente de “nuevo hardware encontrado”. A continuación se muestran los pasos a seguir para la instalación:

1. En el primer cuadro de dialogo del asistente, especifique que la instalación se realizará desde una ubicación específica y presione “Siguiente” (Figura 4.26).

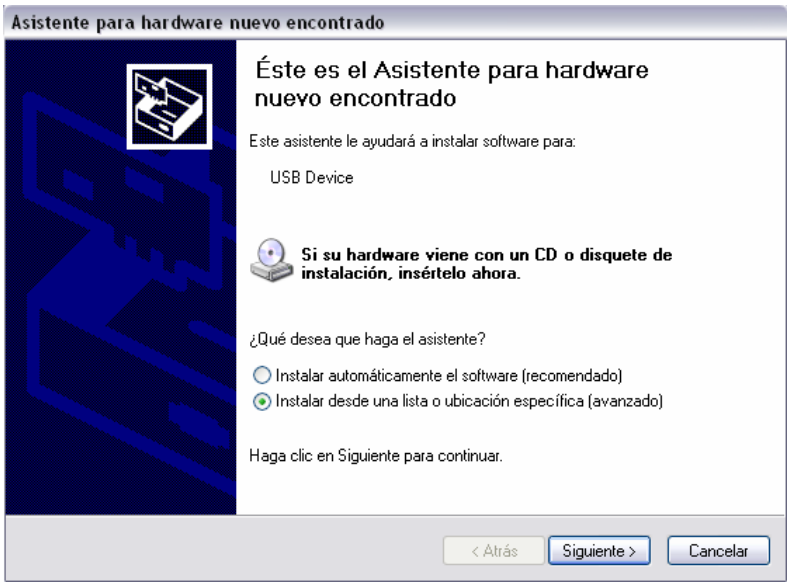


Figura. 4.26. Primer Cuadro de Dialogo de Instalación

2. Los archivos del USB están localizados en la carpeta \usb dentro del paquete de Flip. Ingrese la ruta de la carpeta \usb, presione Aceptar (Figura 4.27).

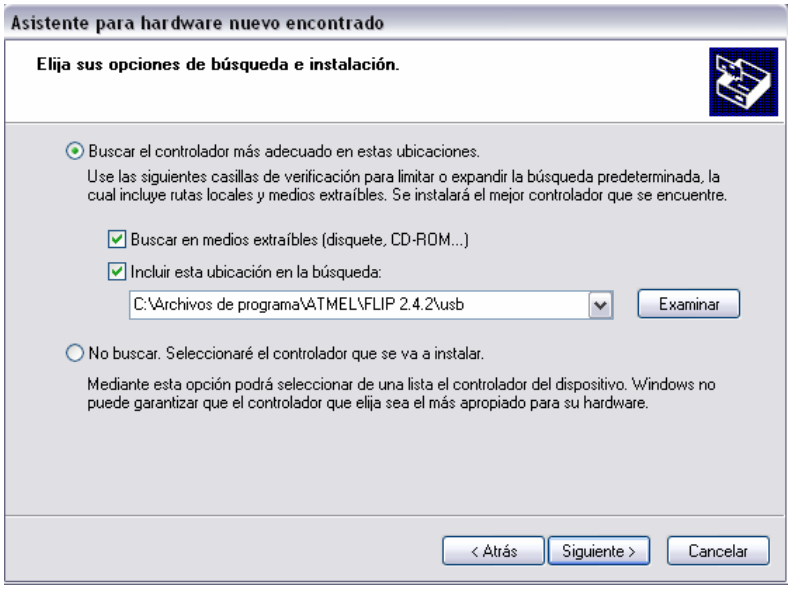


Figura. 4.27. Segundo Cuadro de Dialogo de Instalación

3. El asistente encuentra el controlador y despliega el siguiente cuadro de dialogo. Presione “Siguiente” para comenzar la instalación del controlador (Figura 4.28).

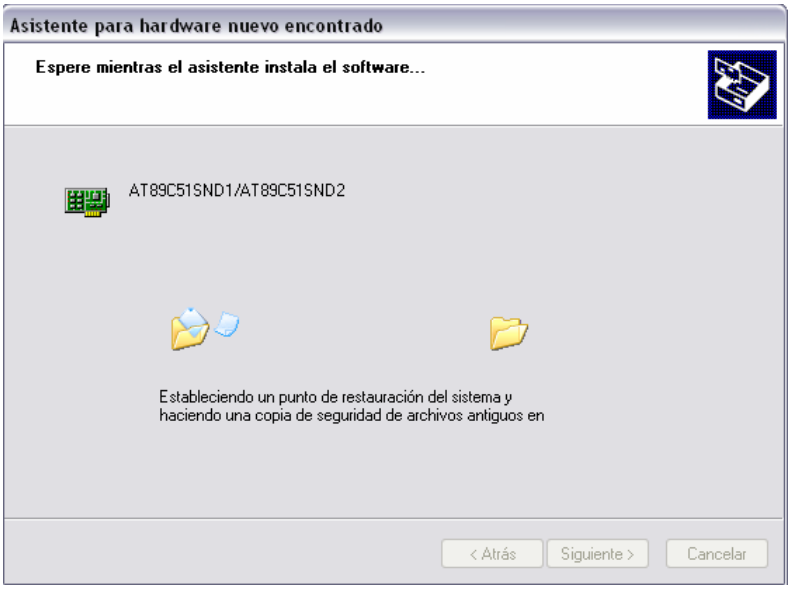


Figura. 4.28. Tercer Cuadro de Dialogo de Instalación

4. Cuando se ha completado la instalación se despliega un cuadro de dialogo mostrando que se ha terminado la operación (Figura 4.29).

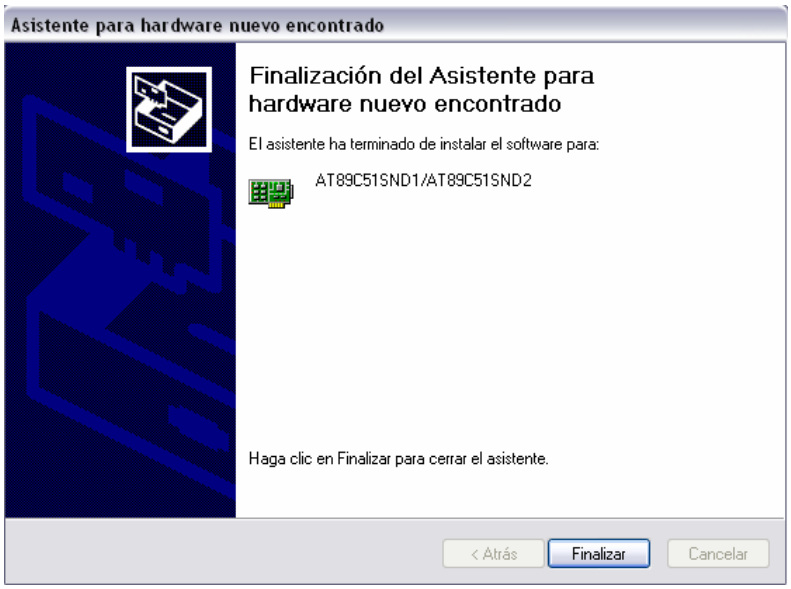


Figura. 4.29. Cuarto Cuadro de Dialogo de Instalación

5. Se puede abrir el *Administrador de Dispositivos* para revisar la instalación correcta del nuevo dispositivo (Figura 4.30).

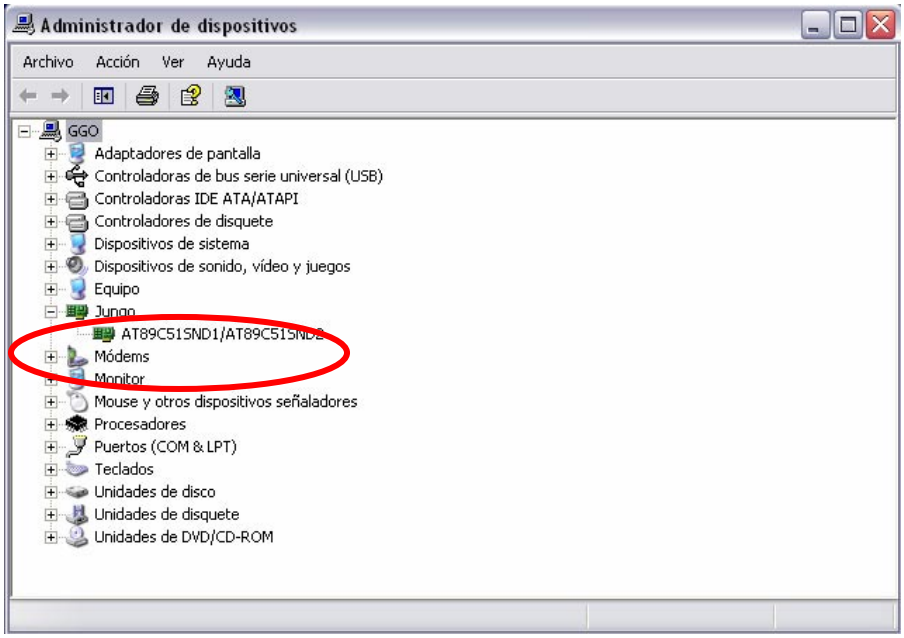


Figura. 4.30. Visualización de Dispositivo Instalado

4.2.3 Entorno de FLIP 2.4.2

Flip consta de una pantalla principal (Figura 4.31) y una pantalla de edición del buffer. En la pantalla principal se pueden ver cuadros de izquierda a derecha que son: Flujo de Operaciones, Información del mapa de memoria o buffer y los parámetros del dispositivo.

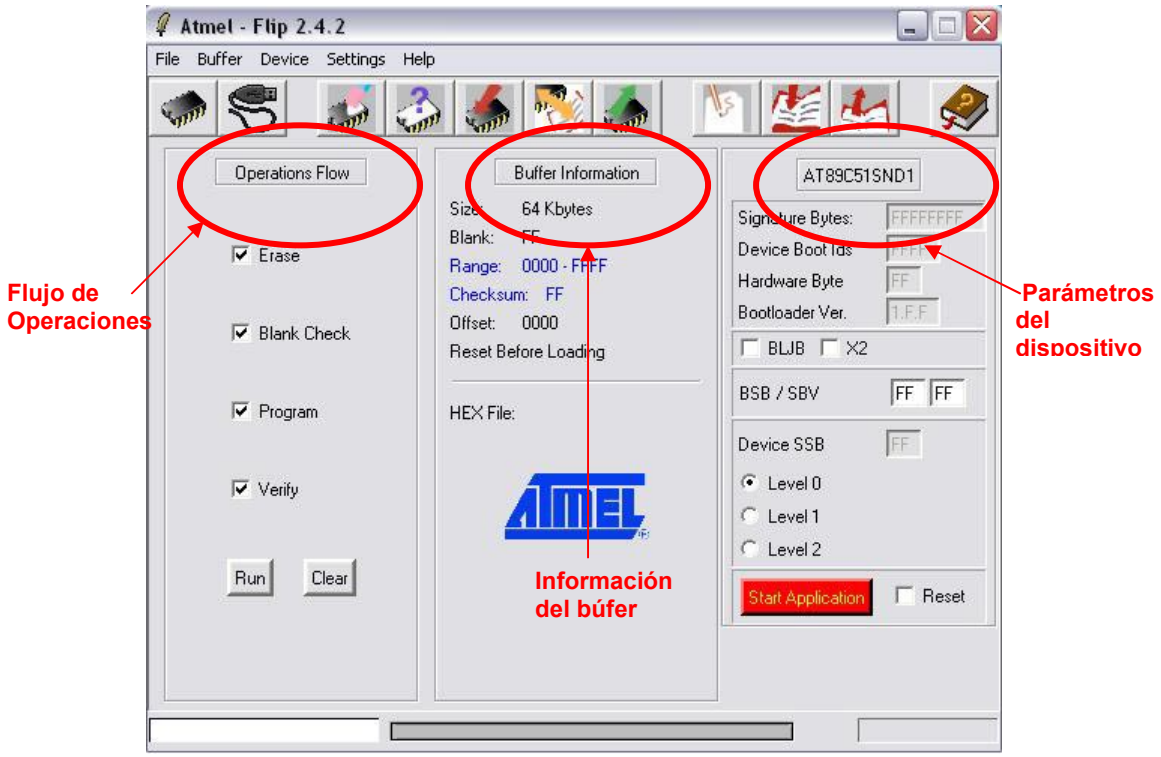


Figura. 4.31. Pantalla Principal

En el menú de Archivo se encuentra 4 opciones:

- *Cargar un Archivo tipo HEX (Load HEX File…)*: Se debe seleccionar el archivo que va a ser cargado desde el buscador (Figura 4.32). Cuando el análisis del archivo HEX se ha completado, Flip actualiza en la ventana principal información como: el rango de direcciones, el nombre y tamaño del archivo HEX.

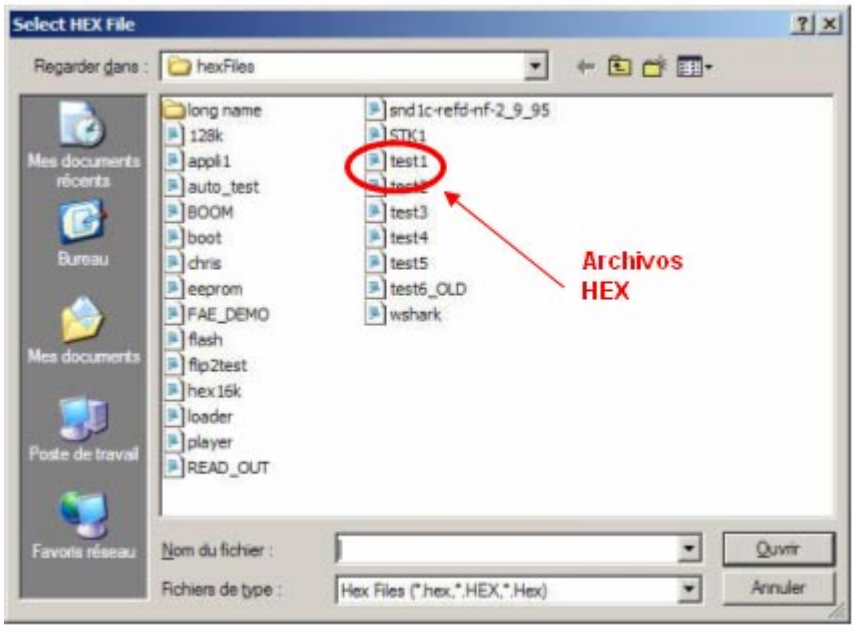


Figura. 4.32. Cargar un archivo HEX

- *Archivos HEX Recientes (Recent HEX Files…)*: Despliega una lista de los archivos tipo HEX que fueron cargados recientemente.
- *Guardar Buffer como (Save Buffer As…)*: Guarda el mapa de memoria que ha sido editado anteriormente como un archivo tipo HEX.
- *Salir (Exit)*: Sale del programa Flip.

El menú de Buffer presenta 2 opciones:

- Edición : Permite entrar a la ventana de edición del buffer o mapa de mapa de memoria
- Opciones: Permite configurar las opciones del buffer como el tamaño y si este debe ser reseteado antes de ser cargado. Además permite la configuración del rango de direcciones que se puede programar. Se despliega una ventana como la que se muestra en la Figura 4.33.

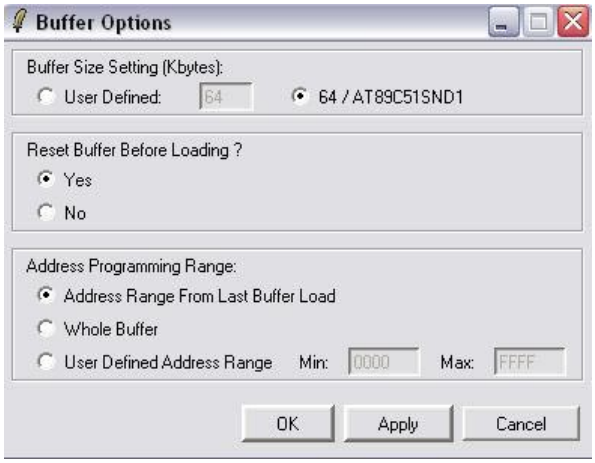


Figura. 4.33. Opciones de Buffer

Por defecto, el tamaño del buffer del dispositivo está definido, pero esto puede ser cambiado siempre y cuando no exceda el tamaño de la memoria del dispositivo.

Dentro del menú de Dispositivo se selecciona el tipo de microcontrolador en el cual se va a trabajar, desplegándose una lista de dispositivos que son compatibles con el software. Se puede seleccionar el dispositivo dando un clic en el botón de la barra de herramientas  , ó presionando la tecla F2.

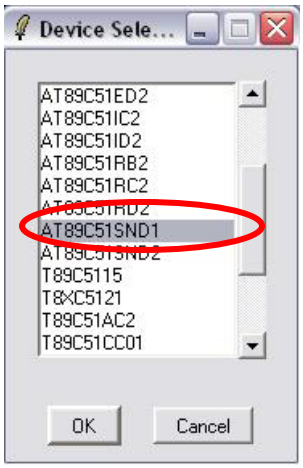


Figura. 4.34. Selección de Dispositivo

Adicionalmente permite seleccionar las tareas que se va a realizar con el microcontrolador y estas son:

- Borrado del Dispositivo  : En el cuadro de diálogo se puede seleccionar el borrado total del dispositivo, ó borrado por bloques de memoria.
- Verificación de Borrado  : Se selecciona el rango de direcciones en donde se va a realizar la verificación. Si el rango de direcciones fue borrado, Flip despliega un mensaje de aprobación, si es que no despliega la primera dirección no borrada.
- Lectura de Memoria  : Se selecciona el rango de direcciones en donde se va a realizar la lectura. Flip lee la memoria del microcontrolador y actualiza el buffer. Aplastando el botón de reset en el cuadro de dialogo, obliga al software a cambiar los valores del rango de memoria a los que se encuentran en la información del buffer.
- Programación  : Una vez que se ha cargado el archivo HEX en el buffer, se puede programar el dispositivo. Si se quiere programar el dispositivo con un buffer vacío o no cargado, se despliega un mensaje para continuar con la acción.
- Verificación  : Flip lee la memoria del dispositivo y la compara con el contenido del buffer, si la verificación fue exitosa despliega un mensaje de aprobación, caso contrario Flip despliega la primera dirección donde se produjo el error.

En el menú de Settings o haciendo clic en la barra de herramientas  se establece el tipo de comunicación que se va a realizar con el microcontrolador. Para comenzar la comunicación mediante USB, se da un clic en *"Open"* para comenzar la sesión de ISP.

Dentro del menú de Settings se encuentra también la opción para entrar en modo de demostración (Demo Mode), en donde se puede realizar cualquier operación sin tener un hardware conectado al PC.

Para acceder a la edición del Buffer, se lo puede realizar haciendo clic en  ó mediante el menú de buffer, desplegándose una pantalla como la que se muestra en la Figura 4.35.

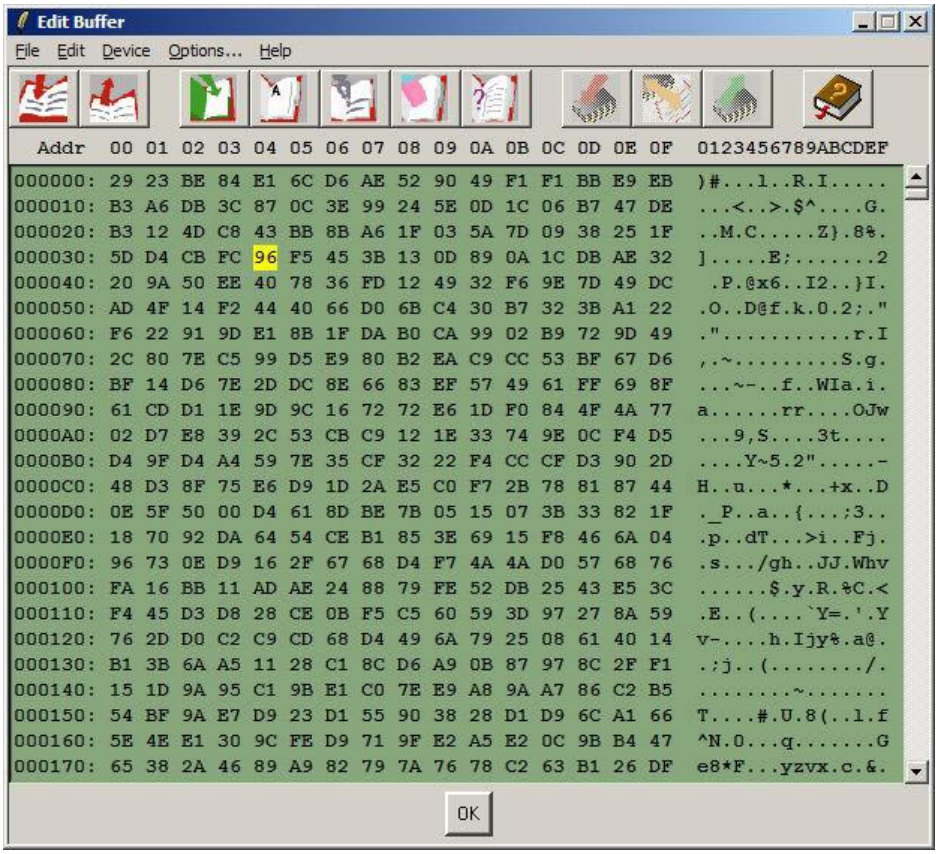


Figura. 4.35. Pantalla de Edición de Buffer

Las opciones de edición del buffer son las siguientes:

- Modificar  : Permite modificar un dato del buffer. Se debe ingresar la dirección y el dato en hexadecimal.
- Mover a la Dirección  : Permite desplazarse a cierta dirección ingresando la misma en el cuadro de diálogo.
- Llenado del Buffer  : Llena un rango de direcciones con un cierto valor. Es de utilidad cuando se necesita que el mismo valor conste a lo largo de ciertas direcciones.

- Datos Aleatorios: Llena el buffer con datos al azar.
- Búsqueda  : Permite buscar un dato en el buffer, se debe especificar si el dato es de tipo ASCII o hexadecimal. Si el dato es encontrado, se muestra la dirección del primer dato encontrado.
- Copiar / Mover: Permite copiar o mover un bloque de memoria, se debe especificar el rango a copiar o mover y la dirección en donde comienza el destino. Si se mueve un bloque de memoria los datos de la dirección fuente son reemplazados con FF.
- Reset  : Borra el contenido del buffer, el valor usual es FF.

4.2.4 Programación del Dispositivo

Producto de la investigación realizada se sugieren los siguientes pasos para la correcta programación del dispositivo:

1. Conectar el hardware al PC en el puerto USB. Prenda el dispositivo mientras presiona el pulsador ISP*(Ver Figura 4.25.)[(3)].
2. Seleccione el dispositivo de la lista (Ver Figura 4.34.): Presione el botón en la barra de herramientas o presione la tecla F2. En el cuadro de dialogo desplegado seleccione el microcontrolador. Cuando el dispositivo es seleccionado, los parámetros del dispositivo son actualizados así como la información del buffer.
3. Seleccione un medio de comunicación: Presione el botón en la barra de herramientas o en el menú. Se selecciona el tipo de conexión, automáticamente Flip ajusta los parámetros. Cuando se establece la comunicación, el cuadro de los parámetros del dispositivo se activa.
4. Cargue el archivo hexadecimal al buffer.
5. Si es necesario editar al buffer, ingrese a la ventana de edición de buffer y modifique los datos necesarios (Ver Figura 4.35.).

[(3)] Al realizar esta acción el microcontrolador ejecuta el programa del BootLoader tal como se muestra en la Figura 1.10.

6. Abra la ventana de opciones de buffer e ingrese la información si es necesario. Los parámetros están establecidos por defecto (Ver Figura 4.33.).
7. Establezca el nivel de seguridad para el dispositivo (Device SSB). El nivel 0 implica que no tiene seguridad. Para trabajar en una aplicación real, se debe establecer el bit de seguridad en el nivel 1 (Ver Figura 4.31.).
8. En el cuadro de Flujo de Operaciones (Operation Flow, Figura 4.31.) marque las operaciones que desea realizar y presione el botón de ejecutar (*"Run"*). Así mismo, puede ejecutar una por una las operaciones haciendo clic en cada botón de la barra de herramientas.
9. Comience la aplicación presionando el botón *"Start Application".*

CAPITULO V

DESARROLLO DEL REPRODUCTOR MP3

5.1 MANEJO DE MEMORIA EXTERNA

Como ya se explicó en el capitulo 1 el microcontrolador AT89C51SND1C posee un espacio de memoria de 64K Bytes de tipo Flash ROM en la cual se almacenará el código del programa, debido a esto es necesario la manipulación de un espacio de memoria de datos externa para almacenar los archivos MP3. La principal razón por la que se emplea una memoria externa se debe al limitado espacio que posee el microcontrolador para almacenar información, debiéndose utilizar necesariamente un dispositivo externo que guarde los archivos MP3 que van a ser reproducidos.

Para el presente proyecto se utilizó una memoria EEPROM AT28BV256 (ver en Anexos las especificaciones de la memoria), de 32Kb de capacidad con un voltaje de alimentación de 3.3 voltios para mantener compatibilidad con los niveles lógicos de voltaje de los demás circuitos integrados. El microcontrolador es capaz de manejar varios tipos de dispositivos de almacenamiento, inicialmente se propuso utilizar una memoria con un voltaje de alimentación de 5 voltios, pero no fue posible debido a que el resto de dispositivos utilizados en el prototipo tenían diferentes niveles lógicos de voltaje.

Para adaptar un espacio de memoria de datos externo al chip se utilizan los Puertos 0 y 2 además la señal de control (ALE), de acuerdo a lo explicado en el capítulo 1 (ver Figura 5.1).

En el Puerto 2 del microcontrolador se encuentran los 8 bits más significativos (MSB) del Bus de Direcciones, las cuales van conectadas directamente a la

memoria EEPROM respetando su respectivo orden. En el Puerto 0 se encuentra multiplexados los 8 bits menos significativos (LSB) del Bus de Direcciones y le Bus de Datos. Estos dos buses son demultiplexados mediante un latch 74HC373, el cual es controlado con la señal proveniente del pin ALE del microcontrolador. De acuerdo como se muestra en la Figura 5.1.

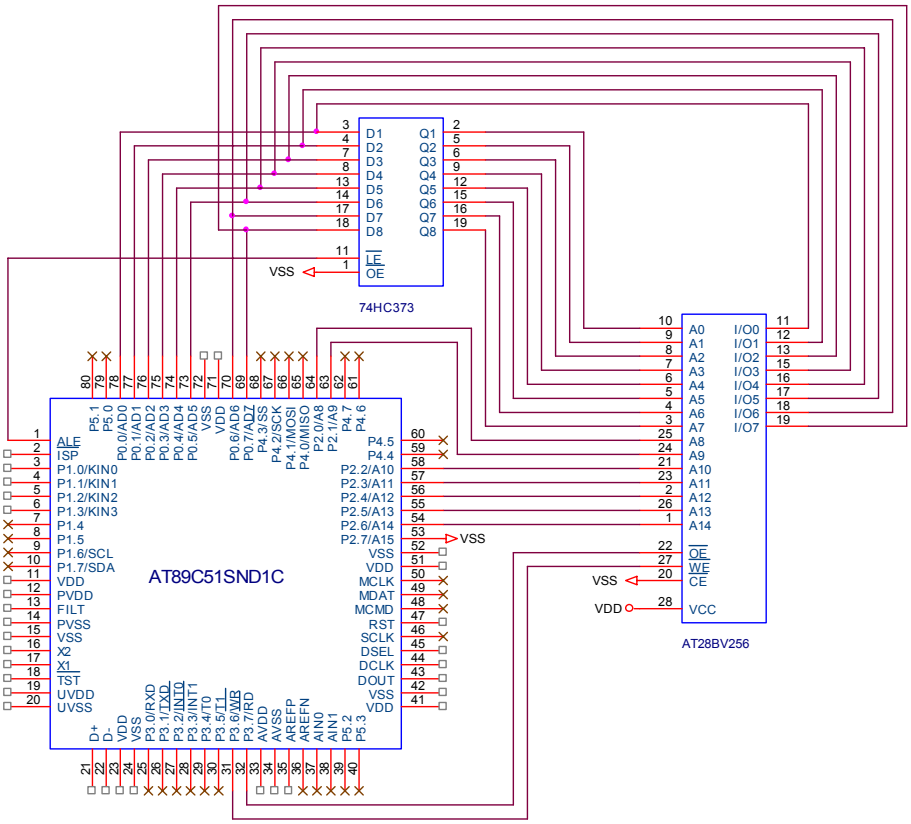


Figura. 5.1. Conexión de memoria

Para acceder a la memoria externa se debe inicializar el registro AUXR que es un registro de control auxiliar, en éste se encuentra el bit EXTRAM, el cual es un habilita la memoria externa. Así, cuando se activa este bit, los puertos P0 y P2 manejan las direcciones y datos de la memoria externa al ejecutar la instrucción XBYTE, como se puede ver a continuación en el siguiente ejemplo:

```
unsigned char a;
…
…

void main()
        {
         AUXR = 0X02;
```

```
                    a=0x00;

        while (1)
                        {
                        a = XBYTE[0x7000];
                }
                        }
```

En este ejemplo se puede ver que primeramente hay que inicializar el registro AUXR con el valor correspondiente para poder habilitar el manejo de memoria externa, esto se hace activando el bit EXTRAM, de acuerdo a la Tabla 5.1.

Registro auxiliar AUXR

7	6	5	4	3	2	1	0
-	EXT16	M0	DPHDIS	XRS1	XRS0	EXTRAM	AO

Tabla. 5.1. Registro AUXR

Lo siguiente es manipular la instrucción XBYTE de acuerdo a las necesidades que se tenga. Esta instrucción permite acceder individualmente a Bytes ubicados en un espacio de memoria externa. Se la puede utilizar para lectura como para escritura de datos. En este caso solo se la utilizó para leer datos guardados previamente en la memoria EEPROM.

5.2 MANEJO DE TECLADO

El microcontrolador AT89C51SND1C posee cuatro pines para entrada de teclado como ya se explicó en la sección 1.7 del Capítulo 1. Cada entrada puede ser programada para activarse tanto en un bajo nivel como en un alto nivel (se puede activar con un 1 lógico o con un 0 lógico). En la aplicación realizada se utilizaron las cuatro entradas de teclado del microcontrolador para poder controlar los niveles de volumen del reproductor MP3 (Vol+ y Vol-), y también el control de reproducción de la canción (Play y Stop).

En la Figura 5.2 se puede ver como se realizó la conexión de los pulsadores para la implementación del teclado de control.

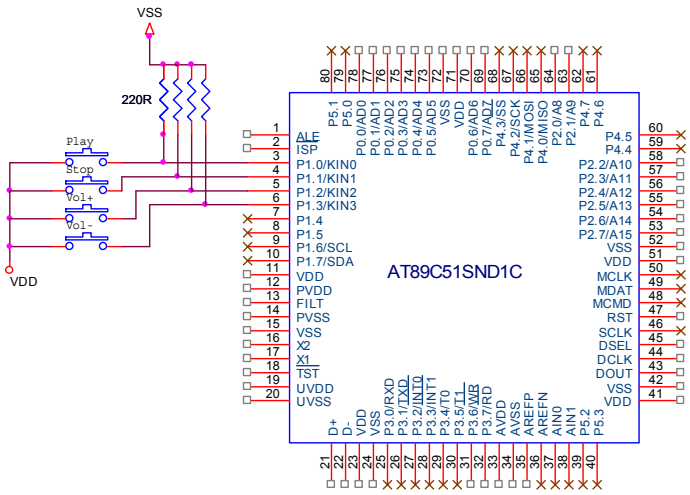


Figura. 5.2. Conexión de teclado

Los pines de entrada para teclado pueden generar cuatro interrupciones. Un bit de interrupción activado EKB en el registro IEN1, permite habilitar o deshabilitar las interrupciones del teclado.

Registro habilitador de interrupciones IEN1

7	6	5	4	3	2	1	0
-	EUSB	-	EKB	EADC	ESPI	E12C	EMMC

Tabla. 5.2. Registro IEN1

Para poder manipular las interrupciones de teclado lo primero que se debe activar las interrupciones globales y las específicas de teclado como se ve a continuación

```
EA = 1;     //Habilita todas las interrupciones
IEN1 = 0x10;     //Habilita interrupciones de teclado poniendo 1 en EKB
```

Lo siguiente es programar el nivel de activación de interrupciones de teclado, esto se hace a través del registro KBCON (ver Tabla 5.3). Los bits del 7 al 4 manejan el control del nivel, poniendo en 1 estos bits la interrupción se activará en nivel alto, al contrario al estar encerados la interrupción se activará en nivel bajo. Los bits restantes del registro KBCON sirven para enmascarar las entradas de interrupción de teclado. A continuación se puede ver cada bit del registro KBCON:

7	6	5	4	3	2	1	0
KINL3	KINL2	KINL1	KINL0	KINM3	KINM2	KINM1	KINM0

Tabla. 5.3. Registro de control de teclado KBCON

En la Figura 5.3 se observa el circuito interno de las entradas de teclado, así se podrá entender mejor el funcionamiento de las mismas.

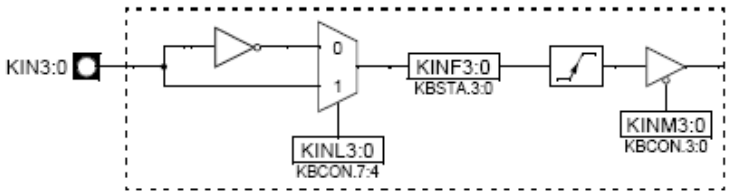


Figura. 5.3. Circuito de entrada de teclado

La siguiente etapa es programar las interrupciones, de acuerdo a la operación que se quiera que realice cada una de estas. Lo primero que se debe hacer es ubicar a la interrupción de teclado (ISR) en un espacio físico de la memoria de programación. En la función de interrupción de teclado lo que se hace es almacenar el valor de las teclas presionadas en una variable, luego se enmascaran los cuatro bits menos significativos para entonces verificar el valor correspondiente a la operación que se debe efectuar. Además se encuentra una bandera de verificación de entrada la cual cambiará de estado cada vez que se ingrese a dicha interrupción, con esta bandera se condicionará la entrada a la función de reproducción de sonido, la que se encuentra fuera de la interrupción.

A continuación se muestra un ejemplo explicativo del código empleado en la interrupción de teclado, basado en la explicación realizada anteriormente:

```
            void KybInt() interrupt 1 using 1
            {
tecla=KBSTA;              //Almacenar valor en la variable tecla
bandera=1;                //activar bandera, se ingresó a interrupción
KBSTA=0x00;               //Resetear estado del teclado
tecla = tecla&0x0F;       //enmascarar valor de tecla
if ( tecla==1)            //Ingresa si el valor de tecla es 1
                {
                  while(tecla==0);      //Verifica si se dejo de presionar
```

```
    Subir_Volumen();            //Ir a función Subir Volumen
  }
 if ( tecla==8)                 //Ingresa si el valor de tecla es 8

{
            while(tecla==0);    //Verifica si se dejo de presionar
    Bajar_Volumen();            //Ir a función Bajar Volumen
    }
if ( tecla==2)                  //Ingresa si el valor de tecla es 2
              {
    Detener_Musica();           //Ir a función Detener Musica
 }
}
```

5.3 DECODIFICADOR MP3

Reloj de PLL

El primer paso para la utilización del módulo MP3, es la configuración del reloj del decodificador mediante la división del reloj PLL, se van a seguir los pasos explicados en el Capítulo 1 sección 1.2.2. Por medio de la Ecuación 5.1 se obtiene el valor del reloj PLL, se debe tener en cuenta que este valor debe ser lo suficientemente grande para poder ser dividido y de ahí obtener los valores de reloj para los módulos de decodificación MP3 y audio.

$$PLLclk = \frac{OSCclk \times (R+1)}{N+1}$$

Ecuación. 5.1.

Se van a suponer los valores de los divisores R = 440 (110111000 en binario) y N = 24 (11000 en binario). El valor del reloj del oscilador es el valor del cristal utilizado en el circuito, el fabricante recomienda el uso de un cristal de 16MHz. Con los valores anteriores se obtiene lo siguiente:

$$PLLclk = \frac{16MHz \times (440+1)}{24+1} = 282.24MHz$$

Ecuación. 5.2.

El valor del divisor R se debe cargar en el registro PLLRDIV que contiene los 8 bits más significativos de R, en los bits 7 y 6 del registro PLLCON se cargan los 2 bits menos significativos del registro R. Así mismo, se deben cargar el valor del divisor N en el registro PLLNDIV. Una vez cargados estos valores en los registros el microcontrolador puede utilizarlos para generar el reloj PLL a la frecuencia deseada. A continuación se muestra la función para cargar los valores de los divisores en los registros correspondientes:

```
extern void SetPLL()
{
PLLRDIV = 0x6E;                 //cargar divisor R, 8 msb del PLL
PLLCON = PLLCON|0x00;           //cargar divisor R, 2 lsb del PLL
PLLNDIV = 0x18;                 //cargar divisor N 7 bits
}
```

Una vez que se han obtenido los valores del reloj PLL, se habilita el mismo siguiendo los pasos explicados en la Figura 1.7. A continuación se presenta el código empleado para la programación del reloj PLL.

```
extern void EnPLL(void)
{
PLLCON=PLLCON|0x02;         //PLLEN=1. Habilita el reloj PLL
PLLCON=PLLCON|0x08;         //PLLRES=1. Resetea el Reloj PLL
PLLCON=PLLCON&0xF7;         //PLLRES=0. Libera el reloj PLL
}
```

Reloj MP3

Para la generación del reloj del decodificador MP3, se debe calcular el divisor para obtener una frecuencia adecuada según los parámetros explicados en el Capítulo 3 sección 3.1.2.2. Se va a suponer un valor de 25 MHz para el reloj MP3, puesto que se necesita una frecuencia mínima de 21 MHz (Ver Tabla 3.1)

$$MP3CLK = \frac{PLLCLK}{MPCD4:0+1}$$

$$25MHz = \frac{282.24MHz}{MPCD4:0+1} \Rightarrow MPCD4:0 \approx 10$$

Ecuación. 5.3.

Los datos reemplazados en la fórmula nos dan como resultado que el valor que debe ser cargado en el registro MP3CLK es de 10. Se debe recalcar que este valor calculado se lo obtuvo después de realizar varias pruebas y probar con una gran cantidad de valores. Se utilizó dicho valor debido a que se aproximaba a un número exacto y no se descartaban muchas cifras decimales.

El último paso para la configuración del módulo MP3, es la activación del decodificador, esto se hace cargando un 1 en el bit 7 MPEN del registro MP3CON, este valor habilita el decodificador MP3.

Diagrama de Flujo de las Funciones Principales e ISR

La ejecución de las diferentes funciones del reproductor, van a depender del botón presionado (interrupción de teclado). Estas funciones van a ser explicadas en los siguientes diagramas de flujo [(4)]:

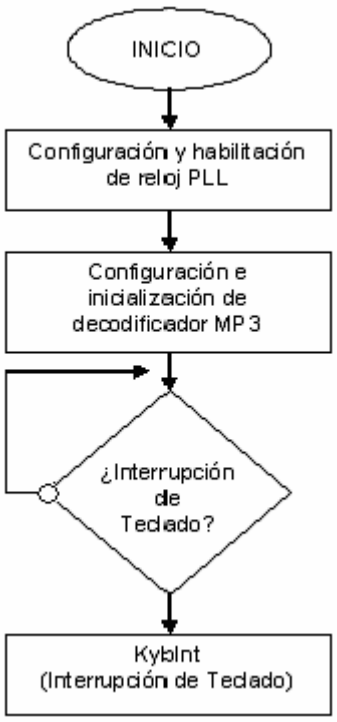


Figura. 5.4. Diagrama de Flujo de Inicialización y Bucle Principal

(4) Un circulo en el Diagrama de Flujo indica que la condición no se a cumplido

Una vez que se haya producido la interrupción de teclado, se debe reconocer que tecla ha producido la interrupción.

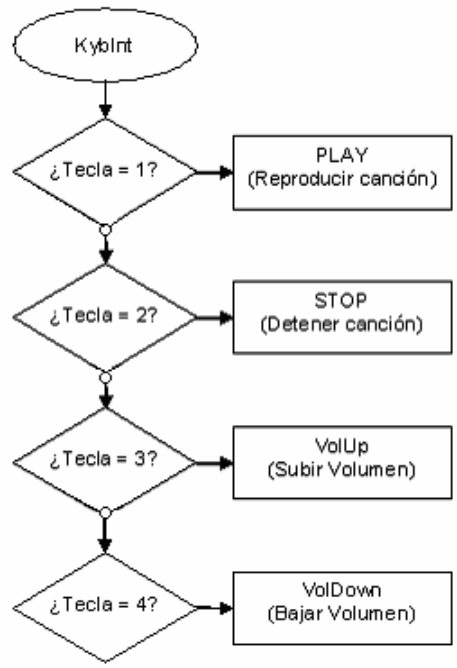


Figura. 5.5. Diagrama de Flujo para Interrupciones de Teclado

Reproducción

Para comenzar la reproducción de la canción, se debe configurar la interfase de audio, la misma que solicita al decodificador MP3 los datos decodificados para ser reproducidos. La función del decodificador MP3, es transformar los datos que son ingresados en el buffer provenientes de un dispositivo de almacenamiento de memoria externo. En el presente proyecto se procedió a llenar este buffer en forma manual utilizando un contador, es decir, se debe ingresar la información del archivo MP3 byte por byte. Cuando los datos se encuentran en el buffer, automáticamente el decodificador reconoce la información de la cabecera del archivo MP3 (en la cabecera se encuentra información como: nombre de la canción, nombre del artista, frecuencia del mp3, etc.), la información que ya forma parte del archivo de audio y finalmente reconoce los datos que indican el fin del archivo MP3, por lo cual sólo es necesario indicar al microcontrolador la primera dirección perteneciente al archivo. Para el control del flujo de información en este buffer, el microcontrolador activa ciertas banderas para solicitar datos y las desactiva cuando el buffer esta lleno; una vez que los datos son decodificados,

son almacenados en otro buffer. Este proceso se realiza una y otra vez, hasta que se encuentre el final de la canción ó hasta que esta sea detenida por el usuario (Tecla Stop).

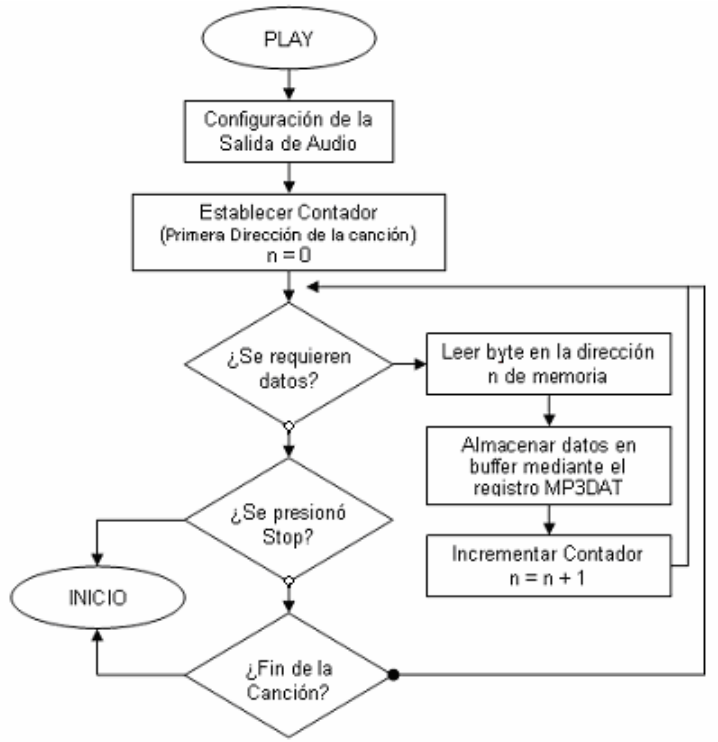


Figura. 5.6. Diagrama de Flujo para Función Play

Control de Volumen

El control de audio consta de un control de volumen de 32 pasos, y un control de ecualización cuyos valores se han configurado en el inicio del programa (Ver en Anexos el programa completo). El control de volumen incrementa o decrementa el valor actual en los registros MP3VOL y MP3VOR, adicionalmente se limitaron los valores para que no excedan los parámetros máximos.

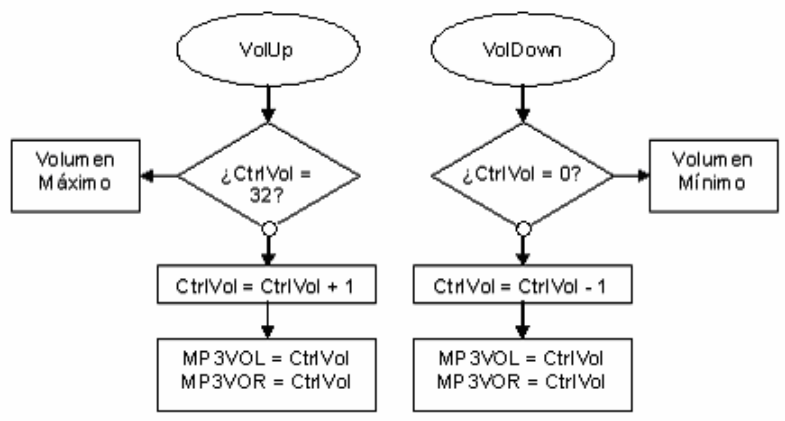


Figura. 5.7. Diagrama de Flujo para Control de Volumen

5.4 INTERFASE DE AUDIO

DAC Serial AD1851

La configuración de la interfase de audio va a depender directamente del Conversor Digital Análogo (DAC) utilizado en el reproductor. El DAC utilizado en el reproductor es el AD1851, que es un conversor serial de 16 bits con formato PCM. Este conversor necesita una alimentación tanto positiva como negativa de 5V, y brinda una salida de audio de ± 3V, con lo cual se evita la conexión de un dispositivo adicional para la amplificación de audio. En la figura siguiente se muestran las conexiones necesarias para el correcto funcionamiento del DAC.

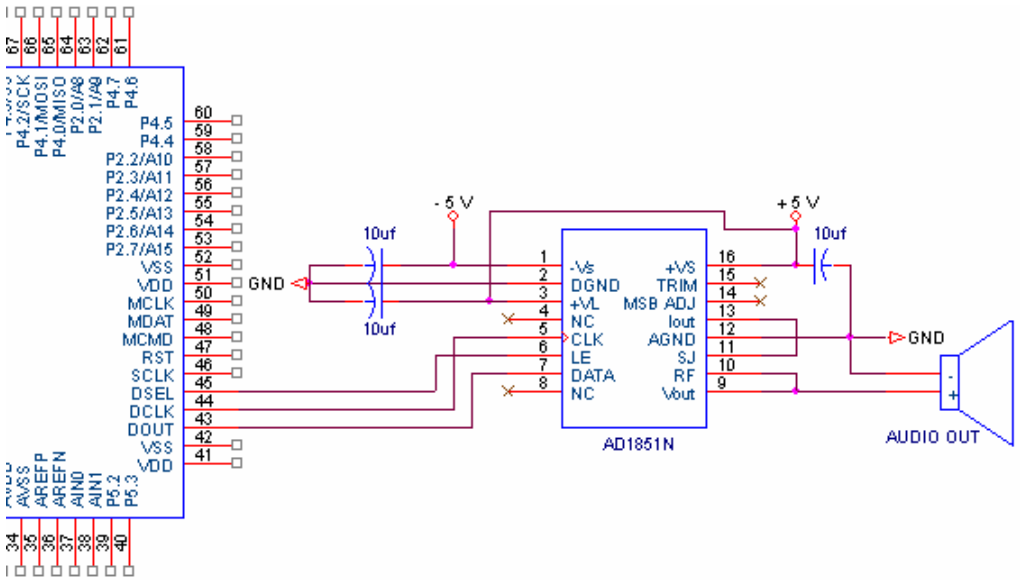


Figura. 5.8. Conexiones del Conversor Digital Análogo.

Los datos transmitidos al DAC, deben ser un flujo de 16 bits seriales con formato MSB justificado. Tres señales deben estar presentes para la correcta operación del DAC, estas son: el reloj (Clock), los datos (Data) y el habilitador (LE). Los bits de entrada están sincronizados con el flanco ascendente de la señal de reloj. Cuando todos los datos han sido cargados, un pulso en bajo en la señal del habilitador actualiza la entrada del DAC. Se debe tener en cuenta al momento de la configuración de la salida de audio del microcontrolador, que la señal de entrada máxima de reloj debe ser de 12.5 MHz. En la figura siguiente se muestra los requerimientos generales de las señales para la transferencia de datos al AD1851.

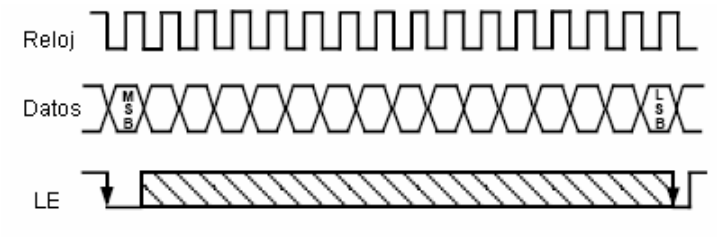


Figura. 5.9. Requerimientos de las Señales para el DAC AD1851

Interfase de Salida de Audio

Teniendo en cuenta las características del DAC, es posible configurar la interfase de audio del microcontrolador. El primer paso es determinar la tasa de bits para la cual el conversor va a trabajar apropiadamente. Según la Tabla 3.16 se debe definir la tasa de bits, poniendo un 1 en el bit HLR del Registro AUDCON0 si la tasa de bits que soporta el DAC se aproxima a 384 multiplicado por la frecuencia de muestreo del archivo MP3. Si el valor se aproxima a 256 por el valor de la frecuencia de muestreo del archivo MP3 se pone un 0 en el bit HLR.

$$Tasa = 256 \times Fs = 256 \times 44.1KHz \approx 11.3MHz$$

Ecuación. 5.4.

$$Tasa = 384 \times Fs = 384 \times 44.1KHz \approx 17MHz$$

Ecuación. 5.5.

La tasa de bits debe ser de 256 frecuencias de muestreo, puesto que un valor superior no podría ser soportado por el conversor. El conversor utilizado en el presente proyecto soporta una tasa máxima de bits de 16 MHz (Ver en Anexos la hoja de especificaciones del DAC1851).

Para la generación del reloj de la interfase de audio, se debe calcular el divisor para obtener una frecuencia adecuada según el valor calculado de la tasa de bits (Ver ecuación 5.4).

$$AUDCLK = \frac{PLLCLK}{AUCD4:0+1}$$

$$11.3MHz = \frac{282.24MHz}{AUCD4:0+1} \Rightarrow AUCD4:0 = 24$$

Ecuación. 5.5.

Este valor debe ser cargado en el registro AUDCLK que almacena los 5 bits para la generación del reloj de audio (Ver Tabla 3.19.).

El formato de los datos transmitidos debe ser configurado según el DAC y según los datos de la Figura 3.2, es decir, que se deben cargar DSIZ = 0 y JUST4:0 = 0, en los registros respectivos para que el formato de salida sea PCM de tipo MSB/LSB de 16 bits. (Ver Figura 5.9). El siguiente diagrama de flujo muestra la configuración de la interfase de salida de audio:

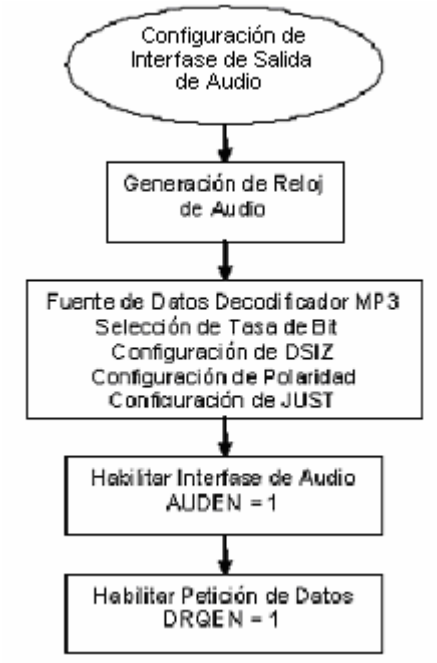


Figura. 5.10. Diagrama de Flujo para Configuración de Interfase de Salida de Audio

Una vez que se ha realizado la configuración de la interfase de salida de audio, comienza la solicitud de los datos decodificados almacenados en el buffer. Estos datos son transformados de un formato de 16 bits paralelos a 16 bits seriales y serán enviados al DAC por el pin de salida de datos (DOUT) y sincronizados con la señal de reloj DCLK. La señal DSEL es la encargada de activar el latch interno del DAC lo cual permite el ingreso de los datos al DAC. Una vez que los datos han sido procesados por el conversor, se va a tener en la salida una señal analógica equivalente al audio del archivo MP3 almacenado en la memoria.

5.5 MANEJO DE LCD

Para la manipulación del LCD se utilizaron el Puerto 4 para el manejo de datos y el Puerto 1 para el control, como se muestra en la siguiente figura:

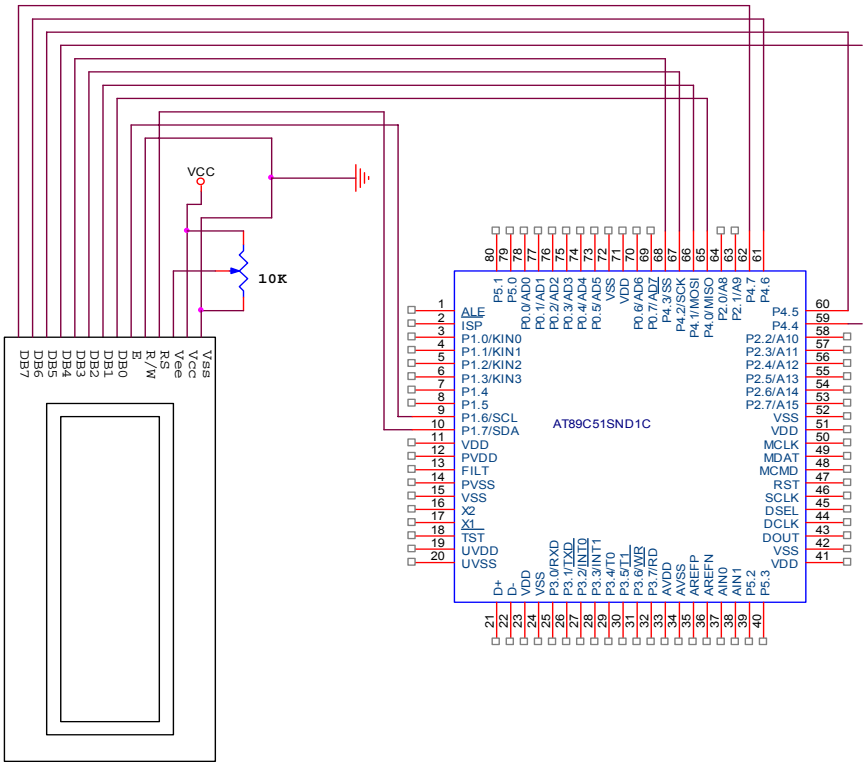


Figura. 5.11. Conexión de LCD

Los pines de control el LCD (RS register selector y E enable), son los que manejan las instrucciones que se deben ejecutar y existe una condición importante que se debe tomar en cuenta referida directamente al tiempo necesario que se necesita para cambiar de un estado a otro en los pines de control, en el caso de que este tiempo sea mas pequeño que el tiempo mínimo requerido, el LCD no ejecutara correctamente la instrucciones, por lo tanto se perderán datos. En la siguiente tabla se pueden encontrar los valores mínimos necesarios de tiempo para una correcta operación del LCD.

Instrucción	Tiempo Mínimo
Borrado de pantalla	2ms.
Escritura de caracteres	40 µs.
Rotar caracteres	40 µs.

Tabla. 5.4. Tiempos de instrucciones básicas

Por lo tanto para el correcto funcionamiento del LCD hay que implementar funciones que puedan realizar retardo de tiempos, en el orden de mili y microsegundos. Para las funciones de retardo de tiempo hay que tomar en

cuenta el cristal con el que está trabajando el microcontrolador, en este caso es de 16MHz, lo siguiente es realizar uno o varios bucles que consuman el tiempo de retardo que se necesita, este tiempo no debe ser exacto pero si aproximado, para lo cual se elaboraron funciones de retardo (delays) puesto que no existían funciones que realicen esta operación (Ver en Anexos, Librarías de Delay)

Funciones para el control del LCD

La principal instrucción que se utiliza es la función de escritura, esta función carga en el puerto de datos (Puerto 4) el byte que se desea enviar al LCD, esta función además manipula el pin de habilitación del LCD (pin E), para así poder realizar una correcta escritura en el dispositivo. A continuación se muestra el código de la función escribir.

```
Void lcd_write(unsigned char c)        // Función de Escritura
{
LCD_EN =1;
LCDPORT = c;
DelayUs(40);
LCD_EN =0;
}
```

Inicialización del LCD

Lo primero que se debe hacer antes de manipular el LCD es inicializarlo y configurarlo, esto se lo hace enviando ciertas instrucciones dependiendo de las necesidades y del dispositivo que se tenga.

La primera instrucción que se debe enviar es comunicar al LCD con que tipo de bus de datos se va a trabajar, de 4 bits o de 8 bits, en nuestro caso se realizo una configuración de 8 bits debido a su facilidad de programación, además hay que indicar si se trabaja con una o dos líneas y por último hay que indicar el tamaño de los caracteres, de 5 x 7 puntos o 5 x 10 puntos. La siguiente instrucción es apagar el LCD, luego prenderlo, fijar el modo de desplazamiento de los caracteres, y finalmente configurar el desplazamiento del cursor. A continuación se presenta el código realizado correspondiente a la inicialización del LCD

```
    Void lcd_init(void)          //Función Para Inicialización del LCD
    {
        LCD_RS = 0;
        lcd_write (0x38);
```

```
        lcd_write (0x04);
        lcd_write (0x0F);
        lcd_write (0x01);
        lcd_write (0x02);
}
```

Funciones del LCD

Además está la función correspondiente a la escritura de cadenas de caracteres, que es una simple variación de la función lcd_write() utilizando un bucle y un puntero.

```
Void lcd_puts(const char * s)     //Función para escritura de cadenas de caracteres
{
   LCD_RS = 1;
   while(*s)
   lcd_write(*s++);
}
```

También se encuentran las funciones de borrado de pantalla y desplazamiento de cursor, las cuales son simples instrucciones que manipulan el LCD.

```
Void lcd_clear(void)          //Función de Borrado de Pantalla
{
   LCD_RS = 0;
   lcd_write(0x1);
   DelayMs(2);
}

Void lcd_goto(unsigned char pos)     //Función de Desplazamiento de cursor
{
   LCD_RS = 0;
   lcd_write(0x80+pos);
}
```

CAPÍTULO VI

CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES

La amplia gama de microcontroladores que existen actualmente en el mercado, facilitan el desarrollo de dispositivos que anteriormente eran de difícil implementación debido a su compleja programación, elevados costos y su complicada adquisición.

En la actualidad microcontroladores como el AT89C51SND1C de la compañía ATMEL, permiten desarrollar dispositivos multimedia capaces de reproducir archivos MP3 con un mínimo de circuitería externa y la posibilidad de interconectarse con prácticamente cualquier dispositivo de almacenamiento.

Existen en el mercado diversos IDE (Entorno de Desarrollo Integrado) que permiten el desarrollo de proyectos, tal es el caso de Keil µVision 2, este software tiene definidas librerías específicas para una amplia gama de microcontroladores. Con esta herramienta la programación del microcontrolador es muy sencilla y se la puede realizar en lenguaje de C o código de ensamblador.

El costo del reproductor MP3 desarrollado, no es elevado en comparación a sus símiles comerciales. La razón de esto se debe a que el valor del microcontrolador utilizado es relativamente económico, el inconveniente se presenta en la adquisición del mismo, puesto que no existen distribuidores en el país, razón por la cual se realizó una importación del microcontrolador y demás componentes del reproductor.

En el Ecuador se tiene la capacidad de desarrollar dispositivos multimedia sin ninguna dificultad como se ha demostrado, siendo estos dispositivos de alta calidad y más económicos que aparatos similares que vienen del exterior, por lo que se debería dedicar mayor tiempo al desarrollo y no solo a la compra de tecnología, con el fin de abaratar costos y generar fuentes de trabajo.

6.2 RECOMENDACIONES

La existencia de una fuente de alimentación negativa para el conversor digital análogo del circuito del reproductor representa una gran limitante, por lo que se recomienda no utilizar un conversor bipolar. Adicionalmente, para lograr una mejor calidad de sonido, el conversor debe ser STEREO.

Debido a que la capacidad de la memoria externa utilizada en el prototipo es muy reducida, se sugiere que para implementar un reproductor completo, se cambie el tipo de dispositivo en donde van a ser almacenados los archivos MP3.

El microcontrolador utilizado trabaja con un nivel de voltaje máximo de 3.3 voltios, así que para reducir el consumo de energía se recomienda que todos los componentes del reproductor trabajen con niveles de voltaje similares

Previamente a la programación tanto en lenguaje C o código de ensamblador, es sugerido estudiar minuciosamente el funcionamiento del software Keil µVision2, el cual brinda opciones útiles como son simulación y depuración de los programas.

Antes de realizar la compra de los componentes a utilizarse en cualquier proyecto, se debe hacer un estudio de todas las características eléctricas, dimensionales y de empaquetado que el fabricante ofrece en las hojas técnicas.

Para aprovechar de mejor manera las características que brinda el microcontrolador AT89C51SND1C se sugiere manipular el conversor análogo

digital, integrado en el encapsulado, opción que permite grabar y almacenar sonidos y voz en el dispositivo de memoria.

REFERENCIAS BIBLIOGRÁFICAS

- ATMEL, *Datasheet AT89c51SND1C*, www.atmel.com
- ATMEL, Datasheet AT28BV256, www.atmel.com
- ANALOG DEVICES, *Datasheet AD1851*, www.digikey.com
- www.mpeg.org, Características del formato MPEG I/II Capa 3
- www.iis.fraunhofer.de/amm/techinf/layer3, Características del formato MPEG I/II Capa 3

ANEXO 1

CÓDIGO DEL PROGRAMA

Programa principal

```
#include <at8xc51snd1.h>                    //incluir libreria de microcontrolador
#include <absacc.h>                         //incluye libreria para manejo de memoria externa
#include <stdio.h>                          //incluye libreria para entrada salida

#include <delay.h>                          //incluye libreria de retardo
#include <delay.c>                          //incluye código de retardo
#include <lcd.h>                            //incluye librería para uso de LCD
#include <lcd.c>                            // incluye funciones para uso de LCD


extern  void leer_teclado(void);            //declara función de lectura de teclado
static  void    main (void);                //declara función principal
unsigned long n;                            //declara contador
unsigned char temporal,ctrlvol = 0x02;      //variables
unsigned int tecla;                          // valor de tecla presionada
bit bandera;                                // bandera
int g;                                      //contador

void EnableX2(void)                         //función para habilitar modo X2
{

CKCON=CKCON|0x01;                           //Configuración de registro CKCON para habilitar
                                            //modo X2

}


extern void SetPLL()                        //función para cargar los valores del reloj PLL
{
PLLRDIV = 0x6E;                             //cargar divisor R, 8 msb del PLL
PLLCON = PLLCON|0x00;                       //cargar divisor R, 2 lsb del PLL
PLLNDIV = 0x18;                             //cargar divisor N 7 bits en registro

}


extern void EnPLL(void)                     //Función para habilitar reloj PLL
{

PLLCON=PLLCON|0x02;                         //PLLEN=1 Habilita reloj PLL
PLLCON=PLLCON|0x08;                         //PLLRES=1      Resetea reloj PLL
PLLCON=PLLCON&0xf7;                         //PLLRES=0 Libera reloj PLL

}


extern void SetMPCD(unsigned char mpcd)     //Función para cargar el divisor del reloj MP3
{
MP3CLK=mpcd;                                //Carga el valor del divisor en el registro
}
```

```
extern void SetAUDCLK(unsigned char AUCD)  //Función para cargar el divisor del reloj de Audio
{
AUDCLK=AUCD;                               //Carga el valor del divisor para gnerar reloj de
                                           //audio
}


extern void EnMP3(void)                    //Función para habilitar el decodificador de MP3
{
MP3CON=MP3CON|0x80;                        //carga un 1 en el bit MPEN del registro MP3CON
}


void EnGINT(void)                          //Función para habilitar interrupciones Globales
{
EA=1;                                      //Habilita todas las interrupciones
EAUD=1;                                    //Habilita las interrupciones de audio
}

extern void SetAUDCON()                    //Función para configurar las opciones de salida de
                                           //interfase de audio
{

AUDCON1 = AUDCON1&0x7F;                    //selecciona mp3 como fuente de audio
AUDCON0 = AUDCON0|0x00;                    //HLR. Selección de Tasa de bits a 256·Fs
AUDCON0 = AUDCON0|0x00;                    //DSIZ Formato de salida de 16 bits
AUDCON0 = AUDCON0|0x04;                    //POL Modo PCM
AUDCON0 = AUDCON0&0x07;                    //just MODO MBS/LSB justificado
AUDCON1= AUDCON1&0x7F;                     //SRC selecciona mp3 como fuente de audio
AUDCON1= AUDCON1|0x01;                     //AUDEN Habilita la interfase de audio

}



void VolCtrl(unsigned char VOL,unsigned char VOR)
                                           //Función de Control de Volumen
{
 MP3VOL=0x1F&VOL;                          //Carga Valor de volumen en canal izquierdo
 MP3VOR=0x1F&VOR;                          //Carga Valor de volumen en canal derecho
 }

void EQCtrl(unsigned char BASS,unsigned char MEDIUM,unsigned char TREBLE,bit MPBBST)
                                           //Función para establecer valores de bajos medio y
                                           //agudos. Activa Bass Bost
{
 MP3BAS=0x1E&BASS;                         //Cargar valor de Bajos
 MP3MED=0x1E&MEDIUM;                       //Cargar valor de Medios
 MP3TRE=0x1E&TREBLE;                       //Cargar valor de Agudos
  if(MPBBST)
 MP3CON=0x40|MP3CON;                       //Activar Bass Bost
}

extern void mp3init()                      //Función para inicializar módulo MP3
{
```

```
 SetPLL();                                  //Carga los valores de los divosores N y R del PLL
 EnPLL();                                   //Habilita el PLL
 SetMPCD(10)                                //Carga los valores del reloj de decodificador MP3
 VolCtrl(ctrlvol,ctrlvol);                  //Control del volumen del decodificador
 EQCtrl(0xFF,0xFF,0xFF,1);                  //Control de Bajos, Medios y Agudos del
                                            //decodificador
 SetAUDCLK(24);                             //Carga los valores del reloj de audio
 SetAUDCON();                               //Configura al DAC
 AUDCON1= AUDCON1|0x40;                     //Play
 n=0x0000;                                  //Direccion inicial del MP3
 EnMP3();                                   //Habilita el decodificador MP3
 EnGINT();                                  //Habilita los interrupciones globales
 AUXR = 0X02;                               //Hablita el manejo de memoria externa
}

void mp3play()                              //Función de Reproducción
{
n=0;                                        //Puntero de memoria en dirección 0
do{

    if((MP3STA1 & 0x10) == 0x10){           //MPFREQ = 1 se requieren datos

        while(!((MP3STA1 & 0x08) == 0x08)); //MPBREQ = 0. Si el buffer pide datos pasa a la
                                            //siguiente linea
            temporal=XBYTE[n];              //carga en temporal lo q se encuentra en la
                                            //dirección n
            MP3DAT =temporal;               //carga los datos de temporal en el buffer
            n++;                            //incrementa puntero de dirección

        }

            if(n>0x76B0){                   //si llego al fin de la canción. Solo en el caso del
                                            //proyecto
                n=0;                        // puntero al inicio de la memoria
            }

}while(!(tecla == 2));                      //sigue tocando mientras no se presione stop

}

void mp3volup()                             //Función para aumentar el volumen
{
lcd_goto(0x00);
lcd_clear();
lcd_puts("    Subir Volumen");              //imprime msg en LCD posición 0
if(ctrlvol==31){                            //condición de volumen maximo
        ctrlvol=30;
        lcd_goto(0x00);
        lcd_clear();
        lcd_puts("    Volumen Maximo");     //imprime mensaje ne LCD
        }
ctrlvol= ctrlvol+1;                         //aumenta valor de volumen en 1
```

```
VolCtrl(ctrlvol,ctrlvol);                    //llama a función de control de volumen

}

void mp3voldown()                            //Función para disminuir el volumen
{
lcd_goto(0x00);
lcd_clear();
lcd_puts("    Bajar Volumen");               // imprime mensaje en LCD posición 0
if(ctrlvol==0){                              //condición de volumen mínimo
          ctrlvol=1;
          lcd_goto(0x00);
lcd_clear();
lcd_puts("    Volumen Minimo");              // imprime mensaje en LCD
          }
ctrlvol= ctrlvol-1;                          //disminuye valor de volumen en 1

VolCtrl(ctrlvol,ctrlvol);                    //llama a función de control de volumen


}

 void EnKybIn(void)//
{

          IEN1 = 0x10;                       //activa interrupciones de teclado
}

void KybInt() interrupt 20 using 1           //Función de interrupción de teclado
{
          tecla=KBSTA;                       //guarda tecla presionada
          bandera=1;                         //activa bandera
          KBSTA=0x00;                        //Pone en 0 las banderas de teclado
          tecla = tecla&0x0F;                //enmascara valor de la tecla
          if ( tecla==1)                     //tecla es vol+?
    {
      while(tecla==0);                       //soltó la tecla?
                    mp3volup();              //llama función para subir volumen

    }

           if ( tecla==8)                    //tecla es vol-?
    {
      while(tecla==0);                       //soltó la tecla?
                    mp3voldown();            //llama función para bajar volumen

    }

          if ( tecla==2)                     //tecla stop?
    {

          lcd_clear();                       //borra LCD

           for (g=0; g < 5; g++)             //Imprime Mensaje 5 veces
           {
           lcd_goto(0x00);
```

```
            lcd_puts("    Detener Musica");          // imprime mensaje 5 veces
            DelayMs(250);
            lcd_clear();
            lcd_goto(0x00);
            lcd_puts("    ");
            DelayMs(250);
            lcd_clear();
    }

                    }
}


void main()                                         //función principal
{
EA = 1;                                             //activar interrupciones
IEN1 = 0x10;                                        //activa interrupciones de teclado
EnableX2();                                         //Entra en Modo X2 ->Fcpu = Fper = Fosc
EnKybIn();                                          //función de interrupciones de teclado
KBCON = KBCON&0xF0;                                 //flanco de detección de las interrupciones

IPH0=0x40;                                          //setear prioridad de las interrupciones
IPH1=0x10;
IPL0=0x20;
IPL1=0x10;

          lcd_init();                               // inicializaci¢n LCD
          lcd_clear();                              // borra LCD

          DelayMs(250);
          DelayMs(250);

while (1)
{

          mp3init();                                //llama a función para inicializar modulo mp3
          lcd_clear();                              //borra LCD
          lcd_goto(0x00);                           //cursor en posición 0
          lcd_puts("   Mp3 Player");                //imprime mensaje
          lcd_goto(0x40);                           //salto de linea
          lcd_puts("Presione Play");                // imprime mensaje



if (bandera)                                        //detectar tecla
  {
    if ( tecla==4)                                  //tecla play??
    {
      lcd_clear();                                  //borra LCD
          lcd_goto(0x00);
                                                    //bandera = 0;
          lcd_puts("   Reproduciendo...");          // imprime mensaje
          mp3play();                                //llama a función de reproducción

    }
```

```
    bandera=0;                          //bandera encerada
    }

}

}
```

Librerías del LCD

```
#include <at8xc51snd1.h>
#include <lcd.h>
#include <delay.h>

#define  LCDPORT P4


sbit LCD_RS = P1^7 ;                    //defino bit 7 del puerto 1 como selector de registos
sbit LCD_EN = P1^6 ;                    //defino bit 6 del puerto 0 como habilitador de LCD


#define LCD_STROBE  ((LCD_EN = 1),(LCD_EN=0))


void lcd_write(unsigned char c)         //función para escribir un caracter
{
        LCD_EN =1;
        LCDPORT = c;
        DelayUs(40);
        LCD_EN =0;
}


void lcd_clear(void)                    //función para borrar LCD
{
        LCD_RS = 0;
        lcd_write(0x1);
        DelayMs(2);
}


void lcd_puts(const char * s)           //función para escribir una cadena de caracteres
{
        LCD_RS = 1;
        while(*s)
                lcd_write(*s++);
}
```

```
void lcd_putch(char c)
{
        LCD_RS = 1;
        LCDPORT = c;
        DelayUs(40);
}


void lcd_goto(unsigned char pos)                //función para mover cursor
{
        LCD_RS = 0;
        lcd_write(0x80+pos);
}


void lcd_init(void)                             //función para inicializar LCD
{

LCD_RS = 0;
lcd_write (0x38);
lcd_write (0x04);
lcd_write (0x0F);
lcd_write (0x01);
lcd_write (0x02);

 }


extern void lcd_write(unsigned char);

extern void lcd_clear(void);

extern void lcd_puts(const char * s);

extern void lcd_goto(unsigned char pos);


extern void lcd_init(void);

extern void lcd_putch(char);


#define lcd_cursor(x)     lcd_write(((x)&0x7F)|0x80)
```

Librerías de delays

```
#include        <delay.h>

void
```

```
DelayMs(unsigned char cnt)
{
#if     XTAL_FREQ <= 2MHZ
        do {
                DelayUs(996);
        } while(--cnt);
#endif

#if   XTAL_FREQ > 2MHZ
        unsigned char   i;
        do {
                i = 4;
                do {
                        DelayUs(250);
                } while(--i);
        } while(--cnt);
#endif
}



#ifndef  XTAL_FREQ
#define XTAL_FREQ    16MHZ
#endif

#define MHZ    *1000
#define KHZ    *1
#if     XTAL_FREQ >= 12MHZ

#define DelayUs(x)      { unsigned char _dcnt; \
                          _dcnt = (x)*((XTAL_FREQ)/12MHZ); \
                          while(--_dcnt != 0) \
                                continue; }
#else

#define DelayUs(x)      { unsigned char _dcnt; \
                          _dcnt = (x)/(12MHZ/(XTAL_FREQ))|1; \
                          while(--_dcnt != 0) \
                                continue; }
#endif

extern void DelayMs(unsigned char);
```

Librerías del microcontrolador

```
#ifndef _REGSND1_H_
#define _REGSND1_H_

/*_____ I N C L U D E S _____________________________________________________*/


/*_____ M A C R O S _____________________________________________________*/
```

```
/* RAISONANCE compiler */

#define Sfr(x, y)       sfr x = y
#define Sbit(x, y, z)   sbit x = y^z


/* C51 CORE */

Sfr (A       , 0xE0);
Sfr (ACC     , 0xE0);
Sfr (B       , 0xF0);
Sfr (PSW     , 0xD0);
Sfr (SP      , 0x81);
Sfr (DPL     , 0x82);
Sfr (DPH     , 0x83);

Sbit (CY     , 0xD0, 7);    /* PSW */
Sbit (AC     , 0xD0, 6);
Sbit (F0     , 0xD0, 5);
Sbit (RS1    , 0xD0, 4);
Sbit (RS0    , 0xD0, 3);
Sbit (OV     , 0xD0, 2);
Sbit (F1     , 0xD0, 1);
Sbit (P      , 0xD0, 0);


/* SYSTEM MANAGEMENT */

Sfr (PCON    , 0x87);
Sfr (AUXR    , 0x8E);
Sfr (AUXR1   , 0xA2);
Sfr (NVERS   , 0xFB);


/* PLL & CLOCK */

Sfr (CKCON   , 0x8F);
Sfr (PLLCON  , 0xE9);
Sfr (PLLDIV0 , 0xEE);
Sfr (PLLNDIV , 0xEE);
Sfr (PLLDIV1 , 0xEF);
Sfr (PLLRDIV , 0xEF);


/* INTERRUPT */

Sfr (IEN0    , 0xA8);
Sfr (IPL0    , 0xB8);
Sfr (IPH0    , 0xB7);
Sfr (IEN1    , 0xB1);
Sfr (IPL1    , 0xB2);
Sfr (IPH1    , 0xB3);

Sbit (EA     , 0xA8, 7);    /* IEN0 */
Sbit (EAUD   , 0xA8, 6);
Sbit (EMP3   , 0xA8, 5);
```

```
Sbit (ES     , 0xA8, 4);
Sbit (ET1    , 0xA8, 3);
Sbit (EX1    , 0xA8, 2);
Sbit (ET0    , 0xA8, 1);
Sbit (EX0    , 0xA8, 0);

Sbit (IPLAUD , 0xB8, 6);    /* IPL0 */
Sbit (IPLMP3 , 0xB8, 5);
Sbit (IPLS   , 0xB8, 4);
Sbit (IPLT1  , 0xB8, 3);
Sbit (IPLX1  , 0xB8, 2);
Sbit (IPLT0  , 0xB8, 1);
Sbit (IPLX0  , 0xB8, 0);


/* PORTS */

Sfr (P0      , 0x80);
Sfr (P1      , 0x90);
Sfr (P2      , 0xA0);
Sfr (P3      , 0xB0);
Sfr (P4      , 0xC0);
Sfr (P5      , 0xD8);

Sbit (P0_7   , 0x80, 7);    /* P0 */
Sbit (P0_6   , 0x80, 6);
Sbit (P0_5   , 0x80, 5);
Sbit (P0_4   , 0x80, 4);
Sbit (P0_3   , 0x80, 3);
Sbit (P0_2   , 0x80, 2);
Sbit (P0_1   , 0x80, 1);
Sbit (P0_0   , 0x80, 0);

Sbit (P1_7   , 0x90, 7);    /* P1 */
Sbit (P1_6   , 0x90, 6);
Sbit (P1_5   , 0x90, 5);
Sbit (P1_4   , 0x90, 4);
Sbit (P1_3   , 0x90, 3);
Sbit (P1_2   , 0x90, 2);
Sbit (P1_1   , 0x90, 1);
Sbit (P1_0   , 0x90, 0);

Sbit (SDA    , 0x90, 7);    /* P1 */
Sbit (SCL    , 0x90, 6);
Sbit (KIN3   , 0x90, 3);
Sbit (KIN2   , 0x90, 2);
Sbit (KIN1   , 0x90, 1);
Sbit (KIN0   , 0x90, 0);

Sbit (P2_7   , 0xA0, 7);    /* P2 */
Sbit (P2_6   , 0xA0, 6);
Sbit (P2_5   , 0xA0, 5);
Sbit (P2_4   , 0xA0, 4);
Sbit (P2_3   , 0xA0, 3);
Sbit (P2_2   , 0xA0, 2);
Sbit (P2_1   , 0xA0, 1);
```

```
Sbit (P2_0   , 0xA0, 0);

Sbit (P3_7   , 0xB0, 7);    /* P3 */
Sbit (P3_6   , 0xB0, 6);
Sbit (P3_5   , 0xB0, 5);
Sbit (P3_4   , 0xB0, 4);
Sbit (P3_3   , 0xB0, 3);
Sbit (P3_2   , 0xB0, 2);
Sbit (P3_1   , 0xB0, 1);
Sbit (P3_0   , 0xB0, 0);

Sbit (RD     , 0xB0, 7);    /* P3 */
Sbit (WR     , 0xB0, 6);
Sbit (T1     , 0xB0, 5);
Sbit (T0     , 0xB0, 4);
Sbit (INT1   , 0xB0, 3);
Sbit (INT0   , 0xB0, 2);
Sbit (TXD    , 0xB0, 1);
Sbit (RXD    , 0xB0, 0);

Sbit (P4_7   , 0xC0, 7);    /* P4 */
Sbit (P4_6   , 0xC0, 6);
Sbit (P4_5   , 0xC0, 5);
Sbit (P4_4   , 0xC0, 4);
Sbit (P4_3   , 0xC0, 3);
Sbit (P4_2   , 0xC0, 2);
Sbit (P4_1   , 0xC0, 1);
Sbit (P4_0   , 0xC0, 0);

Sbit (SS_    , 0xC0, 3);    /* P4 */
Sbit (SCK    , 0xC0, 2);
Sbit (MOSI   , 0xC0, 1);
Sbit (MISO   , 0xC0, 0);

Sbit (P5_3   , 0xD8, 3);    /* P5 */
Sbit (P5_2   , 0xD8, 2);
Sbit (P5_1   , 0xD8, 1);
Sbit (P5_0   , 0xD8, 0);


/* FLASH MEMORY */

Sfr (FCON    , 0xD1);


/* TIMERS */

Sfr (TCON    , 0x88);
Sfr (TMOD    , 0x89);
Sfr (TL0     , 0x8A);
Sfr (TL1     , 0x8B);
Sfr (TH0     , 0x8C);
Sfr (TH1     , 0x8D);

Sbit (TF1    , 0x88, 7);    /* TCON */
Sbit (TR1    , 0x88, 6);
```

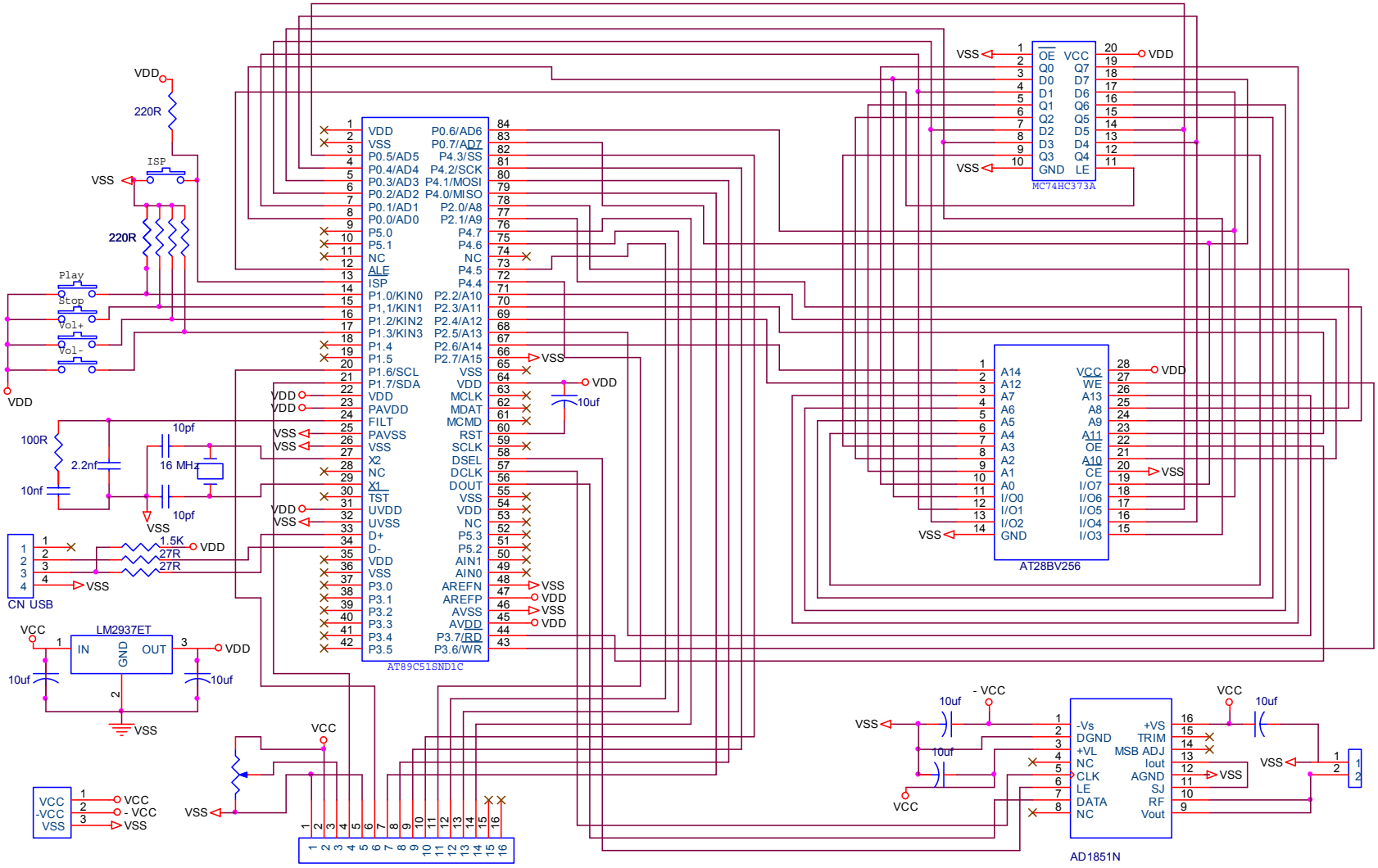
```
Sbit (TF0    , 0x88, 5);
Sbit (TR0    , 0x88, 4);
Sbit (IE1    , 0x88, 3);
Sbit (IT1    , 0x88, 2);
Sbit (IE0    , 0x88, 1);
Sbit (IT0    , 0x88, 0);


/* WATCHDOG */

Sfr (WDTRST  , 0xA6);
Sfr (WDTPRG  , 0xA7);


/* MP3 DECODER */

Sfr (MP3CON  , 0xAA);
Sfr (MP3STA  , 0xC8);
Sfr (MP3STA1 , 0xAF);
Sfr (MP3DAT  , 0xAC);
Sfr (MP3ANC  , 0xAD);
Sfr (MP3VOL  , 0x9E);
Sfr (MP3VOR  , 0x9F);
Sfr (MP3BAS  , 0xB4);
Sfr (MP3MED  , 0xB5);
Sfr (MP3TRE  , 0xB6);
Sfr (MP3CLK  , 0xEB);
Sfr (MP3DBG  , 0xAE); /* hidden register */

Sbit (MPANC  , 0xC8, 7);    /* MP3STA */
Sbit (MPREQ  , 0xC8, 6);
Sbit (ERRLAY , 0xC8, 5);
Sbit (ERRSYN , 0xC8, 4);
Sbit (ERRCRC , 0xC8, 3);
Sbit (MPFS1  , 0xC8, 2);
Sbit (MPFS0  , 0xC8, 1);
Sbit (MPVER  , 0xC8, 0);


/* AUDIO INTERFACE */

Sfr (AUDCON0 , 0x9A);
Sfr (AUDCON1 , 0x9B);
Sfr (AUDSTA  , 0x9C);
Sfr (AUDDAT  , 0x9D);
Sfr (AUDCLK  , 0xEC);


/* USB CONTROLLER */

Sfr (USBCON  , 0xBC);
Sfr (USBADDR , 0xC6);
Sfr (USBINT  , 0xBD);
Sfr (USBIEN  , 0xBE);
Sfr (UEPNUM  , 0xC7);
Sfr (UEPCONX , 0xD4);
```

```
Sfr (UEPSTAX , 0xCE);
Sfr (UEPRST  , 0xD5);
Sfr (UEPINT  , 0xF8);
Sfr (UEPIEN  , 0xC2);
Sfr (UEPDATX , 0xCF);
Sfr (UBYCTX  , 0xE2);
Sfr (UFNUML  , 0xBA);
Sfr (UFNUMH  , 0xBB);
Sfr (USBCLK  , 0xEA);

Sbit (EP3INT , 0xF8, 3);    /* UEPINT */
Sbit (EP2INT , 0xF8, 2);
Sbit (EP1INT , 0xF8, 1);
Sbit (EP0INT , 0xF8, 0);


/* MMC CONTROLLER */

Sfr (MMDAT   , 0xDC);
Sfr (MMCMD   , 0xDD);
Sfr (MMSTA   , 0xDE);
Sfr (MMMSK   , 0xDF);
Sfr (MMCON0  , 0xE4);
Sfr (MMCON1  , 0xE5);
Sfr (MMCON2  , 0xE6);
Sfr (MMINT   , 0xE7);
Sfr (MMCLK   , 0xED);


/* IDE CONTROLLER */

Sfr (DAT16H  , 0xF9);


/* UART */

Sfr (SCON    , 0x98);
Sfr (SBUF    , 0x99);
Sfr (SADDR   , 0xA9);
Sfr (SADEN   , 0xB9);
Sfr (BDRCON  , 0x92);
Sfr (BRL     , 0x91);

Sbit (SM0    , 0x98, 7);    /* SCON */
Sbit (FE     , 0x98, 7);
Sbit (SM1    , 0x98, 6);
Sbit (SM2    , 0x98, 5);
Sbit (REN    , 0x98, 4);
Sbit (TB8    , 0x98, 3);
Sbit (RB8    , 0x98, 2);
Sbit (TI     , 0x98, 1);
Sbit (RI     , 0x98, 0);


/* SPI CONTROLLER */
```

```
Sfr (SPCON   , 0xC3);
Sfr (SPSTA   , 0xC4);
Sfr (SPDAT   , 0xC5);


/* I2C CONTROLLER */

Sfr (SSCON   , 0x93);
Sfr (SSSTA   , 0x94);
Sfr (SSDAT   , 0x95);
Sfr (SSADR   , 0x96);


/* KEYBOARD */

Sfr (KBCON   , 0xA3);
Sfr (KBSTA   , 0xA4);


/* ADC CONVERTER */

Sfr (ADCON   , 0xF3);
Sfr (ADDL    , 0xF4);
Sfr (ADDH    , 0xF5);
Sfr (ADCLK   , 0xF2);


#endif       /* _REGSND1_H_ */
```

ANEXO 2

DIAGRAMA ESQUEMÁTICO


AT89C51SND1C
MC74HC373A
AT28BV256
AD1851N
LM2937ET
CN USB
ISP
Play
Stop
Vol+
Vol-
VDD
VSS
VCC
- VCC
220R
100R
2.2nf
10nf
10pf
16 MHz
1.5K
27R
10uf

ANEXO 3

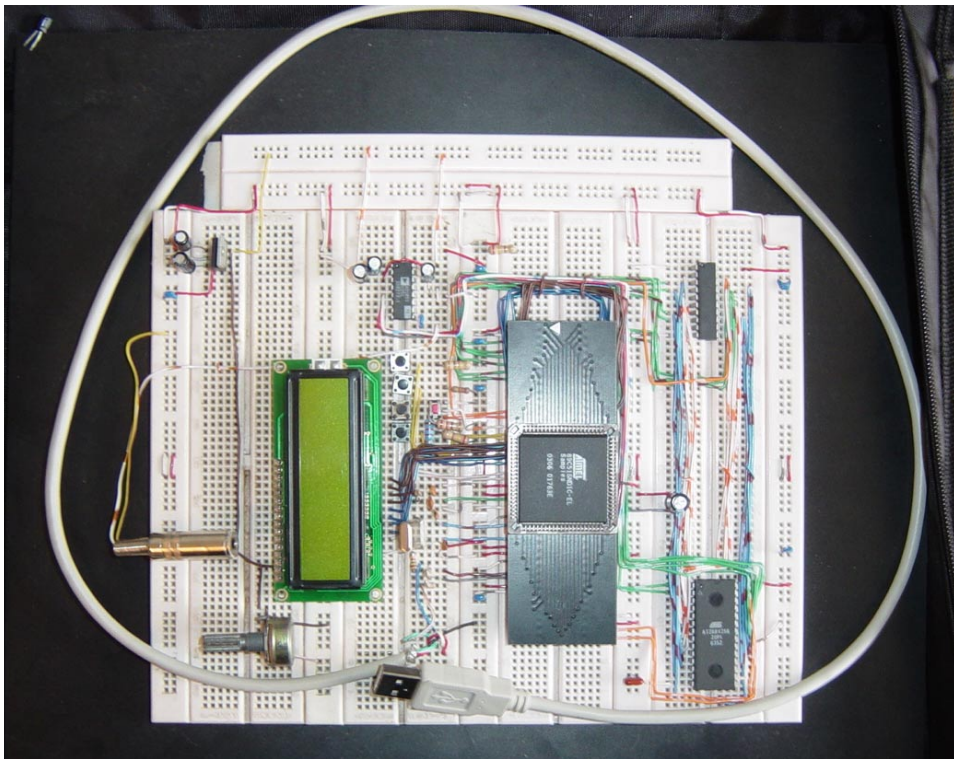
LISTADO DE ELEMENTOS

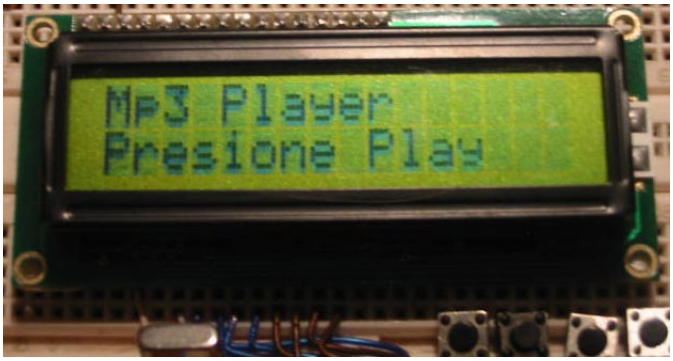
LISTADO DE ELEMENTOS

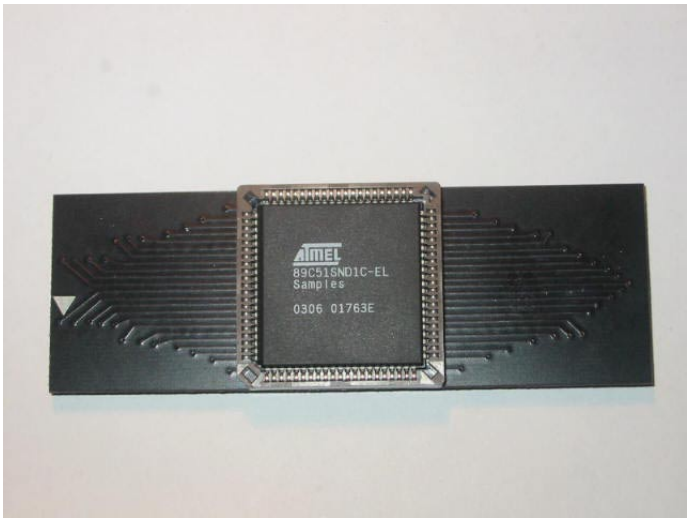
Cantidad	Descripción
1	Microcontrolador AT89C51SND1C
1	Adaptador para encapsulado PLCC
1	Memoria AT28BV256
1	DAC – AD1851N
1	Latch MC74HC373A
1	LCD
1	Regulador de voltaje LM2937ET
1	Cristal 16MHz
1	Jack Stereo Hembra
1	Conector USB
5	Pulsadores N.O.
6	Capacitor 10uf
2	Capacitor 10pf
1	Capacitor 10nf
5	Resistencia 220Ω
1	Resistencia 1.5KΩ
1	Resistencia 100Ω
2	Resistencia 27Ω
1	Potenciómetro 10KΩ

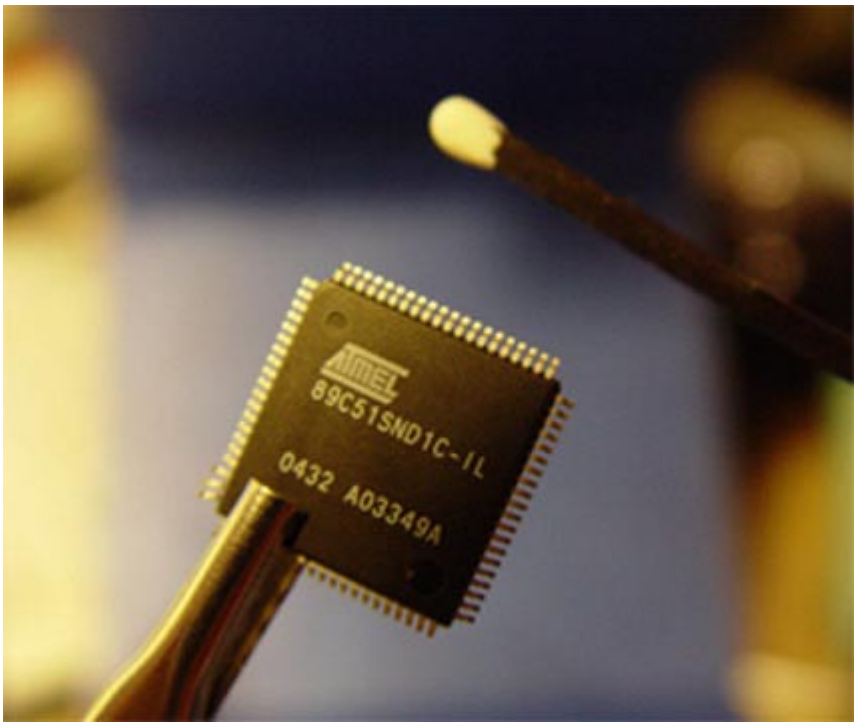
ANEXO 4

FOTOGRAFÍAS




Mp3 Player
Presione Play


AIMEL
89C51SND1C-EL
Samples
0306 01763E


AIMEL
89C51SND1C-IL
0432 A03349A

ÍNDICE DE FIGURAS

1.1. Diagrama de Bloques del Microcontrolador AT89C51SND1C 3
1.2. Microcontrolador AT89C51SND1 4
1.3. Conexión del cristal 12
1.4. Diagrama de bloques del oscilador 13
1.5. Diagrama de bloques del PLL 14
1.6. Conexión del filtro PLL 14
1.7. Diagrama de flujo para la programación del PLL 15
1.8. Memoria de programa o código 19
1.9 Arquitectura memoria flash 20
1.10 Inicialización del microcontrolador 21
1.11. Memoria de datos 23
1.12. Interfaz de memoria 25
1.13. Diagrama de bloques de la interfaz de teclado 32
1.14. Circuito de entrada de teclado 32
2.1. Codificador según la norma ISO 11172-3 41
2.2. Decodificador según la norma ISO 11172-3 42
3.1. Diagrama de Bloques del decodificador MP3 45
3.2. Diagrama de sincronismos de datos 46
3.3. Formato de salida de audio 51
4.1. Diagrama de Bloques del Ciclo de Desarrollo del Proyecto 61
4.2. Pantalla Principal de Keil µVision2 65
4.3. Ventana de Proyecto 66
4.4. Base de Datos de Dispositivos 67
4.5. Barra de Construcción de Proyecto 67
4.6. Opciones del Target 68
4.7. Barra de Edición 68

4.8. Estado de la Línea en la ventana del editor 69
4.9. Menú de Proyecto 76
4.10. Selección del Microcontrolador 77
4.11. Acceso a Opciones de la aplicación 77
4.12. Opciones del Target, Salida 78
4.13. Opciones de Target, Target 79
4.14. Agregar Archivos a un Grupo 80
4.15. Ventana de salida 81
4.16. Botón de “Debug” 82
4.17. Pantalla Principal - Modo de Depuración 83
4.18. Performance Analyzer 84
4.19. Configuración de PA (Performance Analyzer) 85
4.20. Resultados del Performance Analyzer 85
4.21. Visualización del Estado del Puerto 86
4.22. Ventana de memoria externa 87
4.23. Estado del puerto 2 87
4.24. Visualización de variables 88
4.25. Conexiones Básicas de Hardware. 89
4.26. Primer Cuadro de Dialogo de Instalación 90
4.27. Segundo Cuadro de Dialogo de Instalación 91
4.28. Tercer Cuadro de Dialogo de Instalación 91
4.29. Cuarto Cuadro de Dialogo de Instalación 92
4.30. Visualización de Dispositivo Instalado 92
4.31. Pantalla Principal 93
4.32. Cargar un archivo HEX 94
4.33. Opciones de Buffer 95
4.34. Selección de Dispositivo 95
4.35. Pantalla de Edición de Buffer 97
5.1. Conexión de memoria 101
5.2. Conexión de teclado 103
5.3. Circuito de entrada de teclado 104

5.4. Diagrama de Flujo de inicialización y bucle principal 107
5.5. Diagrama de Flujo para Interrupciones de Teclado 108
5.6. Diagrama de Flujo para Función Play 109
5.7. Diagrama de Flujo para Control de Volumen 110
5.8. Conexiones del Conversor Digital Análogo. 110
5.9. Requerimientos de las Señales para el DAC AD1851 111
5.10. Diagrama de Flujo para Configuración de Salida de Audio 113
5.11. Conexión de LCD 114

ÍNDICE DE TABLAS

1.1. Descripción de puertos 4
1.2. Descripción de señales de reloj 5
1.3. Descripción de las señales del Timer 0 y Timer 1 6
1.4. Descripción de señales de la interfaz de audio 6
1.5. Descripción de las señales del controlador USB 7
1.6. Descripción de la señal de interfaz multimedia 7
1.7. Descripción de las señales del UART 8
1.8. Descripción de las señales del controlador SPI 8
1.9. Descripción de las señales del controlador TWI 9
1.10. Descripción de las señales del conversor A/D 9
1.11. Descripción de las señales de la interfaz de teclado 9
1.12. Descripción de las señales de acceso externo 10
1.13. Descripción de las señales del sistema 10
1.14. Descripción de las señales de energía 11
1.15. Valores típicos 12
1.16. Valores típicos 14
1.17 Registros de control del la módulo de reloj 16
1.18 Registros de control del módulo señal PLL 17
1.19 Registros del divisor n 18
1.20 Registros del divisor r 18
1.21. Registro AUXR1 22
1.22. Combinaciones para selección de bancos 24
1.23. Selección de RAM 24
1.24. Señales de memoria externa 25
1.25. Registro AUXR 26
1.26. Registros principales 27

1.27. Registro de sistema 27
1.28. Registro PLL 27
1.29. Registro memoria flash 27
1.30. Registro decodificador MP3 28
1.31. Registro interfaz de audio 28
1.32. Registro interfaz IDE 28
1.33. Registro interfaz de teclado 28
1.34. Registro PCON 29
1.35. Modos de operación especial 30
1.36. Valores de capacitor 31
1.37. Registro KBCON 33
1.38. Registro KBSTA 33
2.1. Comparación de formatos de calidad de audio 38
2.2. Resumen de datos de las tres capas 42
3.1. Frecuencia de reloj de MP3 46
3.2. Control de volumen 47
3.3. Control de ecualización 47
3.4. Frecuencia de muestreo de la trama MP3 49
3.5. Registro MP3CON 52
3.6. Registro MP3STA 53
3.7. Registro MP3DAT 54
3.8. Registro MP3STA1 54
3.9. Registro MP3ANC 55
3.10. Registro MP3VOL 55
3.11. Registro MP3VOR 55
3.12. Registro MP3BAS 55
3.13. Registro MP3MED 56
3.14. Registro MP3TRE 56
3.15. Registro MP3CLK 56
3.16. Registro AUDCON0 56
3.17. Registro AUDCON1 57

3.18. Registro AUDDAT 58
3.19. Registro AUDCLK 58
4.1. Herramientas de Keil uVision 64
4.2. Menú y Comandos de Archivo (File) 70
4.3. Menú y Comandos de Edición (Edit) 70
4.4. Menú y Comandos de Vista (View) 71
4.5. Menú y Comandos de Proyecto (Project) 72
4.6. Menú y Comandos de Depuración (Debug) 72
4.7. Menú de Periféricos (Peripherals) 73
4.8. Menú de herramientas (Tools) 74
4.9. Menú de SVCS 74
4.10. Menú de Ventana (Window) 74
4.11. Menú de Ayuda (Help) 75
4.12. Bytes utilizados de Memoria 80
5.1. Registro AUXR 102
5.2. Registro IEN1 103
5.3. Registro de control de teclado KBCON 104
5.4. Tiempos de instrucciones básicas 114

GLOSARIO

Microcontrolador: Circuito integrado o chip que incluye en su interior las tres unidades funcionales de un ordenador: CPU, Memoria y Unidades de E/S, es decir, se trata de un computador completo en un solo circuito integrado. Aunque sus prestaciones son limitadas, además de dicha integración, su característica principal es su alto nivel de especialización.

MP3: Formato de audio digital comprimido con pérdida desarrollado por el Moving Picture Experts Group (MPEG) para formar parte de la versión 1 (y posteriormente ampliado en la versión 2) del formato de video MPEG. Su nombre es el acrónimo de MPEG-1 Audio Layer 3.

PCM: Siglas de Pulse Code Modulation (Modulación de Código de Pulso)

I2S: El I2S es un bus serie diseñado para dispositivos de audio digital. Es una interfase de 3 hilos, con las señales de datos y clock del audio por separado. Consiste de tres líneas de bus serie: una línea con dos canales de datos “Time-Division Multiplexing”, una línea de selección de “word”, y una línea de “clock”..

MMC: MultiMedia Card. Tipo de tarjetas de memoria, presentes en muchos dispositivos electrónicos, como cámaras digitales. Tarjeta tipo flash que sirve para almacenamiento de diversos tipos de archivos, que posee un estándar abierto regido por la MMC Association

ISP: In – System Programming (Programación en Sistema)

PLL: Siglas de phase lock loop es un componente que genera un reloj de salida por la sincronización de él mismo con un reloj de entrada y permite la multiplicación y división de frecuencias.

DAC: Conversor de digital a analógico

USB: El Bus de Serie Universal (USB, de sus siglas en inglés Universal Serial Bus) provee un estándar de bus serie para conectar dispositivos a una computadora (usualmente a una PC).

SFR: Siglas de Special Function Registers (Registros de Funciones Especiales).

Kbps: Es la abreviatura de Kilo bits por segundo. Se usa en telecomunicaciones e informática, para medir la velocidad de transferencia de información a través de una red.

IDE: Entorno de Desarrollo Integrado.

Buffer: Es una ubicación de la memoria en una computadora o en un instrumento digital reservada para el almacenamiento temporal de información digital, mientras que está esperando ser procesada

LCD: Siglas de "Liquid Crystal Display", pantalla de cristal líquido. Tipo de pantalla de bajo consumo utilizada sobre todo en ordenadores portátiles. Consiste en un conjunto de puntos muy pequeños, situados entre dos filtros polarizados, que pueden estar en modo transparente u opaco. Una fuente de luz, por detrás de la pantalla se encarga de que la imagen sea visible.

Sangolquí, 13 de octubre de 2005

ELABORADO POR:

_____	_____
Guillermo Cisneros V.	**Santiago Padilla G.**
El Decano	**El Secretario Académico**
_____	_____
Ing. Xavier Martínez Carrera	**Dr. Jorge Carvajal R**
Tcrn. de E.M.	