ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE INGENIERÍA

GENERADOR ARBITRARIO DE FORMAS DE ONDA CON LA AYUDA DE UNA TARJETA DE DESARROLLO DSP Y EL PROGRAMA LABVIEW

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL

FERNANDO JAVIER JÁCOME SAGÑAY

DIRECTOR: DR. DIEGO BENÍTEZ

Quito, octubre 2003

DECLARACIÓN

Yo, Fernando Javier Jácome Sagñay, declaro bajo juramento que el trabajo aquí descrito es de mí autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que se han consultado las referencias bibliográficas que se presentan en este documento.

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

FERNANDO JAVIER JÁCOME SAGÑAY

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por el señor Fernando Javier Jácome Sagñay, bajo mi supervisión.

DR. DIEGO BENÍTEZ

DIRECTOR DEL PROYECTO

Fernando

Dedicado a:

Este Proyecto se lo dedico primordialmente a Dios,
quién siempre ha estado apoyándome.
A mis padres y mis hermanos
a quienes quiero mucho.

Fernando

INDICE

CAPITULO 3:

DISEÑO DE LOS INSTRUMENTOS VIRTUALES DESARROLLADOS EN LABVIEW Y EL PROGRAMA DE GENERACIÓN DE FORMAS DE ONDA EN EL ADSP – 2181 39

FIGURAS

TABLAS

RESUMEN

La utilización de ciertas señales básicas se usa a menudo para probar y medir a las características de un sistema. Con la aplicación de señales básicas a un sistema, se logra conocer sus efectos.

El sistema escogido para probar sus características puede ser un filtro digital o analógico que no permita el paso de ruido eléctrico. Para probar la eficiencia de un filtro digital o analógico se necesita generar y aplicar señales con ruido eléctrico y observar su respuesta.

La generación de estas señales se puede realizar mediante el uso de un computador, en el que se seleccione el tipo de forma de onda a aplicar en el sistema.

En las paginas siguientes se describe la implementación de un Generador Arbitrario de Formas de Onda, con el cual se puede dibujar formas de onda e incluir ruido mediante el computador y, posteriormente, generar eléctricamente la señal requerida utilizando una tarjeta de desarrollo DSP.

PRESENTACIÓN

En el presente Proyecto de Titulación se implementa un Generador Arbitrario de Formas de Onda. El objetivo impuesto es el de desarrollar una interfaz de usuario que permita dibujar la forma de onda deseada o seleccionarla desde una librería, dentro de un rango de Amplitudes y frecuencias variables, y poder generarlas eléctricamente.

El tipo de Hardware y Software que se utiliza para la generación de las formas de onda se explica en detalle en el Capitulo 2, en donde se describen las características de los elementos utilizados.

El diseño e implementación de la interfaz de usuario para el control y selección del Generador Arbitrario de Formas de Onda, y el programa para la generación eléctrica, se explica en el Capítulo 3.

Por último, se exponen y discuten, las pruebas y los resultados de la generación eléctrica de distintas formas de onda creadas desde la interfaz de usuario, y ciertos efectos obtenidos, en el Capítulo 4.

CAPITULO 1

INTRODUCCIÓN

La generación de ondas desde PC es muy utilizada en cualquier sistema de test o medición. La generación de ondas Senoidales, Triangulares, Cuadradas y cierto tipo de ruido es muy utilizado en este tipo de pruebas.

Una señal senoidal es usada a menudo para probar sistemas de audio. También se pueden usar señales con ruido para probar la eficiencia y calidad de un filtro analógico y/o digital.

Un instrumento que realice todas estas señales y otras adicionales es un Generador Arbitrario de Formas de Onda. Para Implementar este Generador Arbitrario se puede utilizar un instrumento virtual programado dentro de una PC.

El objetivo del presente Proyecto de Titulación es realizar un Generador Arbitrario de Formas de Onda, con el cual se pueda generar eléctricamente una forma de onda arbitraria. Para el objetivo planteado se utilizó el programa Labview, debido a que utiliza programación gráfica y se puede depurar, a todo el programa de usuario, de una forma rápida. Además de las características propias del software, este posee librerías de generación de formas de onda y librerías de comunicación serial.

El Generador Arbitrario posee dos partes constitutivas: la interfaz de usuario en la PC, con la cual se puede dibujar o seleccionar a la forma de onda que se quiere generar; y el hardware, el cual es una tarjeta de desarrollo DSP en cuyo interior se encuentra un microprocesador, el ADSP-2181 de la marca Analog Devices, cuya arquitectura esta diseñada para realizar algoritmos de procesamiento digital de señales; con este microprocesador se realizará la comunicación serial con la PC para obtener los datos de las formas de onda dibujadas en la interfaz de usuario, y además, controlará el proceso de generación eléctrica, mediante un

conversor digital a analógico en el interior de un CODEC (Codificador/ Decodificador), ubicado en la Tarjeta de desarrollo DSP.

Debido a las limitaciones de hardware de la tarjeta de desarrollo DSP utilizada, la generación de las formas de onda se limita a una amplitud de +1 a –1 Voltios y a una frecuencia entre 10 y 4000 Hz, en pasos regulares dependiendo del rango en que se encuentra la frecuencia deseada de forma de onda. Si la frecuencia se encuentra entre 10 y 100 Hz, el paso será de 1 Hz. Si la frecuencia se encuentra entre 100 y 1000 Hz, el paso se modificará a 10 Hz. Y si la frecuencia se encuentra entre 1000 y 4000 Hz , el paso se cambiará a 500 Hz.

La interfaz de usuario permitirá al operador dibujar una forma de onda deseada en la pantalla de la PC, realizar una selección de una forma de onda básica desde un menú o dibujar una forma de onda por medio de una fórmula. Además, comunicará los valores de un periodo de la forma de onda arbitraria hacia la tarjeta de desarrollo DSP, esta se encargará de generar eléctricamente y en forma periódica a los valores trasmitidos.

1.1 EL SOFTWARE UTILIZADO: LABVIEW DE NATIONAL INSTRUMENTS

El software utilizado para realizar la interfaz de usuario es Labview 6.1. Se seleccionó este software, debido a que se puede observar el flujo de datos en el programa y realizar un seguimiento visual de la ejecución del mismo.

El programa se basa en la construcción de Instrumentos Virtuales (VI), los cuales emulan a los instrumentos físicos. Los instrumentos virtuales tienen la libertad de ser programados para varios objetivos, y gracias a la flexibilidad que poseen, se pueden realizar modificaciones rápidas y una fácil implementación.

Además, permite el cálculo de los valores de las formas de ondas básicas y su trasmisión utilizando comunicación serial, mediante la ejecución de librerías de uso dedicado.

1.2 EL HARDWARE UTILIZADO: LA TARJETA DE DESARROLLO EZ KIT LITE DE ANALOG DEVICES

La tarjeta de desarrollo EZ KIT LITE de Analog Devices posee un microprocesador DSP, el ADSP-2181, que maneja datos de 16 bits directamente y posee una arquitectura mejorada para realizar multiplicaciones y adiciones sucesivas lo más rápido posible. La tarjeta de desarrollo tiene integrado un CODEC (Codificador / Decodificador) que utiliza dos conversores Analógico a Digital y dos conversores Digital a Analógico. Este CODEC esta diseñado para ser utilizado con señales de sonido, para poder reproducir y grabar estas señales, y puede ser usado en conjunto con el microprocesador para generar señales de similares características.

En la arquitectura del microprocesador se encuentran dos puertos seriales sincrónicos; uno de ellos se lo utiliza para realizar una interfaz con el CODEC, y poder generar o almacenar señales; y el puerto serial restante se lo emplea para la comunicación con la PC. Debido a las características antes mencionadas, se escogió a este microprocesador como el más apto para la función de Generador Arbitrario de Formas de Ondas. El resto de dispositivos integrados en la tarjeta se explica en detalle en los siguientes Capítulos.

CAPITULO 2

DESCRIPCIÓN DE LOS ELEMENTOS USADOS EN LA IMPLEMENTACIÓN DEL GENERADOR ARBITRARIO DE FORMAS DE ONDA

2.1 INTRODUCCIÓN

En este Capítulo se describen tanto el hardware como el software utilizado para la implementación del Generador Arbitrario de Formas de Onda, desarrollado en el presente Proyecto de Titulación. El hardware utilizado es una tarjeta de desarrollo EZ KIT LITE, la misma que consta de: el microprocesador ADSP–2181; una memoria EPROM desde donde se descarga un programa llamado Monitor, que realiza una prueba general a todos los registros del DSP; un CODEC (Codificador / Decodificador) AD1847 que consta de dos conversores Análogo/Digital (A/D) y dos conversores Digital/Análogo (D/A) de modulación Sigma-Delta ($\Sigma\Delta$); y un acoplador de voltajes ADM232AAR empleado en la comunicación serial asincrónica RS-232, emulada por el puerto serial sincrónico SPORT1 y controlada desde el programa Monitor. Por medio de esta comunicación serial se descargan los programas de usuario y datos hacia la memoria del microprocesador.

El software utilizado en la interfaz de usuario es Labview 6.1 de National Instruments que utiliza programación gráfica y posee Instrumentos Virtuales (VIs) de generación de señales y comunicación serial.

2.2 ARQUITECTURA DEL ADSP–2181

2.2.1 DESCRIPCIÓN GENERAL

El ADSP-2181 es un microprocesador programable que tiene una arquitectura base optimizada para realizar algoritmos de procesamiento digital de señales y

para aplicaciones de procesamiento numérico de alta velocidad [1]. Adicionalmente a la arquitectura base, posee ciertos periféricos en su interior como son: dos Puertos seriales, un Temporizador, dos Puertos DMA (Acceso Directo de Memoria) y pines de uso general. La alimentación de voltaje es de +5 Vdc. Además, el ADSP-2181 ejecuta 33 MIPS (Millones de Instrucciones por segundo), posee la capacidad de realizar múltiples instrucciones en un solo ciclo, y maneja directamente palabras de datos de 16 bits.

El microprocesador ADSP-2181 posee: tres Unidades Computacionales, un secuenciador de programa y dos generadores de direcciones de datos, en su arquitectura base. Los periféricos incluidos en el chip son: dos puertos DMA, un temporizador, dos puertos seriales, interrupciones externas y banderas de uso general accesibles al usuario. También posee en su interior dos tipos de memoria: memoria RAM de programa, en la cual se almacenan las instrucciones del programa de usuario; y memoria RAM de datos, para guardar valores y parámetros.

El microprocesador posee 5 buses internos. El Bus de direcciones de memoria de programa (PMA) y el Bus de direcciones de memoria de datos (DMA), los cuales son usados internamente para dotar de las direcciones asociadas con memoria de programa y de datos. El Bus de datos de memoria de programa (PMD) y el Bus de datos de memoria de datos (DMD), los cuales son usados para trasladar los datos asociados con los espacios de memoria respectivos. Y el bus de resultados (R), que transfiere resultados entre las Unidades Computacionales. Todos estos buses de datos y direcciones son multiplexados en un bus externo de direcciones y en un bus externo de datos.

Tanto el bus PMA y el bus DMA manejan palabras de 14 bits, permitiendo direccionar hasta 16 K palabras. El bus PMD maneja palabras de 24 bits y permite acceder a las instrucciones de programa; y el bus DMD maneja palabras de 16 bits y sirve para transferir el contenido de ciertos registros a localidades de memoria de datos.

Las ventajas que presentan los microprocesadores DSP frente a otros microprocesadores o microcontroladores son [1]:

- Desarrolla múltiples operaciones en paralelo, en un solo ciclo, como son:
    - Generar la siguiente dirección de programa
    - Traer la siguiente instrucción
    - Realizar uno o dos movimientos de datos
    - Actualizar uno o dos punteros de datos
    - Realizar una operación computacional
- Realiza operaciones mientras la arquitectura base esta procesando, como son:
    - Recibir y trasmitir datos a través de los puertos seriales
    - Recibir y trasmitir datos a través del puerto interno de 16 bits
- *Aritmética rápida y flexible*.- Gracias al multiplicador / acumulador se puede realizar una cantidad grande de operaciones de multiplicación y suma o multiplicación y resta, sucesivas.
- Obtención de dos operandos en un solo ciclo
- *Buffer Circulares en Hardware*.- Los algoritmos DSP, como son filtros digitales, requieren de buffers circulares de datos. La arquitectura del ADSP-2181 posee hardware para manejar punteros y facilitar la implementación de buffers circulares.

2.2.2 DESCRIPCIÓN DEL DIAGRAMA DE BLOQUES DEL ADSP-2181

El diagrama de bloques mostrado en la Figura. 2.1 [1], muestra las tres Unidades Computacionales: La Unidad Aritmética y Lógica (ALU), que realiza operaciones aritméticas y lógicas básicas; el Multiplicador / Acumulador (MAC), el cual ejecuta multiplicaciones con adición o multiplicaciones con substracción, en un solo ciclo; y la Unidad de Desplazamiento, utilizada en operaciones de desplazamiento lógico o aritmético, normalización, denormalización, y derivación de exponente para implementación de aritmética de punto flotante.

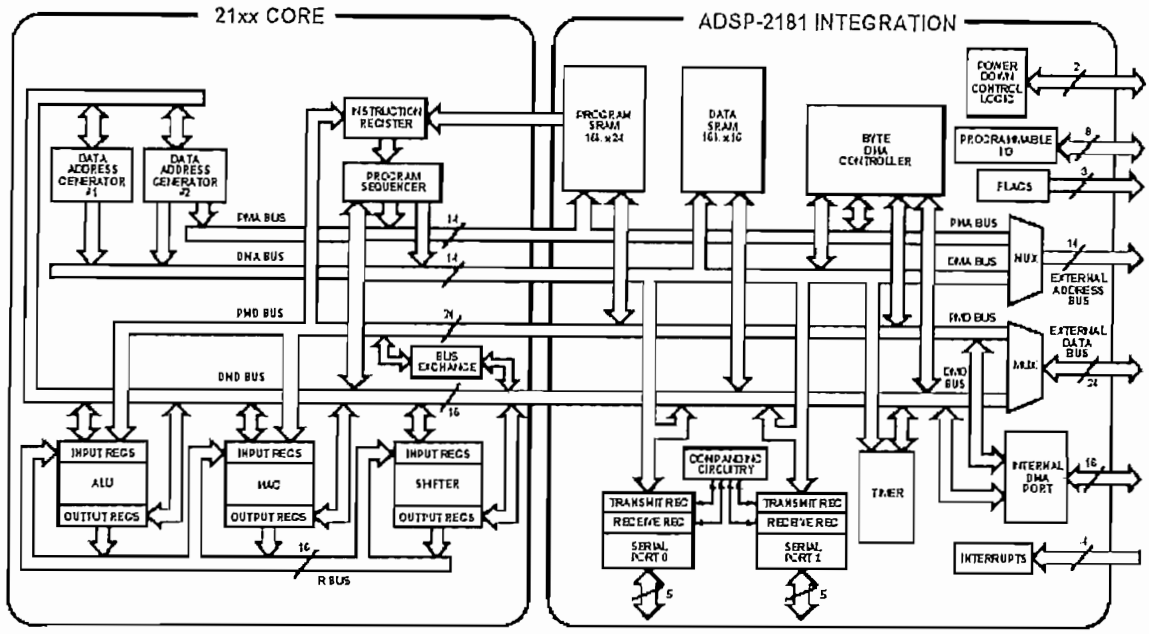


Figura 2.1 Diagrama de Bloques del ADSP-2181

También se puede observar al Secuenciador de programa, y a los Generadores de Direcciones de Datos (DAGs) los cuales proveen de los recursos necesarios para el funcionamiento de cada una de las Unidades Computacionales. El Secuenciador de Programa provee la dirección de la siguiente instrucción de programa y controla el flujo de la ejecución. En cambio cada DAG provee de la dirección del operando a usarse en las unidades computacionales; para obtener el operando posee 4 punteros de datos, cada puntero posee un registro de modificación y de longitud, empleados para direccionamiento directo e indirecto de localidades de memoria, y para direccionamiento hacia buffer circulares.

La unidad de intercambio de bus (Bus Exchange) permite trasladar datos de memoria de datos (16 bits) a memoria de programa (24 bits) y viceversa.

La memoria interna SRAM esta dividida en 16 k palabras de 16 bits para datos y 16 k palabras de 24 bits para programa.

Uno de los dos puertos DMA (Acceso Directo a Memoria), es el puerto interno DMA de 16 bits (IDMA), al cual se lo utiliza para interconexión a sistemas

externos, y utiliza 21 bits de entrada y salida; 5 bits son señales de control y los restantes bits se multiplexan para direcciones y datos. El segundo puerto DMA, es el puerto Byte DMA (BDMA) que se utiliza para acceso a memorias de 8 bits. Este puerto es bidireccional y puede acceder a memorias externas de hasta 4 megabytes, en donde se puede almacenar instrucciones o tablas de datos. Además es muy usado para descargar al programa de usuario, después de un reset.

El ADSP-2181 posee dos puertos seriales sincrónicos (SPORTs). Los puertos seriales indicados en el diagrama de bloques poseen una sección de trasmisión y una sección de recepción, que trabajan independientemente.

El temporizador (Timer) produce una interrupción cada cierto intervalo programado de tiempo. El intervalo programado es un múltiplo del tiempo utilizado en realizar un ciclo en el microprocesador.

Adicionalmente posee 12 interrupciones; 6 externas (1 activada por flanco, 2 activadas por nivel, y 3 son configurables) y 6 internas (1 del Timer, 2 de los puertos seriales, 1 del puerto BDMA y el resto por el circuito de bajo consumo). También posee 13 pines para acceso del usuario; 8 pines configurables de entrada / salida, 3 pines para banderas solo de salida y los demás provienen del puerto serial 1 al configurarse alternativamente como banderas de entrada / salida.

Existen 3 modos de operación extras al modo normal, estos son: Powerdown (Bajo Consumo), Idle y Slow Idle. El modo Powerdown decrementa el consumo de potencia a 1mW mediante controles en hardware. El modo Idle es activado por software, y espera en bajo consumo hasta que se realice una operación; y el Modo Slow Idle, disminuye la frecuencia de la señal de reloj del microprocesador, y entra en modo de bajo consumo.

2.2.3 UNIDADES COMPUTACIONALES Y FUNCIONALES

2.2.3.1 Unidad Aritmética y Lógica (ALU)

La Unidad Aritmética y Lógica (ALU) maneja datos de 16 bits, que pueden expresarse en diferentes formatos, como son:

- Datos de 16 bits sin signo, solo positivos (Unsigned).
- Dato de 16 bits con signo, en complemento de dos (Signed).
- Datos de 16 bits en representación fraccional (Formato 1.15).

El Formato 1.15 es un caso especial de la representación fraccional, y significa que utiliza 1 bit para la parte entera y 15 bits para la parte fraccional, por ejemplo:

Formato 1.15 (Hexadecimal)	**Decimal**
0x0001	0.000031
0x7FFF	0.999969
0XFFFF	-0.000031
0X8000	-1.000000

El peso de cada bit en formato 1.15, se muestra en la Figura 2.2 [1].

-2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}

Figura 2.2 Peso de cada bit en números con formato 1.15

Las funciones aritméticas básicas de la ALU son: sumar, restar, negar, incrementar, decrementar y dar a un número, su valor absoluto, con operandos de 16 bits. Las funciones lógicas de la ALU son: AND, OR, XOR Y NOT.

Como se indica en la Figura 2.3 [1], la ALU posee puertos de entrada de 16 bits conformados por: el puerto X y el puerto Y. El puerto de salida es el puerto R. El puerto X puede aceptar a dos fuentes: el archivo de registro AX y el bus de resultado R que está conectado a los registros de salida de cada una de las Unidades Computacionales.

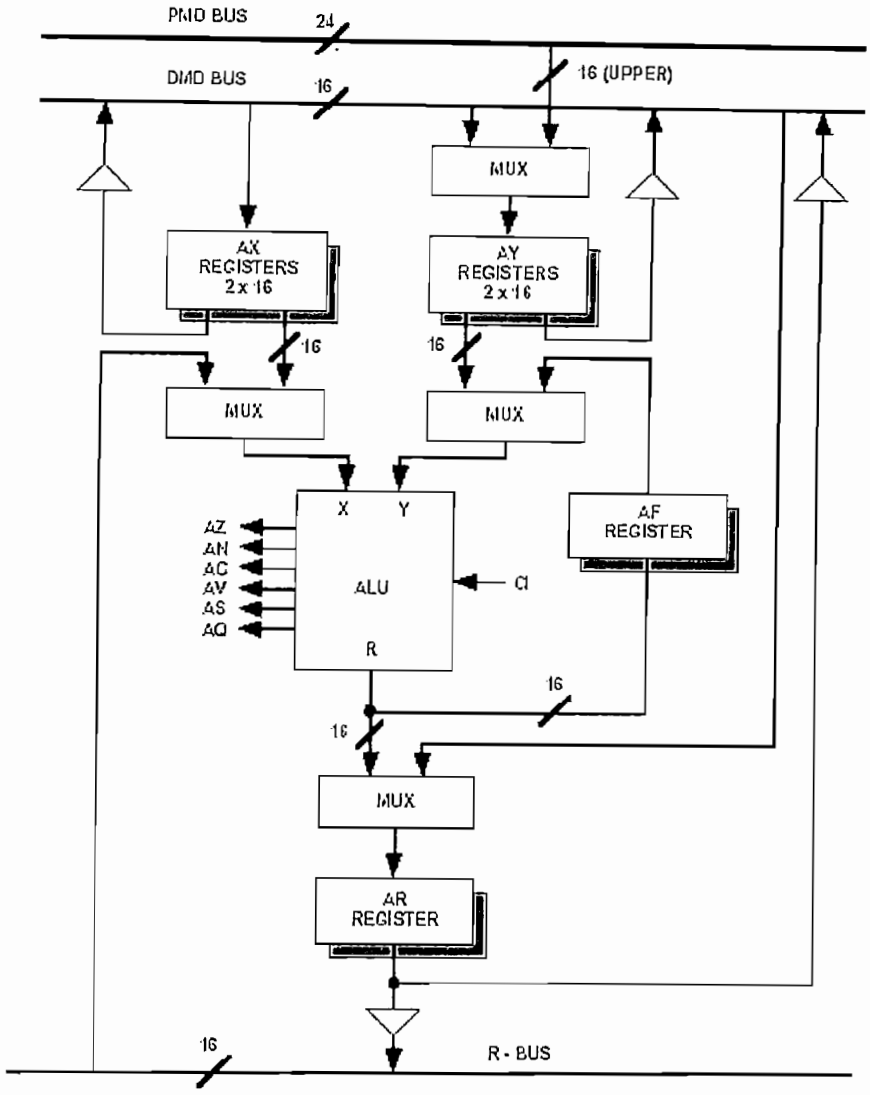


Figura 2.3 Diagrama de Bloques de la ALU

El archivo de registro AX posee dos registros: AX0 y AX1 que pueden ser escritos o leídos desde el bus DMD y también desde el bus PMD, por medio de la unidad de intercambio de bus; además, estos nombres (AX0 y AX1), son usados en las instrucciones correspondientes a la ALU. El puerto Y puede aceptar también dos fuentes: el archivo de registro AY y el registro de realimentación AF. El archivo de registro AY posee dos registros: AY0 y AY1, que igualmente pueden ser escritos o leídos desde el bus DMD o desde PMD por medio de la unidad de intercambio.

Los bits de estado que posee la ALU se encuentran en el registro ASTAT (registro de Estado) y son:

- Bit de estado cero (AZ)
- Bit de estado negativo (AN)
- Bit de carry (AC)
- Bit de Sobreflujo (AV)
- Bit de estado de signo de entrada de X (AS)
- Bit de cociente (AQ)

Además, la ALU posee dos bancos de registros de: AR, AF, AX, y AY; solo un banco se puede utilizar a la vez. El cambio de banco de registros es usado cuando se ejecuta una subrutina de atención de interrupción, y así, no alterar los datos del otro banco de registros, usados en el programa principal.

2.2.3.2 Multiplicador/Acumulador (MAC)

La función del Multiplicador/Acumulador (MAC) es la de proveer multiplicaciones a alta velocidad; además realiza multiplicación con adición acumulativa, multiplicación con substracción acumulativa, saturación y funciones de seteo a cero.

En la Figura 2.4 [1] se muestra el MAC; este posee dos puertos de entrada X y Y, y un puerto de salida P. El puerto de entrada X se puede cargar desde dos fuentes: el archivo de registro MX y el bus de resultado R. El archivo de registro MX posee dos registros: MX0 y MX1, que se pueden escribir o leer desde el bus DMD o PMD. El puerto de entrada Y posee dos fuentes: el archivo de registro MY y el registro de realimentación MF, que utiliza los bits 16-31 del resultado, y lo realimenta al puerto Y. El archivo de registro MY posee dos registros: MY0 y MY1, leídos o escritos desde el bus DMD o PMD. El puerto de salida P es de 32 bits y lleva el resultado de la multiplicación, este resultado puede pasar al Adicionador/ Substractor de 40 bits. El Adicionador/Substractor suma el nuevo producto (P) con el anterior almacenado en el registro MR, que tiene el resultado final, o lo deja pasar desde (P) a MR. El registro MR esta constituido de 3 partes: el registro

MR0, MR1 de 16 bits Y MR2 de 8 bits que pueden ser leídos desde el bus DMD o el bus PMD.

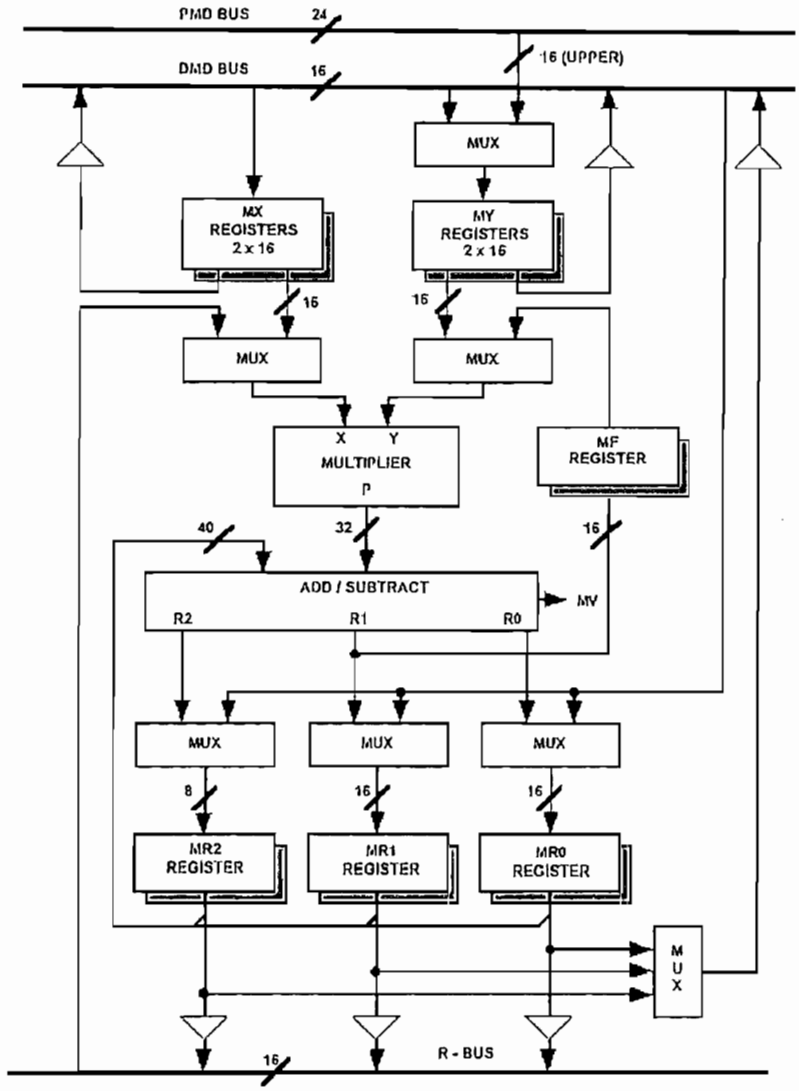


Figura 2.4 Diagrama de Bloques del MAC

El MAC puede funcionar en dos modos: Modo Fraccional (1.15), en el cual los operandos deben estar en formato 1.15; y Modo Entero (16.0), es decir 16 enteros.

Las funciones estándar del MAC son: Multiplicación (X * Y), Multiplicación con adición (MR + [X*Y]), Multiplicación con Substracción (MR – [X*Y]) y seteo a cero (MR = 0).

2.2.3.3 Unidad de Desplazamiento (Barrel Shifter)

La Unidad de Desplazamiento realiza desplazamientos lógicos (LSHIFT) y aritméticos (ASHIFT); y además realiza normalización (NORM), detección de exponente (EXP), y ajuste de exponente de bloque (EXPADJ). En la Figura 2.5 [1] se observa el diagrama de bloques de la Unidad de Desplazamiento. La entrada del desplazador es un dato de 16 bits y su salida es de 32 bits, debido a que puede realizar desplazamientos de 16 lugares. La entrada tiene una fuente, el registro de entrada del desplazador SI, que acepta datos de 16 bits; y la salida es por medio del registro de salida SR, que es de 32 bits. El registro SR esta dividido en dos registros: SR0 y SR1. La unidad de desplazamiento posee un banco alternativo de registros de SE, SB, SI, SR1 y SR0.

Además, la unidad de desplazamiento posee: el registro de exponente SE, el cual mantiene el exponente en procesos de Normalización (procesos de conversión de punto fijo a punto flotante) y en procesos de Denormalización (procesos de conversión de punto flotante a punto fijo); y el registro de bloque SB, el cual retiene el exponente de un bloque de datos. La presencia de un exponente en la entrada del desplazador, se realiza mediante el Detector de Exponente.

El desplazamiento de la entrada depende de dos señales: la señal de control de código (C) y la señal HI/LO. La señal de control de código (C) define la dirección del desplazamiento, si el valor cargado a este es positivo, el desplazamiento es hacia la izquierda; y si es negativo, es hacia la derecha. El valor numérico del control de código (C) indica los lugares a desplazarse. El control de código (C) puede ser cargado desde 2 fuentes: el registro SE y el valor indicado en la instrucción. La señal HI/LO indica el punto de referencia para el desplazamiento, si es HI, el punto de referencia es el registro SR1; y si es LO, el punto de referencia es el registro SR0.

La Unidad de Desplazamiento se puede llenar de ceros, a los espacios restantes, a la derecha o izquierda del valor de entrada, mediante el bit de extensión (X), el

cual tiene tres fuentes: el bit más significativo de la entrada, el bit AC del registro ASTAT o un valor en la instrucción de cero.

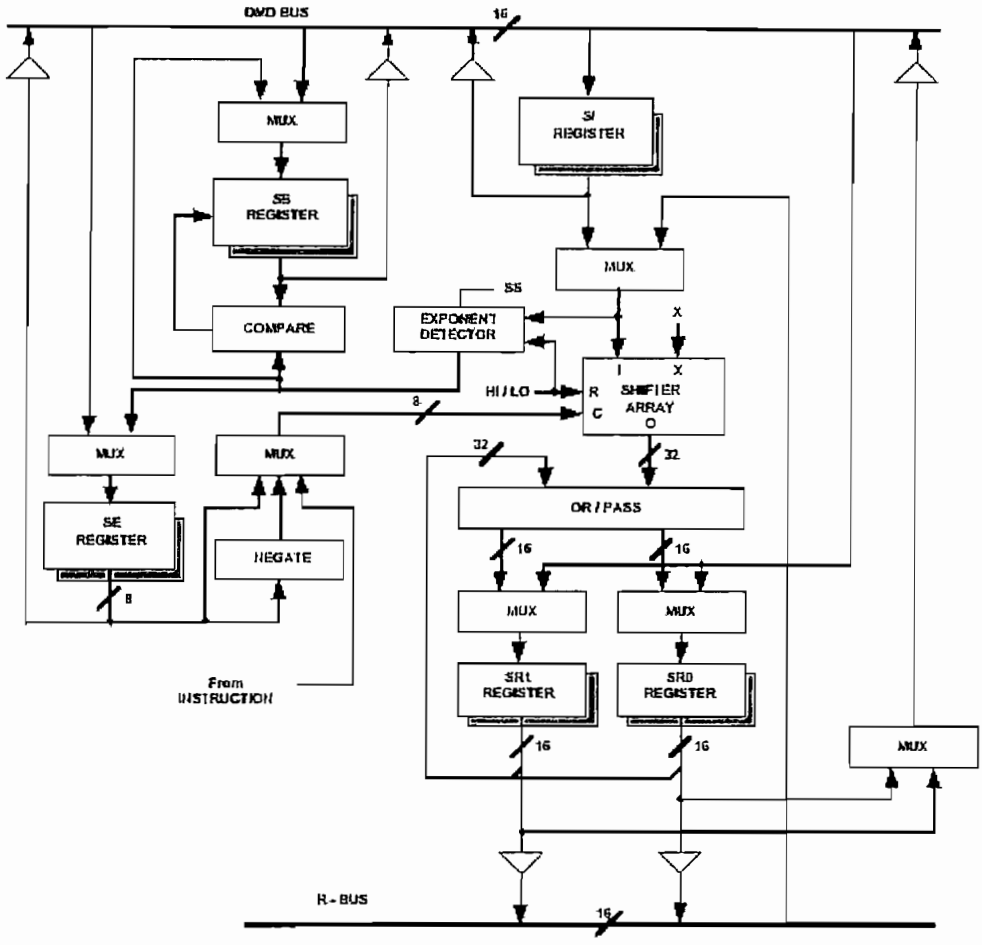


Figura 2.5 Diagrama de Bloques de la Unidad de Desplazamiento

La lógica OR/PASS realiza una función OR de la salida con el registro SR o deja pasar la salida hacia el registro SR. Esta función es muy utilizada en operaciones de desplazamiento lógico y aritmético.

La diferencia entre un desplazamiento lógico y un desplazamiento aritmético, es el valor de los bits restantes del desplazamiento. Para un desplazamiento lógico, el valor de los bits restantes es cero; y para un desplazamiento aritmético es uno, a continuación se muestra un ejemplo:

Desplazamiento Lógico.

SI = 0xB6A3; (se carga al registro SI con el valor indicado).
SR = LSHIFT SI BY –5 (HI); (LSHIFT: Desplazamiento Lógico, referencia a SR1)
Entrada: 10110110 10100011.
Valor de desplazamiento: -5 (hacia la izquierda).
SR: 00000**101 10110101 00011**000 00000000.

Desplazamiento Aritmético.

SI = 0xB6A3; (se carga al registro SI con el valor indicado).
SR = ASHIFT SI BY –5 (HI); (ASHIFT: Desplaz. Aritmético, referencia a SR1).
Entrada: 10110110 10100011.
Valor de desplazamiento: -5 (hacia la izquierda).
Referencia (HI): La referencia es el registro SR1 (16 bits mas hacia la izquierda).
SR: 11111**101 10110101 00011**000 00000000.

2.2.3.4 Secuenciador de Programa

El Secuenciador de Programa controla el flujo de la ejecución y provee las direcciones del programa. Este esta constituido de un controlador de interrupciones, y una lógica de estado y condición. El Secuenciador de Programa permite la ejecución secuencial, de lazos, realiza servicios de interrupción, y realiza saltos en un solo ciclo con instrucciones como Jump (salto condicional) y Call (salto incondicional). Las instrucciones usadas en control de flujo de programa son: DO UNTIL, JUMP, CALL, RTS, RTI e IDLE.

El Secuenciador de Programa preobtiene la siguiente instrucción mientras se ejecuta la anterior. La lógica de selección de la siguiente dirección proviene de 4 fuentes: El Incrementador del Contador de Programa (PC), la Pila del PC, el registro de instrucciones o el Controlador de Interrupciones.

El Incrementador del PC es usado cuando la ejecución del programa es secuencial. La Pila del PC es utilizada cuando se retorna de una subrutina de servicio a una interrupción. El registro de instrucciones es empleado cuando se tiene un salto (JUMP). El controlador de interrupciones provee de la dirección de la siguiente instrucción cuando atiende a una interrupción, generalmente salta a la localidad del vector de la interrupción. Otra fuente puede ser los registros Índice (I4 - I7) cuando se realiza saltos indirectos.

El Contador de Programa (PC) es un registro de 14 bits que contiene la dirección de la instrucción ejecutándose. Mientras se ejecute secuencialmente el programa, la salida del PC se realimenta hacia el Incrementador del PC, que aumenta en 1 a la dirección que contiene el PC; la salida del Incrementador es la dirección de la próxima instrucción. La Pila del PC posee 16 palabras de 14 bits y en esta se coloca la salida del Incrementador cuando se ejecuta una instrucción CALL, DO UNTIL o se ejecuta interrupciones.

Otra parte importante del Secuenciador de Programa es el Contador de Lazos y la Pila de Lazos. El contador de lazos (CNTR) es un registro de 14 bits, con decremento automático, que indica el número de lazos a realizarse. La Pila del Contador de lazos posee 4 palabras de 14 bits, y es utilizada cuando se realizan lazos anidados. El bit CE (Counter Expired) de la Lógica de Selección, indica si se ha terminado de realizar los n lazos indicados en el registro CNTR (Counter). El registro CNTR es cargado desde DMD.

El Comparador de Lazos es el encargado de comparar la instrucción generada por el secuenciador, con la última instrucción del lazo; si cumple con la condición de terminación, sale del lazo. La Pila del comparador de lazos tiene 4 palabras de 18 bits, los 14 bits para la dirección de la ultima instrucción del lazo y los 4 bits poseen la condición de terminación.

Las interrupciones generadas, producen un salto de la ejecución de programa a las localidades del vector de cada una de las interrupciones. Cada localidad del vector de interrupciones tiene 4 localidades subsiguientes, que pueden ser

usadas para trasladar la ejecución hacia subrutinas de atención a interrupción, al terminar la subrutina debe tener la instrucción RTI para regresar el control de ejecución al programa principal. En la Figura 2.6 [1] se indica el diagrama de bloques del Secuenciador de Programa.

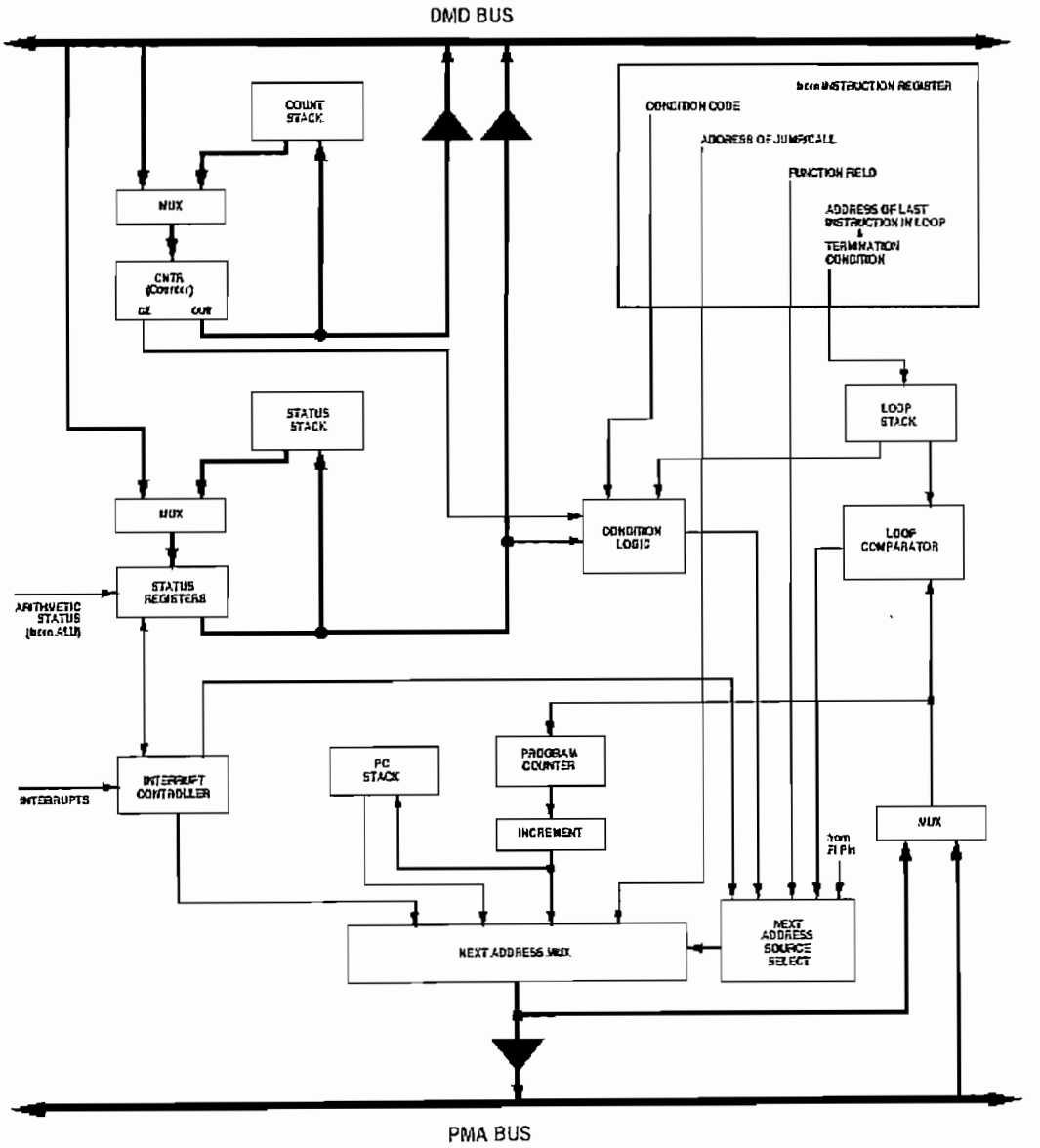


Figura 2.6 Diagrama de Bloques del Secuenciador de Programa

La Tabla 2.1 [1] muestra las localidades de cada una de las interrupciones.

Fuente de Interrupción	*Dirección del Vector de Interrupción*
RESET	0000 (prioridad mas alta)
POWERDOWN	002C
IRQ2	0004
IRQL1 (sensitiva a nivel)	0008
IRQL0 (sensitiva a nivel)	000C
SPORT0 (Trasmisión)	0010
SPORT0 (Recepción)	0014
IRQE (sensitiva en flanco)	0018
Byte DMA interrupción	001C
SPORT1 (Trasmisión) o IRQ1	0020
SPORT1 (Recepción) o IRQ0	0024
TIMER	0028 (prioridad mas baja)

Tabla 2.1 Direcciones de las localidades de los vectores de interrupción del ADSP-2181

Cada interrupción es mantenida hasta terminar de ejecutar la instrucción actual, luego se compara el pedido de interrupción con el registro de enmascaramiento IMASK; si no esta enmascarada la interrupción, se ejecuta el servicio de atención a interrupción, produciéndose un almacenamiento del valor del contador de programa; al terminar la interrupción, se realiza una recuperación de este valor, para regresar a la dirección inicial de donde salto hacia la localidad del vector de interrupción.

La configuración de las interrupciones se realiza con tres registros: el registro ICNTL (registro de Control de Interrupciones) de 5 bits, que determina el anidamiento de las interrupciones y configura a IRQ2, IRQ1, y a IRQO como sensitiva a nivel (si es cero Lógico) o a flanco (si es uno Lógico); el registro IMASK (registro de Enmascaramiento de interrupciones) habilita o deshabilita cada interrupción interna o externa, cada bit representa una interrupción, si es cero lógico, la interrupción es enmascarada y no es atendida, y si es uno lógico, la interrupción es ejecutada; y el registro IFC, que fuerza una interrupción o limpia

una interrupción pendiente por flanco. La interrupción IRQE es sensitiva a flanco y la interrupción IRQL1 y IRQL2 son sensitivas a nivel.

Para control del programa se provee de una pila de Estado y de registros de Estado los cuales son: ASTAT (registro de Estado Aritmético), SSTAT (registro de Estado Aritmético), MSTAT (registro de Estado del Modo Usado), con estos registros se monitorea el estado del procesador.

2.2.3.5 Generadores de Direcciones de Datos (DAGs)

El microprocesador ADSP-2181 posee dos DAGs, uno es utilizado para dar direcciones de memoria de programa y otro es destinado para dar direcciones de memoria de datos, los cuales actúan al mismo tiempo. Además los DAGs poseen direccionamiento indirecto y poseen la característica de realizar buffers circulares. Un buffer circular es un conjunto de palabras de 16 bits con una longitud definida, que al termino de utilizar la última palabra, definida en la longitud, regresa a la palabra inicial del buffer, todo esto implementado en hardware.

Los registros de los DAGs se dividen en tres clases: los registros de Modificación (M) de 14 bits, se dividen en dos grupos: M0-M3 (DAG1) y M4-M7 (DAG2); los registros de Longitud (L) de 14 bits, que indica la longitud del buffer circular; y los registros de Índice, usados como punteros en direccionamiento indirecto y en buffers circulares, se dividen en dos grupos: I0-I3 (DAG1) y I4-I7 (DAG2). Para la implementación de buffers circulares, el registro L debe contener la longitud de palabras del buffer; y para direccionamiento lineal, debe ser igual a uno. En la Figura 2.7 [3] se indica el funcionamiento de un buffer circular.

El direccionamiento indirecto se realiza colocando una dirección en el registro I a utilizarse como puntero, y se lo modifica desplazándolo con el registro M. Tanto la declaración de la dimensión de un buffer lineal como circular, se lo realiza en el archivo fuente con extensión .DSP, por medio de directivas.

Un ejemplo de direccionamiento indirecto se indica a continuación:

I3 = 0x3B46;

M2 = 0;

L3 = 0;

AX0 = DM (I3,M2); (Traspasa el contenido apuntado por I3 a AXO).

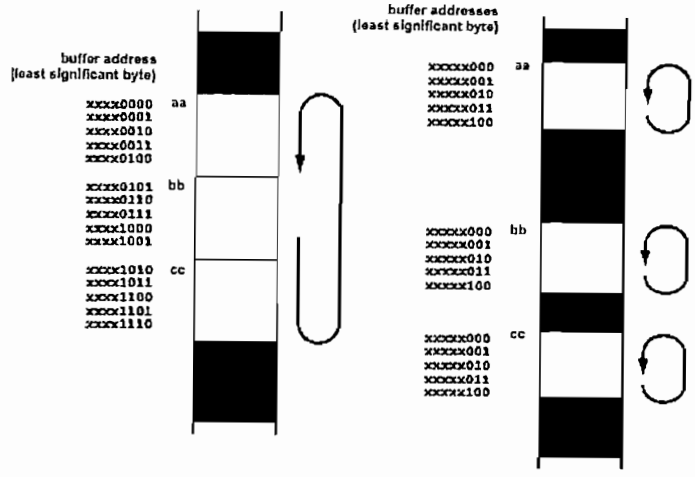


Figura 2.7 Funcionamiento de un Buffer Circular

Otra parte importante es la Unidad de Intercambio de Bus, que permite trasladar datos desde memoria de datos hacia memoria de programa y viceversa. Debido a que el bus de memoria de programa maneja datos de 24 bits, al trasladar desde el bus de memoria de 16 bits se utiliza al registro PX para cargar los 8 restantes bits.

2.2.3.6 Puertos Seriales (SPORTs)

El ADSP-2181 posee dos puertos seriales sincrónicos SPORT0 y SPORT1. La descripción de los pines se muestra en la Tabla 2.2 [1] a continuación:

Nombre del Pin	***Función***
SCLK	Señal de Reloj Serial
RFS	Trama de sincronización de recepción
TFS	Trama de sincronización de trasmisión
DR	Recepción de dato serial
DT	Trasmisión de dato serial

Tabla 2.2 Descripción de los pines de un Puerto Serial

Cada sección de trasmisión y recepción de cada puerto serial es independiente, permitiendo realizar una comunicación Full-Duplex. Cada bit es enviado o recibido en forma sincrónica con la señal de reloj serial, la cual se puede generar o recibir externamente. Puede o no operar con señales de sincronización de trama que pueden ser generadas internamente o recibidas externamente desde un dispositivo, como sucede con el Codec (AD1847), las cuales se usan para indicar que inicia una trasmisión o recepción de datos.

Además, los puertos seriales poseen la propiedad de AUTOBUFERADO, que permite la trasmisión o recepción de un buffer completo de datos en forma automática, mediante la declaración del uso de registros I, L y M, como punteros automáticos. Con la habilitación de este modo, al completar la trasmisión o recepción completa de datos de un buffer, se produce una interrupción.

También posee la función Multicanal, con la cual se puede escoger selectivamente a canales de datos. Un canal de datos es una determinada palabra de un bloque de datos. Si el bloque esta constituido de 24 palabras, con la función Multicanal, se tendrían 24 canales. Con este modo de operación, un flujo de bits seriales puede ser dividido en 24 o 32 canales. De esta forma se puede recibir o trasmitir un determinado canal.

El SPORT1 se puede configurar alternativamente como 2 interrupciones externas; IRQO e IRQ1; y, los pines del puerto serial se convierten en banderas de entrada y banderas de salida. La señal de reloj SCLK, generada internamente o externamente, no se usa en la configuración alternativa del SPORT1.

Existe una interrupción para trasmisión y recepción, para cada uno de los SPORTs. El registro usado en la trasmisión, es el registro TX, y el registro RX es para recepción.

En la Figura 2.8 [1] se muestra el diagrama de bloques de un puerto serial. En esta Figura se observa las secciones independientes de trasmisión y recepción, y junto a ellas la circuitería de generación y recepción de la señal de reloj serial. El

procedimiento a seguir por los puertos seriales, para trasmisión. es el siguiente: los datos a trasmitir se cargan en el bus DMD por medio del registro TX; luego, por medio de la señal de sincronización de trama para trasmisión (TFS), los datos cargados en el registro TX, pasan al registro de desplazamiento de trasmisión para luego enviarlos externamente; el bit más significativo es enviado primero y los demás son enviados con el flanco positivo del reloj serial (SCLK); al terminar de trasmitir los bits se genera una interrupción.

En cambio al recibir datos, estos se cargan en el registro interno de recepción. Al completar con la recepción de todos los bits, la palabra es escrita en el registro RX, con lo cual se realiza una interrupción de recepción.

Otra parte del diagrama de bloques muestra el circuito para Companding en hardware, el cual permite modulación en código por pulso (PCM), por medio de métodos de decodificación μ-law y A-law. Estos métodos sirven para cuantización de formas de onda analógicas, muy usadas en trasmisión de señales de voz.

La programación del puerto serial se inicia habilitando al puerto mediante el registro de control de sistema, mapeado en memoria de datos en la dirección 0x3FFF. El bit 12 habilita el SPORT0, el bit 11 habilita al SPORT1, y el bit 10 configura alternativamente al SPORT1. Luego se debe indicar la longitud de la palabra serial, establecida en los bits 0-3 del registro de control del SPORT0 o SPORT1.

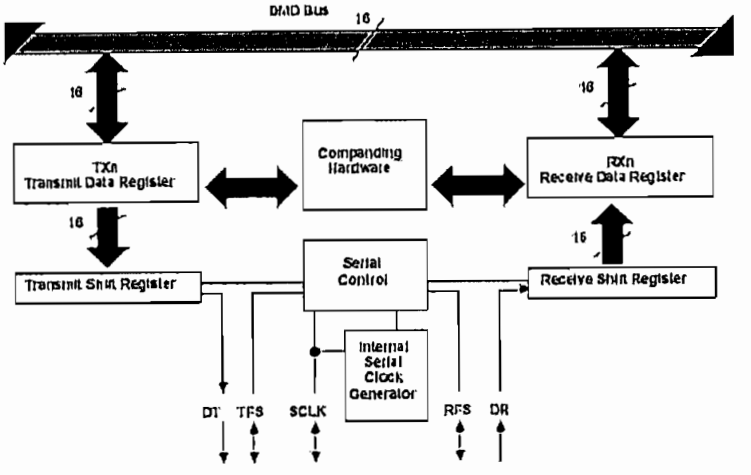


Figura 2.8 Diagrama de Bloques de un Puerto Serial (SPORT)

Posteriormente, se configura al puerto serial para recibir o generar la señal de sincronización de trama, en los bits 8 y 9 del registro de control del SPORT, y por último se configura, en los bits 10 y 12, al modo normal o alternante de trama.

El modo de Autobuferado es configurado en el registro de control de Autobuferado del SPORT0 o SPORT1, que se encuentra mapeado en memoria en la dirección 0x3FF3 para SPORT0 Y 0x3FEF para el SPORT1. En este registro es configurada la habilitación de este modo; por medio del bit 1, para la sección de trasmisión, y con el bit 0 para recepción; además se declaran los registros I, L y M usados exclusivamente para este modo de operación. No se pueden utilizar estos registros declarados, para ningún otro propósito, mientras este habilitado el Modo de Autobuferado.

La función Multicanal utiliza el registro de control del SPORT y los registros de habilitación de palabra Multicanal del SPORT, mapeados en memoria de datos en las direcciones 0x3FF7 – 0x3FFA. Cada bit indica si un canal es ignorado o no. En la Figura 2.9 [1] se muestra una recepción serial en modo Multicanal, ignorando la palabra 1, y con una longitud de palabra de 4 bits.

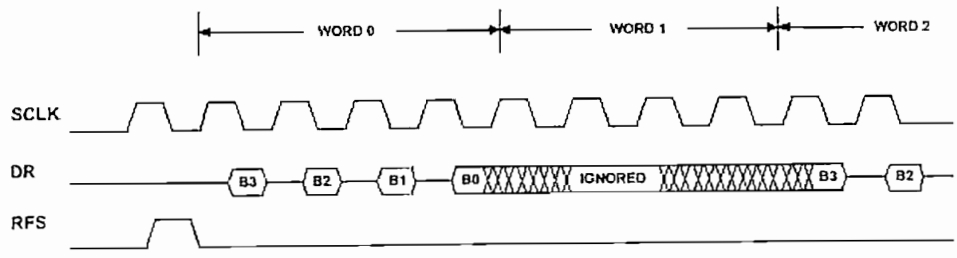


Figura 2.9 Ejemplo de Recepción Serial en modo Multicanal

2.2.3.7 Temporizador (TIMER)

El Temporizador (Timer) puede generar interrupciones periódicas, en un determinado tiempo, basándose en la duración de un ciclo del procesador. Cuando el timer es habilitado, mediante el bit 5 del registro MSTAT, el valor almacenado en un registro de cuenta de 16 bits (TCOUNT) es decrementado en

1, en cada ciclo del procesador. Si el valor del registro de cuenta llega a cero, se genera una interrupción; luego de lo cual, el registro de cuenta es recargado, con el mismo valor de inicio de cuenta, desde el registro de periodo (TPERIOD) de 16 bits. El decremento del valor del registro de cuenta puede ser configurado con el valor del registro de escala (TSCALE) de 8 bits, el cual indica cuantos ciclos del microprocesador deben transcurrir para que se reste en 1 al valor del registro de cuenta. El registro de cuenta (TCOUNT) se encuentra ubicado en la dirección 0x3FFC de memoria de datos, mientras que el registro de periodo (TPERIOD) esta en la dirección 0x3FFD, y el registro de escala (TSCALE) se ubica en la dirección 0x3FFB. En la Figura 2.10 [1] se muestra el diagrama de bloques del Temporizador.

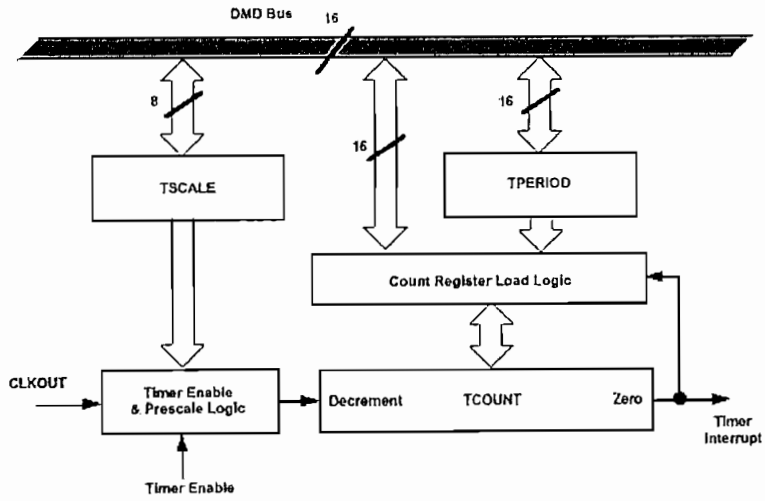


Figura 2.10 Diagrama de bloques del Temporizador (TIMER)

2.2.3.8 Interfaz de Sistema del ADSP-2181

Las señales externas, generadas o recibidas por el microprocesador, son la interfaz del sistema. Una de esas señales es la señal de entrada de reloj del microprocesador, que se produce con un cristal de cuarzo entre los pines CLKIN y XTAL. Otra entrada, es la señal de RESET, la cual detiene la ejecución del microprocesador y realiza un reset del hardware.

Las señales de interrupción externa son señales introducidas por el usuario y sirven para responder a ciertos eventos. Las interrupciones externas pueden ser sensitivas a flanco o sensitivas a nivel. El microprocesador posee las siguientes interrupciones externas: IRQ2, IRQL0, IRQL1, son sensitivas a nivel; mientras, IRQE es sensitiva a flanco.

Además posee pines utilizados como banderas de salida, controladas por software, las cuales son os pines: FL0, FL1 Y FL2. Y también tiene disponible 8 pines configurables como entrada o salida, de propósito general, los cuales son los pines PF7 – 0.

2.2.3.9 Interfaz de Memoria del ADSP-2181

La interfaz de Memoria es la definición de espacios de memoria externa y el enlace con ellas.

La memoria tanto de programa como de datos es memoria RAM, con una Arquitectura Harvard. El microprocesador posee 16 K palabras de 16 bits de Memoria RAM de Datos y 16 K palabras de 24 bits de Memoria RAM de Programa.

Existen 4 espacios de memoria: Memoria de Datos, Memoria de Programa, Memoria Byte y Memoria I/O; que pueden ser accedidas mediante las señales DMS , PMS, BMS Y IOMS, en los respectivos pines. Las líneas de control de transferencia de datos es el pin RD, utilizada para procesos de Lectura, y el pin WR para procesos de escritura. El espacio de Memoria I/O es usado para establecer a periféricos mapeados en memoria externa, y tiene un espacio máximo de 2K, con palabras de 16 bits. El espacio de Memoria Byte es muy utilizado para cargar instrucciones de memoria de Programa después de un reset.

El espacio de Memoria de Programa es configurado de acuerdo al estado del pin MMAP y del registro de capas de memoria de programa PMOVLAY. El espacio de memoria de programa es configurado de acuerdo al registro de capas de memoria

de datos DMOVLAY, además se reserva 32 localidades de memoria de datos desde la dirección 0x3FE0 hasta 0x3FFF para registros de sistema, mapeados en memoria. En la figura 2.11 [1] se muestra el diagrama de Interfaz de Memoria y en la Figura 2.12 [1] los mapas de memoria de datos y de programa. Las señales BR, BG Y BGH son usadas como petición para usar el bus externo multiplexado de direcciones y datos para acceder a los buses internos.

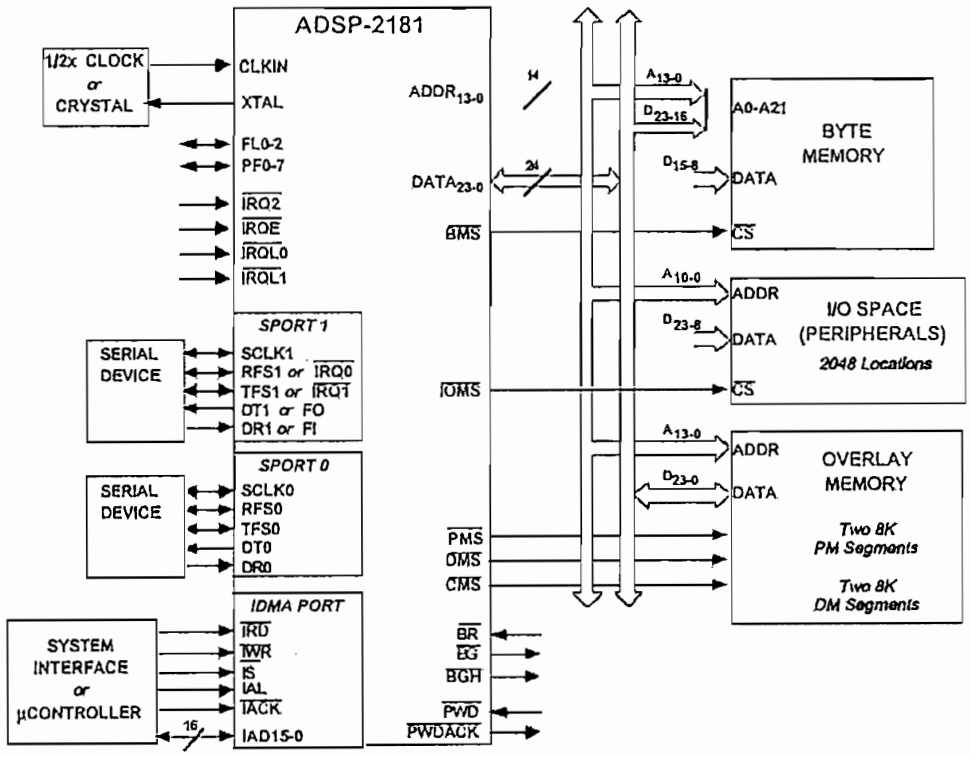


Figura 2.11 Diagrama de Interfaz de Memoria

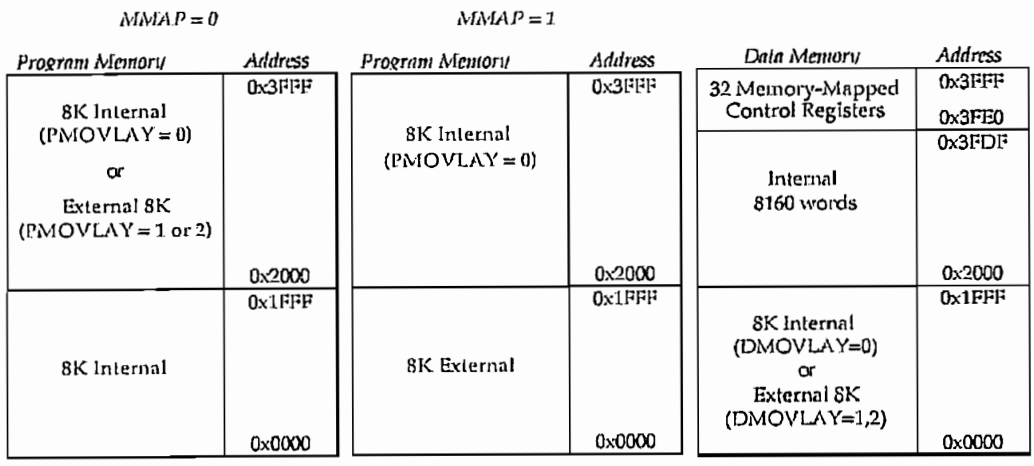


Figura 2.12 Mapa de memoria de Programa y Mapa de memoria de Datos

2.2.3.10 Puertos de Acceso Directo a Memoria (DMA)

Los puertos de acceso a memoria permiten una transferencia de datos desde memoria externa hacia la memoria interna del ADSP-2181. Existen dos tipos de puertos DMA y son los siguientes: El puerto BDMA (Byte DMA) y el puerto IDMA (Internal DMA).

El puerto BDMA (Byte DMA) es de 8 bits, los datos usan las líneas 15 – 8 del bus de datos, y las direcciones se proveen desde las líneas 13 – 0 del bus de direcciones y las líneas 23 – 16 del bus de datos. Maneja hasta 256 paginas, cada una de 16K palabras de 8 bits. Este puerto es usado más a menudo para carga automática de instrucciones de programa y datos después de un reset, esto se logra configurando los pines MMAP y BMODE como se muestra en la Tabla 2.3 [1] a continuación:

Pin MMAP	*Pin BMODE*	*Método de Carga*
0	0	Carga a través del Puerto BDMA
0	1	Carga a través del Puerto IDMA
1	-	No hay descarga después de un Reset

Tabla 2.3 Selección del modo de descarga a memoria de programa

Además, como fuente de las instrucciones para la descarga, se debe tener una memoria EPROM. El formato de los datos para este tipo de memorias, se logra mediante una herramienta de software llamada PROM Splitter, y el almacenamiento se logra mediante un grabador de memorias EPROM.

La carga desde el puerto BDMA de las instrucciones de programa se realiza por medio de los siguientes registros: el registro de direccionamiento BDIR, que indica la carga desde el puerto o el almacenamiento hacia el puerto BDMA (seteado a 0 es carga desde BDMA); el registro de paginamiento BMPAGE, que indica la pagina de inicio de memoria para transferencia (seteado a 0 para pagina 0); los

registros BEAD y BIAD indican la localidad de memoria externa e interna para la transferencia; el registro BTYPE indica el tipo de datos (00 indica palabras de 24 bits); el registro BCR = 1 detiene la ejecución del DSP durante la carga, con esto se tiene la transferencia de las primeras 32 localidades de memoria de programa desde la memoria externa, con lo cual se configura al puerto BDMA para que sigua con la carga de memoria de programa faltante.

El puerto IDMA es un puerto paralelo de entrada y salida, que permite leer o escribir la memoria interna desde o hacia un sistema host. La lectura y escritura se pueden realizar en cualquier momento durante la ejecución del programa, y no necesita atención del microprocesador, en el traspaso de datos. Los pines del puerto IDMA se muestran en la Tabla 2.4[1] a continuación:

Pin	*I/O*	*Función*
$\overline{IRD}$	I	IDMA PORT READ STROBE
$\overline{IWR}$	I	IDMA PORT WRITE STROBE
$\overline{IS}$	I	IDMA PORT SELECT
IAL	I	IDMA PORT ADDRESS LATCH ENABLE
IAD (0 –15)	I/O	IDMA PORT ADDRESS/DATA BUS
$\overline{IACK}$	0	IDMA PORT ACCESS READY ACKDGE

Tabla 2.4 Descripción de los pines del puerto IDMA

El proceso de transferencia empieza con el pin IS, el cual selecciona la operación del puerto, Si se activa el pin IS y el pin IRD se produce una lectura, y si se activa el pin IS y el pin IWR se produce una escritura. Los registros usados para la transferencia son los siguientes: el registro IDMAA, utilizado para indicar la dirección, que se mantendrá en el bus de direcciones; y el registro IDMAD que indica el tipo de memoria de destino, si es 1, se dirigirá a memoria de datos, y si es 0, se dirigirá a memoria de programa. En la Figura 2.13 [1] se muestra el proceso de una trasferencia IDMA.

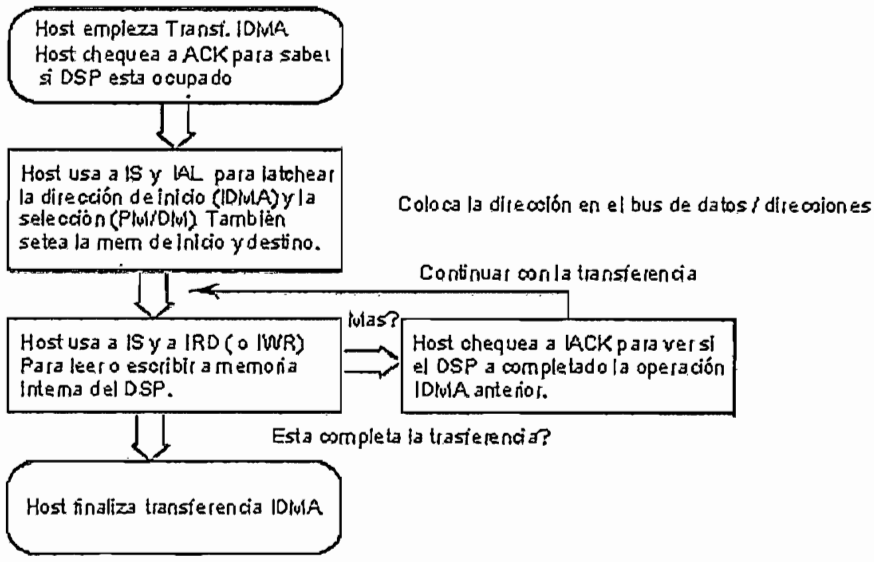


Figura 2.13 Proceso de Transferencia de datos desde un Host hacia memoria interna

2.3 DESCRIPCIÓN DEL HARDWARE Y SOFTWARE DE LA TARJETA DE DESARROLLO EZ KIT LITE PARA EL ADSP-2181.

El Hardware de la tarjeta de desarrollo consta de las siguientes partes:

- ADSP-2181 Microprocesador DSP
- CODEC (Codificador/Decodificador) AD1847
- Circuito de interfaz serial AD232AAR
- EPROM de 256 Kbytes
- Teclas de usuario
- Conectores de expansión
- Jumpers configurables por usuario

Un esquemático de la ubicación de las partes constituyentes en la tarjeta de desarrollo se muestra en la Figura 2.14 [3]. En este esquemático se puede observar a la ubicación del microprocesador ADSP-2181, el Codec AD1847, la

memoria EPROM, los botones de usuario, los conectores, y jumpers de configuración.

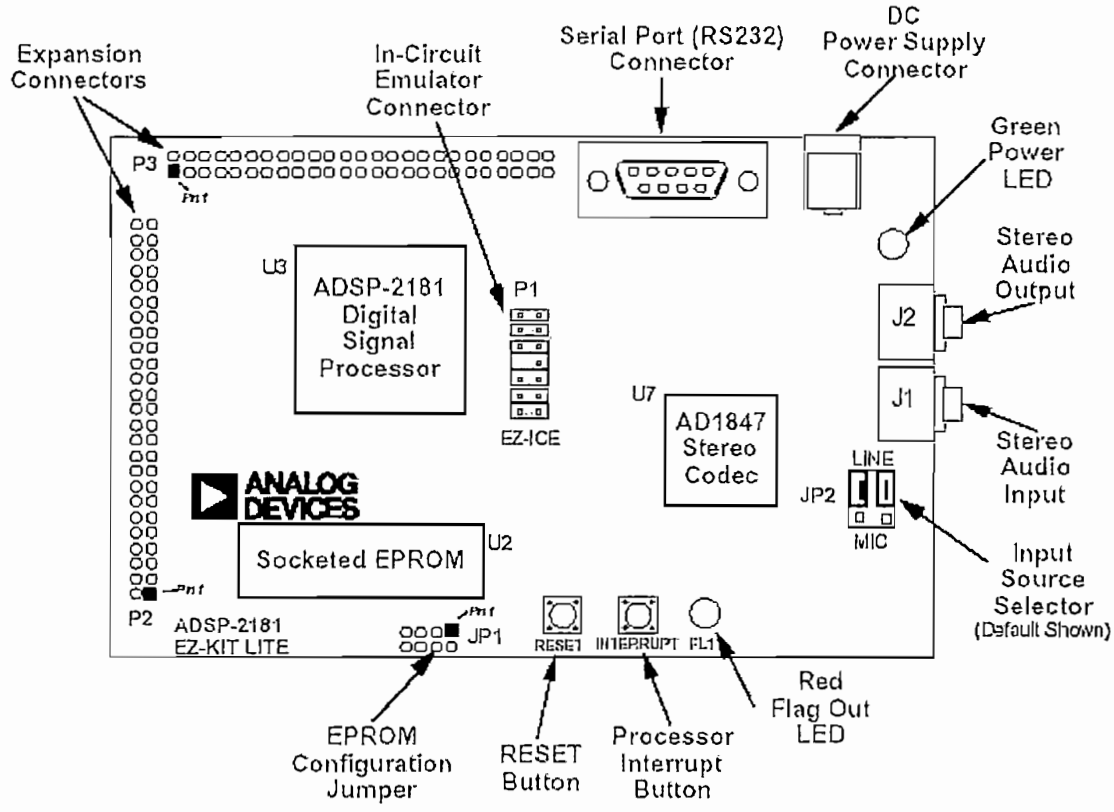


Figura 2.14 Esquemático de las partes constituyentes de la tarjeta EZ KIT LITE

La tarjeta puede conectarse a un puerto serial y por medio de un programa llamado Monitor, guardado en la memoria EPROM; y un programa Host en la PC, se logra descargar el programa de usuario. También se puede remover la memoria EPROM del zócalo, y colocar otra memoria que posea un programa dedicado del usuario, para poderlo descargar desde el puerto BDMA, y no utilizar a los programas mencionados anteriormente.

El software de desarrollo que acompaña a la tarjeta de desarrollo son: el programa Host y los utilitarios, los cuales son : un Constructor de Sistema (Builder System), que indica la ubicación, designada por el usuario, de las instrucciones de programa y la ubicación de datos, dentro de la memoria del microprocesador; u Ensamblador (Assembler), el cual revisa la correcta sintaxis del código fuente de los archivos que contienen las instrucciones de programa, y que tienen una

extensión .DSP o .ASM; un Enlazador (Linker), el cual enlaza todos los archivos que han sido ensamblados, y los une para formar un archivo ejecutable .EXE, o en las versiones actuales :DXE o .DJE; un Conformador de programas para memorias EPROM (PROM SPLITTER) y un Simulador como es el VisualDSP ++ .

El elemento principal de la tarjeta de desarrollo es el ADSP-2181KS, además posee una EPROM que usa el puerto BDMA para descarga del programa Monitor que emula y controla una comunicación serial asincrónica RS-232 para la descarga de programas desde PC. El CODEC AD1847 se usa para la conversión A/D y D/A de las señales analógicas, las cuales pueden tener dos fuente configurables: nivel de micrófono y nivel de Línea. Estas opciones se seleccionan mediante el jumper JP2, como se indica en la Figura 2.15 [7].

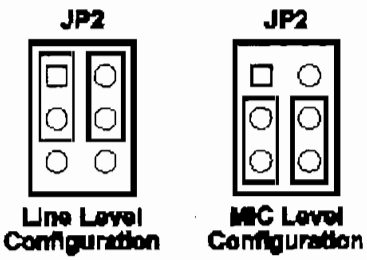


Figura 2.15 Configuración del Tipo de Señal de Entrada

La memoria EPROM puede ser retirada del zócalo y ser remplazada por otra de diferente tamaño de acuerdo a la configuración del jumper JP1 como se indica en la Tabla 2.5 [7].

Además posee un conector DB-9 para comunicación serial. Un conector para fuente de alimentación de voltaje DC de 9 voltios. Un conector J1, jack estereo de 3.5 mm, para entrada de audio estereo. Un conector J2, jack estereo de 3.5 mm, para salida de audio estereo. Un conector P1 , header de 14 pines, para conexión con tarjetas emuladoras. Un led verde usado como indicador de Poder, y un led Rojo conectado al pin FL1, utilizado como bandera de salida de usuario. Un pulsador S2, conectado a la interrupción IRQE, para uso general, y un pulsador S1, de Reset. Además, el puerto serial 1 (SPORT 1), configurado en modo

alternativo, se conecta a un acoplador de voltaje AD232AAR para emular una comunicación serial asincrónica RS-232.

Los conectores P3 y P4 son headers de 50 pines para acceso a todas las señales internas del ADSP-2181, como por ejemplo; el puerto IDMA o los pines configurables de uso general.

JP1	TIPO DE MEMORIA	TAMAÑ0
4 3 2 1 JP1	27C256	32 Kbytes x 8 bits
4 3 2 1 JP1	27C512 27C010	64 Kbytes x 8 bits 128 Kbytes x 8 bits
4 3 2 1 JP1	27C020 27C040 27C080	256 Kbytes x 8 bits 512 Kbytes x 8 bits 1 Mbyte x 8 bits

Tabla 2.5 Configuración de JP1 para diferentes tipos de EPROM

2.3.1 CODIFICADOR / DECODIFICADOR (CODEC) AD1847

El Codec AD1847 codifica y decodifica las señales de entrada y salida, analógicas. El Codec trasmite y recibe las muestras de las señales digitalizadas hacia o desde el ADSP-2181 mediante un bus serial sincrónico. La comunicación se realiza mediante el puerto serial 0 (SPORT 0) del ADSP-2181 y el puerto serial del AD1847. La alimentación del Codec es única de +5VDC, logrando generar señales analógicas AC de amplitud máxima de ± 1 voltio. Para la codificación y decodificación usa 2 conversores A/D y 2 conversores D/A, con modulación $\Sigma\Delta$.

La frecuencia de muestreo es configurable desde un valor limite inferior de 5.5 Khz. hasta 48 Khz., dependiendo de los circuitos osciladores a cristal conectados. La interfaz de datos utiliza un esquema TDM (Time División Multiplex) que es compatible con el modo Multicanal del SPORT0, configurado con 32 slots de

tiempo de 16 bits. En la Figura 2.16 [6] se indica un diagrama de bloques del Codec AD1847, en este se observa el puerto serial para comunicación con el ADSP-2181, los conversores A/D y D/A para codificación y decodificación, circuitos de ganancia y atenuación programable y circuitos de companding con leyes A-law y μ-law.

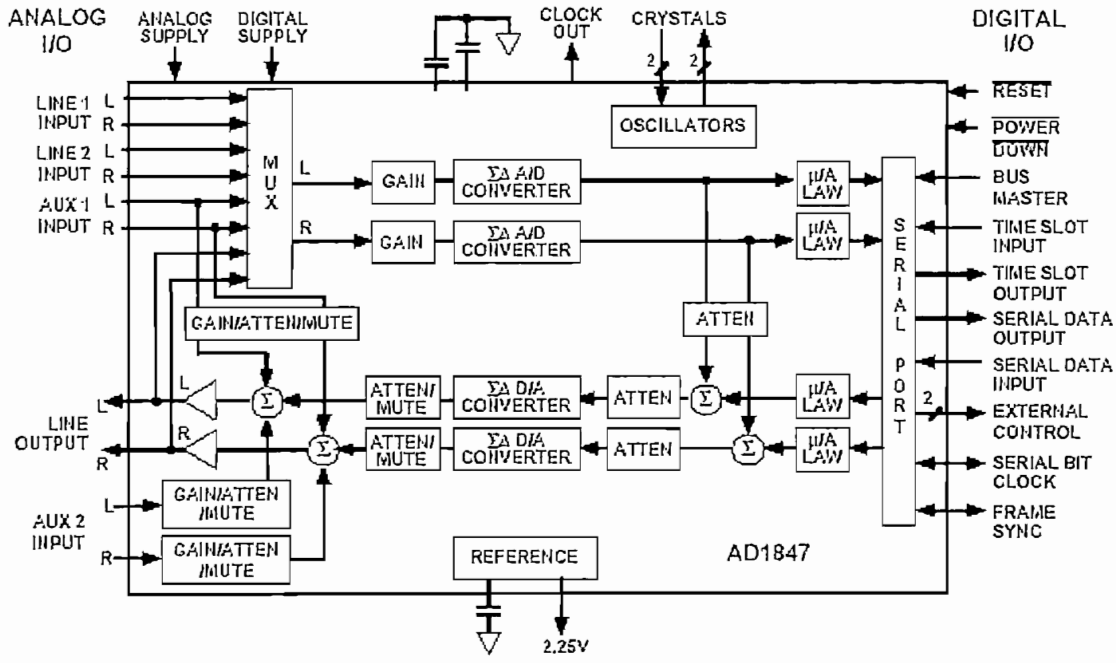


Figura 2.16 Diagrama de Bloques del Codec AD1847

Las señales de entrada AUX 1 y AUX 2 pueden mezclarse con las señales analógicas de salida.

Los registros de control del AD1847 accesibles a usuario son: 6 registros de 16 bits (Palabra de Control, Palabra de Estado, Datos de Entrada de los canales izquierdo y derecho y Datos de Salida de los canales izquierdo y derecho) y 13 registros de 8 bits (Registros indexados). La Palabra de Estado (Status Word) indica el estado del CODEC y es enviado hacia el ADSP-2181. La Palabra de Control da información del control al Codec, que se envía desde el ADSP-2181. Los datos de entrada, son los datos digitalizados del canal izquierdo y canal derecho que se guardan en dos registros de 16 bits, y que se enviarán al microprocesador DSP; y los datos de salida, son los datos de las muestras digitalizadas que se quiere generar y que son enviadas desde el procesador DSP

hacia el Codec, estos datos se encuentran en 2 registros de 16 bits, uno para el canal derecho y otro para el canal izquierdo.

Los registros indexados (de 8 bits) son usados para configuración del Codec como son: la configuración de la comunicación serial, la frecuencia de muestreo, el formato de datos, etc. Al inicio se debe realizar la configuración del Codec mediante estos registros, previamente habilitando el modo de Cambio, mediante el envío de la Palabra de control con el bit MCE seteado a 1.

El Codec posee un modo de Autocalibración, este modo calibra a los DACs y ADCs, y se recomienda habilitarlo después de configurar al Codec (después de salir del Modo de cambio). Para conocer la terminación del modo de Autocalibración, se debe chequear el bit ACI de la palabra de Estado, debido a que el proceso puede durar algunos milisegundos. La configuración de la comunicación serial depende del pin TSSEL del Codec. En la interfaz serial con el ADSP-2181 se tiene TSSEL = 1, en este modo se tiene que los 6 registros comparten 3 slots de tiempo, pero la trasmisión y recepción de datos ocurren simultáneamente (Sistema "2-wires"), es decir en el slot 0 se realiza la recepción de la palabra de Control en el pin SDI del Codec y también se trasmite la palabra de estado por el pin SDO. En la Tabla 2.6 [6] se indican los slots de tiempo y las palabras asignadas en el sistema "2-wires".

Slot de tiempo	*Nombre del Registro*
0	Palabra de Control
1	Datos de Entrada del canal izquierdo
2	Datos de Entrada del canal derecho
0	Palabra de Estado
1	Datos de Salida del canal izquierdo
2	Datos de Salida del canal derecho

Tabla 2.6 Slots de tiempo de los registros en sistema "2-wires"

En la Figura 2.17 se muestra la interfaz serial entre el Codec AD1847 y el ADSP-2181. El ADSP-2181 usa el puerto serial 0 configurado en modo Multicanal de 32

slots de tiempo (canales), al igual que el AD1847. La señal de reloj serial y la señal de sincronización de trama es generada por el AD1847. El AD1847 trasmite y recibe 3 palabras de 16 bits en cada muestra de la señal analógica, que esta siendo digitalizada con una cierta frecuencia de muestreo.

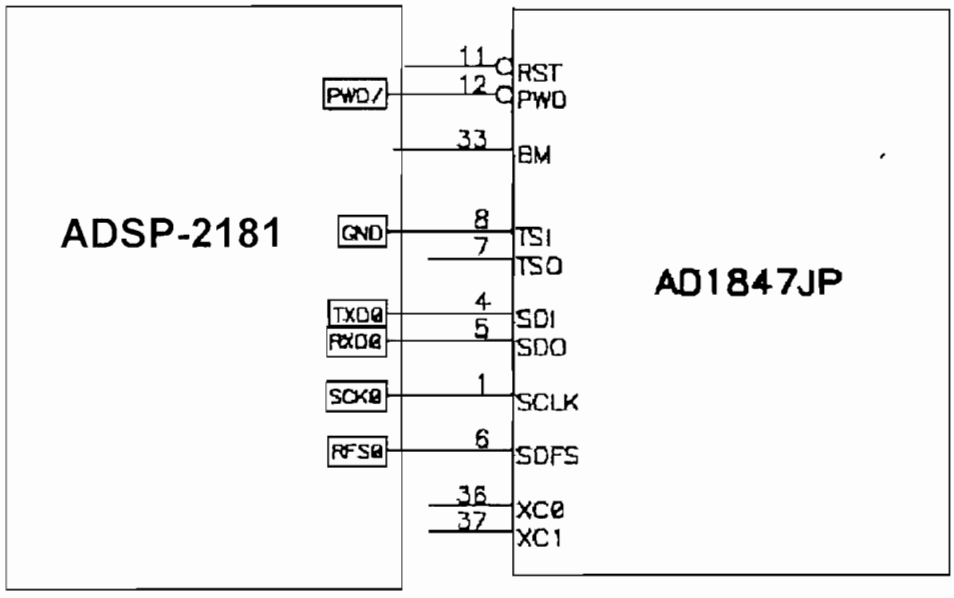


Figura 2.17 Interfaz Serial entre el ADSP-2181 y el AD1847

2.3.1.1 Principio de funcionamiento de Conversores A/D y D/A con modulación Sigma-Delta (ΣΔ)

El principio de conversores A/D con modulación sigma-delta se basa en obtener un flujo de ceros y unos lógicos en equilibrio, dependiendo de la amplitud de la señal de entrada analógica. Luego, se suma el número de unos lógicos generados en un cierto número de ciclos de tiempo, y este número nos indicará una cantidad relativa a la señal de entrada. Por ejemplo; si la señal de entrada es cero voltios, existirá un mismo número de ceros lógicos que de unos lógicos, es decir existirán 500 valores de uno lógico en 1000 ciclos de reloj de un contador.

En la Figura 2.18 [8] se muestra el diagrama de bloques de un conversor A/D con modulación sigma-delta (ΣΔ). La señal analógica de entrada se aplica a un capacitor cuyo voltaje se compara con el voltaje de tierra (0 Voltios), dependiendo

del voltaje del comparador, un selector permite la inserción de carga positiva o negativa, mediante un circuito de realimentación. Si el voltaje en el capacitor aumenta, existirán más unos lógicos en la salida del circuito comparador, el cual habilitará la inserción de carga negativa para compensación de carga del capacitor hasta llegar a un equilibrio. Los unos lógicos generados se introducen en un contador con una señal de reloj para contar cada uno lógico, el número de unos lógicos nos indicara el nivel y el signo de la señal de entrada. Si la señal de entrada es negativa existirán mas ceros que unos, debido a que se inyectara más carga positiva hacia el capacitor.

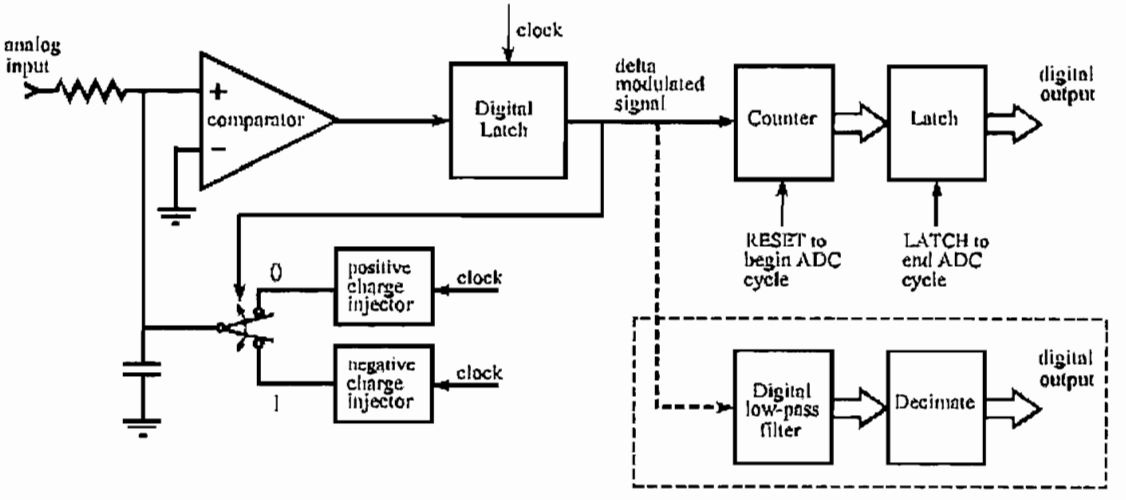


Figura 2.18 Diagrama de Bloques de un conversor A/D con modulación Sigma-Delta (ΣΔ)

En vez de utilizar un contador de unos lógicos se utilizan filtros digitales pasabajos que en este tipo de señal, con modulación delta, se obtiene la señal digitalizada de la señal analógica de entrada. Un Decimador tiene la función de ignorar ciertos unos lógicos para ajustar a cierta cantidad de bits al ADC.

Un conversor D/A con este tipo de modulación únicamente obtiene la señal modulada de algún circuito generador, y a esta señal únicamente se le aplica un filtro pasabajos de primer orden para su reconstrucción en la señal original analógica.

2.4 DESCRIPCIÓN DEL PROGRAMA LABVIEW DE NATIONAL INSTRUMENT

Labview es un programa de National Instruments que utiliza programación gráfica y programación por medio de flujo de datos.

Posee dos partes esenciales: el Panel Frontal y el Diagrama de Bloques. En el panel frontal se pueden ver las interfaces de usuario, los datos y formas de onda; mientras, en el diagrama de bloques se encuentra el código de programación en una forma gráfica. Los programas realizados en Labview se llaman VI (Virtual Instruments) y se denominan virtuales porque imitan a los instrumentos verdaderos. En la Figura 2.19 se observa el panel frontal y el diagrama de bloques de un VI.

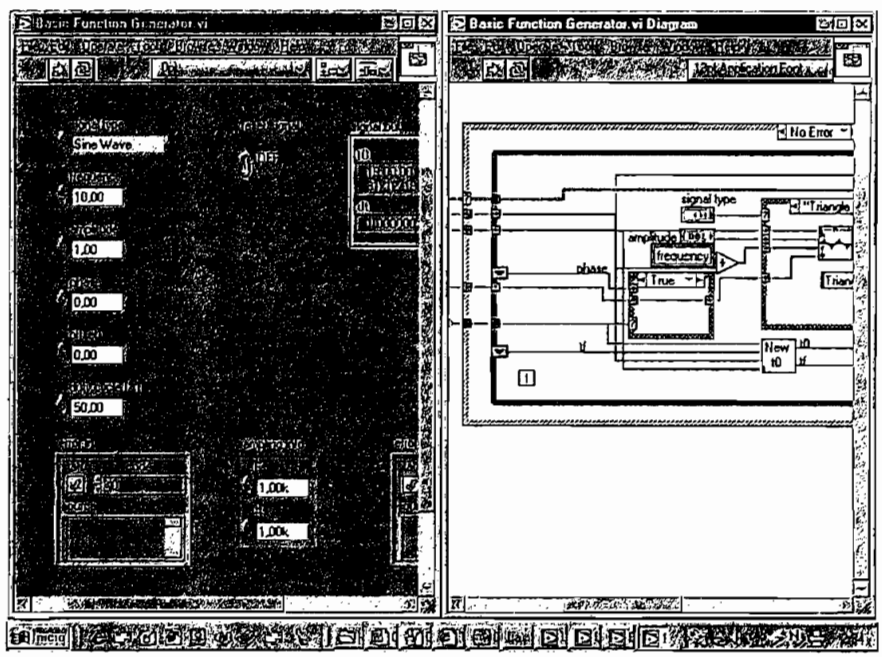


Figura 2.19 Panel Frontal y Diagrama de Bloque de un VI (Virtual Instrument)

Cada VI posee en su parte superior un icono y un Conector de panel asociado a ese icono. Un icono es la representación grafica simbólica del VI y el conector de panel son las conexiones posibles que se puede realizar al VI para que este

funcione como SubVI. Un SubVI es un conjunto de VIs usados en el diagrama de bloques de otro VI, generalmente usados para acciones repetitivas que requieren de gran espacio de programación, utilizadas como subrutinas. En la Figura 2.20 se muestra al icono y al conector de panel de un VI.

Figura 2.20 Icono y Conector de Panel de un VI

2.5 SUMARIO

Los microprocesadores ADSP-2181 son especializados en Procesamiento Digital de Señales, debido a su gran velocidad de operación y también a su arquitectura. Por tal motivo son los más requeridos en Algoritmos Digitales de síntesis de señales analógicas. Con su interfaz serial hacia la PC y hacia un CODEC lo hace el más optimo para recibir datos generados de formas de onda y trasmitidos desde Labview, recibidos y controlados en el ADSP-2181, y generados eléctricamente por medio de un Codec AD1847.

CAPITULO 3

DISEÑO DE LOS INSTRUMENTOS VIRTUALES DESARROLLADOS EN LABVIEW Y EL PROGRAMA DE GENERACIÓN DE FORMAS DE ONDA EN EL ADSP – 2181

3.1 INTRODUCCIÓN

En este Capítulo se describen los Instrumentos Virtuales diseñados para realizar la implementación del Generador Arbitrario de Formas de Onda, y los algoritmos utilizados para su funcionamiento. Además se describe el programa desarrollado en el ADSP -2181 para la generación eléctrica de las formas de onda trasmitidas desde Labview.

La interfaz gráfica con el usuario esta constituida por 7 pantallas. La primera pantalla que aparece es la pantalla de presentación inicial. La segunda pantalla es el Menú Principal, en donde se puede escoger entre el Menú de Generación y el Menú de Comunicación. El Menú de Generación posee 3 opciones, cada una con una pantalla propia; y el Menú de Comunicación, muestra una pantalla para conocer el estado de la comunicación con el microprocesador DSP, esta pantalla indica los errores ocurridos en la transferencia e indica la finalización de la misma.

El programa en el DSP configura ciertos registros para la funcionalidad propia del sistema e inicializa algunas localidades de memoria de datos utilizadas en la configuración del CODEC (Conversores A/D y D/A) para la generación de las formas de onda, la cual se realiza mediante comunicación serial sincrónica.

Además toma las muestras trasmitidas para un periodo de la forma de onda, desde Labview, y ubicadas en ciertas localidades de memoria de datos del DSP, y las trasmite hacia el CODEC en forma periódica.

3.2 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL PRESENTACIÓN_INICIAL.VI DEL GENERADOR ARBITRARIO DE FORMAS DE ONDA

La pantalla inicial del Generador de Formas de Onda es el panel frontal del instrumento virtual **Presentación_Inicial.VI,** e indica el título del Proyecto de Titulación y "Escuela Politécnica Nacional" con su respectivo escudo.

Posee dos opciones: **Iniciar Sesión y Salir.** Al escoger *Iniciar Sesión* se abrirá el panel frontal del instrumento virtual Menú_Principal.VI, que es la siguiente pantalla de interfaz gráfica. Si se presiona *Salir* se detiene la ejecución del Instrumento virtual. En la Figura 3.1 se muestra el panel frontal de Presentación_Inicial.VI.

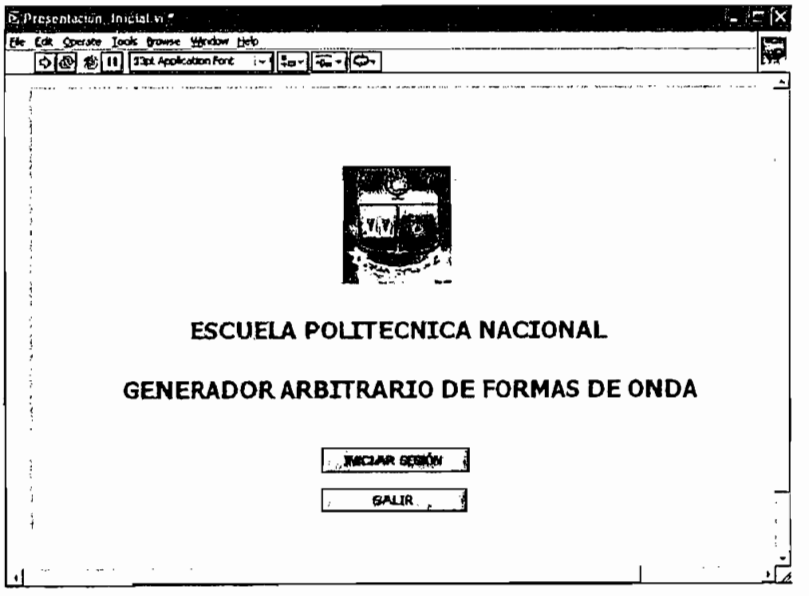


Figura 3.1 Panel Frontal de Presentación_Inicial.VI

3.3 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL MENÚ_PRINCIPAL.VI

El Menú Principal se muestra al Iniciar la sesión desde la Presentación Inicial. Esta pantalla es el panel frontal del instrumento virtual **Menú_Principal.VI.** Dentro

del Menú Principal existen dos submenús: un Menú de Generación y un Menú de Comunicación. Además posee un indicador de Gráficos de Forma de Onda, el cual mostrará la forma de onda a generar, que es construida desde los Submenús de Generación. Los submenús de Generación son 3: *Crear desde Pantalla, Crear desde Librerías y Crear desde Formula,* al escoger cualquiera de estas opciones se mostrará una pantalla diferente.

El Submenú de Comunicación tiene una opción, *Enviar Datos Al Sistema DSP*, al escoger esta opción aparecerá una pantalla de comunicación serial de los datos de la forma de onda a generar.

Además de los menús se pueden observar: indicadores de la frecuencia de la forma de Onda, la frecuencia de muestreo y el número de muestras de la forma de onda creada. En la Figura 3.2, se muestra el panel frontal de Menú_Principal.VI y en la Figura 3.3 se muestra su correspondiente diagrama de bloques.

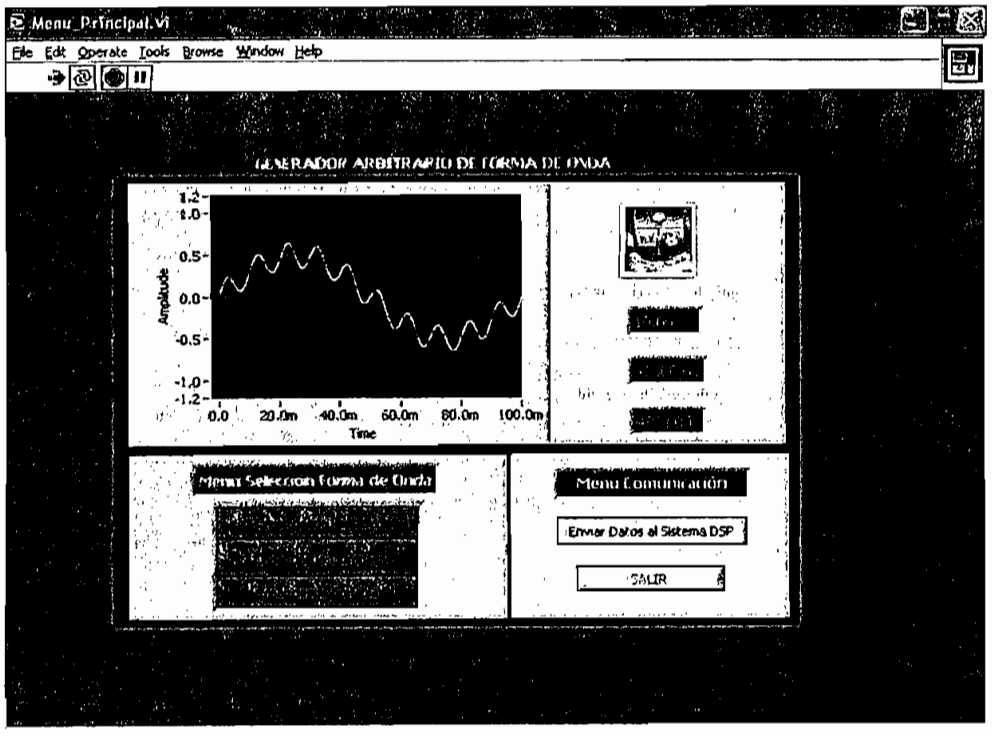


Figura 3.2 Panel Frontal de Menú_Principal.VI

En el Diagrama de Bloques se pueden observar algunos iconos utilizados. Cada icono representa un SubVI que se utiliza como una subrutina, la cual se ejecuta si se escoge alguna de las opciones de los Submenus en el Menu Principal. En la Figura 3.3 se observa el icono del SubVI *Crear_desde_Pantalla.VI*, el icono del subVI *Convertir_a_ASCII.VI* y el icono del subVI de Trasmisión Serial *Comunicación_Serial_DSP_Labview.VI*.

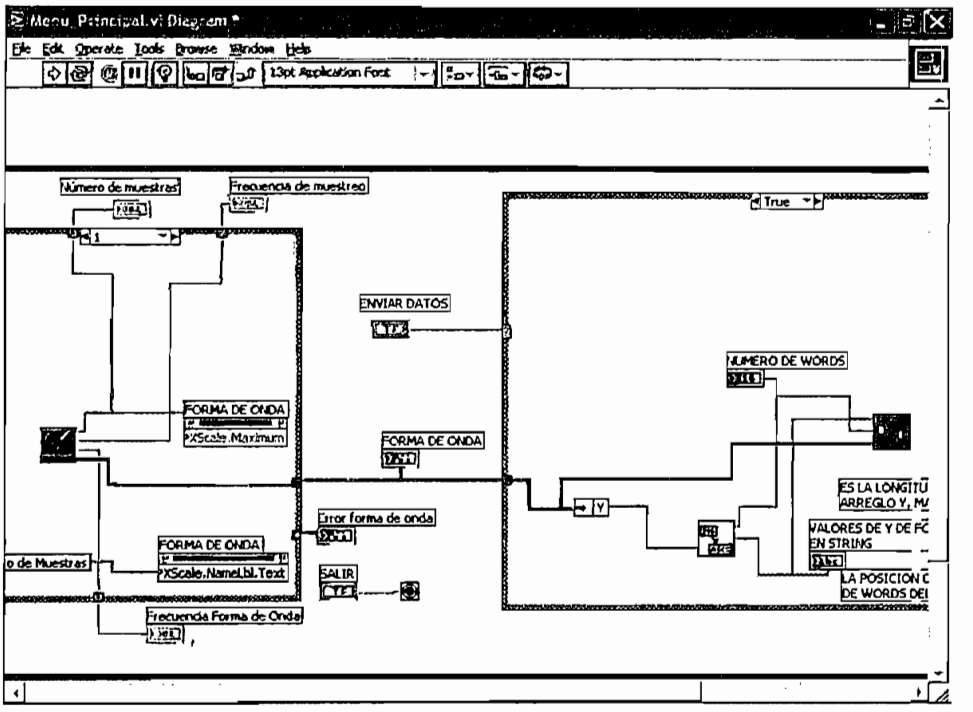


Figura 3.3 Diagrama de Bloques de Menú_Principal.VI

3.4 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL CREAR_DESDE_PANTALLA.VI

Al escoger la opción Crear desde Pantalla desde el Menú Principal, se despliega una pantalla, la cual posee un indicador de gráfico de Forma de Onda. En este caso la creación de la forma de onda se produce escogiendo la frecuencia deseada y presionando el botón izquierdo del mouse sobre el indicador de la forma de onda, al realizar esta acción, aparecerá una línea continua sobre el indicador, que seguirá al movimiento del mouse.

El Algoritmo utilizado se basa en la posición que proporcionan los cursores del indicador de gráfico de forma de onda. Estos cursores se mueven presionando el botón izquierdo del mouse y arrastrándolo sobre el indicador. Al mover el mouse, estos cursores se mueven e indican la posición que tienen en el indicador de forma de onda. Posteriormente, el dato de posición del cursor es almacenado como una muestra de la forma de onda, y luego se dibujan todas las muestras sobre el mismo indicador.

En la Figura 3.4 [9] se indican los elementos de un indicador de Gráfico de Forma de Onda.

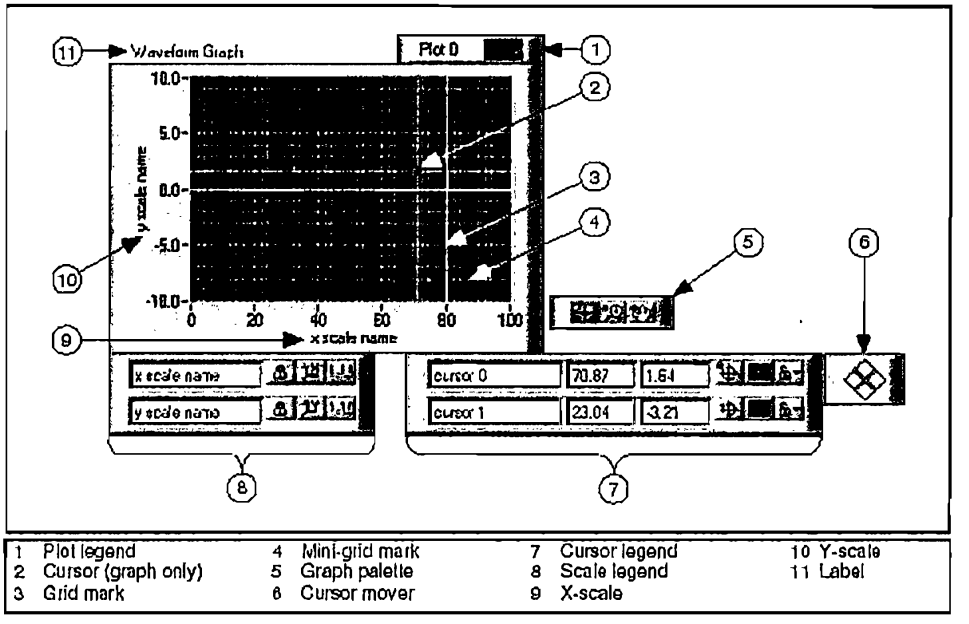


Figura 3.4 Elementos Constitutivos de un indicador de forma de onda

El algoritmo toma el valor de la escala en Y de la posición vertical, y el valor de la escala en X de la posición horizontal del cursor 0, mediante un *Nodo de Propiedad* del Indicador de Forma de Onda, y lo almacena en un arreglo de datos. Un Nodo de Propiedad puede setear o leer ciertas propiedades de los VIs, de los objetos de los VIs y de las aplicaciones. Un arreglo de datos, en Labview, es un conjunto de datos ordenados de un mismo tipo. Un arreglo es semejante a una

matriz con elementos de la misma clase, es decir todos los elementos son caracteres, o todos son números o todos son números de punto flotante, etc.

Teniendo los valores de la escala en Y de la posición vertical del cursor 0 se puede construir un Cluster de Forma de Onda. Un Cluster de Forma de Onda es un conjunto de datos de diferente tipo que definen a una forma de onda. Este cluster de Forma de Onda está constituido de un arreglo de los datos de Y de la forma de onda (valores de amplitud), un valor inicial en X y un incremento de la escala en X (ΔX) entre muestras. Definiendo este cluster se lo envía hacia el indicador de Forma de Onda para su graficación.

En la Figura 3.5 se indica el Panel Frontal de Crear_desde_Pantalla.VI y en la Figura 3.6 se indica una parte del Diagrama de Bloques respectivo.

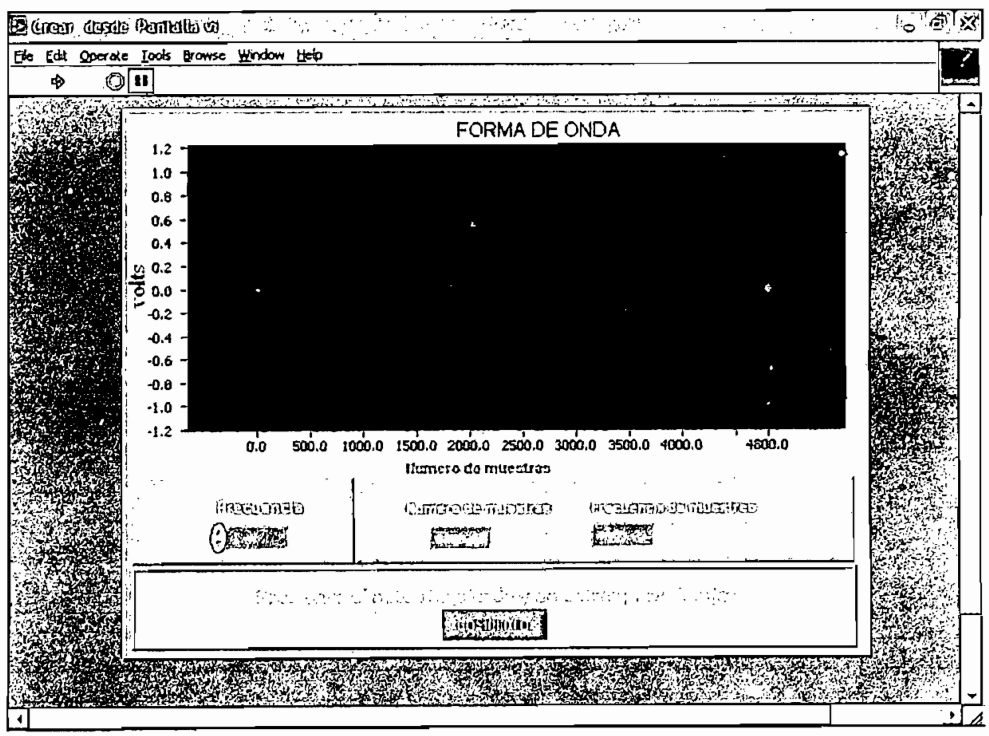


Figura 3.5 Panel Frontal de Crear_desde_Pantalla.VI

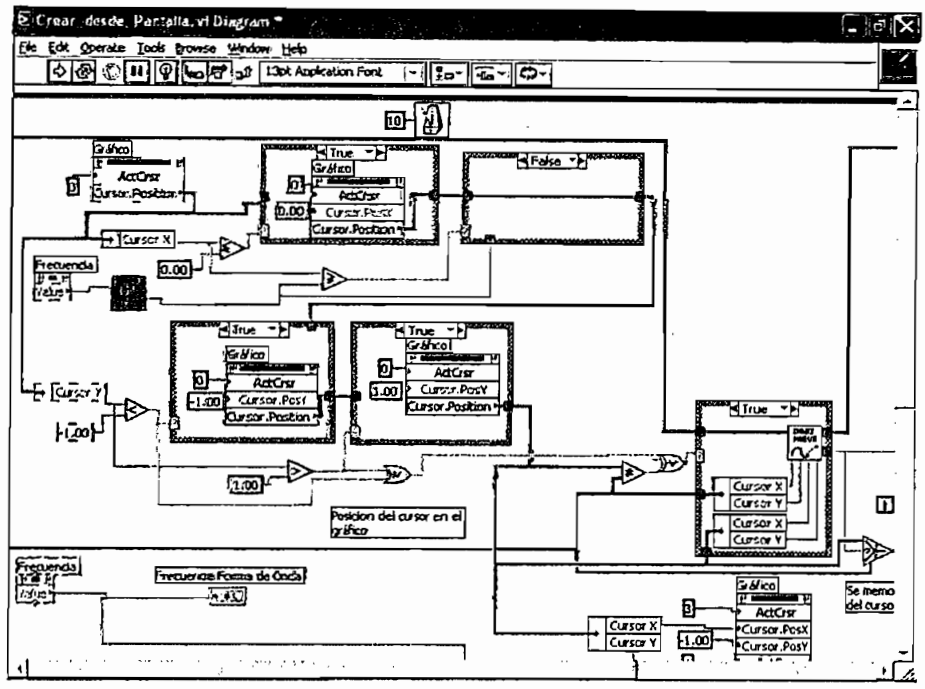


Figura 3.6 Diagrama de Bloques de Crear_desde_Pantalla.VI

En el diagrama de bloques, mostrado en la Figura 3.6, se puede observar a 2 SubVIs: el SubVI *Calcular_Número_Muestras.VI* y el subVI *Dibujar_Nuevo_Punto.VI.* El SubVI Calcular_Número_Muestras.VI, calcula el número de puntos que se dibujarán en el indicador de gráfico de forma de onda y el número de datos a trasmitir hacia el sistema DSP. El SubVI Dibujar_Nuevo_Punto.Vi utiliza el valor en la escala X de la posición horizontal y el valor en la escala Y de la posición vertical, del cursor 0; y la posicion actual y anterior, para modificar los puntos del arreglo de datos de Y (valores de amplitud) de la forma de onda deseada, que se graficará.

En la Figura 3.7 se puede observar el diagrama de Flujo de Crear_desde_Pantalla.VI .

El valor en la escala en Y de la posición vertical del cursor 0, se usa para calcular el valor en el arreglo de datos de Y (valores de amplitud) de la forma de onda deseada.

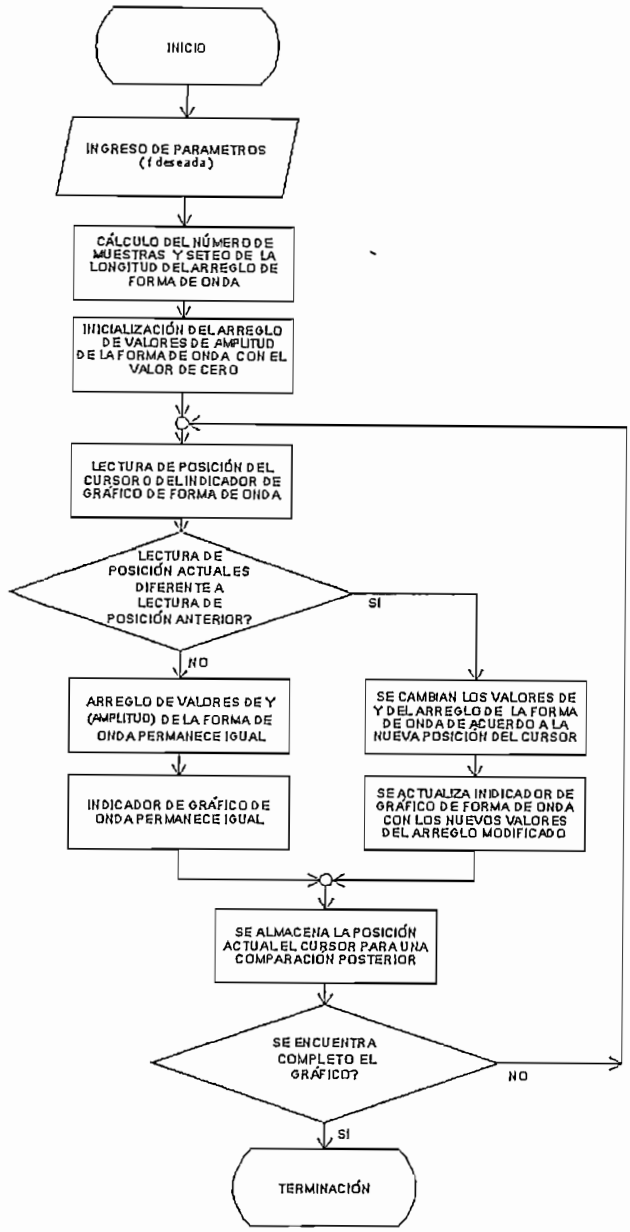


Figura 3.7 Diagrama de Flujo de Crear_desde_Pantalla.VI

3.4.1 DESCRIPCIÓN DEL ALGORITMO IMPLEMENTADO EN EL SUBVI CALCULAR_NÚMERO_MUESTRAS.VI

El SubVI Calcular_Número_Muestras.VI es utilizado por el instrumento virtual Crear_desde_Pantalla.VI, y es el encargado de calcular el número de muestras de la forma de onda que serán el número de puntos a graficar. El número de muestras se calcula dividiendo la frecuencia de muestreo con la frecuencia de la

forma de onda. Esta afirmación se demuestra con la Ecuación 3.1 [11] y 3.2 [11]. La frecuencia de muestreo se seleccionó tomando la mayor frecuencia de muestreo que posee el CODEC en el sistema DSP, y que es 48 kHz.

$$n.T_s = T \qquad \text{Ec. 3.1}$$

$$n = \frac{T}{T_s} = \frac{\frac{1}{f}}{\frac{1}{f_s}} = \frac{f_s}{f} \qquad \text{Ec. 3.2}$$

Donde:

n = Número de muestras

T = Periodo de la Forma de Onda

Ts = Periodo de Muestreo

f = Frecuencia de la forma de Onda

fs = Frecuencia de muestreo

En la Figura 3.8 se muestra una señal analógica y una señal digitalizada. Las muestras de la señal digitalizada tiene un tiempo entre ellas, ese tiempo es el periodo de muestreo. Si se multiplica el número de muestras por el periodo de muestreo se tendrá un tiempo total que será igual al periodo de la forma de onda analógica.

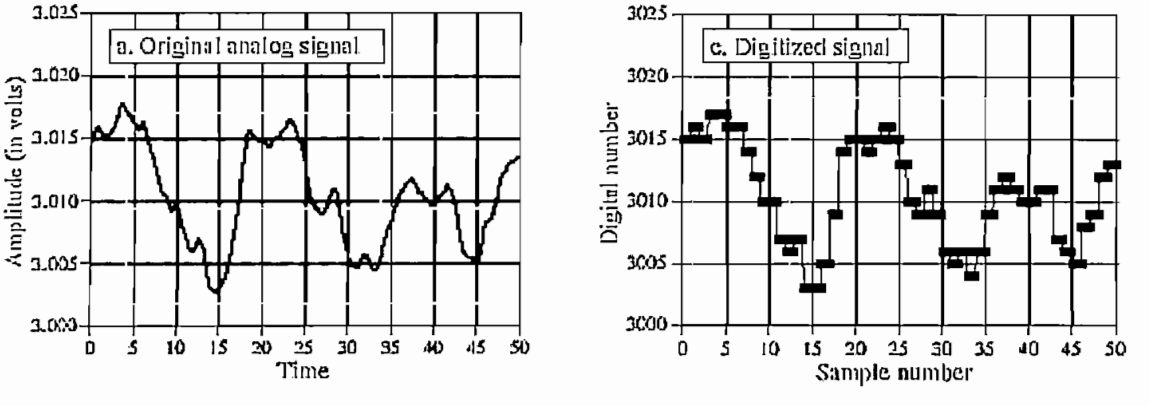


Figura 3.8 Formas de Onda Analógica y Digitalizada

El algoritmo, implementado en este SubVI, obtiene el intervalo de tiempo entre muestras, mediante la división del periodo de la forma de onda para el número de muestras. Tanto el intervalo entre muestras, como el periodo de la forma de onda, son salidas del SubVI.

En la Figura 3.9 se muestra el Diagrama de Bloques del SubVI Calcular_Número_Muestras.VI .

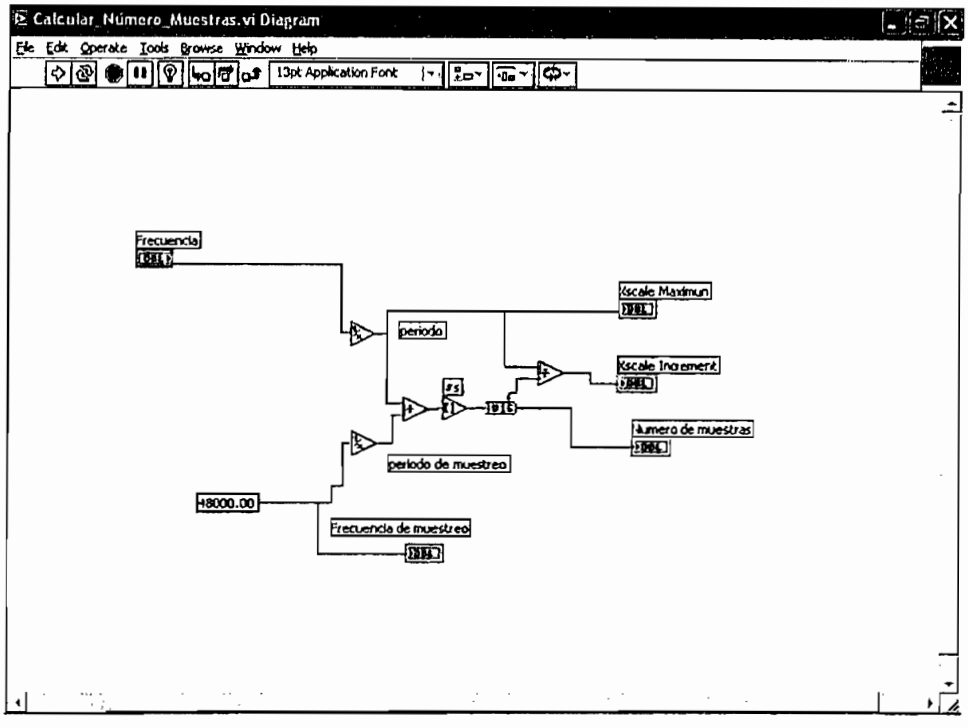


Figura 3.9 Diagrama de Bloques del SubVI Calcular_Número_Muestras.VI

Además del número de muestras, se tienen dos salidas opcionales: Xscale Increment y Xscale Maximun. Xscale Maximun, indica el valor máximo que se debe tener en la escala de X, este valor es el periodo de la Forma de Onda deseada; y, Xscale Increment, indica el incremento de la escala de X (ΔX), cuyo valor es el periodo de muestreo e indica el intervalo de tiempo entre muestras. Estos dos valores son necesarios para setear las propiedades del indicador de

grafico a utilizarse, en el cual se mostrará un periodo de la forma de onda deseada.

El diagrama de Flujo del SubVI Calcular_Número_Muestras.VI, se muestra en la Figura 3.10 a continuación.

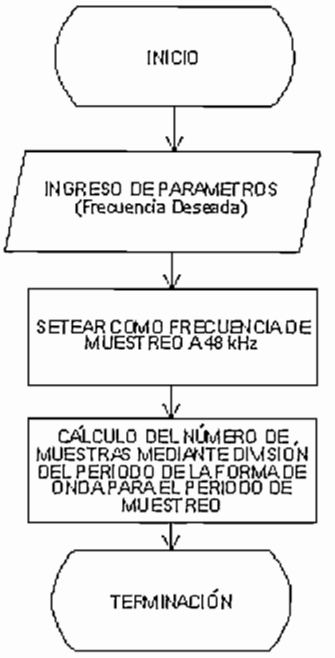


Figura 3.10 Diagrama de Flujo del SubVI Calcular_Número_Muestras.VI

3.4.2 DESCRIPCIÓN DEL ALGORITMO IMPLEMENTADO EN EL SUBVI DIBUJAR_NUEVO_PUNTO.VI

El SubVI Dibujar_Nuevo_Punto.VI es también utilizado por el instrumento Virtual Crear_desde_Pantalla.VI. Este SubVI calcula el arreglo de datos de amplitud de la forma de onda; cada valor del eje Y, en el indicador de gráfico, representa un valor de amplitud, por lo tanto, referenciar a datos de amplitud es lo mismo que decir datos de Y de la forma de onda. El orden de cada dato de Y del arreglo de forma de onda, indica su posición en la escala de X, esto quiere decir, que el primer elemento del arreglo de datos de Y (primer valor de amplitud) se encuentra implícitamente en el valor cero de la escala de X, el segundo elemento se encuentra en un valor de cero más el intervalo en X (ΔX) calculado, el cual es el

periodo de muestreo; y el siguiente se encuentra a dos intervalos en X; y asi con los demás valores del arreglo.

Para saber cuantos datos de Y, de la forma de onda, se deben incluir en el arreglo de datos, se debe calcular cuantos puntos existirán entre el valor en la escala de X, de la posición horizontal inicial del cursor 0; y el nuevo valor en la escala de X, de la nueva posición horizontal del cursor 0. Calculando la distancia entre la posición horizontal inicial y la nueva posición, y dividiendo el resultado para el periodo de muestreo, se obtiene el número de puntos a incluir en el arreglo de datos de Y de la forma de onda. Como posición inicial, se hace referencia a una posición del cursor desde la cual se empezó a moverlo; y como nueva posición, a aquella en la cual se detiene al cursor 0.

El SubVI encuentra el mayor valor en X de las posiciones horizontales, actual y anterior; posteriormente resta el mayor valor encontrado con el menor, y el resultado lo divide para el incremento en X del Grafico de Forma de onda (periodo de muestreo). El resultado de lo indicado anteriormente es únicamente el número de puntos (cantidad de valores de amplitud) a incluir en el arreglo de datos de Y de la Forma de Onda, no se calculan los valores del arreglo de datos de Y (valores de amplitud), entre la posición actual y anterior el cursor 0. Todo lo anteriormente indicado es para saber cuantos valores del arreglo de Y (valores de amplitud) de la Forma de Onda se deben calcular.

Por otro lado, con el valor en la escala de Y de la posición vertical, de la posición actual y anterior del Cursor 0; y el número de puntos a añadir, calculados anteriormente, se puede utilizar el SubVI Interpolación_Lineal.VI, para encontrar los nuevos valores del arreglo de datos de Y de la Forma de Onda. En la Figura 3.11 se puede observar el Diagrama de Flujo del SubVI Dibujar_Nuevo_Punto.VI .

Otra función del algoritmo es encontrar la ubicación, dentro del arreglo de datos de Y de la Forma de Onda, desde donde se localizaran los nuevos datos de Y a graficar. Comparando el valor en la escala de X de la posición horizontal, actual y

anterior del cursor 0, se puede encontrar cual es el menor valor, el cual indicará la ubicación desde donde se posicionarán los nuevos datos de Y.

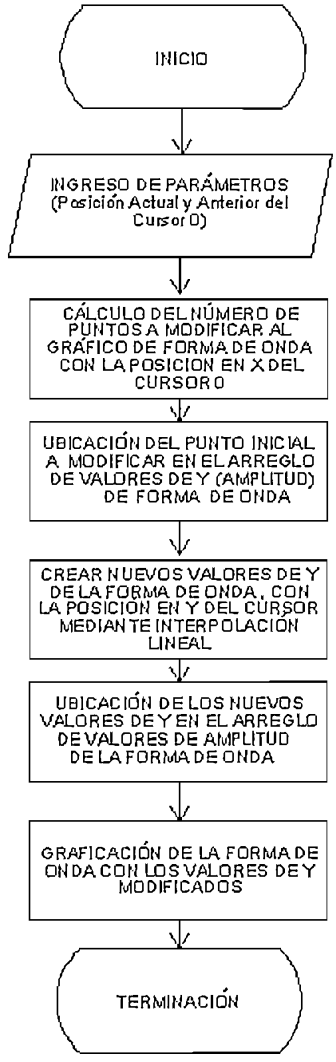


Figura 3.11 Diagrama de Flujo del subVI Dibujar_Nuevo_Punto.VI

Debido a la velocidad de ejecución de Labview, la forma de onda se observa de una manera suave, especialmente en gráficos con un número de muestras muy grande, debido a formas de onda con frecuencias bajas.

Este Algoritmo se ejecuta únicamente si la posición actual del cursor es diferente de la posición anterior que fue almacenada. Esto quiere decir, que la gráfica mostrada se altera, si el cursor se mueve.

En la Figura 3.12 se muestra el Diagrama de Bloques el SubVI Dibujar_Nuevo_Punto.VI .

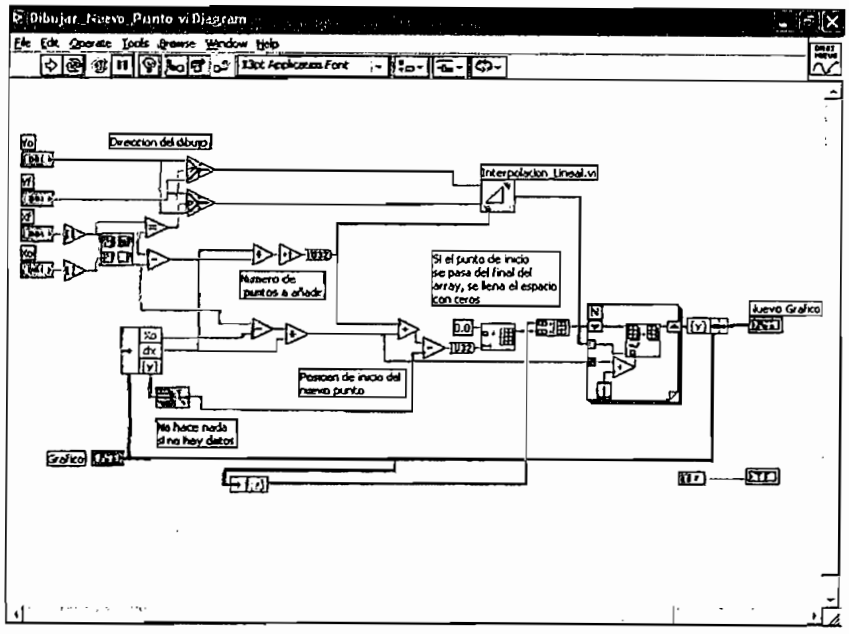


Figura 3.12 Diagrama de Bloques del SubVI Dibujar_Nuevo_Punto.VI

3.4.2.1 Descripción del algoritmo Implementado en el SubVI Interpolación_Lineal.VI

El SubVI Interpolación_Lineal.VI es utilizado en el SubVI Dibujar_Nuevo_Punto.VI. El algoritmo implementado realiza una Interpolación Lineal, con el valor en la escala de Y, de la posición vertical actual y anterior del cursor 0. En la Figura 3.13 se explica gráficamente el algoritmo implementado.

El algoritmo Dibujar_Nuevo_Punto.VI, envía como parámetros de entrada del subVI Interpolación_Lineal.VI a Yf, el cual es el mayor valor en la escala de Y, del resultado de comparar la posición vertical actual y anterior del cursor 0; y a Yo

como el menor valor. Además, como otro parámetro de entrada, se envía el número de muestras a modificar.

Con Yf, Yo, y el número de muestras a modificar; se calcula la pendiente de la fórmula que servirá para la interpolación lineal. Con la pendiente y el valor de Yo, se puede encontrar el valor del dato de Y (valor de amplitud) de una muestra, dos muestras, tres muestras, etc, de la forma de Onda. La pendiente se calcula restando Yf – Yo, y dividiendo al resultado, para el número de muestras a modificar. En la Ecuación . 3.3 y 3.4 se indica lo anteriormente dicho.

$$\text{pendiente} = m = \frac{Y_f - Y_o}{\text{número de puntos a modificar}} \qquad \text{Ec. 3.3}$$

$$Y_k = Y_0 + m.k \qquad k = 0,\ldots,n = \text{número de puntos a añadir} \qquad \text{Ec. 3.4}$$

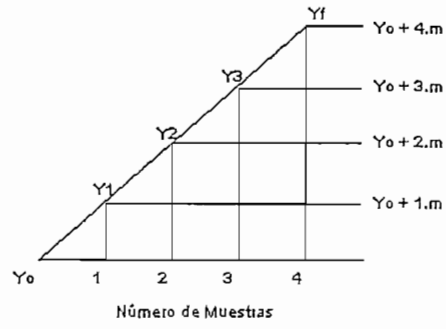


Figura 3.13 Gráfica Explicativa de una Interpolación Lineal

Si el usuario no llega a dibujar en todo el espacio de la pantalla, asignado para el periodo deseado de forma de onda, el algoritmo determina la última posición del cursor 0 en la escala de X; y encuentra su correspondiente elemento en el arreglo de datos de Y (arreglo de valores de amplitud). Al encontrar el elemento en el arreglo, empieza a llenar de ceros a los demás espacios restantes, esto se hace debido a que se debe tener un valor por cada muestra.

La dimensión del arreglo de datos de Y, es el número de muestras calculado con el SubVI Calcular_Número_Muestras.VI.

En la Figura 3.14 se muestra el diagrama de flujo del algoritmo de Interpolación Lineal.

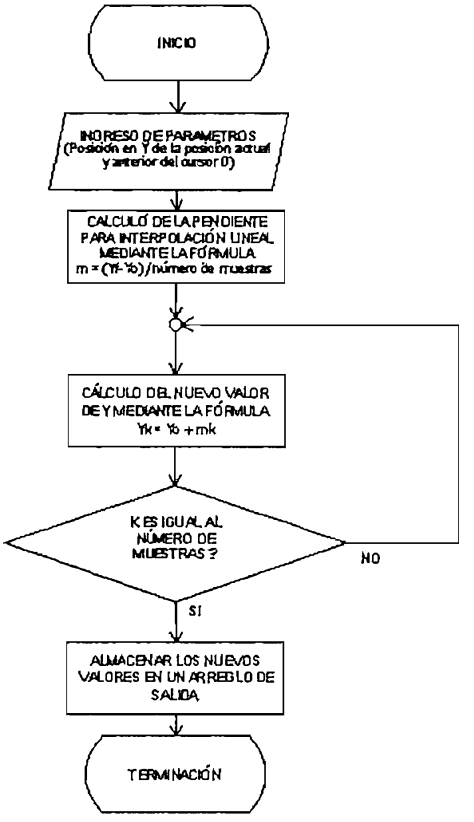


Figura 3.14 Diagrama de Flujo de Interpolación_Lineal.VI

En la Figura 3.15 se puede ver el Diagrama de Bloques correspondiente al diagrama de Flujo anterior.

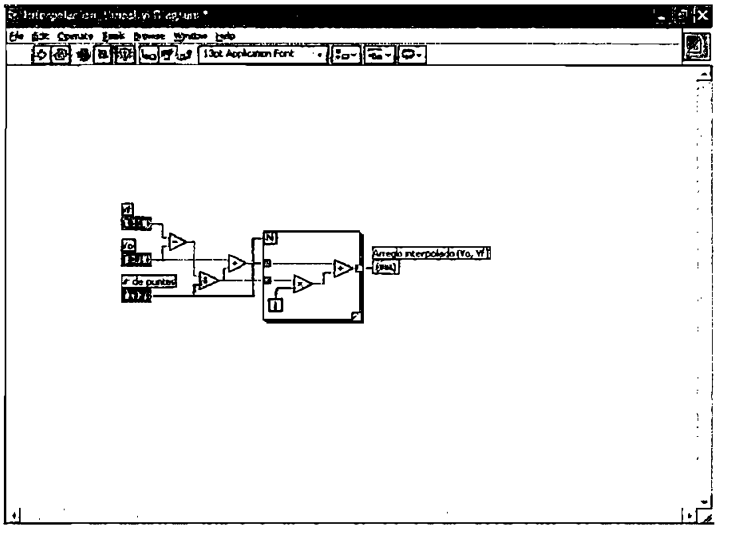


Figura 3.15 Diagrama de Bloques de Interpolación_Lineal.VI

3.5 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL CREAR_DESDE_LIBRERÍAS.VI

Otra opción del Submenú Generación, del Menú Principal, es la de *Crear desde Librerías*. Al escoger esta opción, se mostrará una pantalla correspondiente a el panel frontal del instrumento virtual Crear_desde_Librerías.VI, en esta pantalla se puede escoger una de entre varias formas de ondas básicas como son: Onda Senoidal, Onda Triangular, Onda Cuadrada, Onda Diente de Sierra y Ruido Blanco Uniforme.

Se tienen tres indicadores de forma de onda, dos son independientes y se puede escoger una forma de onda diferente para cada uno, el tercer indicador de forma de onda muestra el resultado de las operaciones básicas aplicadas sobre las dos anteriores. Las operaciones básicas disponibles en la pantalla del VI son: sumar la forma de onda 1 con la forma de onda 2 y dividir el resultado para dos; restar la forma de onda 1 con la forma de onda 2; y por último, incluir la forma de onda 2 a continuación de la forma de onda 1; además, se tiene la opción de mostrar únicamente a la forma de onda 1 o la forma de onda 2.

Como otra opción interesante, se pueden generar formas de onda con ruido blanco uniforme, mediante la opción sumar y escogiendo, en la forma de onda 1, a cualquier forma de onda básica y, en la forma de onda 2, a Ruido Blanco Uniforme. En la Figura 3.16 se indica a una Onda Senoidal con Ruido Blanco Uniforme, dibujada en el panel frontal del instrumento virtual Crear_desde_Librerías.VI .

El algoritmo implementado usa un SubVI, el cual se encarga de generar las formas de ondas básicas, el cual se llama Generador_Básico_de_Funciones.VI. El resto del algoritmo se encarga de realizar las operaciones básicas a las formas de onda generadas por este SubVI.

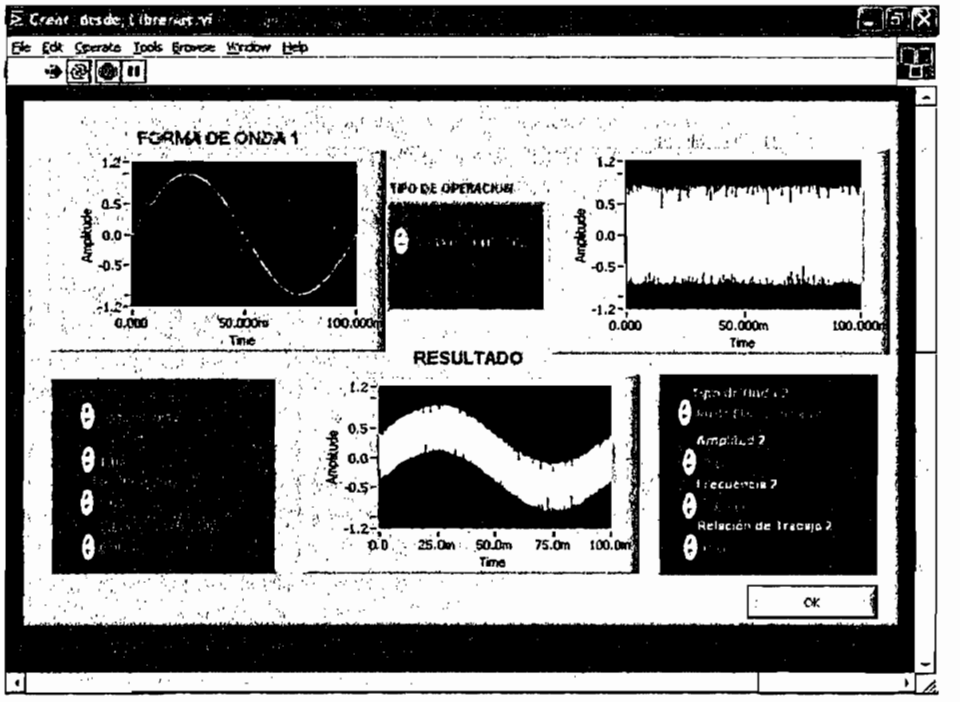


Figura 3.16 Panel Frontal de Crear_desde_Librerías.VI

Antes de generar las formas de ondas básicas, se debe conocer la menor de entre las dos frecuencias deseadas, con el motivo de saber cuantos periodos, de la forma de onda con mayor frecuencia, caben en la forma de onda con menor frecuencia. Este paso se debe a que las operaciones se realizan punto por punto, y con un incremento en X igual para las dos formas de onda, si una forma de onda tiene un periodo de 1 segundo, y se quiere sumar con una forma de onda con un periodo de 1ms, aunque ambas tienen una frecuencia de muestreo igual, no se podrá, debido a que el número de muestras en el periodo de 1 segundo es mucho mayor al número de muestras que existen en la forma de onda con el periodo de 1 ms, para igualar el número de muestras se deberá tener 1000 periodos de la forma de onda de 1ms, con esto se tendrá igual número de muestras y se podrá realizar una operación punto por punto.

Después de encontrar el número de periodos de las dos formas de onda, se envían los parámetros de entrada hacia el generador básico de funciones, el cual generará el número de muestras de los periodos indicados. Después de este paso, se realiza la operación deseada sobre las dos formas de onda generadas y

por ultimo se grafica en el indicador de Resultado. En la Figura 3.17 se muestra el diagrama de flujo de Crear_desde_Librerías.VI .

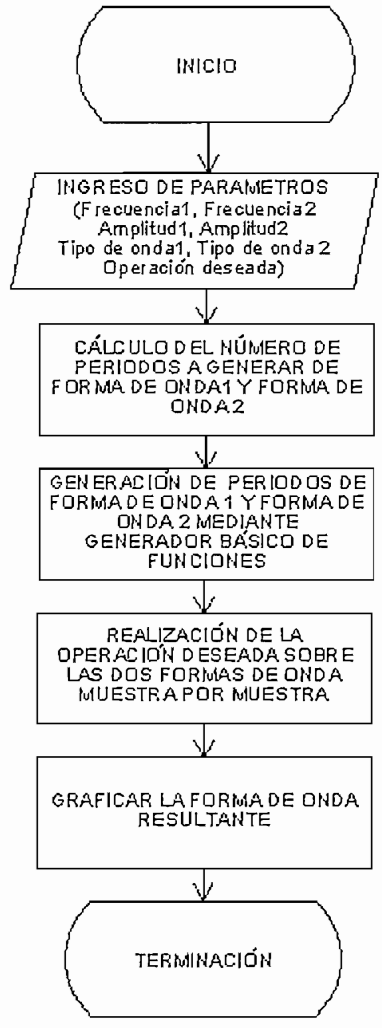


Figura 3.17 Diagrama de Flujo de Crear_desde_Librerías.VI

Una parte del diagrama de bloques del VI Crear_desde_Librerías.VI se indica en la Figura 3.18. En la parte derecha se puede observar el icono del subVI que genera las formas de ondas básicas, este es el instrumento virtual Generador_Básico_de_Funciones.VI .

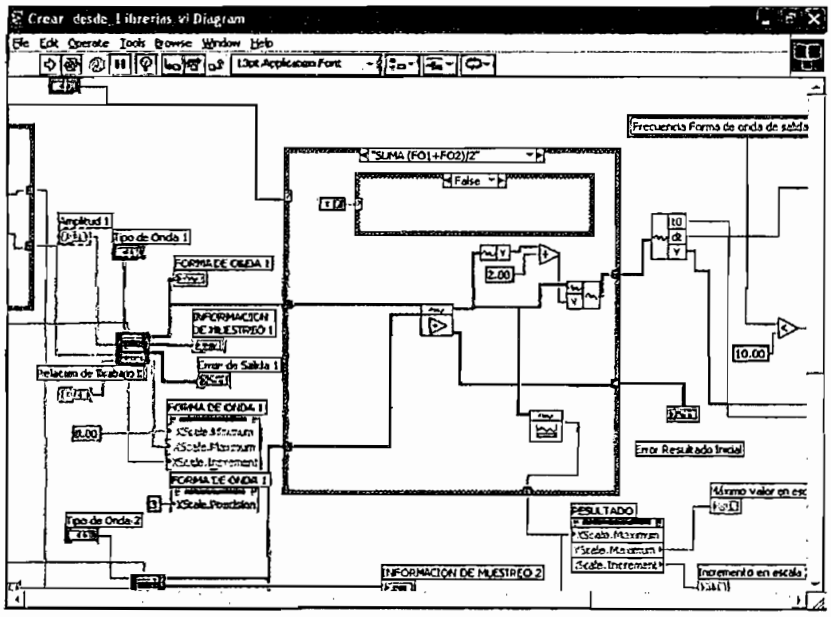


Figura 3.18 Diagrama de Bloques de Crear_desde_Librerías.VI

3.5.1 DISEÑO E IMPLEMENTACIÓN DEL SUBVI PARA GENERACIÓN BÁSICA DE FUNCIONES

El SubVI utilizado para la generación de funciones básicas es llamado Generador_Básico_de_Funciones.VI. Este subVI calcula el valor en Y de cada una de las muestras de la forma de onda indicada. Los parámetros de entrada a este SubVI son: Tipo de onda, frecuencia, amplitud, frecuencia, y número de periodos. La frecuencia de muestreo es por defecto de 48 KHz.

La generación de los valores de Y, de la forma de onda seleccionada, se realiza mediante VIs propios de Labview para generación de ondas; estos VIs se pueden encontrar en la paleta de funciones, dentro de la librería *ANALIZE,* y en la sublibrería *WAVEFORM GENERATION*; en esta sublibrería se pueden encontrar VIs para generar formas de ondas básicas como: Onda Senoidal (Sine Waveform), Onda Triangular (Triangle Waveform), Onda diente de Sierra (Sawtooth Waveform), entre otras.

El algoritmo implementado se encarga de dar la información de muestreo y el número de periodos a los VIs para generación de formas de onda. Con la información de muestreo, los Vis de generación calculan los valores en la escala de Y (amplitud) de las formas de ondas básicas. En la Figura 3.19 se muestra el Diagrama de Bloques del SubVI Generador_Básico_de_Funciones.VI .

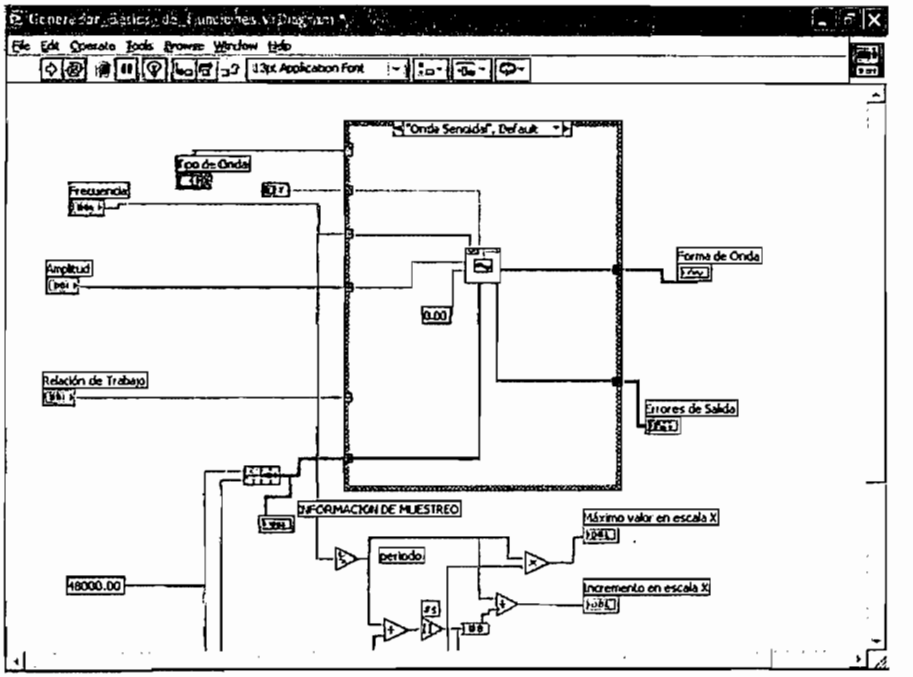


Figura 3.19 Diagrama de Bloques de Generador_Básico_de_Funciones.VI

3.6 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL CREAR_DESDE_FÓRMULA.VI

La última opción, desde el submenú de generación, es *Crear desde Fórmula*. Esta acción se realiza mediante el instrumento virtual Crear_desde_Fórmula.VI, con el cual se puede dibujar una forma de onda mediante el ingreso de su fórmula.

El algoritmo implementado utiliza el VI de Labview llamado *Formula Waveform*, ubicado en la sublibreria *WAVEFORM GENERATION*. El algoritmo básicamente indica la información de muestreo y traslada la fórmula, ingresada por el usuario, hacia este VI. La fórmula introducida debe tener cierta sintaxis, por tal motivo se

encuentra el SubVI llamado Ayuda_Sintaxis_Fórmula.VI. El panel frontal del SubVI Ayuda_Sintaxis_Fórmula.VI indica ciertas variables y el formato de la fórmula a introducir.

En la Figura 3.20 se expone el diagrama de bloques del instrumento virtual Crear_desde_Formula.VI, y en la Figura 3.21 se muestra el correspondiente panel frontal.

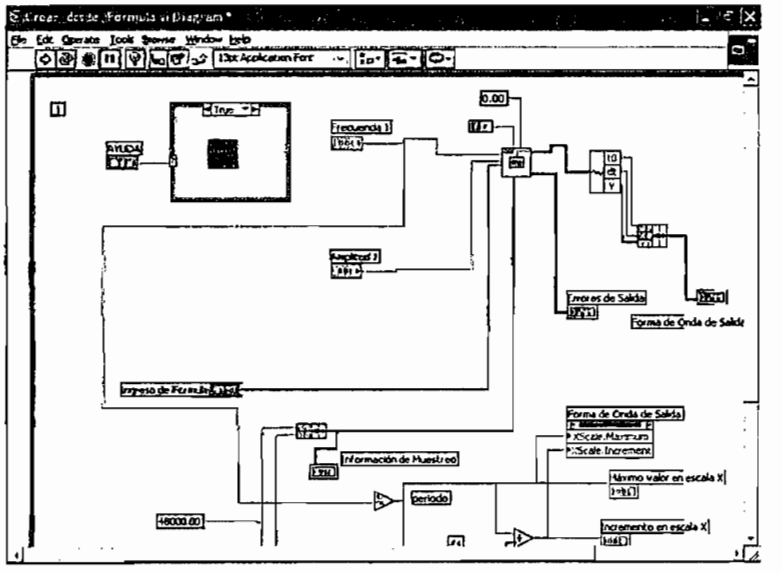


Figura 3.20 Diagrama de Bloques de Crear_desde_Fórmula.VI

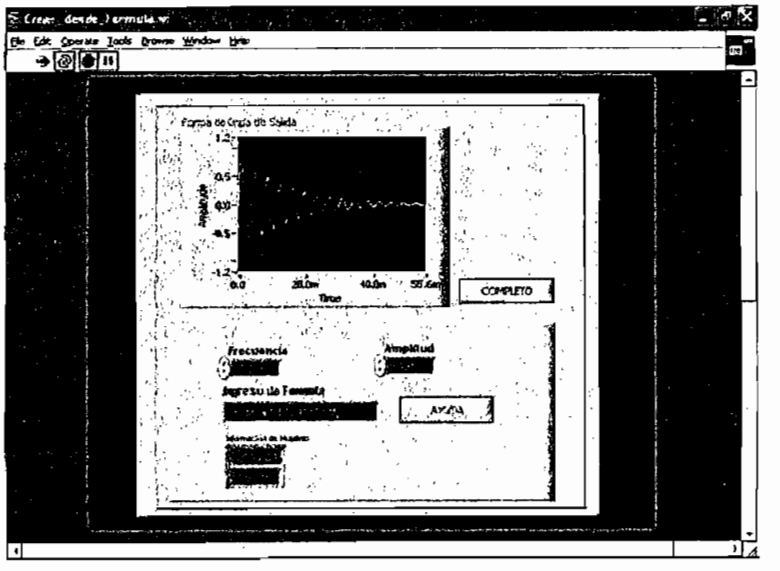


Figura 3.21 Panel Frontal de Crear_desde_Fórmula.VI

El panel frontal del subVI Ayuda_Sintaxis_Fórmula.VI, indica el formato de las variables a usar y el nombre de las funciones disponibles para construir la fórmula de la forma de onda deseada, este se indica en la Figura 3.22.

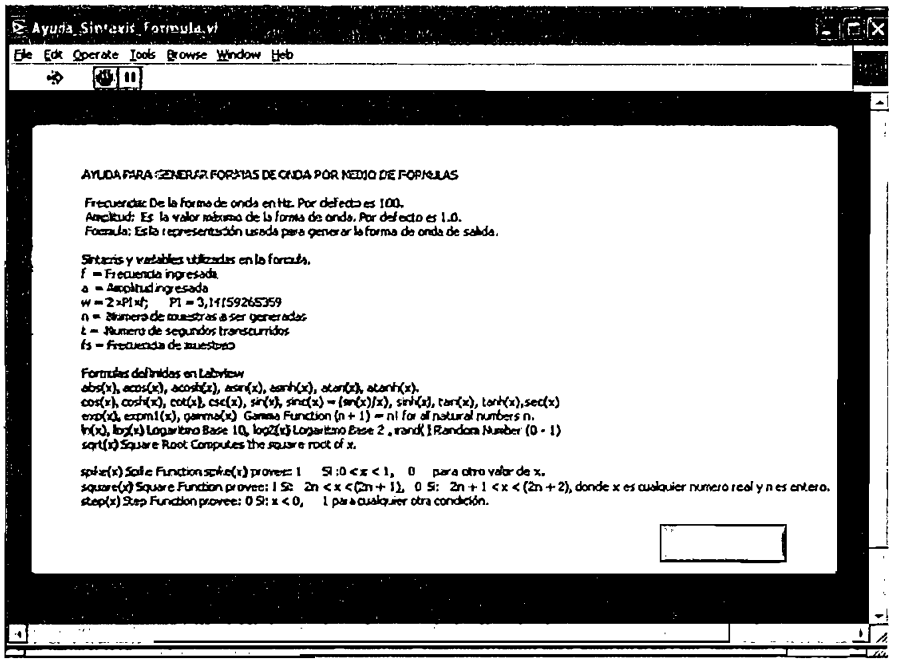


Figura 3.22 Panel Frontal de Ayuda_Sintaxis_Fórmula.VI

3.7 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL CONVERTIR_A_ASCII.VI

El instrumento virtual Convertir_a_ASCII.VI se encarga de convertir los valores de Y de la forma de onda construida a formato ASCII.

El primer paso del algoritmo es convertir los valores de Y, de la forma de onda, en formato de doble precisión (IEEE 64-bits) a números en formato entero con signo de 16 bits, esto se realiza mediante la función *To Word Integer,* que se encuentra en la paleta de funciones, en la librería *NUMERIC*, dentro de la sublibreria *CONVERSIÓN*. Previo a la conversión, se tiene que realizar un escalamiento. El escalamiento se debe al formato que acepta el CODEC desde el sistema DSP.

Para el CODEC el valor de 1 Voltio significa un valor de 7FFF. El valor de 7FFF, para un entero con signo de 16 bits, es 32767, y el rango de los valores de Y de la forma de onda esta entre –1 y +1, al multiplicar cada uno de los valores de Y por el valor de 32767 se obtiene el escalamiento requerido.

Posterior al escalamiento y cambio de formato, se debe formar un nuevo arreglo con los valores de Y; además, se debe incluir el numero de valores de Y en 16 bits, el cual es un parámetro que recibirá el sistema DSP para saber la longitud del arreglo, y también se incluye un número que indica la frecuencia escogida de 48000 Hz como frecuencia de muestreo, este dato es 1.

En la Figura 3.23 se muestra el diagrama de flujo del algoritmo implementado, y en la Figura 3.24, el diagrama de bloques correspondiente.

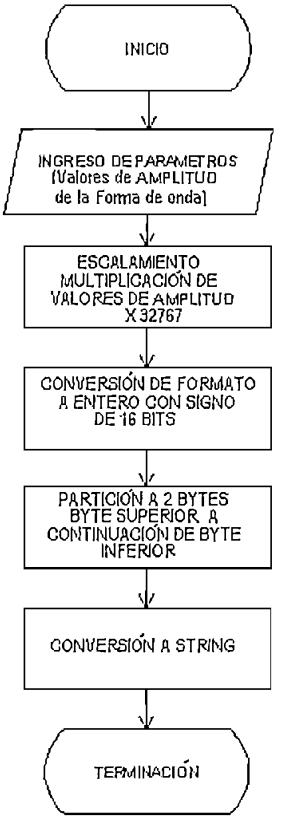


Figura 3.23 Diagrama de Flujo de Convertir_a_ASCII.VI

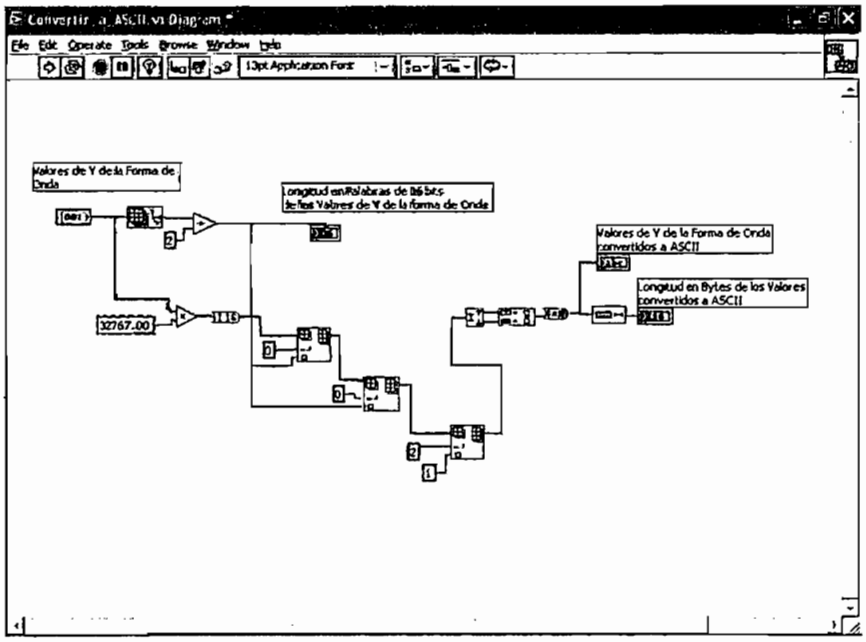


Figura 3.24 Diagrama de Bloques de Convertir_a_ASCII.VI

Después del proceso de escalamiento y del cambio a entero de 16 bits, cada valor se debe particionar en 2 bytes, y colocar uno a continuación de otro dentro del arreglo. Por último se debe convertir, a cada valor del arreglo, en un formato "string" (ASCII).

3.8 DESCRIPCIÓN Y DISEÑO DEL INSTRUMENTO VIRTUAL COMUNICACIÓN_SERIAL_DSP_LABVIEW.VI

El último de los instrumentos virtuales, es el encargado de trasmitir los valores de Y de la forma de onda, previamente transformados a ASCII.

El ingreso al panel frontal es mediante el submenú Trasmisión. Al escoger la opción de *Enviar Datos al sistema DSP,* se realiza la conversión a ASCII por medio del SubVI Convertir_a_ASCII.VI, al terminar la conversión se muestra el panel frontal del VI Comunicación_Serial_Dsp_Labview.VI. El Panel frontal muestra la forma de onda a trasmitir; a los indicadores de la respuesta enviada por el DSP hacia Labview; a los datos convertidos a ASCII de la forma de Onda,

a los errores producidos y al estado de la trasmisión. También posee leds indicadores de la ejecución completa de trasmisión de ciertos datos hacia el DSP. Si el DSP envía una respuesta correcta, el led se prenderá indicando que no hubo un error.

Al enviar un conjunto de datos, el instrumento virtual espera la respuesta del sistema DSP; si la respuesta no llega durante 5 segundos, posteriores a la finalización correcta de una trasmisión, se mostrará un cuadro de información, anunciando una trasmisión incorrecta.

Cuando se produce una falla en la trasmisión de datos, se debe salir hacia el menú principal y escoger nuevamente la opción de *Enviar datos al sistema DSP*, previamente corrigiendo la causa del error producido.

Si la trasmisión se ha producido sin errores, aparecerá un mensaje de trasmisión completa, y después de algunos segundos, se empezará a realizar la generación eléctrica de la forma de onda deseada, por parte del sistema DSP.

El cambio de color de un led, en el panel frontal, indicará la trasmisión completa de un conjunto de datos. Los leds se encuentran ubicados en forma vertical uno a continuación de otro, indicando el orden de trasmisión. El primer led, indica el estado correcto del sistema DSP.

Las fases, que indican los leds, para asegurar una trasmisión correcta de las muestras de la forma de onda son las siguientes:

- Conocer el estado del sistema DSP
- Trasmisión de las muestras de los valores de Y
- Trasmisión de ciertos datos a ciertas localidades del sistema DSP que sirven de configuración del CODEC
- Trasmisión de ciertos datos para inicializar al DSP
- Enviar la tabla de vectores de interrupción del DSP
- Enviar el programa que realiza la generación eléctrica

- Y por último, enviar un comando para traspasar el control hacia el programa de usuario en el DSP

Al completar la última fase, el instrumento virtual no podrá enviar nuevamente los datos a menos que ,desde el sistema DSP, se presione un pulsante para regresar el control al programa de trasmisión serial, controlado por el Programa Monitor.

En la Figura 3.25 se muestra el panel frontal del instrumento virtual Comunicación_Serial_DSP_Labview.VI.

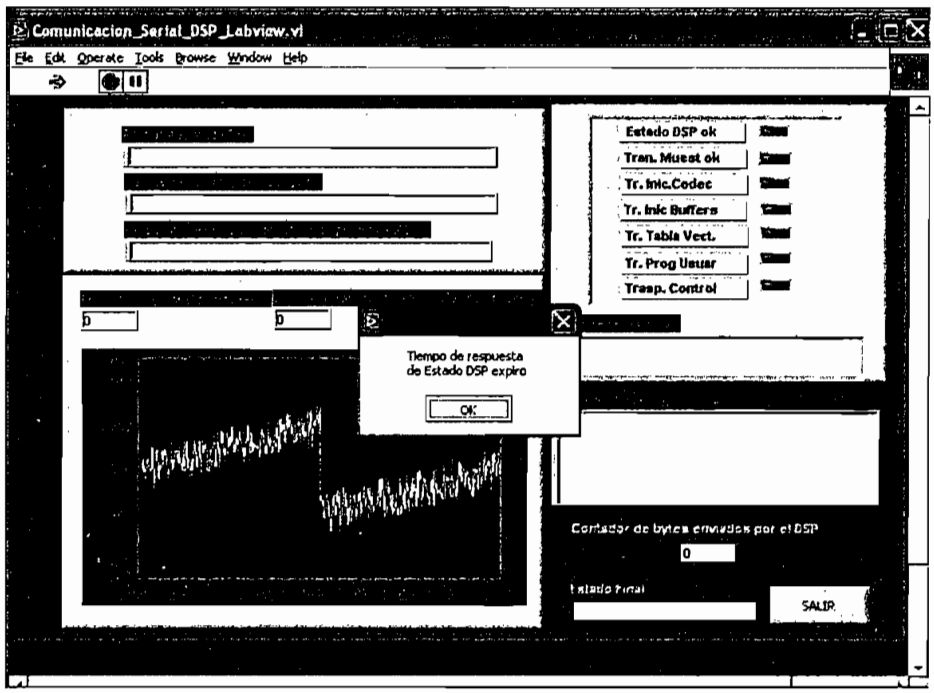


Figura 3.25 Panel Frontal de Comunicación_Serial_DSP_Labview.VI

El algoritmo implementado realiza una comunicación serial entre el instrumento virtual y el programa MONITOR, en el sistema DSP. El programa MONITOR viene como una herramienta de software en la tarjeta de desarrollo DSP. Este programa está almacenado en la memoria EPROM que se encuentra en la tarjeta de desarrollo DSP. Al realizar un "power on" o un "reset", automáticamente se carga el programa almacenado en la memoria EPROM, por medio del puerto BDMA, hacia la memoria de programa del DSP. Este programa MONITOR realiza

una prueba a todos los registros del DSP, procede a inicializar al CODEC y realiza una emulación de comunicación serial asincrónica por medio del timer y las interrupciones generadas por el puerto sincrónico serial 1 del DSP en configuración alternante.

Después de emular la comunicación serial asincrónica, el programa MONITOR espera por ciertos comandos que le indican que acción debe realizar. Las acciones que realiza el programa Monitor son: enviar el estado del DSP; recibir valores y ubicarlos en ciertas localidades de memoria de datos, como sucede con los valores de Y de la forma de onda, recibiendo dos paquetes de 8 bits; también puede recibir datos de 24 bits, en tres paquetes, los cuales pueden ser las instrucciones del programa de usuario, a las que se les puede ubicar en ciertas posiciones de memoria de programa; incluso puede recibir comandos, indicando el envío de valores desde ciertas localidades de memoria de datos o memoria de programa en el DSP, hacia Labview en la PC, por medio de comunicación serial.

El programa MONITOR puede llamar al programa descargado desde Labview como una subrutina, pero debe recibir un comando de traspaso de control. El momento que se envía ese comando desde Labview, el programa MONITOR deshabilita interrupciones, y toma la nueva tabla de vectores de interrupciones del programa de usuario y la traspasa hacia las primeras localidades de memoria de programa, no sin antes guardar su tabla de vectores de interrupción en ciertas localidades de memoria de datos reservadas. Por tal motivo, a más de enviar el programa de usuario debe enviarse la tabla de vectores de interrupción y los datos para inicializar a ciertos registros del CODEC y del DSP. Si no se realiza esta operación previa, el sistema DSP no realizará ninguna acción.

El primer paso que realiza el algoritmo del instrumento virtual de comunicación serial es la configuración del puerto serial. Los parámetros se escogieron de acuerdo a los establecidos en el programa MONITOR para emulación de la comunicación serial.

La velocidad de comunicación se establece a 9600 bits/seg, 8 bits para datos, 1 bit de parada y sin paridad. La longitud del buffer del puerto serial se estableció a una longitud de 12000 bytes, debido a la longitud máxima del arreglo de los valores de y de la forma de onda.

Luego de la configuración del puerto serial, se verifica si existe algún error en la configuración del puerto serial. Si no existe ningún error en la configuración, se inicia la comunicación enviando el comando que revisa el estado del sistema DSP. Este comando esta compuesto de tres caracteres ASCII, cuyo código es: "$$$" [3]; al recibir estos caracteres, por parte del sistema DSP, este envía una respuesta hacia Labview, la cual lleva información del resultado que realiza el programa MONITOR a todos los registros internos. Esta respuesta es mostrada en un indicador de caracteres en el panel frontal.

El instrumento virtual al recibir esta respuesta inicia la siguiente secuencia, si no recibe la respuesta por parte del sistema, espera durante 5 segundos, si durante este periodo no recibe respuesta, aparecerá un cuadro de información indicando que no se puede leer el estado del sistema DSP e indicará que la trasmisión fue incompleta, y no se realizará ninguna otra acción, se debe salir y volver a iniciar la transferencia.

Si la respuesta por parte del sistema DSP llega hasta el programa Labview, indicando que el sistema DSP se encuentra bien, se prenderá el led indicador mostrando que la fase de trasmisión serial indicada fue realizada, después de esto, se inicia la trasmisión del arreglo de datos de Y de la forma de onda construida.

El comando que le indica al programa MONITOR que se iniciará una transferencia de valores hacia ciertas localidades de memoria de datos, esta constituido de tres caracteres ASCII los cuales son: "$DD" [3], además se debe anexar, a estos tres caracteres, la localidad de inicio de memoria de datos en donde se almacenarán, también debe incluirse el número de datos que se enviarán, y por último se debe anexar los datos de Y de la Forma de Onda.

El programa MONITOR al recibir estos datos, los dirige hacia la localidad de inicio, y los almacena a continuación de esta. El valor que indica cuantos datos se enviaron, se utiliza como contador de datos recibidos. Después de realizar esta operación se envía una respuesta hacia Labview, indicando cuantos bytes fueron enviados, y que la operación se realizó; luego de lo cual, regresa a esperar un nuevo comando desde Labview.

El instrumento virtual en Labview espera durante 5 segundos la respuesta del sistema DSP, si no llega la respuesta de terminación de transferencia de datos de amplitud de la forma de onda, se mostrará un cuadro de información indicando que no se realizó la trasmisión; si llega la respuesta, se prenderá el led indicador de realización de esa operación.

La siguiente secuencia es la trasmisión de los datos de inicialización del CODEC, los cuales se deben almacenar en ciertas localidades de memoria de datos establecidas al realizar el programa de generación eléctrica de la forma de onda, esta operación es semejante a la trasmisión de los valores de amplitud de la forma de onda, únicamente cambia la dirección de la localidad de memoria en donde se almacenarán. El comportamiento es semejante al anterior, si la respuesta no llega se muestra un cuadro de información y se interrumpe el proceso de trasmisión serial; si llega la respuesta desde el DSP, se prenderá el led indicador de realización.

Luego, se envían los datos de inicialización de los registros de funcionamiento del DSP hacia localidades de memoria de datos, con el mismo procedimiento anterior. Posteriormente, se debe trasmitir la tabla de vectores de interrupción del programa de generación eléctrica hacia ciertas localidades de memoria establecidas por el programa MONITOR. Debido a que la Tabla de vectores de interrupción son instrucciones y tienen una longitud de 24 bits, se debe enviar cada instrucción en tres bytes. Las instrucciones de la Tabla de vectores de interrupción se logró obtener mediante el Programa depurador VisualDSP.

El programa de depuración Visual DSP posee una ventana para revisar los valores de la memoria de datos y la memoria de programa; con esta ventaja, se obtienen los valores, en formato de 24 bits, de la tabla de vectores de interrupción, luego de lo cual, se digitaron en el diagrama de bloques en la secuencia de trasmisión de datos de la Tabla de vectores de interrupción.

Después de enviar la Tabla de vectores de interrupción y recibir la respuesta por parte del Sistema DSP se realiza la siguiente secuencia, la cual es la transmisión de las instrucciones del programa de generación eléctrica hacia ciertas localidades de memoria de programa.

Debido a que las instrucciones tienen una longitud de 24 bits se deben enviar en tres bytes. El comando que se debe enviar desde Labview hacia el DSP, para trasmisión de instrucciones, son tres caracteres y son "$DP" [3], a continuación de los tres caracteres se debe enviar la dirección en donde se va a almacenar la primera instrucción, la longitud en bytes de las instrucciones a transmitir y posterior a esto se debe anexar el byte superior, luego el byte intermedio y el byte inferior de cada una de las instrucciones una a continuación de otra. Los valores en formato Hexadecimal de las instrucciones se obtuvieron de la ventana de memoria de programa del depurador VisualDSP, al tener estos datos se digitaron en el diagrama de Bloques de la secuencia de trasmisión de programa de usuario para poder trasmitirlos.

Al tener todos los valores de Y de la forma de onda almacenados en la memoria de datos y el programa de generación en la memoria de programa del sistema DSP , se inicia la última secuencia la cual envía un comando al Sistema DSP que indica al programa MONITOR que debe empezar la ejecución del programa de generación eléctrica.

El comando para iniciar la generación es "$GO" [3] el cual indica al programa MONITOR que debe llamar al programa de generación como una subrutina, al hacer esto el programa MONITOR ya no realizará ninguna acción. Al terminar

esta secuencia se indicará que la trasmisión fue completa y posteriormente se cerrará el acceso al puerto serial.

En casi todas las secuencias se utiliza básicamente dos VIs de Labview que realizan la escritura hacia el puerto serial y la lectura desde el puerto serial, estos VIs se encuentran en la paleta de funciones (correspondiente a la comunicación serial) en la librería *Instrument I/O* dentro de la sublibrería *I/O Compatibility* con el nombre de *Serial Port Read.VI* para la lectura el puerto serial, y con el nombre de *Serial Port Write.VI* para la escritura hacia el puerto serial.

En el momento que se realiza una trasmisión completa, no se puede realizar una nueva trasmisión sin antes presionar el pulsante conectado en el pin de interrupción externa IRQE en el sistema DSP, el realizar esta acción el programa de usuario devuelve el control de ejecución a el programa MONITOR y se puede enviar una nueva forma de onda para ser generada. En la Figura 3.26 se muestra una parte del diagrama de bloques del VI Comunicación_Serial_DSP_Labview.VI y en la Figura 3.27 se muestra del diagrama de flujo correspondiente.

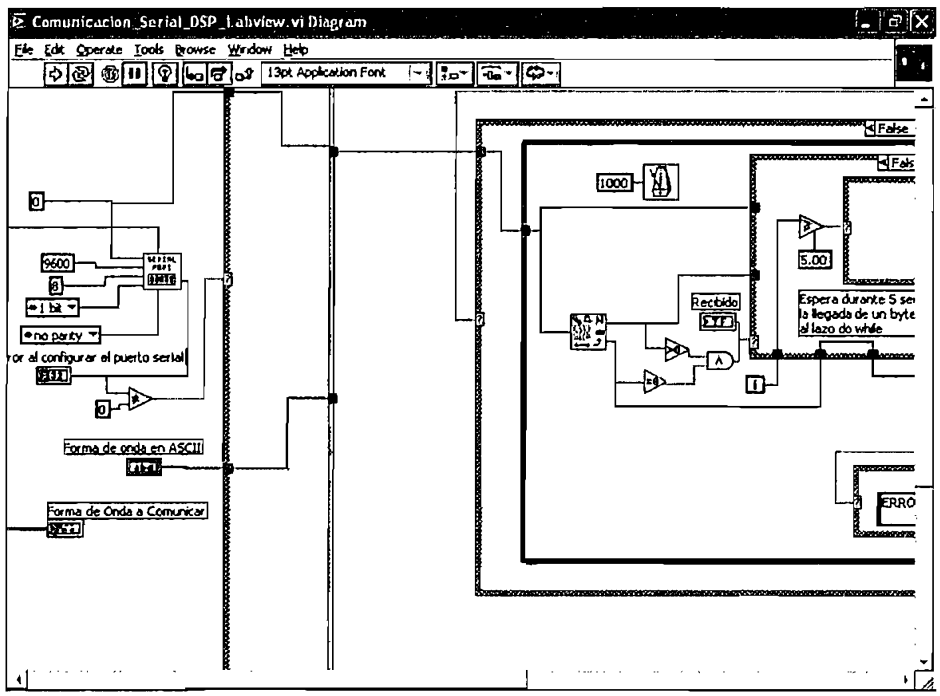


Figura 3.26 Parte del Diagrama de Bloques de Comunicación_Serial_DSP_Labview.VI

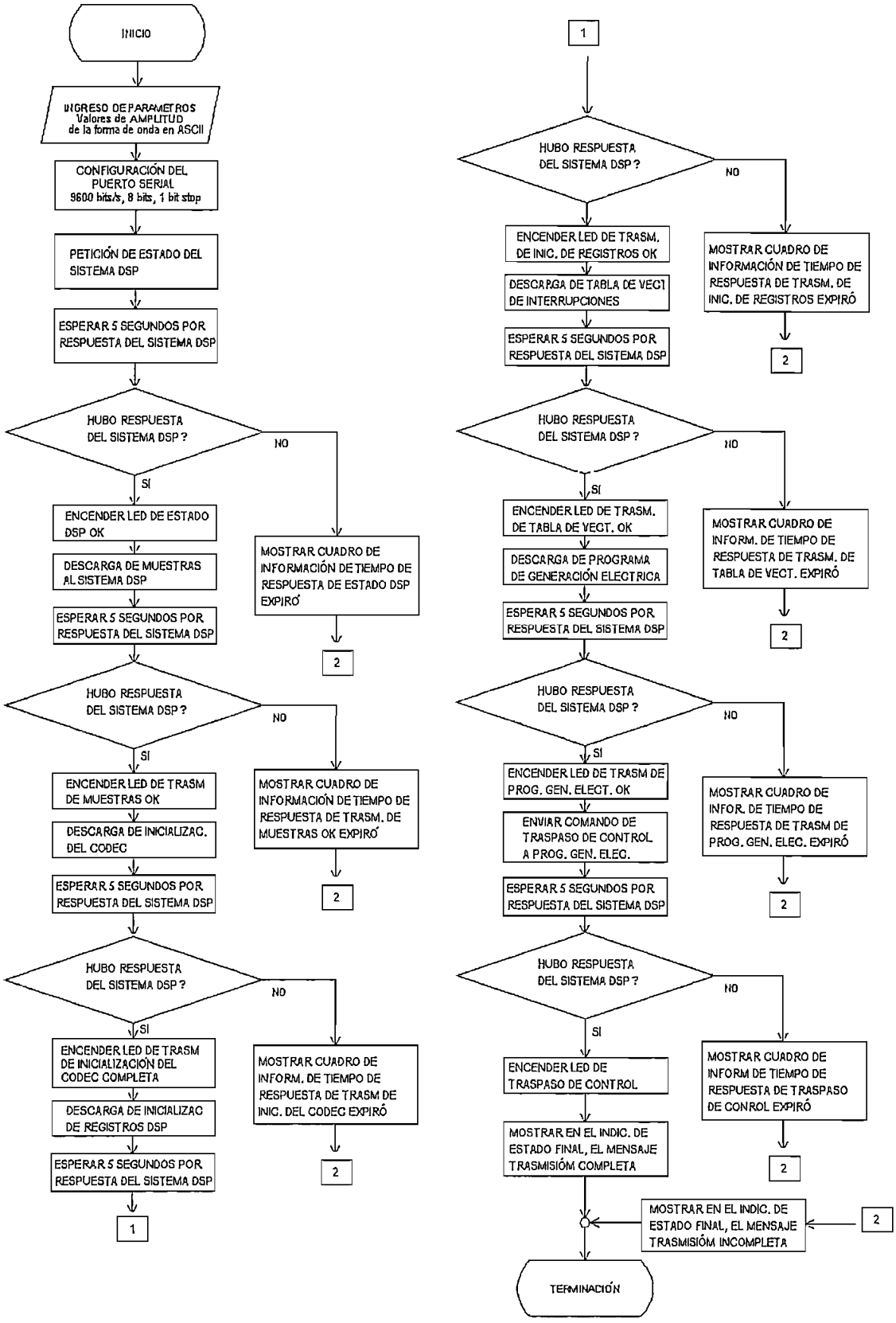


Figura 3.27 Diagrama de Flujo de Comunicación_Serial_DSP_Labview.VI

3.9 DESCRIPCIÓN Y DISEÑO DEL PROGRAMA DE GENERACIÓN ELÉCTRICA GENARBFOR.DSP EN EL SISTEMA DSP

El programa de generación eléctrica que se envía desde Labview hacia la memoria de programa del sistema DSP se llama GENARBFOR.DSP, este programa es el encargado de tomar las muestras de la forma de onda trasmitidas desde Labview y almacenadas en memoria de datos, y enviarlas hacia el CODEC por medio del puerto serial 0 para su generación eléctrica. Este programa posee instrucciones que son ejecutadas por el sistema DSP.

Todos los programas creados para ser utilizados en el sistema DSP requieren de ciertos pasos previos antes de generar el programa final. Además se requiere de ciertas Herramientas de software para su creación.

La primera herramienta de software es el Constructor de Aplicaciones (Aplication Builder), con esta herramienta se define el tipo de procesador a utilizar y las localidades de memoria de datos y de programa a utilizar con su respectiva longitud. Esta herramienta es un programa ejecutable que acepta como parámetro de ingreso un archivo de texto con la extensión .SYS, este archivo de texto debe contener ciertas instrucciones y directivas que indicarán al constructor de aplicaciones en donde debe ubicar y asignar las localidades de memoria de datos y programa de acuerdo a los límites asignados en memoria para el tipo de procesador DSP, también definido en este. Esta herramienta es llamada desde el comando de DOS Bld21.exe, y especificando el nombre del archivo .SYS y ciertos parámetros adicionales. La salida del constructor de aplicaciones es un archivo .ACH el cual será utilizado por otra herramienta de software.

El siguiente paso es escribir el código en assembler del programa de usuario mediante un archivo de texto con la extensión .DSP. Este archivo deberá tener las instrucciones para realizar la acción deseada. Luego mediante la herramienta Enzamblador (ASSEMBLER) se revisan los errores de sintaxis y el apropiado uso

de instrucciones y asignaciones de memoria. Para utilizar esta herramienta se debe ejecutar desde DOS el programa ASM21.exe, anexando el archivo .DSP y el archivo .ACH. La salida del "assembler" son ciertos archivos que indicarán los errores de sintaxis, como es el archivo .Lst, y las asignaciones de memoria, como es el archivo .Map .

Por último se utiliza la herramienta Enlazador (LINKER) para enlazar varios archivos .DSP. Esta herramienta se utiliza llamándola desde DOS mediante el programa Link.exe y especificando los nombres de los archivos .DSP. La salida de esta herramienta es el archivo .EXE que se puede utilizar en el depurador VisualDSP. En versiones recientes todo el proceso se unificó en un ambiente gráfico, mediante una interfaz del programa Visual DSP ++; aunque el formato del constructor de aplicaciones cambio, al igual que las extensiones de los archivos. En el presente Proyecto de titulación se usó la versión anterior del VisualDSP, y se siguió el proceso anterior descrito.

A continuación, se muestra la definición del archivo .SYS utilizado en el programa GENARBFOR. En este archivo se indica el tipo de DSP a utilizar y la definición de las localidades de memoria a utilizar, también se indican las localidades de memoria para las muestras de la forma de onda y las localidades de memoria de programa para el programa de usuario.

```
.SYSTEM GENAR_system;
.ADSP2181;
.SEG/PM/RAM/ABS=0/CODE/DATA     mem_prog[14336];
.SEG/DM/RAM/ABS=0/DATA          muestras[5000];
.SEG/DM/RAM/ABS=5000/DATA       mem_datos[10871];
.ENDSYS;
```

Este programa toma las muestras de las formas de onda creadas y trasmitidas desde Labview hacia las localidades de memoria de datos 0x0000 a 0x1388 (4800 muestras máximo) del DSP.

El Codec de audio AD1847 incluido en la tarjeta de desarrollo genera una trasmisión serial de los datos de entrada analógico convertidos a digital cada cierto periodo, este tiempo es el periodo de muestreo, que se tiene que indicar en el modo de inicialización del Codec, mediante la transmisión serial de ciertos registros de formato, desde el DSP hacia el Codec. Ciertas localidades de memoria de datos poseen los valores de inicialización del Codec, estos datos fueron enviados desde Labview hacia el sistema DSP.

El programa en el DSP no lee ningún dato analógico convertido a digital desde el Codec, pero aprovecha la interrupción generada, dada por la frecuencia de muestreo, para enviar los datos de las muestras de las formas de onda desde el DSP hacia el CODEC, y así poder convertirlos de digital a analógico.

El programa empieza declarando un buffer de trasmisión y un buffer de recepción, cada buffer posee tres localidades de memoria de datos. El buffer de transmisión debe enviar varios datos de inicialización a ciertos registros de control, mediante comunicación serial sincrónica, para tal motivo, se utiliza a el registro I0 y L0 del DSP para apuntar al inicio del buffer de recepción, e I1 y L1 para apuntar al inicio el buffer de trasmisión.

Además se utilizan los registro: I3 y L3, para apuntar hacia los códigos de inicialización del Codec. Posteriormente se configura al puerto serial 0 para utilizar Autobuferado, para recibir trama de sincronización externa y reloj serial externo desde el CODEC. La característica de Autobuferado realiza una interrupción cuando se ha trasmitido un buffer completo de datos.

Posteriormente se configura al sistema y a la memoria del DSP. Se habilita la interrupción del puerto serial 0 y se limpian interrupciones pendientes. Al configurar al puerto serial 0, al sistema y a la memoria del DSP; se inicia la transferencia de códigos de inicialización hacia el Codec. Se desenmascara la interrupción de trasmisión, y se utiliza una bandera de estado, cuando la bandera de estado cambie de 1 a 0 se habrá terminado la inicialización del CODEC. Al

inicio del programa se setea la bandera de estado a 1, se apunta hacia el buffer de trasmisión, y se trasmite el código que indica al Codec el envío de un parámetro de inicialización, este parámetro se encontrará en la primera localidad de memoria del buffer de trasmisión, las dos restantes localidades de memoria sirven para escribir valores hacia los canales izquierdo y derecho del Codec, pero se setean a cero para no generar ninguna señal, al inicializar el Codec.

Cuando se han trasmitido los tres valores de las localidades del buffer de trasmisión, se genera una interrupción debido a la característica de autobuferado. El valor de inicio del buffer de trasmisión es de 16 bits, el Codec reconoce a los 8 bits inferiores como bits de inicialización y a los 8 bits superiores, como bits que indican a que registro se inicializará.

El primer código de inicialización define al registro de control de entrada del canal izquierdo, el siguiente código de inicialización setea al registro de entrada del canal derecho; luego, los cuatro próximos códigos, setean a los registros de control de los canales auxiliares 1 y 2, los dos siguientes códigos setean al registro de control del DAC, del canal izquierdo y derecho; el siguiente código indica el formato del dato que recibirá el Codec y la frecuencia de muestreo, y los siguientes parámetros inicializan la forma del sistema de comunicación.

En cada envío de un código de inicialización se genera una interrupción. Dentro de la subrutina de atención a la interrupción del puerto serial 0 se chequea si se ha enviado el último código de inicialización, si se ha terminado de enviar todos los códigos de inicialización hacia el Codec, la subrutina coloca el valor de 0 a la bandera de estado. El programa principal revisa esta bandera de estado, si el valor es 1 se encierra en un lazo y no prosigue, al cambiar a 0, sale de ese lazo y prosigue con la siguiente instrucción.

Antes de terminar de enviar los códigos de inicialización del Codec, se setea un bit de autocalibración de los conversores A/D y D/A. Después de enviar los códigos de inicialización se lee el bit que indica la realización de Autocalibración,

si este bit no cambia de estado, el programa principal, se encierra en un lazo y no prosigue la ejecución.

Una vez que la Autocalibración esta completa, se activan los canales DAC izquierdo y derecho escribiendo a los respectivos registros indexados por medio del buffer de trasmisión. Al realizar esto se limpian interrupciones pendientes, y el programa principal se encierra en un lazo que revisa a una bandera de retorno, esta bandera normalmente esta en 0, pero si se presiona el botón conectado a la interrupción externa IRQE, se produce un salto a la subrutina de atención a esta interrupción, en esta subrutina se setea a la bandera de retorno a 1; al realizar esto, el programa principal termina, y retorna el control al programa MONITOR, ya que este llama al programa de usuario como si fuera una subrutina. Mientras no se presione el botón conectado a IRQE, el programa principal se encierra en un lazo.

El DSP, al término de un periodo de muestreo, genera una interrupción de recepción. En la subrutina de atención a esta interrupción, se trasladan los valores de los datos de amplitud de la forma de onda, trasmitidos y almacenados en memoria de datos, hacia el buffer de trasmisión; y se envían hacia el CODEC, mediante un apuntador a las localidades en que se encuentran estos valores.

También se define un contador de muestras, si la cantidad del contador es igual a la cantidad del número de muestras de la forma de onda, quiere decir que se ha llegado a la última muestra; al detectar la última muestra, se indica al puntero de las localidades, que regrese a la primera localidad en donde esta almacenado el valor de amplitud de la primera muestra de la forma de onda.

En la Figura 3.28 se muestra el diagrama de Flujo del programa principal GENARBFOR.DSP, y en las Figuras 3.29, 3.30 y 3.31 se muestran los diagramas de flujo de las subrutinas de atención de trasmisión, recepción e interrupción externa IRQE.

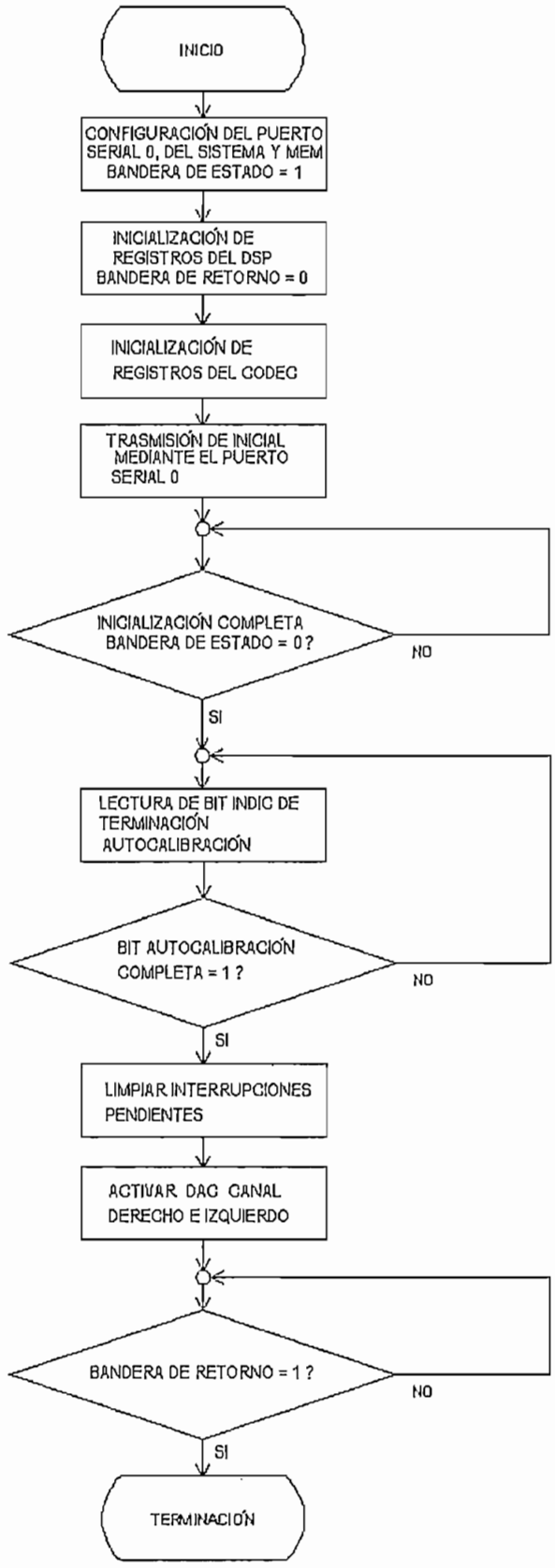


Figura 3.28 Diagrama de Flujo del Programa Principal GENARBFOR.DSP

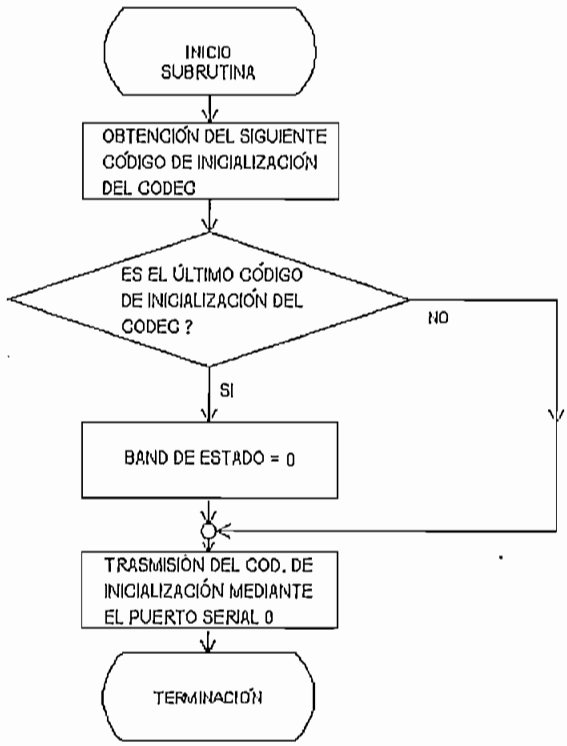


Figura 3.29 Diagrama de Flujo de la Subrutina de Atención a la Interrupción de Trasmisión

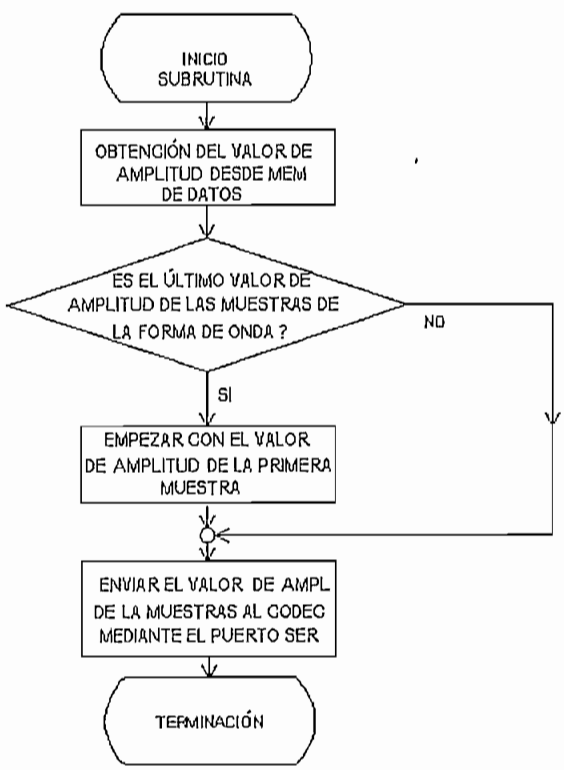


Figura 3.30 Diagrama de Flujo de la Subrutina de Atención a la Interrupción de Recepción

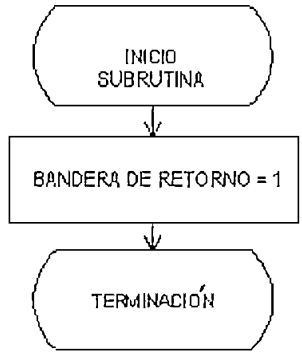


Figura 3.31 Diagrama de Flujo de la Subrutina de Atención a la Interrupción Externa IRQE

3.10 SUMARIO

En el presente Capítulo se han expuesto el diseño y la implementación, tanto del programa de selección en Labview como el programa de generación eléctrica en el DSP. Labview posee muchas herramientas para la generación de señales y para realizar comunicación serial. En cambio el sistema DSP maneja datos de 16 bits y posee dos puertos seriales sincrónicos , uno para emular una comunicación serial asincrónica y el otro para comunicarse con el CODEC. Por lo tanto la utilización de ambas herramientas en conjunto son ideales para el objetivo de generación arbitraria de formas de onda.

CAPITULO 4

PRUEBAS Y RESULTADOS

4.1 INTRODUCCIÓN

En este Capítulo se presentan los resultados obtenidos de las pruebas realizadas a cada una de las opciones del Generador Arbitrario de Formas de Onda. Las pruebas se realizaron con un PC Pentium III de 866 MHz con 256 MB de memoria RAM, en donde se ejecuta el programa de Interfaz de usuario del Generador arbitrario de Formas de Onda en Labview. Además se encuentra conectado en el puerto serial, la tarjeta de desarrollo EZ-KIT LITE, en la cual se encuentra el microprocesador ADSP-2181, encargado del control del CODEC para la generación eléctrica. La salida analógica del CODEC, por donde se obtiene la forma de onda eléctricamente, se encuentra conectada hacia un osciloscopio marca TEKTRONIX TDS100. El osciloscopio TEKTRONIX TDS100 posee una interfaz gráfica para la descarga de las formas de onda mostradas en su pantalla hacia la PC por medio de un cable serial RS-232. La interfaz gráfica para la familia de osciloscopios TEKTRONIX TDS se llama WAVESTAR y la versión utilizada para estas pruebas es la 1.0.

El procedimiento a seguir es ejecutar el Generador Arbitrario de Forma por medio de Labview, al llegar al Menú Principal se escoge una opción en el menú Selección de Forma de Onda, se construye formas de onda con cada una de las opciones con diferentes frecuencias y con diferentes amplitudes. El rango predeterminado de Amplitud esta entre +1 y –1 voltios, debido a las limitaciones especificadas en la hoja de datos del CODEC. El rango predeterminado de frecuencia esta entre 10 Hz y 4000 Hz, y solo se deben generar frecuencias en las cuales la división de la frecuencia de muestreo para la frecuencia deseada sea un número entero, debido a que este resultado es el número de muestras. Al producirse lo contrario, al tener un valor decimal, existirá un error en frecuencia de la forma de onda, ya que al completar un periodo de la forma de onda se obtendrá una última muestra que se encuentra en el número de muestras por el periodo de

muestreo, mientras en verdad el último valor de la forma de onda se encuentra entre esa última muestra y la siguiente debido al valor decimal del número de muestras obtenido. Si se quiere generar periódicamente una forma de onda con un número de muestras que no es un número entero, se podrá ver un pequeño cambio brusco entre el último valor y el primer valor del siguiente periodo de la forma de onda. El error que se produce en frecuencias bajas es casi insignificante, por lo que se pueden generar frecuencias entre 10 Hz y 100 Hz en pasos de 1 Hz, luego de este valor el error es insignificante con pasos de 10 Hz, entre frecuencias entre 100 Hz y 1000 Hz. Al pasar los 1000 Hz, se deben tener pasos de 500 Hz, debido a que luego de los 1000 Hz, la frecuencia deseada se acerca a la frecuencia de muestreo y el error es mas notable; con un paso de 500Hz se obtienen errores muy pequeños o no se tiene errores. El error máximo en el periodo de la forma de onda deseada es de ± 1/2 periodo de muestreo (±10 us), pero únicamente si el resultado de la división entre la frecuencia de muestreo y la frecuencia deseada no es un número entero.

En la Figura 4.1 se muestra la interfaz gráfica para la obtención de las formas de onda mostradas en el osciloscopio TEKTRONIX TDS 1000 hacia la PC, llamada WAVESTAR.

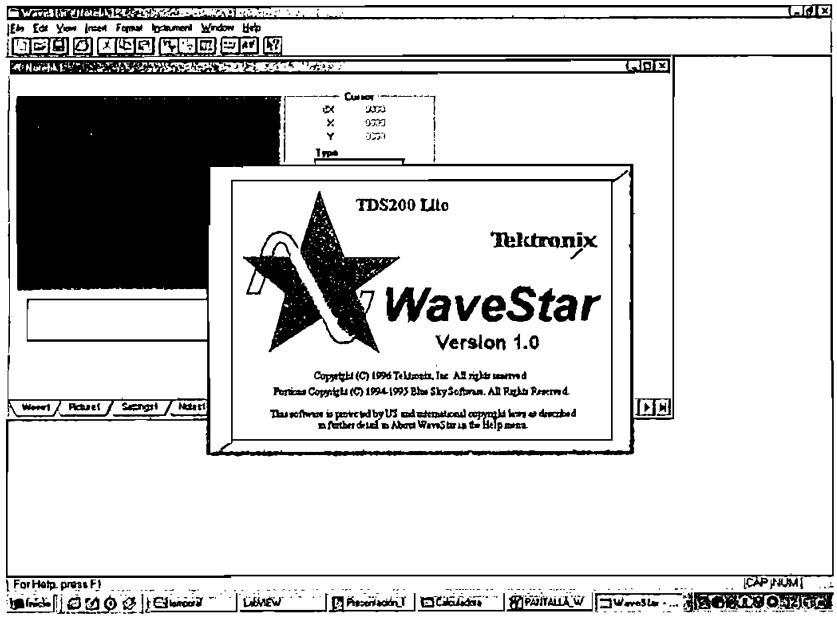


Figura 4.1 Presentación Inicial de la interfaz gráfica WAVESTAR

El límite inferior de la frecuencia deseada se determinó en base al número de muestras obtenidas con la frecuencia de muestreo de 48000 Hz, ya que se obtiene un número de 4800 muestras, si la frecuencia escogida es igual a 10 Hz. Al escoger una frecuencia inferior a esta, por ejemplo, si la frecuencia escogida para la forma de onda fuese 5 Hz, se debería tener un número de 9600 muestras y cada una ocuparía una localidad de memoria de datos de 16 bits en el sistema DSP, llegando a sobrepasar la asignación de memoria para los datos de amplitud.

La memoria de datos del ADSP-2181 tiene una capacidad de 16K palabras de 16 bits pero algunas localidades son registros mapeados en memoria, otras localidades son usadas para variables que necesita el programa MONITOR y otras son localidades que se usan en el programa de generación eléctrica, por lo tanto no se deben trasmitir 16000 muestras.

Además, el CODEC esta diseñado para generar señales de sonido entre 20 Hz y 20000 Hz, al generar señales con frecuencias menores de 20 Hz, con el Generador Arbitrario de Formas de Onda, se está cruzando el limite inferior recomendado de diseño del CODEC.

El límite superior del rango de frecuencias predeterminado se estableció debido a que el número de muestras obtenido con una frecuencia escogida de 4000 Hz, con una frecuencia de 48000 Hz, es 12; y al pasar los 4000 Hz, se obtiene un número de muestras muy reducido que distorsiona la señal a construir.

Al construir la forma de onda, esta se debe trasmitir hacia el sistema DSP, si la trasmisión fue completa, después de unos pocos segundos, se empezará a generar eléctricamente. Al trasmitir la forma de onda, se trasmite también el programa que realiza la generación eléctrica, luego de lo cual ya no se necesitará al puerto serial y se puede desconectar al sistema DSP. El momento que el DSP se encuentra generando eléctricamente la forma de onda, se puede conectar el cable serial RS-232 desde el osciloscopio hacia la PC y descargar, por medio del puerto serial, las formas de onda mostradas en el osciloscopio hacia la interfaz gráfica WAVESTAR. En las siguientes secciones se muestra las gráficas de la

formas de ondas tomadas desde el osciloscopio con la interfaz gráfica WAVESTAR y los dibujos de las formas de onda construidas desde la interfaz gráfica de usuario del Generador Arbitrario de Forma de Onda en Labview, construidas con las diferentes opciones.

4.2 PRUEBAS REALIZADAS CON LA OPCIÓN CREAR DESDE PANTALLA DEL GENERADOR ARBITARIO DE FORMAS DE ONDA

Mediante la opción Crear desde Pantalla, se pueden construir formas de onda arrastrando el cursor, presionando el botón izquierdo del mouse. Esta acción dibujará una línea sobre una pantalla en donde se puede ver la forma de onda creada. Las pruebas se realizaron construyendo una forma de onda aleatoria sobre esta pantalla de visualización.

La Figura 4.2 muestra la forma de onda construida en la interfaz del Generador Arbitrario de Forma de Onda, y la Figura 4.3 muestra la forma de onda tomada desde el osciloscopio TEKTRONIX con la interfaz gráfica WAVESTAR.

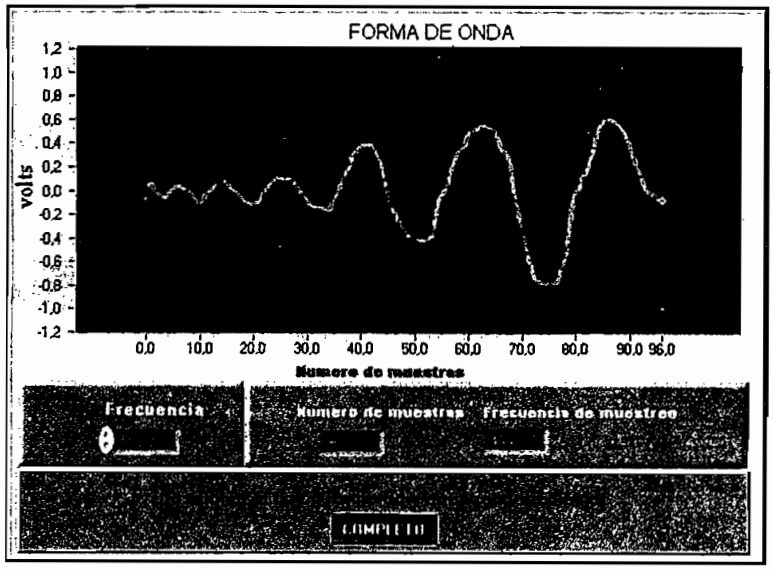


Figura 4.2 Construcción de Onda con la opción Crear desde Pantalla (10 Hz, 0.7 Voltios)

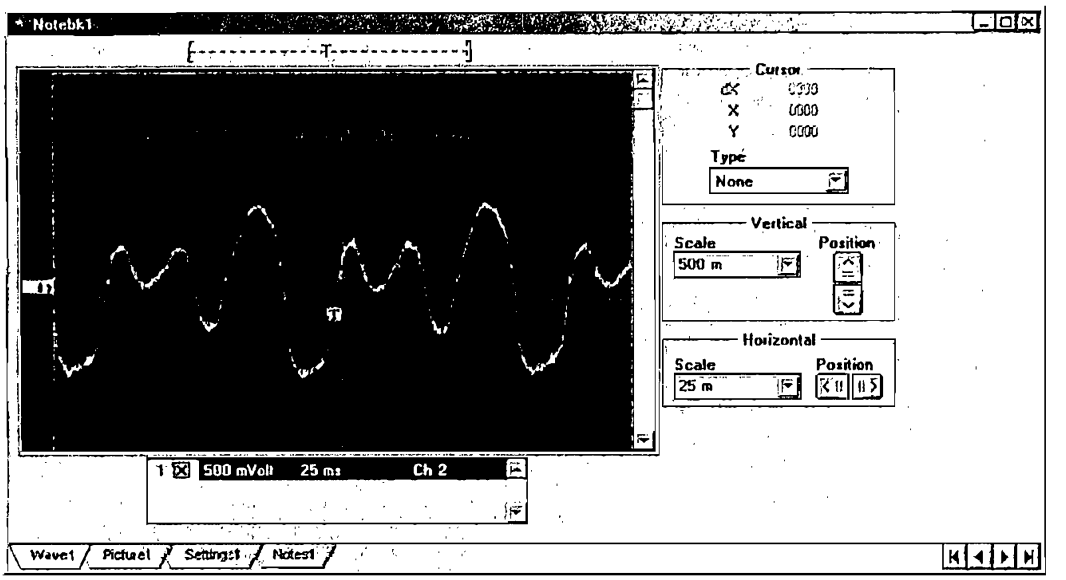


Figura 4.3 Forma de Onda obtenida desde el osciloscopio TEKTRONIX (10 Hz, 0.7 Voltios)

En la Figura 4.2 se puede observar que la amplitud es de aproximadamente 0.7 Voltios, y con un frecuencia escogida de 10 Hz, cuyo periodo es 100 ms.

Comparando la forma de onda tomada desde el osciloscopio, mostrada en la Figura 4.3, se puede observar que el periodo de la onda en el osciloscopio es de 4 divisiones en la escala de tiempo, y además se indica, en la parte derecha, los valores de cada división tanto en la escala horizontal (Tiempo) y la escala vertical (Voltios), por tanto el periodo de la onda es de 4 divisiones x 25ms/división = 100 ms, cuyo periodo corresponde a 10 Hz; y también se puede observar que se esta generando en forma periódica.

El nivel de cero voltios del canal se indica en el extremo izquierdo y se encuentra desplazado una subdivisión. En la escala vertical se encuentran 5 subdivisiones, en una división que corresponde a 500 mV, entonces cada subdivisión es 100 mV; si se cuenta las subdivisiones se obtiene 7 x 100 mV = 0.7 Voltios, teniendo en cuenta que el nivel de cero voltios se encuentra desplazado a 100 mV. Con esto se demuestra la igualdad tanto en frecuencia como en amplitud de la forma de onda construida mediante la interfaz del Generador Arbitrario de Forma de Onda, y la Forma de onda eléctrica tomada desde el osciloscopio. En la Figura

4.4 se muestra la forma de onda creada desde pantalla con una frecuencia escogida de 68 Hz, y en la Figura 4.5 se muestra la forma de onda captada desde el osciloscopio por medio del puerto serial y la interfaz gráfica WAVESTAR.

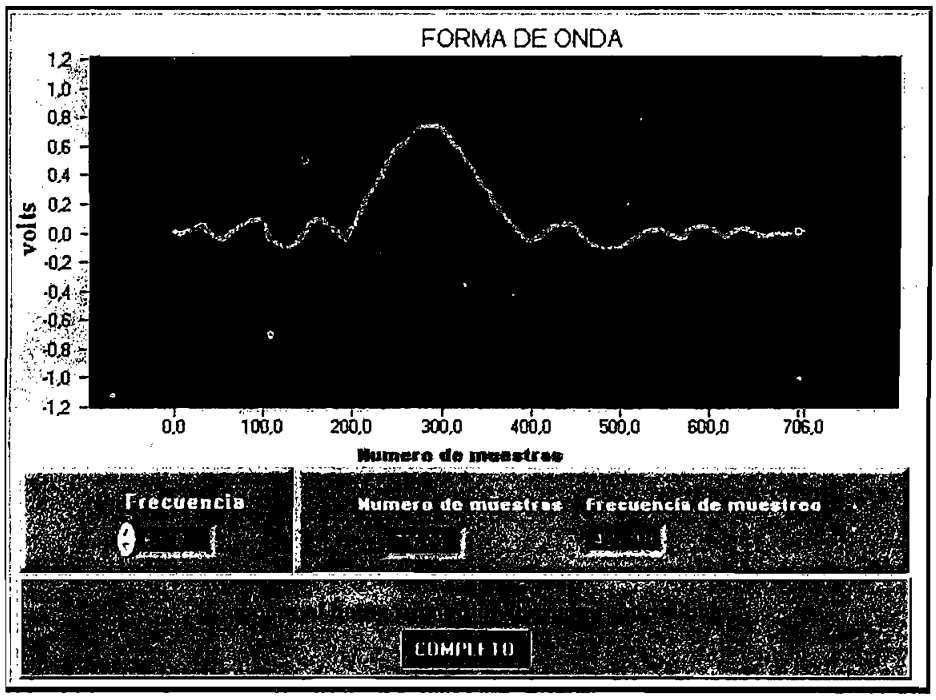


Figura 4.4 Construcción de Onda con la opción Crear desde Pantalla (68 Hz, 0.7 Voltios)

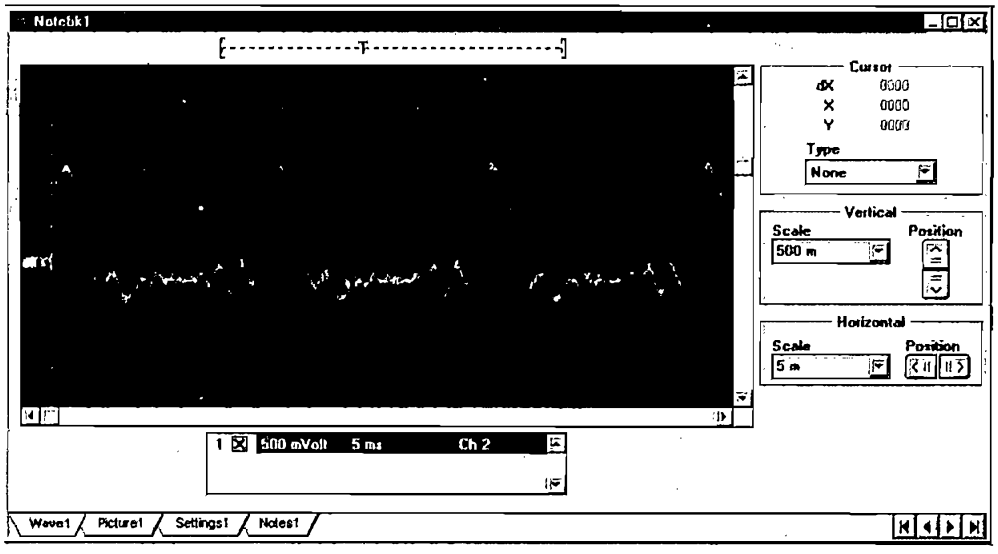


Figura 4.5 Forma de Onda obtenida desde el osciloscopio TEKTRONIX (68 Hz, 0.7 Voltios)

Para observar el efecto del reducido número de muestras para frecuencias altas en la Figura 4.6 se muestra una forma de onda con una frecuencia escogida de 4000 Hz y en la Figura 4.7 se encuentra la forma de onda mostrada en el osciloscopio.

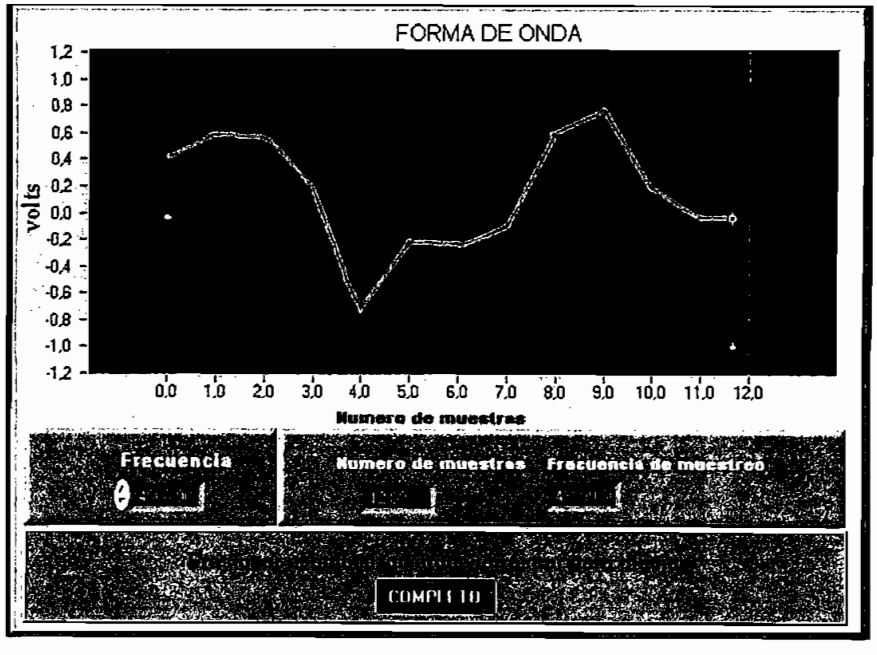


Figura 4.6 Construcción de Onda con la opción Crear desde Pantalla (4000 Hz, 0.8 Voltios)

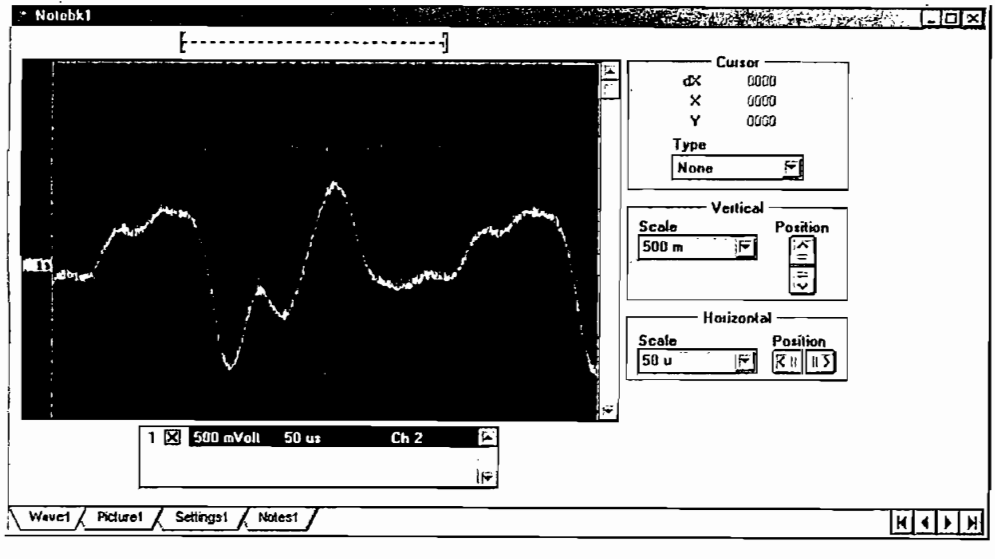


Figura 4.7 Forma de Onda obtenida con el osciloscopio TEKTRONIX (4000 Hz, 0.8 Voltios)

Analizando la Figura 4.6 se puede observar que al tener una frecuencia escogida de 4000 Hz se tiene únicamente 12 muestras, y el VI únicamente realiza una interpolación lineal entre un punto anterior y un punto posterior. Por tal motivo se muestra la forma de onda constituida por líneas rectas.

Se puede observar en la Figura 4.7 que la forma de onda es semejante, pero la unión de los puntos es continua, esto es debido a los capacitores usados en los filtros de salida del CODEC.

4.3 PRUEBAS REALIZADAS CON LA OPCIÓN CREAR DESDE LIBRERÍA DEL GENERADOR ARBITARIO DE FORMAS DE ONDA.

La opción de Crear desde Librería también posee los rangos predeterminados en amplitud y frecuencia, anteriormente explicados.

En esta opción se pueden generar formas de ondas conocidas como: ondas Senoidales, ondas Triangulares, ondas Diente de Sierra y ondas Cuadradas. Además se puede realizar una operación básica con las dos formas de onda. Las operaciones básicas son la de sumar y restar entre ondas de diferente frecuencias.

En la opción de suma entre formas de onda se puede sumar ruido blanco uniforme sobre una de las formas básicas antes mencionadas.

Si se quiere sumar formas de onda de diferente frecuencia, se mostrará un periodo de la forma de onda con menor frecuencia, y los periodos respectivos de la forma de onda con mayor frecuencia. En el resultado de la operación se observará el periodo de la forma de onda con menor frecuencia.

En la Figura 4.8 se muestra la selección de la generación de una onda senoidal de 10 Hz, y en la Figura 4.9 se muestra la forma de onda eléctrica tomada desde el osciloscopio.

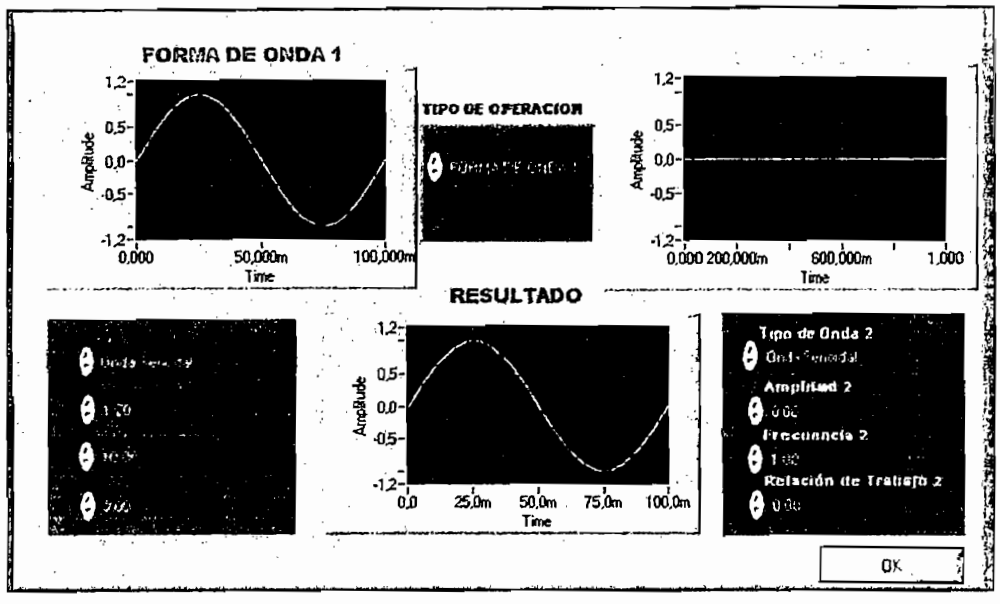


Figura 4.8 Onda Senoidal con la opción Crear desde Librería (10 Hz, 1 Voltio)

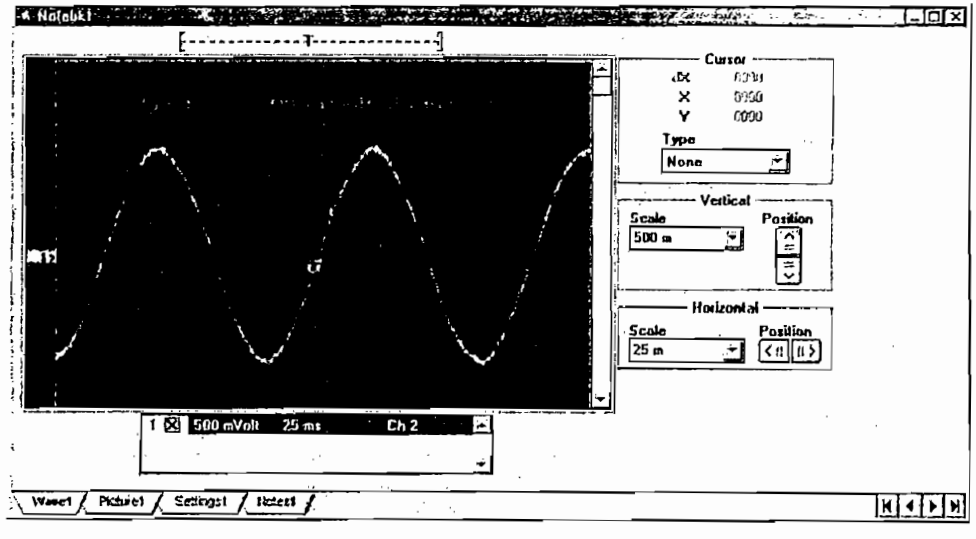


Figura 4.9 Onda Senoidal obtenida desde el osciloscopio TEKTRONIX (10 Hz, 1 Voltio)

Una opción interesante se muestra en la Figura 4.10 y es la selección de la generación de una onda senoidal de 25 Hz con una amplitud de 1 V, adicionando

Ruido Blanco Uniforme de 1000 Hz con una amplitud de 0.5 V. En la Figura 4.11 se muestra la forma de onda eléctrica resultante tomada desde el osciloscopio.

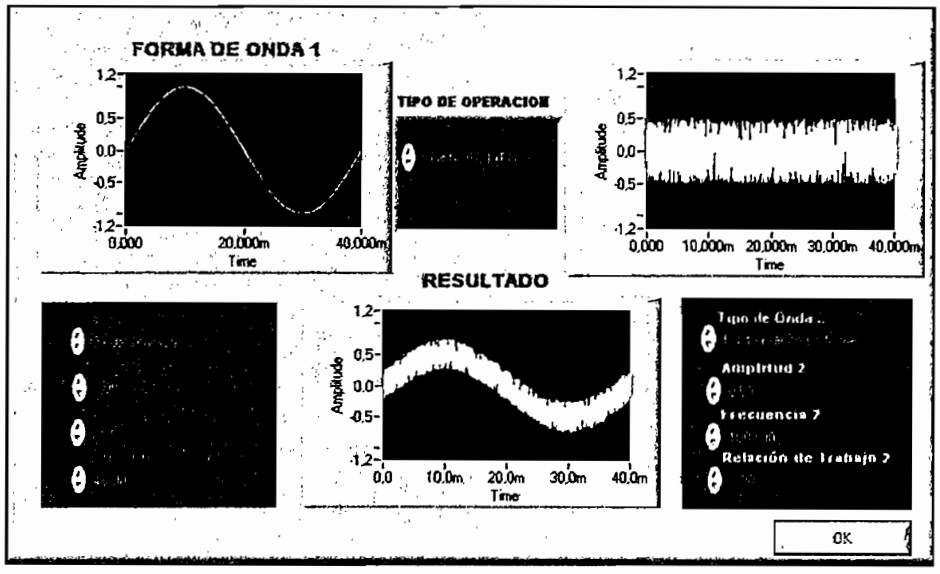


Figura 4.10 Onda Senoidal (25 Hz, 1 V) con Ruido Blanco Uniforme (1000 Hz, 0.5 V)

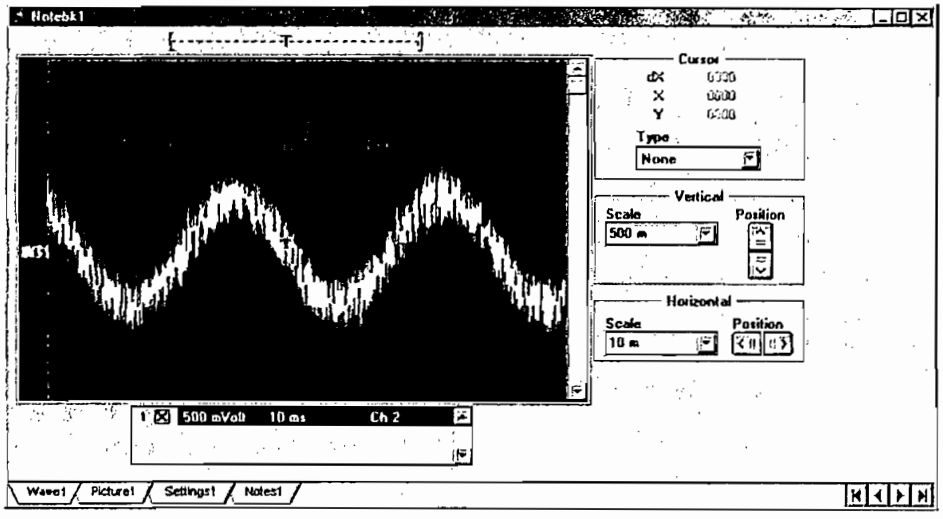


Figura 4.11 Onda Senoidal con Ruido Blanco Uniforme desde el osciloscopio (25 Hz, 1 V)

Otra Forma de onda básica se muestra en la Figura 4.12, en la cual se observa la generación de una onda triangular de 100 Hz con una amplitud de 1 V. La forma de onda tomada desde el osciloscopio se muestra en la Figura 4.13.

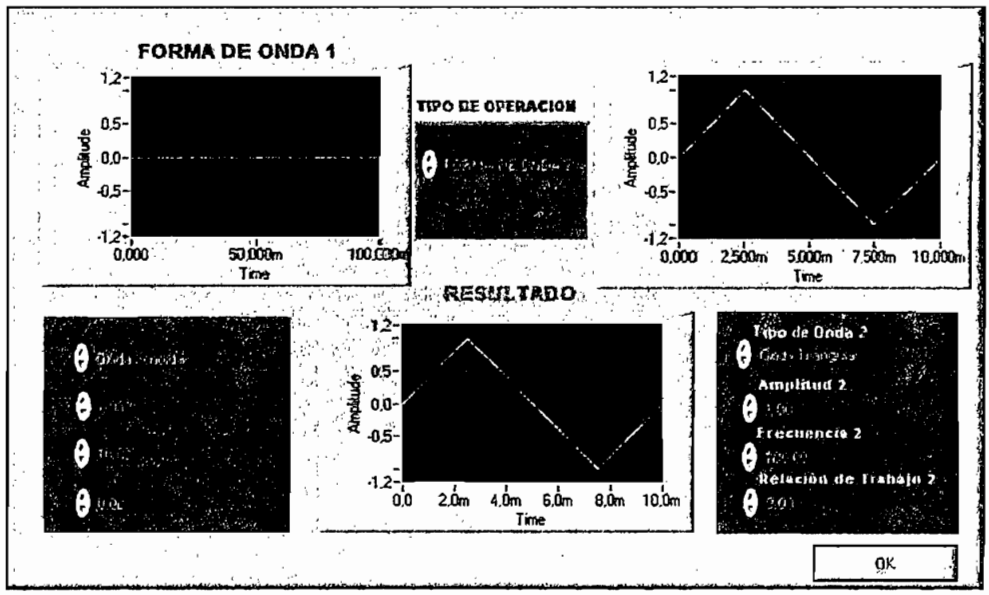


Figura 4.12 Onda Triangular con la opción Crear desde Librería (100 Hz, 1 Voltio)

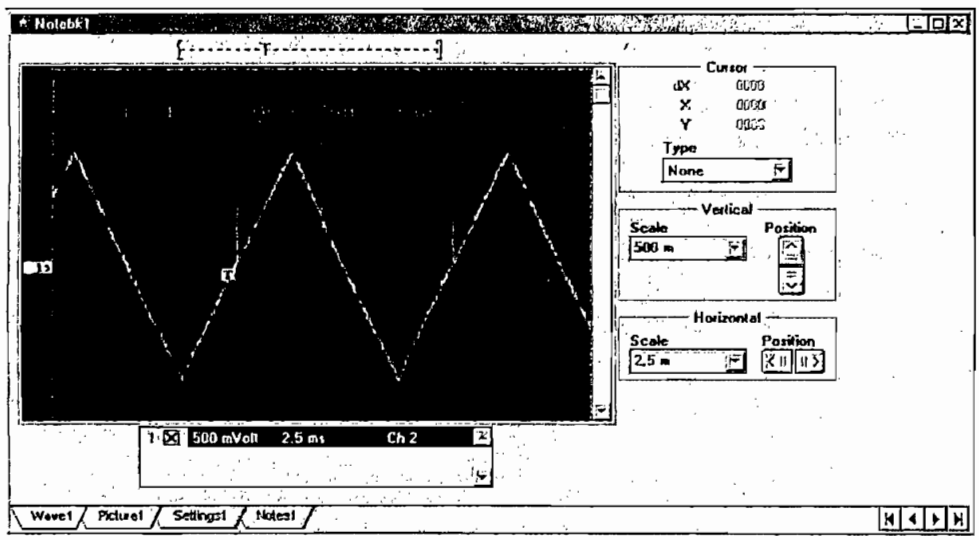


Figura 4.13 Onda triangular obtenida desde el osciloscopio TEKTRONIX (100 Hz, 1 Voltio)

Los picos mostrados en la Figura 4.13 son posiblemente un error en la trasmisión de una muestra de la forma de onda.

La aplicación de una operación básica entre formas de onda, se observa en la Figura 4.14, en la cual se indica la forma de onda resultante de sumar una onda senoidal de 25 Hz con amplitud de 1 V con una onda senoidal de 1000 Hz con una amplitud de 0.5 V. La amplitud máxima de la señal resultante es 1.5/2 = 0.75 V. La señal resultante captada desde el osciloscopio se muestra en la Figura 4.15.

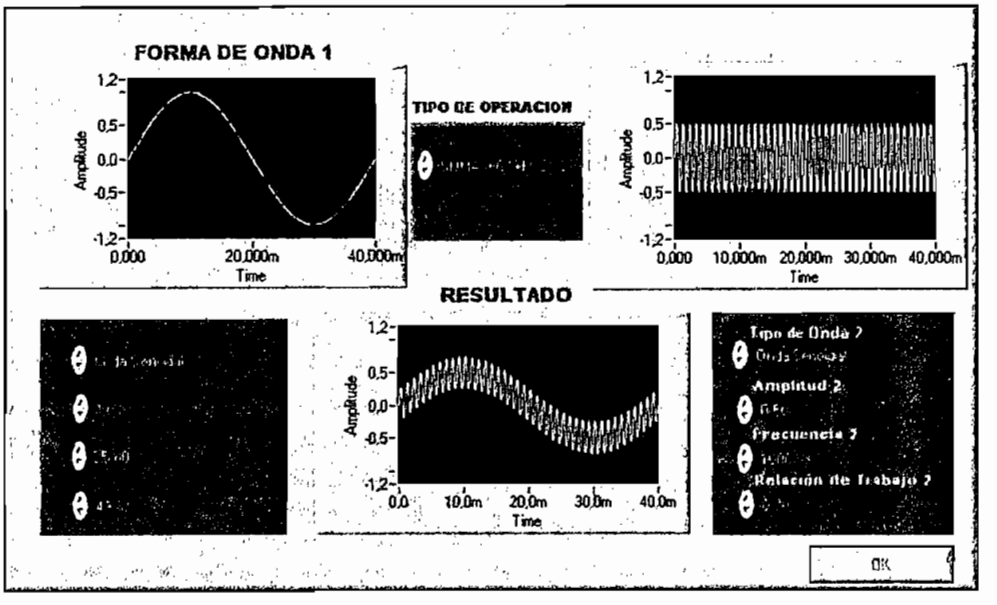


Figura 4.14 Suma de una Senoidal (25 Hz, 1 V) con una Senoidal (1000 Hz, 0.5 V)

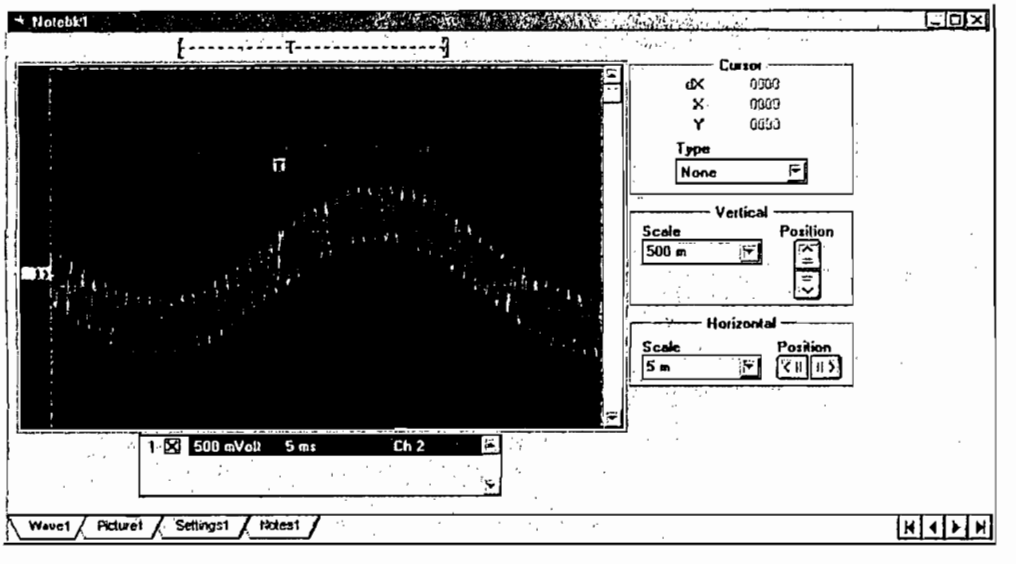


Figura 4.15 Onda Resultante de la suma de Senoidales captada desde el osciloscopio

En la Figura 4.16 se tiene la forma de onda resultante de restar una onda senoidal de 2000 Hz de amplitud 1V con una onda triangular de 100 Hz de 1 V. La forma de onda descargada desde el osciloscopio se muestra en la Figura 4.17.

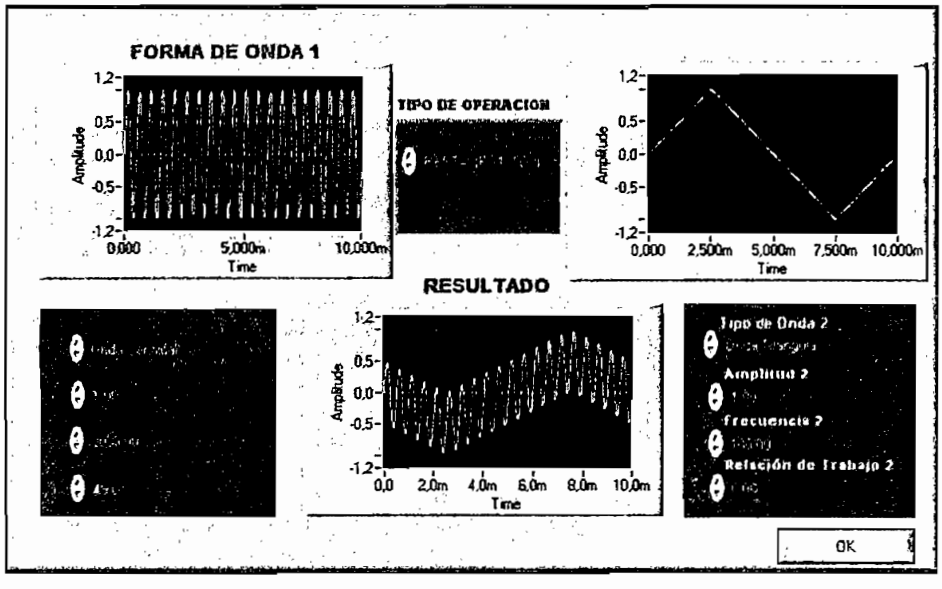


Figura 4.16 Onda Senoidal (2000 Hz, 1 V) menos Onda Triangular (100 Hz, 1 V)

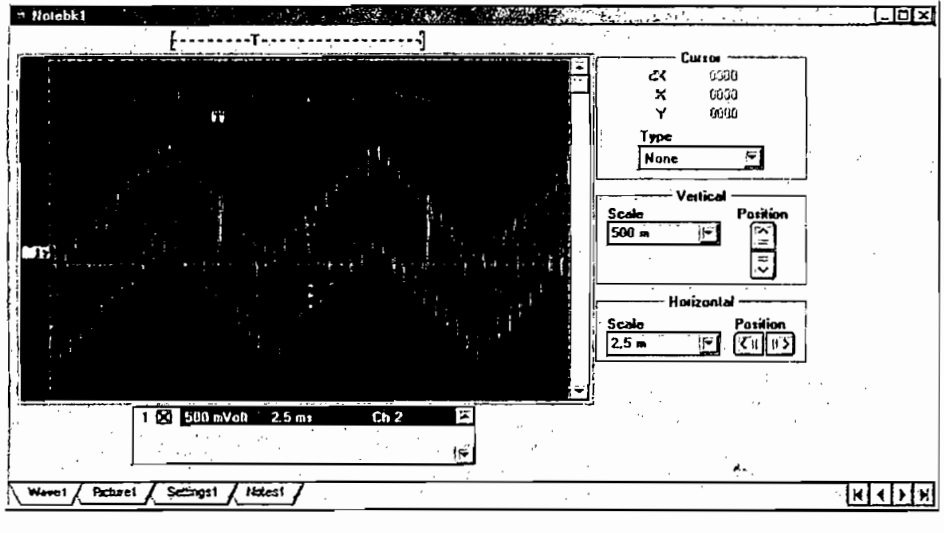


Figura 4.17 Onda Senoidal menos Onda triangular tomada desde el osciloscopio

En la siguiente Figura 4.18 se muestra la construcción de una forma de onda Cuadrada de 200 Hz con una relación de trabajo de 0.49, y en la Figura 4.19 se muestra la forma de onda obtenida por medio del osciloscopio.

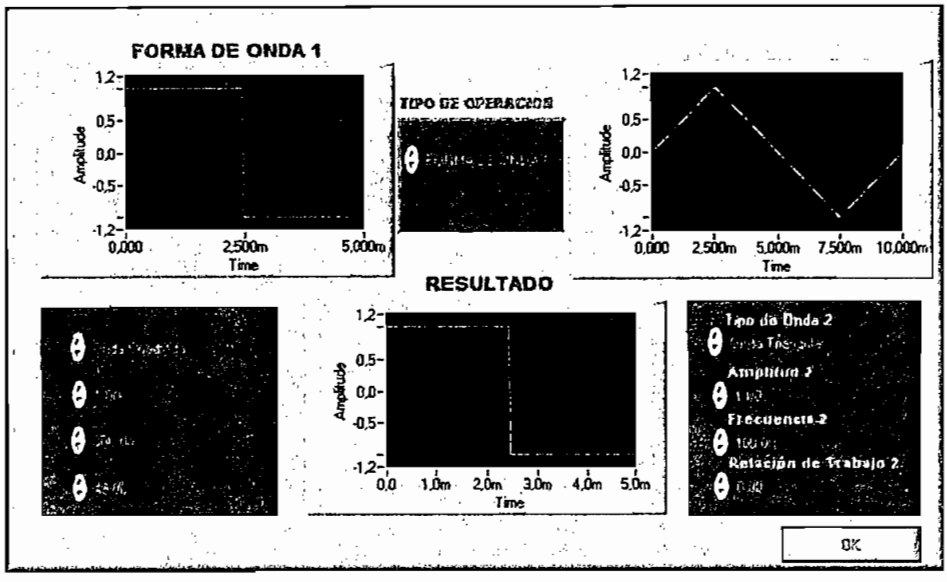


Figura 4.18 Onda Cuadrada (200 Hz, 1 Voltio, relación de trabajo = 0.49)

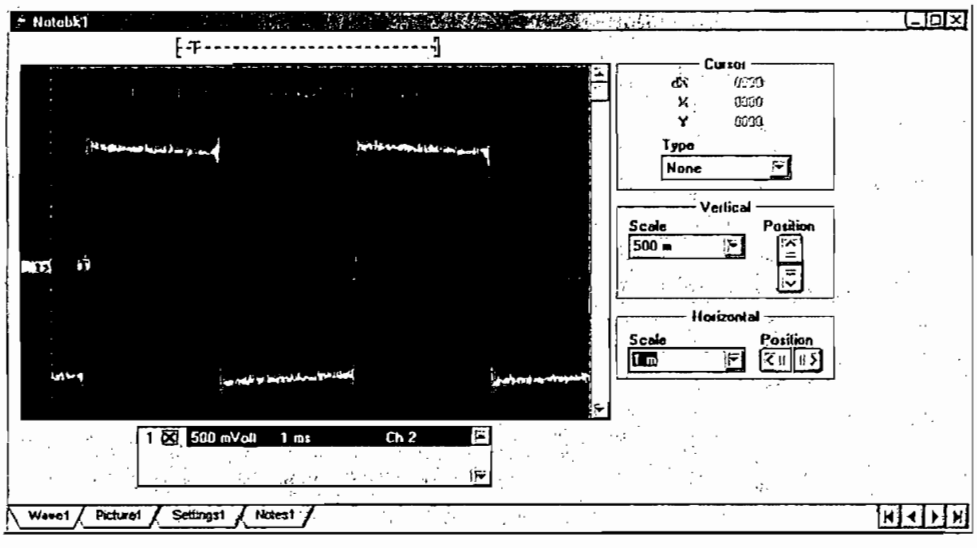


Figura 4.19 Onda Cuadrada (200 Hz, 1V) obtenida desde el osciloscopio

En las siguientes Figuras 4.20 y 4.21 se muestra el efecto que se produce al generar señales cuadradas de baja frecuencia mediante la tarjeta de desarrollo DSP. En la Figura 4.20 se indica la construcción de una forma de onda cuadrada de 10 Hz con una relación de trabajo de 0.49 mediante la interfaz del generador arbitrario, y en la Figura 4.21 se indica la forma de onda correspondiente obtenida desde el osciloscopio, en la cual se observa la diferencia.

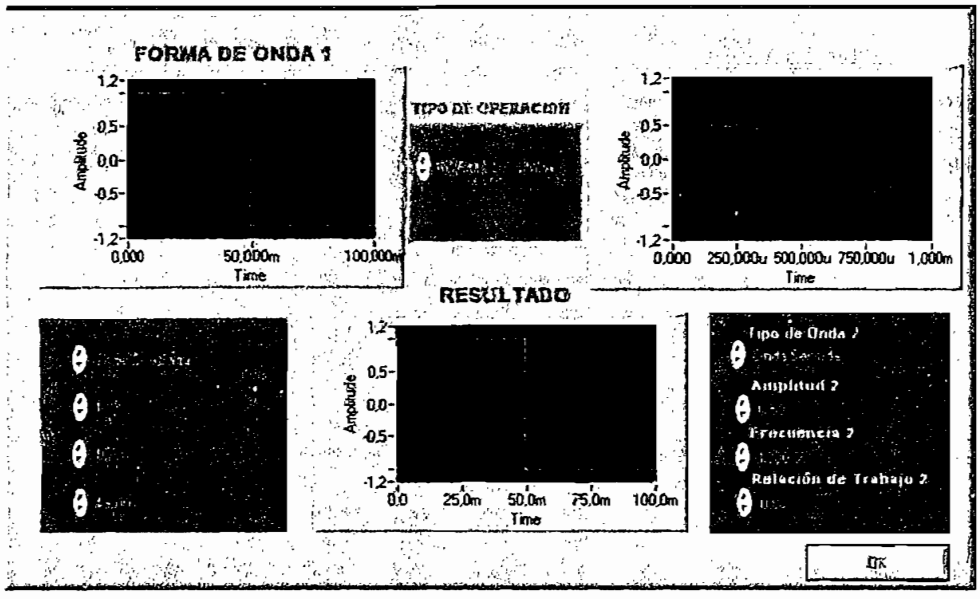


Figura 4.20 Onda Cuadrada (10 Hz, 1 Voltio, relación de trabajo = 0.49)

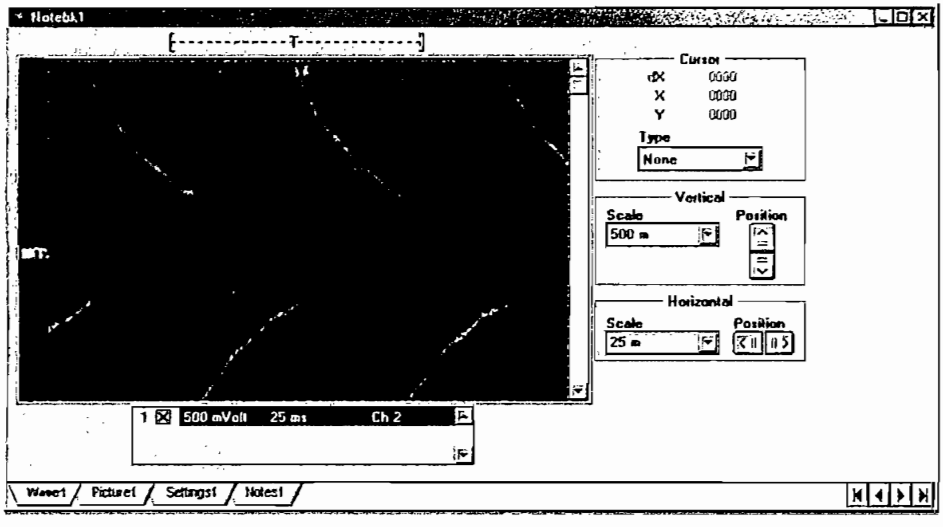


Figura 4.21 Onda Cuadrada (10 Hz, 1V) obtenida desde el osciloscopio

El efecto mostrado en la Figura 4.21 es el decaimiento de la señal en el nivel de 1 Voltio, esto es debido a que en la salida del CODEC, en la tarjeta de desarrollo DSP, se encuentra un circuito de acoplamiento AC que posee filtros con capacitores que no permiten el paso de la componente continua. En el caso de una onda cuadrada de 200 Hz, el nivel de continua decae muy poco debido a que el cambio de nivel positivo a negativo es muy rápido y la amplitud de 1 Voltio no permanece mucho tiempo constante. En el caso de una onda cuadrada con frecuencia de 10 Hz, el nivel positivo de 1 V se mantiene durante un periodo mayor y se empieza a notar el efecto del capacitor, que posee el Filtro en la salida del CODEC. En la Figura 4.22 se muestra el tipo de filtro de salida a continuación del CODEC en la tarjeta de desarrollo DSP.

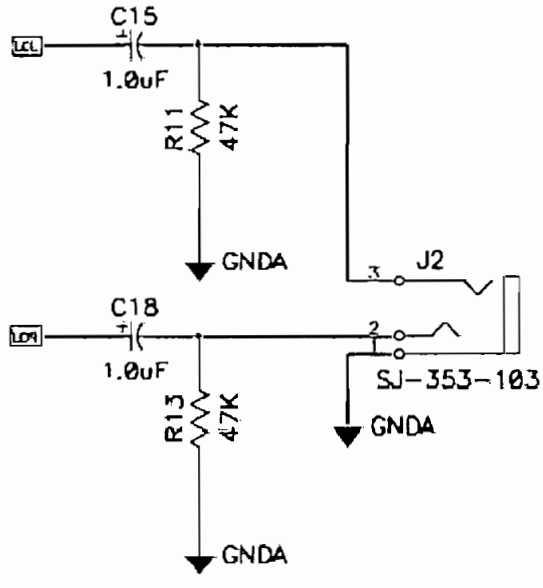


Figura 4.22 Filtros a la salida del Codec AD1847

En el circuito de acoplamiento AC, mostrado en la Figura 4.22 se puede observar el capacitor que no permite el paso de la componente continua. Tanto el canal izquierdo como derecho posee este tipo de circuito, por lo tanto no existirá nivel de continua en los dos canales. Por lo mencionado anteriormente se demuestra que en la tarjeta de desarrollo del DSP no se puede generar señales constantes o DC debido a los filtros de salida que posee.

4.4 PRUEBAS REALIZADAS CON LA OPCIÓN CREAR DESDE FÓRMULA DEL GENERADOR ARBITARIO DE FORMAS DE ONDA

Con la opción Crear desde fórmula se pueden generar formas de ondas ingresando una fórmula deseada; en las pruebas realizadas, se construye una forma de onda senoidal envuelta de una exponencial, tanto creciente como decreciente. Además, se genera una forma de onda con las características anteriores pero con una frecuencia de 2000 Hz, y se puede observar el efecto que produce el construir una forma de onda con un número de muestras muy bajo. En la Figura 4.23 se puede observar la interfaz de la opción Crear desde Fórmula y también al gráfico de la onda senoidal envuelta de una exponencial decreciente con una frecuencia de 10 Hz.

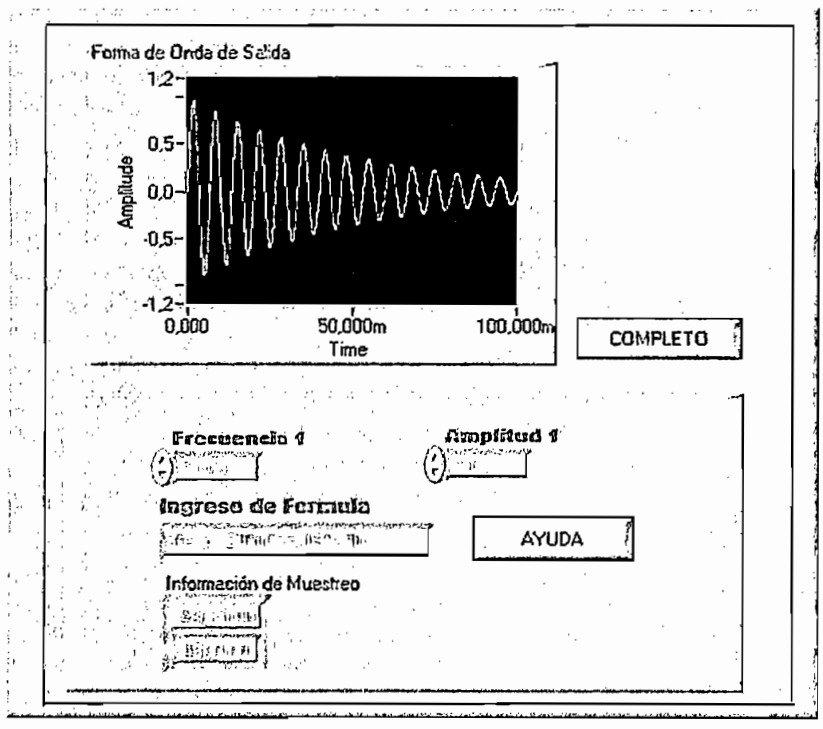


Figura 4.23 Onda Senoidal exponencialmente decreciente (10 Hz, 1 Voltio)

La fórmula introducida a*exp(-2*t*f)*sin(15*w*t) define a una senoidal envuelta de una exponencial decreciente. Se puede observar la letra **a** que representa la amplitud deseada de la forma de onda, **w** es 2xπxf donde **f** es la frecuencia introducida, y **t** es el número de segundos transcurridos. La función exp(-2*t*f) realiza la operación e^{-2xtxf} y la función sin(15*w*t), genera datos de 15 periodos de una forma de onda senoidal con frecuencia de 10 Hz. Al multiplicar ambas funciones con la amplitud deseada, se obtiene la forma de onda mostrada en la interfaz gráfica. Todos los parámetros y funciones disponibles en Labview, se especifican en la pantalla de Ayuda de sintaxis, al presionar *Ayuda* en la pantalla de Crear_desde_Fórmula.VI. En la Figura 4.24 se muestra la forma de onda obtenida con el osciloscopio.

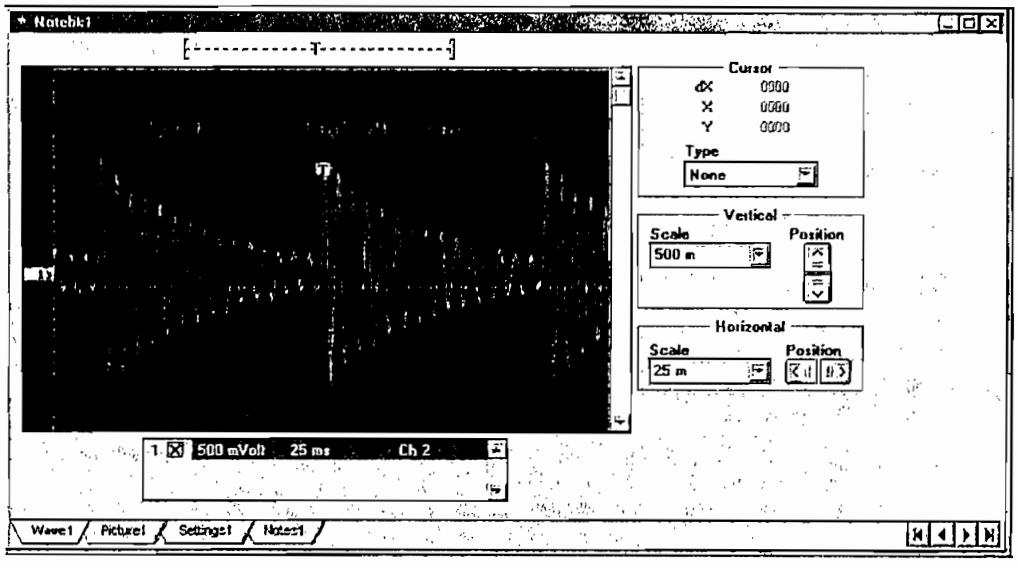


Figura 4.24 Onda Senoidal exponencialmente decreciente obtenida desde osciloscopio

En la siguiente Figura 4.25 se puede observar una señal Senoidal exponencialmente creciente, generada desde la interfaz gráfica del Generador Arbitrario de Forma de Onda; y en la Figura 4.26 se muestra la forma de onda generada eléctricamente y en forma periódica obtenida desde el osciloscopio.

Al aumentar la frecuencia deseada se obtendrá una forma de onda deformada como se muestra en la Figura 4.27, en donde se puede observar una forma de

onda senoidal exponencialmente creciente con una frecuencia de 2000 Hz. Debido a los filtros que posee el CODEC, se obtiene una forma continua de la onda, aunque se utilice pocas muestras de la forma de onda. El número de muestras obtenido para la forma de onda es de 24, tanto el número de muestras como la frecuencia de muestreo se indican en el panel frontal. La forma de onda captada desde el osciloscopio se muestra en la Figura 4.28 .

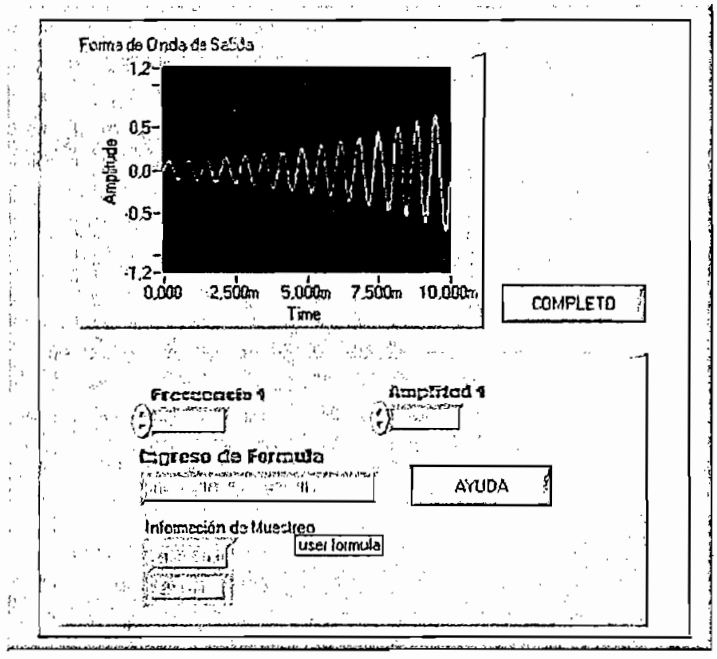


Figura 4.25 Onda Senoidal exponencialmente creciente (100 Hz, 0.7 Voltios)

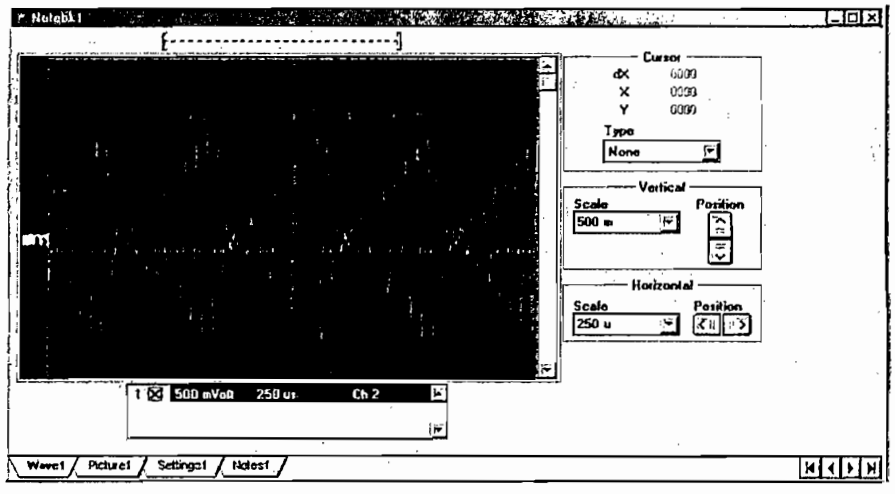


Figura 4.26 Onda Senoidal exponencialmente creciente (100Hz, 0.7V) obtenida desde osciloscopio

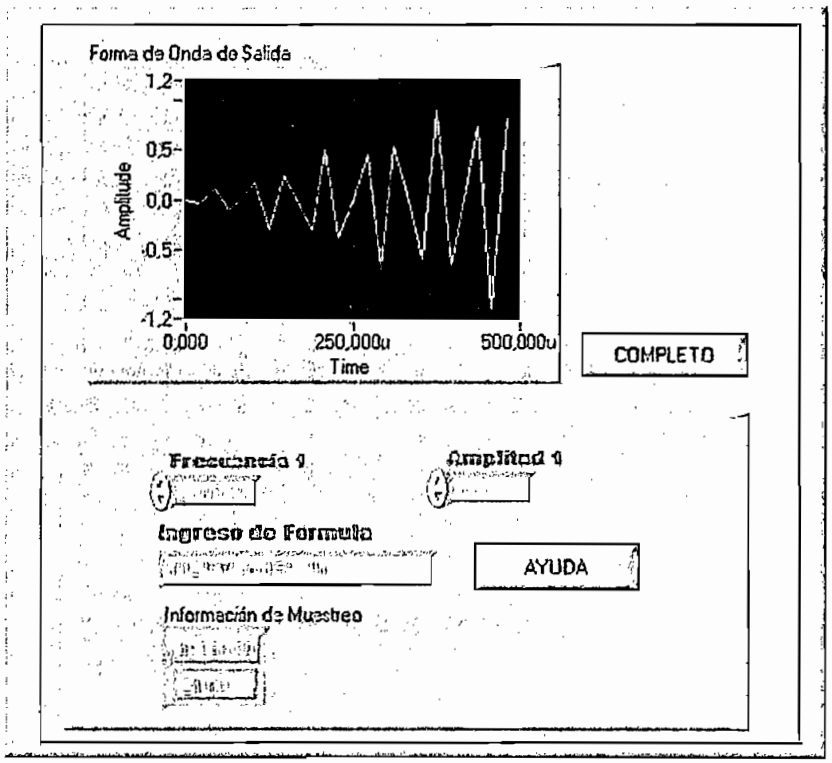


Figura 4.27 Onda Senoidal exponencialmente creciente (2000 Hz, 0.9 Voltios)

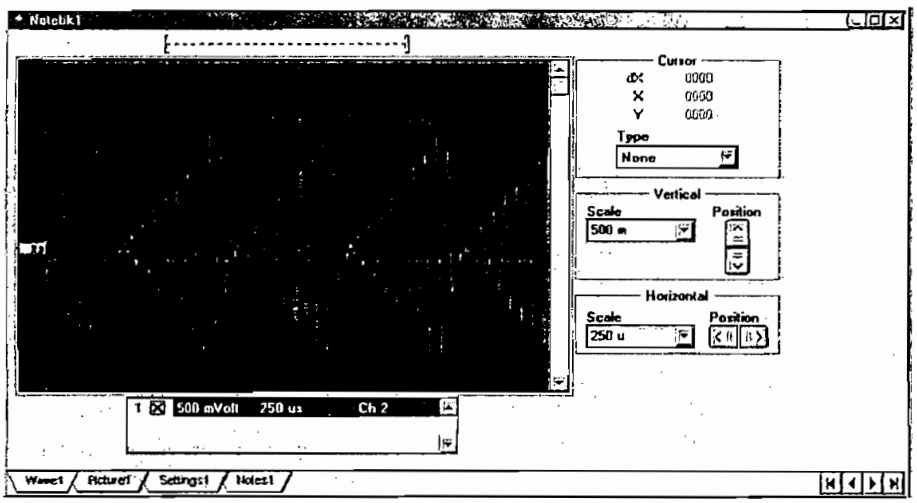


Figura 4.28 Onda Senoidal exponencialmente creciente (2000 Hz, 0.9V) obtenida desde osciloscopio

4.5 SUMARIO

En el presente Capítulo se han presentado los resultados de la generación eléctrica de las distintas formas de onda creadas desde la interfaz de usuario del Generador Arbitrario de Forma de Onda. En los resultados se tiene especial interés en el resultado obtenido al generar formas de ondas cuadradas y demostrar que no se puede generar señales constantes o DC.

Además se logra generar señales básicas con ruido y realizar operaciones con formas de onda de diferente frecuencia.

Finalmente se muestra la generación eléctrica de una forma de onda definida por una fórmula y los efectos que produce al tener bajo número de muestras en la forma de onda a generar.

CAPITULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- Un microprocesador DSP de la familia 21XX maneja datos de 16 bits con formato alternativo de 1 bit entero y 15 bits fraccionarios, esta cualidad permite manejar valores de muestras de formas de onda de gran precisión, y generar formas de onda muy semejantes a las generadas por el programa ejecutándose en Labview, el cual maneja datos con formato IEEE de 64 bits.

- La tarjeta de desarrollo DSP EZ KIT LITE es una herramienta para desarrollar algoritmos de procesamiento digital de señales, por lo que es ideal para la generación eléctrica de formas de onda. Pero debido a la funcionalidad del CODEC (integrado en la tarjeta de desarrollo), el cual fue diseñado para manipular señales de sonido, no se pueden generar señales que posean una componente continua, de un valor constante o señales DC. Esto debido a que en la salida del CODEC, en el hardware de la tarjeta de desarrollo, se tienen circuitos de acoplamiento AC que no permiten pasar ningún tipo de señal DC. Por este motivo con el CODEC integrado en la tarjeta de desarrollo, no se puede generar ningún tipo de señal DC, esto es una desventaja para la implementación de controladores. En futuras aplicaciones se puede reemplazar al CODEC, con un conversor D/A con interfase serial para realizar cualquier tipo de controlador.

- Los VIs de Labview para la generación de Formas de Onda entregan datos del tiempo de inicio de la forma de onda, el incremento de tiempo entre muestras (el cual sería el periodo de muestreo impuesto), y los valores de amplitud de las formas de onda, de acuerdo a la información de muestreo que se ingrese en los VIs de generación; si se quiere realizar una operación básica sobre dos formas de onda de diferente frecuencia, *obligatoriamente* ambas formas de onda deben tener la misma frecuencia de muestreo y el

mismo número de muestras, esto es debido a que Labview al realizar alguna operación sobre ambas formas de onda, toma los valores de amplitud de una forma de onda en orden de tiempo con los valores de amplitud de la otra forma de onda. Si una forma de onda posee diferente periodo de muestreo que otra, se tendrá un valor de amplitud en un diferente instante que el valor de amplitud de la otra forma de onda. Si el número de muestras no es el mismo para ambas formas de onda, se tendrá un número diferente de valores de amplitud, y al realizar una operación, no se tendrán los elementos correspondientes, y se producirá un error.

- Al generar una forma de onda con un periodo que no sea un múltiplo del periodo de muestreo, se producirá un error en la frecuencia deseada. Esto es debido a que el número de muestras no será un número entero sino un número fraccionario, lo que quiere decir que el último valor de amplitud de la forma de onda para que esta sea periódica, se generará entre el último y el primer periodo de muestreo del siguiente periodo de la forma de onda. Como las muestras se generan en cada periodo de muestreo, el último valor generado será el valor del último periodo de muestreo, y este no es el último valor de amplitud de la forma de onda, debido a que el último valor no esta destinado a generarse justo en el instante del periodo de muestreo sino en un instante posterior o anterior, produciéndose un error.

- El momento de realizar la comunicación desde el DSP hacia el CODEC, se definen ciertos registros que serán de uso dedicado para la comunicación. Estos registros definidos (I, M. L) son usados para la característica de Autobuferado, y al momento de definir a estos registros no se los debe utilizar para ninguna otra función dentro del programa del DSP, ya que estos registros se los puede utilizar también como punteros a localidades de memoria en direccionamiento indirecto. Si se utiliza un registro definido para la función de Autobuferado, se provocará un error en la ejecución en el programa del DSP. Este error es muy común debido a que estos registros se definen al inicio del programa, y al desarrollar el programa principal se los puede utilizar casualmente.

- Debido a que el microprocesador posee memoria de datos y memoria de programa RAM, al apagar o dar un reset a la tarjeta de desarrollo se perderá el programa trasmitido desde Labview, por esta razón se debe trasmitir nuevamente, junto con las muestras de los valores de amplitud de la forma de onda, a el programa que realiza la generación eléctrica con el CODEC. También el programa Monitor se perderá, pero este programa se descarga desde la memoria EPROM en cada reset o al energizar a la tarjeta DSP.

5.2 RECOMENDACIONES

- Para realizar frecuencias menores a 10 Hz, en futuros proyectos, se puede conectar a la tarjeta de desarrollo DSP una memoria externa para guardar los valores de amplitud de las muestras de las formas de onda trasmitidas desde Labview, por medio de los conectores de acceso a ciertos pines que posee la tarjeta de desarrollo DSP. Con el uso de memoria externa se puede enviar una cantidad superior a los 16000 muestras, las cuales superan el número de localidades de memoria de datos que posee el DSP.

- Varias aplicaciones en control se pueden realizar con los microprocesadores DSP debido a su velocidad de procesamiento matemático y a su habilidad de manejar datos de 16 bits, entre tantas se puede sugerir: Control de dispositivos mediante comandos por voz, Control Adaptivo, Control y estimación de parámetros en tiempo real de motores de Inducción, etc

- Una de las limitaciones en el manejo de los microprocesadores DSP es su empaque (PQFP) con 128 pines, sería una gran ventaja implementar una materia que explique como realizar y diseñar circuitos impresos que utilicen esta clase de empaque, y no utilizar tarjetas de desarrollo para realizar proyectos específicos.

BIBLIOGRAFÍA

[1] ANALOG DEVICES. **ADSP-2100 Family User's Manual**. Third Edition. Canadá. September 1995.

[2] ANALOG DEVICES. **ADSP-218X DSP Instruction Set Reference**. First Edition. Canadá. February 2001.

[3] ANALOG DEVICES. **ADSP-2100 Family. EZ-KIT Lite Reference Manual**. Canadá. 1995.

[4] ANALOG DEVICES. **ADSP-2181 Data Sheet**. Usa. 1997.

[5] ANALOG DEVICES. **ADSP-2100 Family Development Tools Data Sheet**. Usa. 1995.

[6] ANALOG DEVICES. **Serial-Port 16-Bit SoundPort Stereo Codec AD1847 DataSheet**. Usa. 1997.

[7] ANALOG DEVICES. **ADSP-2181 EZ-KIT Lite Evaluation System Manual**. Second Edition. August 2002.

[8] SMITH Steven W. **The Scientist and Engineer's Guide to Digital Signal Processing Electronic files.** Second Edition. 1999.

[9] NATIONAL INSTRUMENTS. **LABVIEW User Manual**. November 2001.

[10] IFEACHOR, Emmanuel; JERVIS, Barrie. **Digital Signal Processing**. Second Edition. Prentice Hall. England. 2002.

[11] PROAKIS, John G.; MANOLAKIS, Dimitris G., **Tratamiento Digital de Señales**. Tercera Edición. 1998.

[12] VÁSQUEZ YÉPEZ Oscar Fernando. **Implementación de un Laboratorio Básico de Procesamiento Digital de Señales.** Proyecto de Titulación, EPN. Octubre 1997.

ANEXO 1

HOJA DE DATOS DEL ADSP-2181

ANALOG DEVICES

DSP Microcomputer

ADSP-2181

FEATURES

PERFORMANCE

25 ns Instruction Cycle Time from 20 MHz Crystal @ 5.0 Volts
40 MIPS Sustained Performance
Single-Cycle Instruction Execution
Single-Cycle Context Switch
3-Bus Architecture Allows Dual Operand Fetches in Every Instruction Cycle
Multifunction Instructions
Power-Down Mode Featuring Low CMOS Standby Power Dissipation with 100 Cycle Recovery from Power-Down Condition
Low Power Dissipation in Idle Mode

INTEGRATION

ADSP-2100 Family Code Compatible, with Instruction Set Extensions
80K Bytes of On-Chip RAM, Configured as
16K Words On-Chip Program Memory RAM
16K Words On-Chip Data Memory RAM
Dual Purpose Program Memory for Both Instruction and Data Storage
Independent ALU, Multiplier/Accumulator, and Barrel Shifter Computational Units
Two Independent Data Address Generators
Powerful Program Sequencer Provides
Zero Overhead Looping
Conditional Instruction Execution
Programmable 16-Bit Interval Timer with Prescaler
128-Lead TQFP/128-Lead PQFP

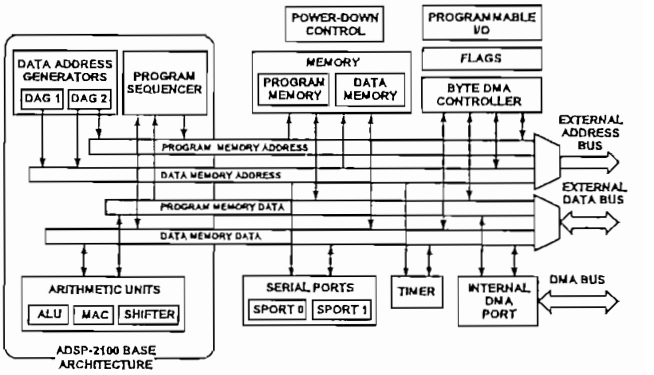
SYSTEM INTERFACE

16-Bit Internal DMA Port for High Speed Access to On-Chip Memory
4 MByte Memory Interface for Storage of Data Tables and Program Overlays
8-Bit DMA to Byte Memory for Transparent Program and Data Memory Transfers
I/O Memory Interface with 2048 Locations Supports Parallel Peripherals
Programmable Memory Strobe and Separate I/O Memory Space Permits "Glueless" System Design
Programmable Wait State Generation
Two Double-Buffered Serial Ports with Companding Hardware and Automatic Data Buffering
Automatic Booting of On-Chip Program Memory from Byte-Wide External Memory, e.g., EPROM, or Through Internal DMA Port
Six External Interrupts
13 Programmable Flag Pins Provide Flexible System Signaling
ICE-Port™* Emulator Interface Supports Debugging in Final Systems

*ICE-Port is a trademark of Analog Devices, Inc.

REV. C

FUNCTIONAL BLOCK DIAGRAM



GENERAL DESCRIPTION

The ADSP-2181 is a single-chip microcomputer optimized for digital signal processing (DSP) and other high speed numeric processing applications.

The ADSP-2181 combines the ADSP-2100 family base architecture (three computational units, data address generators and a program sequencer) with two serial ports, a 16-bit internal DMA port, a byte DMA port, a programmable timer, Flag I/O, extensive interrupt capabilities, and on-chip program and data memory.

The ADSP-2181 integrates 80K bytes of on-chip memory configured as 16K words (24-bit) of program RAM, and 16K words (16-bit) of data RAM. Power-down circuitry is also provided to meet the low power needs of battery operated portable equipment. The ADSP-2181 is available in 128-pin TQFP and 128-pin PQFP packages.

In addition, the ADSP-2181 supports new instructions, which include bit manipulations—bit set, bit clear, bit toggle, bit test—new ALU constants, new multiplication instruction (x squared), biased rounding, result free ALU operations, I/O memory transfers and global interrupt masking for increased flexibility.

Fabricated in a high speed, double metal, low power, CMOS process, the ADSP-2181 operates with a 30 ns instruction cycle time. Every instruction can execute in a single processor cycle.

The ADSP-2181's flexible architecture and comprehensive instruction set allow the processor to perform multiple operations in parallel. In one processor cycle the ADSP-2181 can:

- generate the next program address
- fetch the next instruction
- perform one or two data moves
- update one or two data address pointers
- perform a computational operation

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 617/329-4700 World Wide Web Site: http://www.analog.com
Fax: 617/326-8703

This takes place while the processor continues to:

- receive and transmit data through the two serial ports
- receive and/or transmit data through the internal DMA port
- receive and/or transmit data through the byte DMA port
- decrement timer

Development System

The ADSP-2100 Family Development Software, a complete set of tools for software and hardware system development, supports the ADSP-2181. The System Builder provides a high level method for defining the architecture of systems under development. The Assembler has an algebraic syntax that is easy to program and debug. The Linker combines object files into an executable file. The Simulator provides an interactive instruction-level simulation with a reconfigurable user interface to display different portions of the hardware environment. A PROM Splitter generates PROM programmer compatible files. The C Compiler, based on the Free Software Foundation's GNU C Compiler, generates ADSP-2181 assembly source code. The source code debugger allows programs to be corrected in the C environment. The Runtime Library includes over 100 ANSI-standard mathematical and DSP-specific functions.

The EZ-KIT Lite is a hardware/software kit offering a complete development environment for the entire ADSP-21xx family: an ADSP-2181 evaluation board with PC monitor software plus Assembler, Linker, Simulator, and PROM Splitter software. The ADSP-218x EZ-KIT Lite is a low-cost, easy to use hardware platform on which you can quickly get started with your DSP software design. The EZ-KIT Lite includes the following features:

- 33 MHz ADSP-2181
- Full 16-bit Stereo Audio I/O with AD1847 SoundPort®* Codec
- RS-232 Interface to PC with Windows 3.1 Control Software
- Standalone Operation with Socketed EPROM
- EZ-ICE®* Connector for Emulator Control
- DSP Demo Programs

The ADSP-218x EZ-ICE®* Emulator aids in the hardware debugging of ADSP-218x systems. The emulator consists of hardware, host computer resident software and the target board connector. The ADSP-218x integrates on-chip emulation support with a 14-pin ICE-Port™* interface. This interface provides a simpler target board connection requiring fewer mechanical clearance considerations than other ADSP-2100 Family EZ-ICE®*s. The ADSP-218x device need not be removed from the target system when using the EZ-ICE®*, nor are any adapters needed. Due to the small footprint of the EZ-ICE®* connector, emulation can be supported in final board designs.

The EZ-ICE®* performs a full range of functions, including:

- In-target operation
- Up to 20 breakpoints
- Single-step or full-speed operation
- Registers and memory values can be examined and altered
- PC upload and download functions
- Instruction-level emulation of program booting and execution
- Complete assembly and disassembly of instructions
- C source-level debugging

See the Designing An EZ-ICE®*-Compatible Target System section of this data sheet for exact specifications of the EZ-ICE®* target board connector.

*EZ-ICE and SoundPort are registered trademarks of Analog Devices, Inc.

Additional Information

This data sheet provides a general overview of ADSP-2181 functionality. For additional information on the architecture and instruction set of the processor, refer to the *ADSP-2100 Family User's Manual.* For more information about the development tools, refer to the *ADSP-2100 Family Development Tools Data Sheet.*

ARCHITECTURE OVERVIEW

The ADSP-2181 instruction set provides flexible data moves and multifunction (one or two data moves with a computation) instructions. Every instruction can be executed in a single processor cycle. The ADSP-2181 assembly language uses an algebraic syntax for ease of coding and readability. A comprehensive set of development tools supports program development.

Figure 1 is an overall block diagram of the ADSP-2181. The processor contains three independent computational units: the ALU, the multiplier/accumulator (MAC) and the shifter. The computational units process 16-bit data directly and have provisions to support multiprecision computations. The ALU performs a standard set of arithmetic and logic operations; division primitives are also supported. The MAC performs single-cycle multiply, multiply/add and multiply/subtract operations with 40 bits of accumulation. The shifter performs logical and arithmetic shifts, normalization, denormalization and derive exponent operations. The shifter can be used to efficiently implement numeric format control including multiword and block floating-point representations.

The internal result (R) bus connects the computational units so that the output of any unit may be the input of any unit on the next cycle.

A powerful program sequencer and two dedicated data address generators ensure efficient delivery of operands to these computational units. The sequencer supports conditional jumps, subroutine calls and returns in a single cycle. With internal loop counters and loop stacks, the ADSP-2181 executes looped code with zero overhead; no explicit jump instructions are required to maintain loops.

Two data address generators (DAGs) provide addresses for simultaneous dual operand fetches (from data memory and program memory). Each DAG maintains and updates four address pointers. Whenever the pointer is used to access data (indirect addressing), it is post-modified by the value of one of four possible modify registers. A length value may be associated with each pointer to implement automatic modulo addressing for circular buffers.

Efficient data transfer is achieved with the use of five internal buses:

- Program Memory Address (PMA) Bus
- Program Memory Data (PMD) Bus
- Data Memory Address (DMA) Bus
- Data Memory Data (DMD) Bus
- Result (R) Bus

The two address buses (PMA and DMA) share a single external address bus, allowing memory to be expanded off-chip, and the two data buses (PMD and DMD) share a single external data bus. Byte memory space and I/O memory space also share the external buses.

Program memory can store both instructions and data, permitting the ADSP-2181 to fetch two operands in a single cycle, one from program memory and one from data memory. The ADSP-2181 can fetch an operand from program memory and the next instruction in the same cycle.

In addition to the address and data bus for external memory connection, the ADSP-2181 has a 16-bit Internal DMA port (IDMA port) for connection to external systems. The IDMA port is made up of 16 data/address pins and five control pins. The IDMA port provides transparent, direct access to the DSPs on-chip program and data RAM.

An interface to low cost byte-wide memory is provided by the Byte DMA port (BDMA port). The BDMA port is bidirectional and can directly address up to four megabytes of external RAM or ROM for off-chip storage of program overlays or data tables.

The byte memory and I/O memory space interface supports slow memories and I/O memory-mapped peripherals with programmable wait state generation. External devices can gain control of external buses with bus request/grant signals ($\overline{BR}$, $\overline{BGH}$, and $\overline{BG}$). One execution mode (Go Mode) allows the ADSP-2181 to continue running from on-chip memory. Normal execution mode requires the processor to halt while buses are granted.

The ADSP-2181 can respond to eleven interrupts. There can be up to six external interrupts (one edge-sensitive, two level-sensitive and three configurable) and seven internal interrupts generated by the timer, the serial ports (SPORTs), the Byte DMA port and the power-down circuitry. There is also a master $\overline{RESET}$ signal.

The two serial ports provide a complete synchronous serial interface with optional companding in hardware and a wide variety of framed or frameless data transmit and receive modes of operation. Each port can generate an internal programmable serial clock or accept an external serial clock.

The ADSP-2181 provides up to 13 general-purpose flag pins. The data input and output pins on SPORT1 can be alternatively configured as an input flag and an output flag. In addition, there are eight flags that are programmable as inputs or outputs and three flags that are always outputs.

A programmable interval timer generates periodic interrupts. A 16-bit count register (TCOUNT) is decremented every *n* processor cycles, where *n* is a scaling value stored in an 8-bit register (TSCALE). When the value of the count register reaches zero, an interrupt is generated and the count register is reloaded from a 16-bit period register (TPERIOD).

Serial Ports

The ADSP-2181 incorporates two complete synchronous serial ports (SPORT0 and SPORT1) for serial communications and multiprocessor communication.

Here is a brief list of the capabilities of the ADSP-2181 SPORTs. Refer to the *ADSP-2100 Family User's Manual* for further details.

- SPORTs are bidirectional and have a separate, double-buffered transmit and receive section.
- SPORTs can use an external serial clock or generate their own serial clock internally.
- SPORTs have independent framing for the receive and transmit sections. Sections run in a frameless mode or with frame synchronization signals internally or externally generated. Frame sync signals are active high or inverted, with either of two pulse widths and timings.

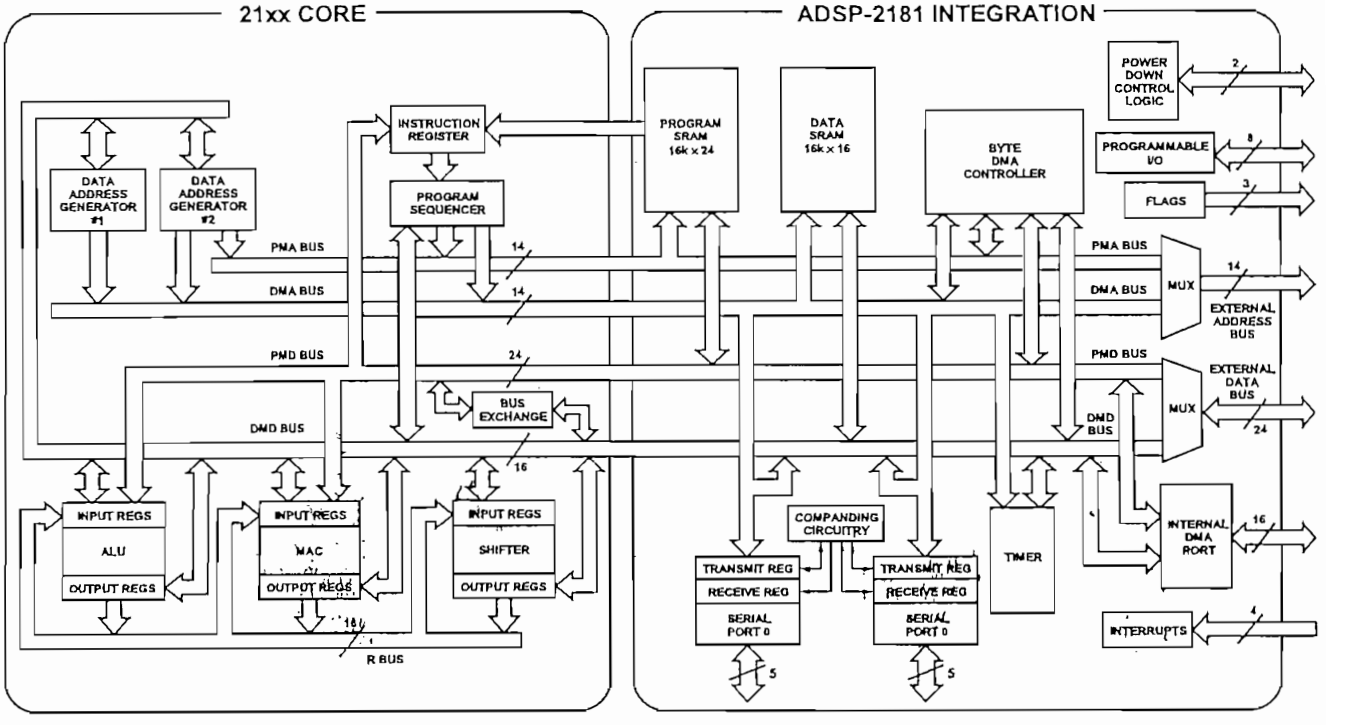


Figure 1. ADSP-2181 Block Diagram

- SPORTs support serial data word lengths from 3 to 16 bits and provide optional A-law and μ-law companding according to CCITT recommendation G.711.
- SPORT receive and transmit sections can generate unique interrupts on completing a data word transfer.
- SPORTs can receive and transmit an entire circular buffer of data with only one overhead cycle per data word. An interrupt is generated after a data buffer transfer.
- SPORT0 has a multichannel interface to selectively receive and transmit a 24 or 32 word, time-division multiplexed, serial bitstream.
- SPORT1 can be configured to have two external interrupts ($\overline{IRQ0}$ and $\overline{IRQ1}$) and the Flag In and Flag Out signals. The internally generated serial clock may still be used in this configuration.

Pin Descriptions

The ADSP-2181 is available in 128-lead TQFP and 128-lead PQFP packages.

PIN FUNCTION DESCRIPTIONS

Pin Name(s)	# of Pins	Input/ Output	Function
Address	14	O	Address Output Pins for Program, Data, Byte, & I/O Spaces
Data	24	I/O	Data I/O Pins for Program and Data Memory Spaces (8 MSBs Are Also Used as Byte Space Addresses)
$\overline{RESET}$	1	I	Processor Reset Input
$\overline{IRQ2}$	1	I	Edge- or Level-Sensitive Interrupt Request
$\overline{IRQL0}$, $\overline{IRQL1}$	2	I	Level-Sensitive Interrupt Requests
$\overline{IRQE}$	1	I	Edge-Sensitive Interrupt Request
$\overline{BR}$	1	I	Bus Request Input
$\overline{BG}$	1	O	Bus Grant Output
$\overline{BGH}$	1	O	Bus Grant Hung Output
$\overline{PMS}$	1	O	Program Memory Select Output
$\overline{DMS}$	1	O	Data Memory Select Output
$\overline{BMS}$	1	O	Byte Memory Select Output
$\overline{IOMS}$	1	O	I/O Space Memory Select Output
$\overline{CMS}$	1	O	Combined Memory Select Output
$\overline{RD}$	1	O	Memory Read Enable Output
$\overline{WR}$	1	O	Memory Write Enable Output
MMAP	1	I	Memory Map Select Input
BMODE	1	I	Boot Option Control Input
CLKIN, XTAL	2	I	Clock or Quartz Crystal Input
CLKOUT	1	O	Processor Clock Output
SPORT0	5	I/O	Serial Port I/O Pins
SPORT1	5	I/O	Serial Port 1 *or* Two External $\overline{IRQs}$, Flag In and Flag Out
$\overline{IRD}$, $\overline{IWR}$	2	I	IDMA Port Read/Write Inputs
$\overline{IS}$	1	I	IDMA Port Select
IAL	1	I	IDMA Port Address Latch Enable
IAD	16	I/O	IDMA Port Address/Data Bus
$\overline{IACK}$	1	O	IDMA Port Access Ready Acknowledge
$\overline{PWD}$	1	I	Powerdown Control
PWDACK	1	O	Powerdown Control
FL0, FL1, FL2	3	O	Output Flags
PF7:0	8	I/O	Programmable I/O Pins
EE	1	*	(Emulator Only*)
$\overline{EBR}$	1	*	(Emulator Only*)
$\overline{EBG}$	1	*	(Emulator Only*)
$\overline{ERESET}$	1	*	(Emulator Only*)
$\overline{EMS}$	1	*	(Emulator Only*)
$\overline{EINT}$	1	*	(Emulator Only*)
ECLK	1	*	(Emulator Only*)
ELIN	1	*	(Emulator Only*)
ELOUT	1	*	(Emulator Only*)
GND	11	–	Ground Pins
VDD	6	–	Power Supply Pins

*These ADSP-2181 pins must be connected *only* to the EZ-ICE® connector in the target system. These pins have no function except during emulation, and do not require pullup or pulldown resistors.

Interrupts

The interrupt controller allows the processor to respond to the eleven possible interrupts and reset with minimum overhead. The ADSP-2181 provides four dedicated external interrupt input pins, $\overline{IRQ2}$, $\overline{IRQL0}$, $\overline{IRQL1}$ and $\overline{IRQE}$. In addition, SPORT1 may be reconfigured for $\overline{IRQ0}$, $\overline{IRQ1}$, FLAG_IN and FLAG_OUT, for a total of six external interrupts. The ADSP-2181 also supports internal interrupts from the timer, the byte DMA port, the two serial ports, software and the power-down control circuit. The interrupt levels are internally prioritized and individually maskable (except power down and reset). The $\overline{IRQ2}$, $\overline{IRQ0}$ and $\overline{IRQ1}$ input pins can be programmed to be either level- or edge-sensitive. $\overline{IRQL0}$ and $\overline{IRQL1}$ are level-sensitive and $\overline{IRQE}$ is edge sensitive. The priorities and vector addresses of all interrupts are shown in Table I.

Table I. Interrupt Priority & Interrupt Vector Addresses

Source of Interrupt	Interrupt Vector Address (Hex)
Reset (or Power-Up with PUCR = 1)	0000 (*Highest Priority*)
Power-Down (Nonmaskable)	002C
$\overline{IRQ2}$	0004
$\overline{IRQL1}$	0008
$\overline{IRQL0}$	000C
SPORT0 Transmit	0010
SPORT0 Receive	0014
$\overline{IRQE}$	0018
BDMA Interrupt	001C
SPORT1 Transmit or $\overline{IRQ1}$	0020
SPORT1 Receive or $\overline{IRQ0}$	0024
Timer	0028 (*Lowest Priority*)

Interrupt routines can either be nested with higher priority interrupts taking precedence or processed sequentially. Interrupts can be masked or unmasked with the IMASK register. Individual interrupt requests are logically ANDed with the bits in IMASK; the highest priority unmasked interrupt is then selected. The power-down interrupt is nonmaskable.

The ADSP-2181 masks all interrupts for one instruction cycle following the execution of an instruction that modifies the IMASK register. This does not affect serial port autobuffering or DMA transfers.

The interrupt control register, ICNTL, controls interrupt nesting and defines the $\overline{IRQ0}$, $\overline{IRQ1}$ and $\overline{IRQ2}$ external interrupts to be either edge- or level-sensitive. The $\overline{IRQE}$ pin is an external edge-sensitive interrupt and can be forced and cleared. The $\overline{IRQL0}$ and $\overline{IRQL1}$ pins are external level-sensitive interrupts.

The IFC register is a write-only register used to force and clear interrupts.

On-chip stacks preserve the processor status and are automatically maintained during interrupt handling. The stacks are twelve levels deep to allow interrupt, loop and subroutine nesting.

The following instructions allow global enable or disable servicing of the interrupts (including power down), regardless of the state of IMASK. Disabling the interrupts does not affect serial port autobuffering or DMA.

ENA INTS;
DIS INTS;

When the processor is reset, interrupt servicing is enabled.

LOW POWER OPERATION

The ADSP-2181 has three low power modes that significantly reduce the power dissipation when the device operates under standby conditions. These modes are:

- Power-Down
- Idle
- Slow Idle

The CLKOUT pin may also be disabled to reduce external power dissipation.

Power-Down

The ADSP-2181 processor has a low power feature that lets the processor enter a very low power dormant state through hardware or software control. Here is a brief list of power-down features. For detailed information about the power-down feature, refer to the *ADSP-2100 Family User's Manual*, "System Interface" chapter.

- Quick recovery from power-down. The processor begins executing instructions in as few as 100 CLKIN cycles.
- Support for an externally generated TTL or CMOS processor clock. The external clock can continue running during power-down without affecting the lowest power rating and 100 CLKIN cycle recovery.
- Support for crystal operation includes disabling the oscillator to save power (the processor automatically waits 4096 CLKIN cycles for the crystal oscillator to start and stabilize), and letting the oscillator run to allow 100 CLKIN cycle start up.
- Power-down is initiated by either the power-down pin ($\overline{PWD}$) or the software power-down force bit.
- Interrupt support allows an unlimited number of instructions to be executed before optionally powering down. The power-down interrupt also can be used as a nonmaskable, edge-sensitive interrupt.
- Context clear/save control allows the processor to continue where it left off or start with a clean context when leaving the power-down state.
- The $\overline{RESET}$ pin also can be used to terminate power-down.
- Power-down acknowledge pin indicates when the processor has entered power-down.

Idle

When the ADSP-2181 is in the Idle Mode, the processor waits indefinitely in a low power state until an interrupt occurs. When an unmasked interrupt occurs, it is serviced; execution then continues with the instruction following the *IDLE* instruction.

Slow Idle

The *IDLE* instruction is enhanced on the ADSP-2181 to let the processor's internal clock signal be slowed, further reducing power consumption. The reduced clock frequency, a programmable fraction of the normal clock rate, is specified by a selectable divisor given in the *IDLE* instruction. The format of the instruction is

IDLE (n);

where n = 16, 32, 64 or 128. This instruction keeps the processor fully functional, but operating at the slower clock rate. While it is in this state, the processor's other internal clock signals, such as SCLK, CLKOUT and timer clock, are reduced by the same ratio. The default form of the instruction, when no clock divisor is given, is the standard *IDLE* instruction.

When the *IDLE (n)* instruction is used, it effectively slows down the processor's internal clock and thus its response time to incoming interrupts. The one-cycle response time of the standard idle state is increased by *n*, the clock divisor. When an enabled interrupt is received, the ADSP-2181 will remain in the idle state for up to a maximum of *n* processor cycles (n = 16, 32, 64 or 128) before resuming normal operation.

When the *IDLE (n)* instruction is used in systems that have an externally generated serial clock (SCLK), the serial clock rate may be faster than the processor's reduced internal clock rate. Under these conditions, interrupts must not be generated at a faster rate than can be serviced, due to the additional time the processor takes to come out of the idle state (a maximum of *n* processor cycles).

SYSTEM INTERFACE

Figure 2 shows a typical basic system configuration with the ADSP-2181, two serial devices, a byte-wide EPROM, and optional external program and data overlay memories. Programmable wait state generation allows the processor to connect easily to slow peripheral devices. The ADSP-2181 also provides four external interrupts and two serial ports or six external interrupts and one serial port.

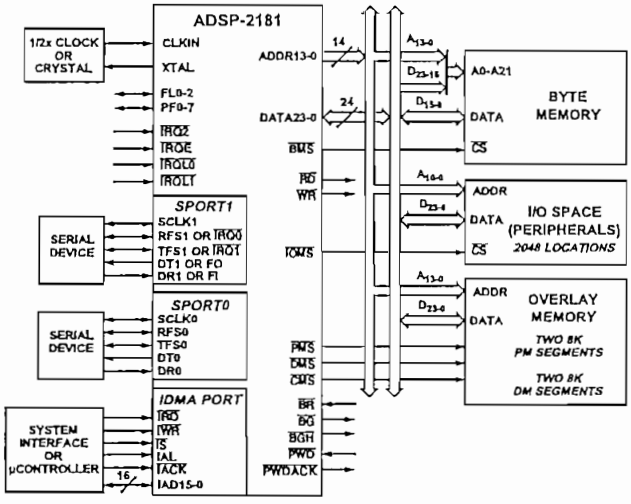


Figure 2. ADSP-2181 Basic System Configuration

Clock Signals

The ADSP-2181 can be clocked by either a crystal or a TTL-compatible clock signal.

The CLKIN input cannot be halted, changed during operation or operated below the specified frequency during normal operation. The only exception is while the processor is in the power-down state. For additional information, refer to Chapter 9, *ADSP-2100 Family User's Manual*, for detailed information on this power-down feature.

If an external clock is used, it should be a TTL-compatible signal running at half the instruction rate. The signal is connected to the processor's CLKIN input. When an external clock is used, the XTAL input *must* be left unconnected.

The ADSP-2181 uses an input clock with a frequency equal to half the instruction rate; a 20.00 MHz input clock yields a 25 ns processor cycle (which is equivalent to 40 MHz). Normally, instructions are executed in a single processor cycle. All device timing is relative to the internal instruction clock rate, which is indicated by the CLKOUT signal when enabled.

Because the ADSP-2181 includes an on-chip oscillator circuit, an external crystal may be used. The crystal should be connected across the CLKIN and XTAL pins, with two capacitors connected as shown in Figure 3. Capacitor values are dependent on crystal type and should be specified by the crystal manufacturer. A parallel-resonant, fundamental frequency, microprocessor-grade crystal should be used.

A clock output (CLKOUT) signal is generated by the processor at the processor's cycle rate. This can be enabled and disabled by the CLKODIS bit in the SPORT0 Autobuffer Control Register.

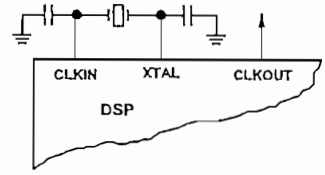


Figure 3. External Crystal Connections

Reset

The $\overline{\text{RESET}}$ signal initiates a master reset of the ADSP-2181. The $\overline{\text{RESET}}$ signal must be asserted during the power-up sequence to assure proper initialization. $\overline{\text{RESET}}$ during initial power-up must be held long enough to allow the internal clock to stabilize. If $\overline{\text{RESET}}$ is activated any time after power-up, the clock continues to run and does not require stabilization time.

The power-up sequence is defined as the total time required for the crystal oscillator circuit to stabilize after a valid V_{DD} is applied to the processor, and for the internal phase-locked loop (PLL) to lock onto the specific crystal frequency. A minimum of 2000 CLKIN cycles ensures that the PLL has locked, but does not include the crystal oscillator start-up time. During this power-up sequence the $\overline{\text{RESET}}$ signal should be held low. On any subsequent resets, the $\overline{\text{RESET}}$ signal must meet the minimum pulse width specification, t_{RSP}.

The $\overline{\text{RESET}}$ input contains some hysteresis; however, if you use an RC circuit to generate your $\overline{\text{RESET}}$ signal, the use of an external Schmidt trigger is recommended.

The master reset sets all internal stack pointers to the empty stack condition, masks all interrupts and clears the MSTAT register. When $\overline{\text{RESET}}$ is released, if there is no pending bus request and the chip is configured for booting (MMAP = 0), the boot-loading sequence is performed. The first instruction is fetched from on-chip program memory location 0x0000 once boot loading completes.

Memory Architecture

The ADSP-2181 provides a variety of memory and peripheral interface options. The key functional groups are Program Memory, Data Memory, Byte Memory and I/O.

Program Memory is a 24-bit-wide space for storing both instruction opcodes and data. The ADSP-2181 has 16K words of Program Memory RAM on chip and the capability of accessing up to two 8K external memory overlay spaces using the external data bus. Both an instruction opcode and a data value can be read from on-chip program memory in a single cycle.

Data Memory is a 16-bit-wide space used for the storage of data variables and for memory-mapped control registers. The ADSP-2181 has 16K words on Data Memory RAM on chip, consisting of 16,352 user-accessible locations and 32 memory-mapped registers. Support also exists for up to two 8K external memory overlay spaces through the external data bus.

Byte Memory provides access to an 8-bit wide memory space through the Byte DMA (BDMA) port. The Byte Memory interface provides access to 4 MBytes of memory by utilizing eight data lines as additional address lines. This gives the BDMA Port an effective 22-bit address range. On power-up, the DSP can automatically load bootstrap code from byte memory.

I/O Space allows access to 2048 locations of 16-bit-wide data. It is intended to be used to communicate with parallel peripheral devices such as data converters and external registers or latches.

Program Memory

The ADSP-2181 contains a 16K × 24 on-chip program RAM. The on-chip program memory is designed to allow up to two accesses each cycle so that all operations can complete in a single cycle. In addition, the ADSP-2181 allows the use of 8K external memory overlays.

The program memory space organization is controlled by the MMAP pin and the PMOVLAY register. Normally, the ADSP-2181 is configured with MMAP = 0 and program memory organized as shown in Figure 4.

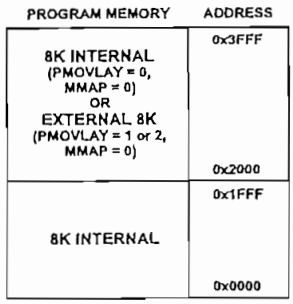


Figure 4. Program Memory (MMAP = 0)

There are 16K words of memory accessible internally when the PMOVLAY register is set to 0. When PMOVLAY is set to something other than 0, external accesses occur at addresses 0x2000 through 0x3FFF. The external address is generated as shown in Table II.

Table II.

PMOVLAY	Memory	A13	A12:0
0	Internal	Not Applicable	Not Applicable
1	External Overlay 1	0	13 LSBs of Address Between 0x2000 and 0x3FFF
2	External Overlay 2	1	13 LSBs of Address Between 0x2000 and 0x3FFF

This organization provides for two external 8K overlay segments using only the normal 14 address bits. This allows for simple program overlays using one of the two external segments in place of the on-chip memory. Care must be taken in using this overlay space in that the processor core (i.e., the sequencer) does not take into account the PMOVLAY register value. For example, if a loop operation was occurring on one of the external overlays and the program changes to another external overlay or internal memory, an incorrect loop operation could occur. In addition, care must be taken in interrupt service routines as the overlay registers are not automatically saved and restored on the processor mode stack.

For ADSP-2100 Family compatibility, MMAP = 1 is allowed. In this mode, booting is disabled and overlay memory is disabled (*PMOVLAY* must be 0). Figure 5 shows the memory map in this configuration.

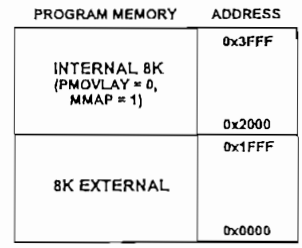


Figure 5. Program Memory (MMAP = 1)

Data Memory

The ADSP-2181 has 16,352 16-bit words of internal data memory. In addition, the ADSP-2181 allows the use of 8K external memory overlays. Figure 6 shows the organization of the data memory.

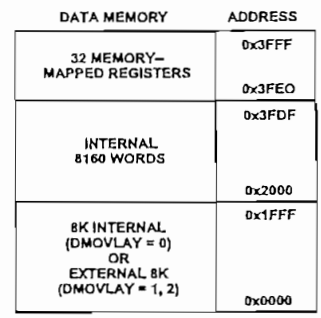


Figure 6. Data Memory

There are 16,352 words of memory accessible internally when the DMOVLAY register is set to 0. When DMOVLAY is set to something other than 0, external accesses occur at addresses 0x0000 through 0x1FFF. The external address is generated as shown in Table III.

Table III.

DMOVLAY	Memory	A13	A12:0
0	Internal	Not Applicable	Not Applicable
1	External Overlay 1	0	13 LSBs of Address Between 0x0000 and 0x1FFF
2	External Overlay 2	1	13 LSBs of Address Between 0x0000 and 0x1FFF

This organization allows for two external 8K overlays using only the normal 14 address bits.

All internal accesses complete in one cycle. Accesses to external memory are timed using the wait states specified by the DWAIT register.

I/O Space

The ADSP-2181 supports an additional external memory space called I/O space. This space is designed to support simple connections to peripherals or to bus interface ASIC data registers. I/O space supports 2048 locations. The lower eleven bits of the external address bus are used; the upper three bits are undefined. Two instructions were added to the core ADSP-2100 Family instruction set to read from and write to I/O memory space. The I/O space also has four dedicated 3-bit wait state registers, IOWAIT0-3, which specify up to seven wait states to be automatically generated for each of four regions. The wait states act on address ranges as shown in Table IV.

Table IV.

Address Range	Wait State Register
0x000-0x1FF	IOWAIT0
0x200-0x3FF	IOWAIT1
0x400-0x5FF	IOWAIT2
0x600-0x7FF	IOWAIT3

Composite Memory Select ($\overline{CMS}$)

The ADSP-2181 has a programmable memory select signal that is useful for generating memory select signals for memories mapped to more than one space. The $\overline{CMS}$ signal is generated to have the same timing as each of the individual memory select signals ($\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$, $\overline{IOMS}$) but can combine their functionality.

When set, each bit in the CMSSEL register, causes the $\overline{CMS}$ signal to be asserted when the selected memory select is asserted. For example, to use a 32K word memory to act as both program and data memory, set the PMS and DMS bits in the CMSSEL register and use the $\overline{CMS}$ pin to drive the chip select of the memory; use either $\overline{DMS}$ or $\overline{PMS}$ as the additional address bit.

The $\overline{CMS}$ pin functions like the other memory select signals, with the same timing and bus request logic. A 1 in the enable bit causes the assertion of the $\overline{CMS}$ signal at the same time as the selected memory select signal. All enable bits, except the $\overline{BMS}$ bit, default to 1 at reset.

Byte Memory

The byte memory space is a bidirectional, 8-bit-wide, external memory space used to store programs and data. Byte memory is accessed using the BDMA feature. The byte memory space consists of 256 pages, each of which is 16K × 8.

The byte memory space on the ADSP-2181 supports read and write operations as well as four different data formats. The byte memory uses data bits 15:8 for data. The byte memory uses data bits 23:16 and address bits 13:0 to create a 22-bit address. This allows up to a 4 meg × 8 (32 megabit) ROM or RAM to be used without glue logic. All byte memory accesses are timed by the BMWAIT register.

Byte Memory DMA (BDMA)

The Byte memory DMA controller allows loading and storing of program instructions and data using the byte memory space. The BDMA circuit is able to access the byte memory space while the processor is operating normally, and steals only one DSP cycle per 8-, 16- or 24-bit word transferred.

The BDMA circuit supports four different data formats which are selected by the BTYPE register field. The appropriate number of 8-bit accesses are done from the byte memory space to build the word size selected. Table V shows the data formats supported by the BDMA circuit.

Table V.

BTYPE	Internal Memory Space	Word Size	Alignment
00	Program Memory	24	Full Word
01	Data Memory	16	Full Word
10	Data Memory	8	MSBs
11	Data Memory	8	LSBs

Unused bits in the 8-bit data memory formats are filled with 0s. The BIAD register field is used to specify the starting address for the on-chip memory involved with the transfer. The 14-bit BEAD register specifies the starting address for the external byte memory space. The 8-bit BMPAGE register specifies the starting page for the external byte memory space. The BDIR register field selects the direction of the transfer. Finally the 14-bit BWCOUNT register specifies the number of DSP words to transfer and initiates the BDMA circuit transfers.

BDMA accesses can cross page boundaries during sequential addressing. A BDMA interrupt is generated on the completion of the number of transfers specified by the BWCOUNT register. The BWCOUNT register is updated after each transfer so it can be used to check the status of the transfers. When it reaches zero, the transfers have finished and a BDMA interrupt is generated. The BMPAGE and BEAD registers must not be accessed by the DSP during BDMA operations.

The source or destination of a BDMA transfer will always be on-chip program or data memory, regardless of the values of MMAP, PMOVLAY or DMOVLAY.

When the BWCOUNT register is written with a nonzero value, the BDMA circuit starts executing byte memory accesses with wait states set by BMWAIT. These accesses continue until the count reaches zero. When enough accesses have occurred to create a destination word, it is transferred to or from on-chip memory. The transfer takes one DSP cycle. DSP accesses to external memory have priority over BDMA byte memory accesses.

The BDMA Context Reset bit (BCR) controls whether the processor is held off while the BDMA accesses are occurring. Setting the BCR bit to 0 allows the processor to continue operations. Setting the BCR bit to 1 causes the processor to stop execution while the BDMA accesses are occurring, to clear the context of the processor and start execution at address 0 when the BDMA accesses have completed.

Internal Memory DMA Port (IDMA Port)

The IDMA Port provides an efficient means of communication between a host system and the ADSP-2181. The port is used to access the on-chip program memory and data memory of the DSP with only one DSP cycle per word overhead. The IDMA port cannot, however, be used to write to the DSP's memory-mapped control registers.

The IDMA port has a 16-bit multiplexed address and data bus and supports 24-bit program memory. The IDMA port is completely asynchronous and can be written to while the ADSP-2181 is operating at full speed.

The DSP memory address is latched and then automatically incremented after each IDMA transaction. An external device can therefore access a block of sequentially addressed memory by specifying only the starting address of the block. This increases throughput as the address does not have to be sent for each memory access.

IDMA Port access occurs in two phases. The first is the IDMA Address Latch cycle. When the acknowledge is asserted, a 14-bit address and 1-bit destination type can be driven onto the bus by an external device. The address specifies an on-chip memory location; the destination type specifies whether it is a DM or PM access. The falling edge of the address latch signal latches this value into the IDMAA register.

Once the address is stored, data can either be read from or written to the ADSP-2181's on-chip memory. Asserting the select line ($\overline{IS}$) and the appropriate read or write line ($\overline{IRD}$ and $\overline{IWR}$ respectively) signals the ADSP-2181 that a particular transaction is required. In either case, there is a one-processor-cycle delay for synchronization. The memory access consumes one additional processor cycle.

Once an access has occurred, the latched address is automatically incremented and another access can occur.

Through the IDMAA register, the DSP can also specify the starting address and data format for DMA operation.

Bootstrap Loading (Booting)

The ADSP-2181 has two mechanisms to allow automatic loading of the on-chip program memory after reset. The method for booting after reset is controlled by the MMAP and BMODE pins as shown in Table VI.

BDMA Booting

When the BMODE and MMAP pins specify BDMA booting (MMAP = 0, BMODE = 0), the ADSP-2181 initiates a BDMA boot sequence when reset is released. The BDMA interface is set up during reset to the following defaults when BDMA booting is specified: the BDIR, BMPAGE, BIAD and BEAD registers are set to 0, the BTYPE register is set to 0 to specify program memory 24 bit words, and the BWCOUNT register is set to 32. This causes 32 words of on-chip program memory to be loaded from byte memory. These 32 words are used to set up the BDMA to load in the remaining program code. The BCR bit is also set to 1, which causes program execution to be held off until all 32 words are loaded into on-chip program memory. Execution then begins at address 0.

Table VI. Boot Summary Table

MMAP	BMODE	Booting Method
0	0	BDMA feature is used in default mode to load the first 32 program memory words from the byte memory space. Program execution is held off until all 32 words have been loaded.
0	1	IDMA feature is used to load any internal memory as desired. Program execution is held off until internal program memory location 0 is written to.
1	X	Bootstrap features disabled. Program execution immediately starts from location 0.

The ADSP-2100 Family Development Software (Revision 5.02 and later) fully supports the BDMA booting feature and can generate byte memory space compatible boot code.

The IDLE instruction can also be used to allow the processor to hold off execution while booting continues through the BDMA interface.

IDMA Port Booting

The ADSP-2181 can also boot programs through its Internal DMA port. If BMODE = 1 and MMAP = 0, the ADSP-2181 boots from the IDMA port. IDMA feature can load as much on-chip memory as desired. Program execution is held off until on-chip program memory location 0 is written to.

The ADSP-2100 Family Development Software (Revision 5.02 and later) can generate IDMA compatible boot code.

Bus Request & Bus Grant

The ADSP-2181 can relinquish control of the data and address buses to an external device. When the external device requires access to memory, it asserts the bus request ($\overline{BR}$) signal. If the ADSP-2181 is not performing an external memory access, then it responds to the active $\overline{BR}$ input in the following processor cycle by:

- three-stating the data and address buses and the $\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$, $\overline{CMS}$, $\overline{IOMS}$, $\overline{RD}$, $\overline{WR}$ output drivers,
- asserting the bus grant ($\overline{BG}$) signal, and
- halting program execution.

If Go Mode is enabled, the ADSP-2181 will not halt program execution until it encounters an instruction that requires an external memory access.

If the ADSP-2181 is performing an external memory access when the external device asserts the $\overline{BR}$ signal, then it will not three-state the memory interfaces or assert the $\overline{BG}$ signal until the processor cycle after the access completes. The instruction does not need to be completed when the bus is granted. If a single instruction requires two external memory accesses, the bus will be granted between the two accesses.

When the $\overline{BR}$ signal is released, the processor releases the $\overline{BG}$ signal, reenables the output drivers and continues program execution from the point where it stopped.

The bus request feature operates at all times, including when the processor is booting and when $\overline{RESET}$ is active.

The $\overline{BGH}$ pin is asserted when the ADSP-2181 is ready to execute an instruction, but is stopped because the external bus is already granted to another device. The other device can release the bus by deasserting bus request. Once the bus is released, the ADSP-2181 deasserts $\overline{BG}$ and $\overline{BGH}$ and executes the external memory access.

Flag I/O Pins

The ADSP-2181 has eight general purpose programmable input/output flag pins. They are controlled by two memory mapped registers. The PFTYPE register determines the direction, 1 = output and 0 = input. The PFDATA register is used to read and write the values on the pins. Data being read from a pin configured as an input is synchronized to the ADSP-2181's clock. Bits that are programmed as outputs will read the value being output. The PF pins default to input during reset.

In addition to the programmable flags, the ADSP-2181 has five fixed-mode flags, FLAG_IN, FLAG_OUT, FL0, FL1 and FL2. FL0-FL2 are dedicated output flags. FLAG_IN and FLAG_OUT are available as an alternate configuration of SPORT1.

BIASED ROUNDING

A mode is available on the ADSP-2181 to allow biased rounding in addition to the normal unbiased rounding. When the BIASRND bit is set to 0, the normal unbiased rounding operations occur. When the BIASRND bit is set to 1, biased rounding occurs instead of the normal unbiased rounding. When operating in biased rounding mode all rounding operations with MR0 set to 0x8000 will round up, rather than only rounding up odd MR1 values. For example:

Table VII.

MR Value Before RND	Biased RND Result	Unbiased RND Result
00-0000-8000	00-0001-8000	00-0000-8000
00-0001-8000	00-0002-8000	00-0002-8000
00-0000-8001	00-0001-8001	00-0001-8001
00-0001-8001	00-0002-8001	00-0002-8001
00-0000-7FFF	00-0000-7FFF	00-0000-7FFF
00-0001-7FFF	00-0001-7FFF	00-0001-7FFF

This mode only has an effect when the MR0 register contains 0x8000; all other rounding operations work normally. This mode allows more efficient implementation of bit-specified algorithms that use biased rounding, for example the GSM speech compression routines. Unbiased rounding is preferred for most algorithms.

Note: BIASRND bit is bit 12 of the SPORT0 Autobuffer Control register.

INSTRUCTION SET DESCRIPTION

The ADSP-2181 assembly language instruction set has an algebraic syntax that was designed for ease of coding and readability. The assembly language, which takes full advantage of the processor's unique architecture, offers the following benefits:

- The algebraic syntax eliminates the need to remember cryptic assembler mnemonics. For example, a typical arithmetic add instruction, such as AR = AX0 + AY0, resembles a simple equation.
- Every instruction assembles into a single, 24-bit word that can execute in a single instruction cycle.
- The syntax is a superset ADSP-2100 Family assembly language and is completely source and object code compatible with other family members. Programs may need to be relocated to utilize on-chip memory and conform to the ADSP-2181's interrupt vector and reset vector map.
- Sixteen condition codes are available. For conditional jump, call, return or arithmetic instructions, the condition can be checked and the operation executed in the same instruction cycle.
- Multifunction instructions allow parallel execution of an arithmetic instruction with up to two fetches or one write to processor memory space during a single instruction cycle.

I/O Space Instructions

The instructions used to access the ADSP-2181's I/O memory space are as follows:

Syntax: IO(*addr*) = *dreg*
dreg = IO(*addr*);

where *addr* is an address value between 0 and 2047 and *dreg* is any of the 16 data registers.

Examples: IO(23) = AR0;
AR1 = IO(17);

Description: The I/O space read and write instructions move data between the data registers and the I/O memory space.

DESIGNING AN EZ-ICE®*-COMPATIBLE SYSTEM

The ADSP-2181 has on-chip emulation support and an ICE-Port™*, a special set of pins that interface to the EZ-ICE®*. These features allow in-circuit emulation without replacing the target system processor by using only a 14-pin connection from the target system to the EZ-ICE®*. Target systems must have a 14-pin connector to accept the EZ-ICE®*'s in-circuit probe, a 14-pin plug.

The ICE-Port™* interface consists of the following ADSP-2181 pins:

$\overline{EBR}$
$\overline{EBG}$
$\overline{ERESET}$
$\overline{EMS}$
$\overline{EINT}$
ECLK
ELIN
ELOUT
EE

These ADSP-2181 pins must be connected *only* to the EZ-ICE®* connector in the target system. These pins have no function except during emulation, and do not require pull-up or pull-down resistors. The traces for these signals between the ADSP-2181 and the connector must be kept as short as possible, no longer than three inches.

The following pins are also used by the EZ-ICE®*:

$\overline{BR}$
$\overline{BG}$
$\overline{RESET}$
GND

The EZ-ICE®* uses the EE (emulator enable) signal to take control of the ADSP-2181 in the target system. This causes the processor to use its $\overline{ERESET}$, $\overline{EBR}$ and $\overline{EBG}$ pins instead of the $\overline{RESET}$, $\overline{BR}$ and $\overline{BG}$ pins. The $\overline{BG}$ output is three-stated. These signals do not need to be jumper-isolated in your system.

The EZ-ICE®* connects to your target system via a ribbon cable and a 14-pin female plug. The ribbon cable is 10 inches in length with one end fixed to the EZ-ICE®*. The female plug is plugged onto the 14-pin connector (a pin strip header) on the target board.

Target Board Connector for EZ-ICE®* Probe

The EZ-ICE®* connector (a standard pin strip header) is shown in Figure 7. You must add this connector to your target board design if you intend to use the EZ-ICE®*. Be sure to allow enough room in your system to fit the EZ-ICE®* probe onto the 14-pin connector.

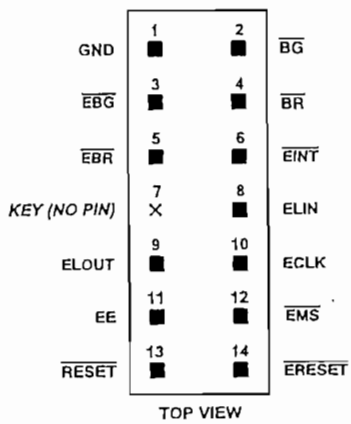


*Figure 7. Target Board Connector for EZ-ICE®**

The 14-pin, 2-row pin strip header is keyed at the Pin 7 location—you must remove Pin 7 from the header. The pins must be 0.025 inch square and at least 0.20 inch in length. Pin spacing should be 0.1 x 0.1 inches. The pin strip header must have at least 0.15 inch clearance on all sides to accept the EZ-ICE®* probe plug. Pin strip headers are available from vendors such as 3M, McKenzie and Samtec.

Target Memory Interface

For your target system to be compatible with the EZ-ICE®* emulator, it must comply with the memory interface guidelines listed below.

PM, DM, BM, IOM and CM

Design your Program Memory (PM), Data Memory (DM), Byte Memory (BM), I/O Memory (IOM) and Composite Memory (CM) external interfaces to comply with worst case device timing requirements and switching characteristics as specified in the DSP's data sheet. The performance of the EZ-ICE®* may approach published worst case specification for some memory access timing requirements and switching characteristics.

Note: If your target does not meet the worst case chip specification for memory access parameters, you may not be able to emulate your circuitry at the desired CLKIN frequency. Depending on the severity of the specification violation, you may have trouble manufacturing your system as DSP components statistically vary in switching characteristic and timing requirements within published limits.

Restriction: All memory strobe signals on the ADSP-2181 ($\overline{RD}$, $\overline{WR}$, $\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$, $\overline{CMS}$ and $\overline{IOMS}$) used in your target system must have 10 kΩ pull-up resistors connected when the EZ-ICE®* is being used. The pull-up resistors are necessary because there are no internal pull-ups to guarantee their state during prolonged three-state conditions resulting from typical EZ-ICE®* debugging sessions. These resistors may be removed at your option when the EZ-ICE®* is not being used.

Target System Interface Signals

When the EZ-ICE®* board is installed, the performance on some system signals changes. Design your system to be compatible with the following system interface signal changes introduced by the EZ-ICE®* board:

- EZ-ICE®* emulation introduces an 8 ns propagation delay between your target circuitry and the DSP on the $\overline{RESET}$ signal.
- EZ-ICE®* emulation introduces an 8 ns propagation delay between your target circuitry and the DSP on the $\overline{BR}$ signal.
- EZ-ICE®* emulation ignores $\overline{RESET}$ and $\overline{BR}$ when single-stepping.
- EZ-ICE®*emulation ignores $\overline{RESET}$ and $\overline{BR}$ when in Emulator Space (DSP halted).
- EZ-ICE®* emulation ignores the state of target $\overline{BR}$ in certain modes. As a result, the target system may take control of the DSP's external memory bus *only* if bus grant ($\overline{BG}$) is asserted by the EZ-ICE®* board's DSP.

Target Architecture File

The EZ-ICE®* software lets you load your program in its linked (executable) form. The EZ-ICE®* PC program can not load sections of your executable located in boot pages (by the linker). With the exception of boot page 0 (loaded into PM RAM), all sections of your executable mapped into boot pages are not loaded.

Write your target architecture file to indicate that only PM RAM is available for program storage, when using the EZ-ICE®* software's loading feature. Data can be loaded to PM RAM or DM RAM.

ADSP-2181–SPECIFICATIONS

RECOMMENDED OPERATING CONDITIONS

Parameter		K Grade Min	K Grade Max	B Grade Min	B Grade Max	Unit
V_{DD}	Supply Voltage	4.5	5.5	4.5	5.5	V
T_{AMB}	Ambient Operating Temperature	0	+70	–40	+85	°C

ELECTRICAL CHARACTERISTICS

Parameter		Test Conditions	K/B Grades Min	Typ	Max	Unit
V_{IH}	Hi-Level Input Voltage[1, 2]	@ V_{DD} = max	2.0			V
V_{IH}	Hi-Level CLKIN Voltage	@ V_{DD} = max	2.2			V
V_{IL}	Lo-Level Input Voltage[1, 3]	@ V_{DD} = min			0.8	V
V_{OH}	Hi-Level Output Voltage[1, 4, 5]	@ V_{DD} = min I_{OH} = –0.5 mA	2.4			V
		@ V_{DD} = min I_{OH} = –100 µA[6]	V_{DD} – 0.3			V
V_{OL}	Lo-Level Output Voltage[1, 4, 5]	@ V_{DD} = min I_{OL} = 2 mA			0.4	V
I_{IH}	Hi-Level Input Current[3]	@ V_{DD} = max V_{IN} = V_{DD}max			10	µA
I_{IL}	Lo-Level Input Current[3]	@ V_{DD} = max V_{IN} = 0 V			10	µA
I_{OZH}	Three-State Leakage Current[7]	@ V_{DD} = max V_{IN} = V_{DD}max[8]			10	µA
I_{OZL}	Three-State Leakage Current[7]	@ V_{DD} = max V_{IN} = 0 V[8]			10	µA
I_{DD}	Supply Current (Idle)[9]	@ V_{DD} = 5.0				
		t_{CK} = 34.7 ns[11]		10		mA
		t_{CK} = 30 ns[11]		12		mA
		t_{CK} = 25 ns[11]		13		mA
I_{DD}	Supply Current (Dynamic)[10]	@ V_{DD} = 5.0 T_{AMB} = +25°C				
		t_{CK} = 34.7 ns[11]		52		mA
		t_{CK} = 30 ns[11]		60		mA
		t_{CK} = 25 ns[11]		70		mA
C_I	Input Pin Capacitance[3, 6, 12]	@ V_{IN} = 2.5 V, f_{IN} = 1.0 MHz, T_{AMB} = +25°C			8	pF
C_O	Output Pin Capacitance[6, 7, 12, 13]	@ V_{IN} = 2.5 V, f_{IN} = 1.0 MHz, T_{AMB} = +25°C			8	pF

NOTES

[1] Bidirectional pins: D0-D23, RFS0, RFS1, SCLK0, SCLK1, TFS0, TFS1, A1-A13, PF0-PF7.

[2] Input only pins: $\overline{RESET}$, $\overline{BR}$, DR0, DR1, $\overline{PWD}$.

[3] Input only pins: CLKIN, $\overline{RESET}$, $\overline{BR}$, DR0, DR1, $\overline{PWD}$.

[4] Output pins: $\overline{BG}$, $\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$, $\overline{IOMS}$, $\overline{CMS}$, $\overline{RD}$, $\overline{WR}$, PWDACK, A0, DT0, DT1, CLKOUT, FL2-0, $\overline{BGH}$.

[5] Although specified for TTL outputs, all ADSP-2186 outputs are CMOS-compatible and will drive to V_{DD} and GND, assuming no dc loads.

[6] Guaranteed but not tested.

[7] Three-statable pins: A0-A13, D0-D23, $\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$, $\overline{IOMS}$, $\overline{CMS}$, $\overline{RD}$, $\overline{WR}$, DT0, DT1, SCLK0, SCLK1, TFS0, TFS1, RFS0, RSF1, PF0–PF7.

[8] 0 V on $\overline{BR}$, CLKIN Inactive.

[9] Idle refers to ADSP-2186 state of operation during execution of IDLE instruction. Deasserted pins are driven to either V_{DD} or GND.

[10] I_{DD} measurement taken with all instructions executing from internal memory. 50% of the instructions are multifunction (types 1, 4, 5, 12, 13, 14), 30% are type 2 and type 6, and 20% are idle instructions.

[11] V_{IN} = 0 V and 3 V. For typical figures for supply currents, refer to Power Dissipation section.

[12] Applies to TQFP package type.

[13] Output pin capacitance is the capacitive load for any three-stated output pin.

Specifications subject to change without notice.

ABSOLUTE MAXIMUM RATINGS*

Supply Voltage ..... -0.3 V to +7 V
Input Voltage ..... -0.3 V to V_{DD} + 0.3 V
Output Voltage Swing ..... -0.3 V to V_{DD} + 0.3 V
Operating Temperature Range (Ambient) .... -40°C to +85°C
Storage Temperature Range ..... -65°C to +150°C
Lead Temperature (5 sec) TQFP ..... +280°C
Lead Temperature (5 sec) PQFP ..... +280°C

*Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only; functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ESD SENSITIVITY

The ADSP-2181 is an ESD (electrostatic discharge) sensitive device. Electrostatic charges readily accumulate on the human body and equipment and can discharge without detection. Permanent damage may occur to devices subjected to high energy electrostatic discharges.

The ADSP-2181 features proprietary ESD protection circuitry to dissipate high energy discharges (Human Body Model). Per method 3015 of MIL-STD-883, the ADSP-2181 has been classified as a Class 1 device.

Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Unused devices must be stored in conductive foam or shunts, and the foam should be discharged to the destination before devices are removed.

TIMING PARAMETERS

GENERAL NOTES

Use the exact timing information given. Do not attempt to derive parameters from the addition or subtraction of others. While addition or subtraction would yield meaningful results for an individual device, the values given in this data sheet reflect statistical variations and worst cases. Consequently, you cannot meaningfully add up parameters to derive longer times.

TIMING NOTES

Switching Characteristics specify how the processor changes its signals. You have no control over this timing—circuitry external to the processor must be designed for compatibility with these signal characteristics. Switching characteristics tell you what the processor will do in a given circumstance. You can also use switching characteristics to ensure that any timing requirement of a device connected to the processor (such as memory) is satisfied.

Timing Requirements apply to signals that are controlled by circuitry external to the processor, such as the data input for a read operation. Timing requirements guarantee that the processor operates correctly with other devices.

MEMORY TIMING SPECIFICATIONS

The table below shows common memory device specifications and the corresponding ADSP-2181 timing parameters, for your convenience.

Memory Device Specification	ADSP-2181 Timing Parameter	Timing Parameter Definition
Address Setup to Write Start	t_{ASW}	A0–A13, $\overline{xMS}$ Setup before $\overline{WR}$ Low
Address Setup to Write End	t_{AW}	A0–A13, $\overline{xMS}$ Setup before $\overline{WR}$ Deasserted
Address Hold Time	t_{WRA}	A0–A13, $\overline{xMS}$ Hold after $\overline{WR}$ Deasserted
Data Setup Time	t_{DW}	Data Setup before $\overline{WR}$ High
Data Hold Time	t_{DH}	Data Hold after $\overline{WR}$ High
$\overline{OE}$ to Data Valid	t_{RDD}	$\overline{RD}$ Low to Data Valid
Address Access Time	t_{AA}	A0–A13, $\overline{xMS}$ to Data Valid

$\overline{xMS}$ = $\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$, $\overline{CMS}$, $\overline{IOMS}$.

FREQUENCY DEPENDENCY FOR TIMING SPECIFICATIONS

t_{CK} is defined as $0.5t_{CKI}$. The ADSP-2181 uses an input clock with a frequency equal to half the instruction rate: a 16.67 MHz input clock (which is equivalent to 60 ns) yields a 30 ns processor cycle (equivalent to 33 MHz). t_{CK} values within the range of $0.5t_{CKI}$ period should be substituted for all relevant timing parameters to obtain the specification value.

Example: $t_{CKH} = 0.5t_{CK} - 7$ ns = 0.5 (25 ns) - 7 ns = 8 ns

ENVIRONMENTAL CONDITIONS

Ambient Temperature Rating:

$T_{AMB} = T_{CASE} - (PD \times \theta_{CA})$
T_{CASE} = Case Temperature in °C
PD = Power Dissipation in W
θ_{CA} = Thermal Resistance (Case-to-Ambient)
θ_{JA} = Thermal Resistance (Junction-to-Ambient)
θ_{JC} = Thermal Resistance (Junction-to-Case)

Package	θ_{JA}	θ_{JC}	θ_{CA}
TQFP	50°C/W	2°C/W	48°C/W
PQFP	41°C/W	10°C/W	31°C/W

POWER DISSIPATION

To determine total power dissipation in a specific application, the following equation should be applied for each output:

$$C \times V_{DD}^2 \times f$$

C = load capacitance, f = output switching frequency.

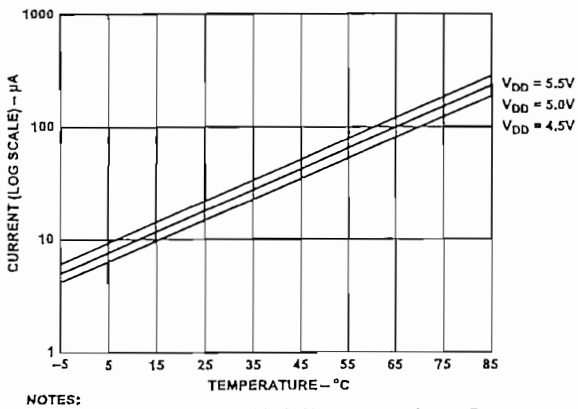
Example:
In an application where external data memory is used and no other outputs are active, power dissipation is calculated as follows:

Assumptions:

- External data memory is accessed every cycle with 50% of the address pins switching.
- External data memory writes occur every other cycle with 50% of the data pins switching.
- Each address and data pin has a 10 pF total load at the pin.
- The application operates at V_{DD} = 5.0 V and t_{CK} = 30 ns.

$$\textit{Total Power Dissipation} = P_{INT} + (C \times V_{DD}^2 \times f)$$

P_{INT} = internal power dissipation from Power vs. Frequency graph (Figure 20).



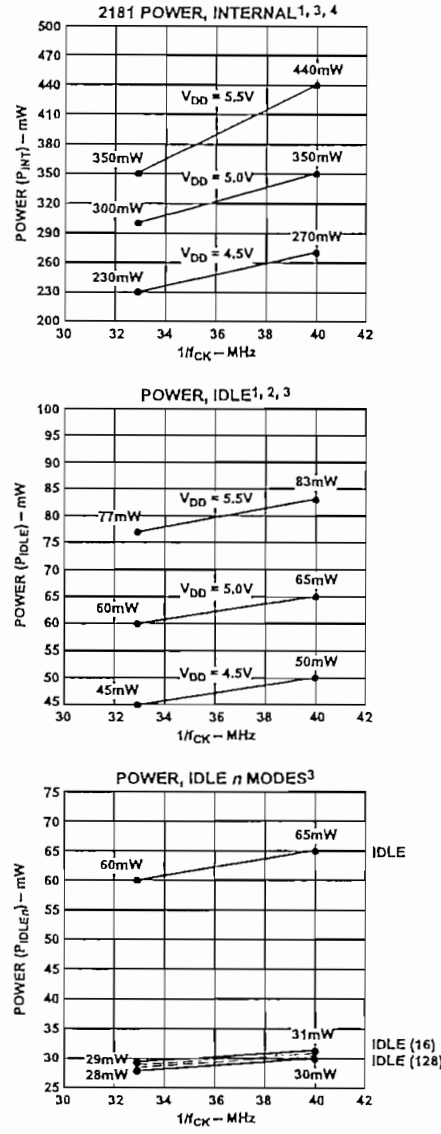
NOTES:
1. REFLECTS ADSP-2181 OPERATION IN LOWEST POWER MODE. (SEE "SYSTEM INTERFACE" CHAPTER OF THE ADSP-2100 FAMILY USER'S MANUAL FOR DETAILS.)
2. CURRENT REFLECTS DEVICE OPERATING WITH NO OUTPUT LOADS.

Figure 8. Power-Down Supply Current (Typical)

$(C \times V_{DD}^2 \times f)$ is calculated for each output:

	# of Pins	× C	× V_{DD}^2	× f		
Address, $\overline{DMS}$	8	× 10 pF	× 5^2 V	× 33.3 MHz	=	66.6 mW
Data Output, $\overline{WR}$	9	× 10 pF	× 5^2 V	× 16.67 MHz	=	37.5 mW
$\overline{RD}$	1	× 10 pF	× 5^2 V	× 16.67 MHz	=	4.2 mW
CLKOUT	1	× 10 pF	× 5^2 V	× 33.3 MHz	=	8.3 mW
						116.6 mW

Total power dissipation for this example is P_{INT} + 116.6 mW.



VALID FOR ALL TEMPERATURE GRADES.
[1]POWER REFLECTS DEVICE OPERATING WITH NO OUTPUT LOADS.
[2]IDLE REFERS TO ADSP-2181 STATE OF OPERATION DURING EXECUTION OF IDLE INSTRUCTION. DEASSERTED PINS ARE DRIVEN TO EITHER V_{DD} OR GND.
[3]TYPICAL POWER DISSIPATION AT 5.0V V_{DD} AND 25°C EXCEPT WHERE SPECIFIED.
[4]I_{DD} MEASUREMENT TAKEN WITH ALL INSTRUCTIONS EXECUTING FROM INTERNAL MEMORY. 50% OF THE INSTRUCTIONS ARE MULTIFUNCTION (TYPES 1, 4, 5, 12, 13, 14), 30% ARE TYPE 2 AND TYPE 6 AND 20% ARE IDLE INSTRUCTIONS.

Figure 9. Power vs. Frequency

CAPACITIVE LOADING

Figures 10 and 11 show the capacitive loading characteristics of the ADSP-2181.

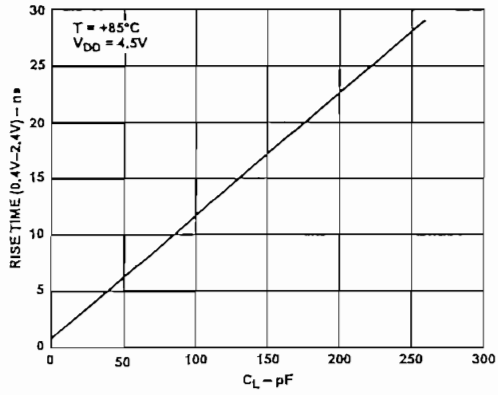


Figure 10. Typical Output Rise Time vs. Load Capacitance, C_L (at Maximum Ambient Operating Temperature)

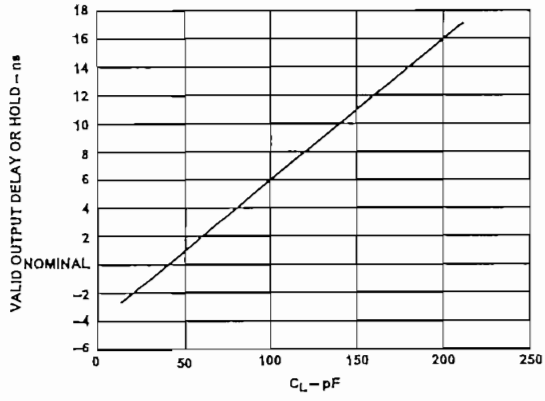


Figure 11. Typical Output Valid Delay or Hold vs. Load Capacitance, C_L (at Maximum Ambient Operating Temperature)

TEST CONDITIONS

Output Disable Time

Output pins are considered to be disabled when they have stopped driving and started a transition from the measured output high or low voltage to a high impedance state. The output disable time (t_{DIS}) is the difference of $t_{MEASURED}$ and t_{DECAY}, as shown in the Output Enable/Disable diagram. The time is the interval from when a reference signal reaches a high or low voltage level to when the output voltages have changed by 0.5 V from the measured output high or low voltage. The decay time, t_{DECAY}, is dependent on the capacitive load, C_L, and the current load, i_L, on the output pin. It can be approximated by the following equation:

$$t_{DECAY} = \frac{C_L \cdot 0.5\,V}{i_L}$$

from which

$$t_{DIS} = t_{MEASURED} - t_{DECAY}$$

is calculated. If multiple pins (such as the data bus) are disabled, the measurement value is that of the last pin to stop driving.

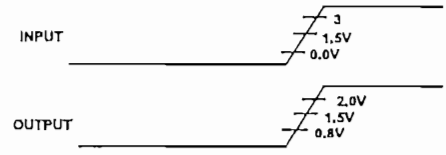


Figure 12. Voltage Reference Levels for AC Measurements (Except Output Enable/Disable)

Output Enable Time

Output pins are considered to be enabled when they have made a transition from a high-impedance state to when they start driving. The output enable time (t_{ENA}) is the interval from when a reference signal reaches a high or low voltage level to when the output has reached a specified high or low trip point, as shown in the Output Enable/Disable diagram. If multiple pins (such as the data bus) are enabled, the measurement value is that of the first pin to start driving.

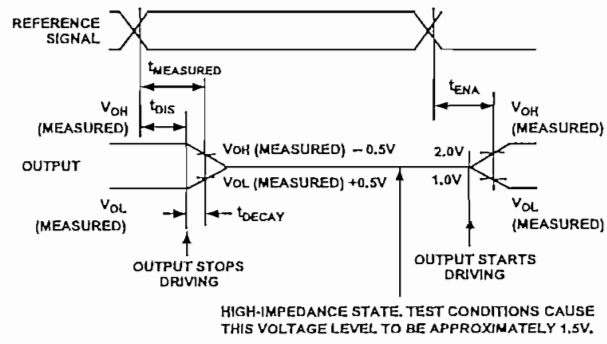


Figure 13. Output Enable/Disable

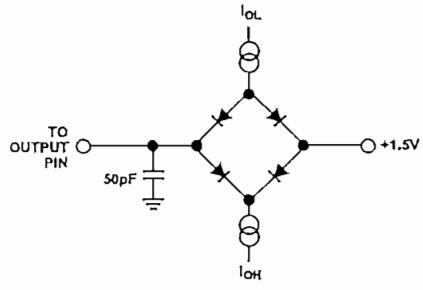


Figure 14. Equivalent Device Loading for AC Measurements (Including All Fixtures)

Parameter		Min	Max	Unit
Clock Signals and Reset				
Timing Requirements:				
t_{CKI}	CLKIN Period	60	150	ns
t_{CKIL}	CLKIN Width Low	20		ns
t_{CKIH}	CLKIN Width High	20		ns
Switching Characteristics:				
t_{CKL}	CLKOUT Width Low	$0.5t_{CK} - 7$		ns
t_{CKH}	CLKOUT Width High	$0.5t_{CK} - 7$		ns
t_{CKOH}	CLKIN High to CLKOUT High	0	20	ns
Control Signals				
Timing Requirement:				
t_{RSP}	$\overline{\text{RESET}}$ Width Low	$5t_{CK}$[1]		ns

NOTE
[1]Applies after power-up sequence is complete. Internal phase lock loop requires no more than 2000 CLKIN cycles assuming stable CLKIN (not including crystal oscillator start-up time).

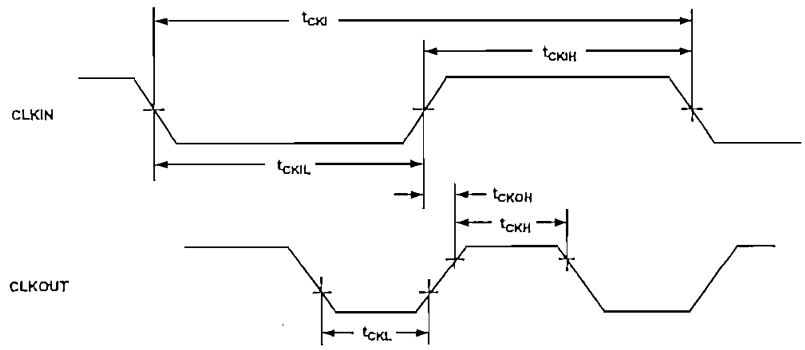


Figure 15. Clock Signals

Parameter		Min	Max	Unit
Interrupts and Flag				
Timing Requirements:				
t_{IFS}	$\overline{IRQx}$, FI, or PFx Setup before CLKOUT Low [1,2,3,4]	$0.25t_{CK} + 15$		ns
t_{IFH}	$\overline{IRQx}$, FI, or PFx Hold after CLKOUT High [1,2,3,4]	$0.25t_{CK}$		ns
Switching Characteristics:				
t_{FOH}	Flag Output Hold after CLKOUT Low[5]	$0.5t_{CK} - 7$		ns
t_{FOD}	Flag Output Delay from CLKOUT Low[5]		$0.15t_{CK} + 5$	ns

NOTES
[1]If $\overline{IRQx}$ and FI inputs meet t_{IFS} and t_{IFH} setup/hold requirements, they will be recognized during the current clock cycle; otherwise the signals will be recognized on the following cycle. (Refer to "Interrupt Controller Operation" in the Program Control chapter of the User's Manual for further information on interrupt servicing.)
[2]Edge-sensitive interrupts require pulse widths greater than 10 ns; level-sensitive interrupts must be held low until serviced.
[3]$\overline{IRQx}$ = $\overline{IRQ0}$, $\overline{IRQ1}$, $\overline{IRQ2}$, $\overline{IRQL0}$, $\overline{IRQL1}$, $\overline{IRQE}$.
[4]PFx = PF0, PF1, PF2, PF3, PF4, PF5, PF6, PF7.
[5]Flag outputs = PFx, FL0, FL1, FL2, Flag_out4.

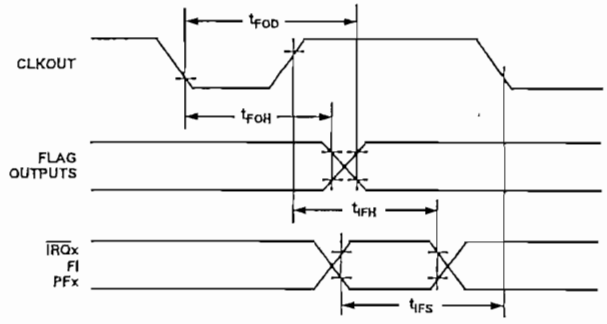


Figure 16. Interrupts and Flags

Parameter		Min	Max	Unit
Bus Request/Grant				
Timing Requirements:				
t_{BH}	$\overline{BR}$ Hold after CLKOUT High[1]	$0.25t_{CK} + 2$		ns
t_{BS}	$\overline{BR}$ Setup before CLKOUT Low[1]	$0.25t_{CK} + 17$		ns
Switching Characteristics:				
t_{SD}	CLKOUT High to $\overline{xMS}$, $\overline{RD}$, $\overline{WR}$ Disable		$0.25t_{CK} + 10$	ns
t_{SDB}	$\overline{xMS}$, $\overline{RD}$, $\overline{WR}$ Disable to $\overline{BG}$ Low	0		ns
t_{SE}	$\overline{BG}$ High to $\overline{xMS}$, $\overline{RD}$, $\overline{WR}$ Enable	0		ns
t_{SEC}	$\overline{xMS}$, $\overline{RD}$, $\overline{WR}$ Enable to CLKOUT High	$0.25t_{CK} - 7$		ns
t_{SDBH}	$\overline{xMS}$, $\overline{RD}$, $\overline{WR}$ Disable to $\overline{BGH}$ Low[2]	0		ns
t_{SEH}	$\overline{BGH}$ High to $\overline{xMS}$, $\overline{RD}$, $\overline{WR}$ Enable[2]	0		ns

NOTES
$\overline{xMS}$ = $\overline{PMS}$, $\overline{DMS}$, $\overline{CMS}$, $\overline{IOMS}$, $\overline{BMS}$.
[1]$\overline{BR}$ is an asynchronous signal. If $\overline{BR}$ meets the setup/hold requirements, it will be recognized during the current clock cycle; otherwise the signal will be recognized on the following cycle. Refer to the *ADSP-2100 Family User's Manual* for $\overline{BR}$/$\overline{BG}$ cycle relationships.
[2]$\overline{BGH}$ is asserted when the bus is granted and the processor requires control of the bus to continue.

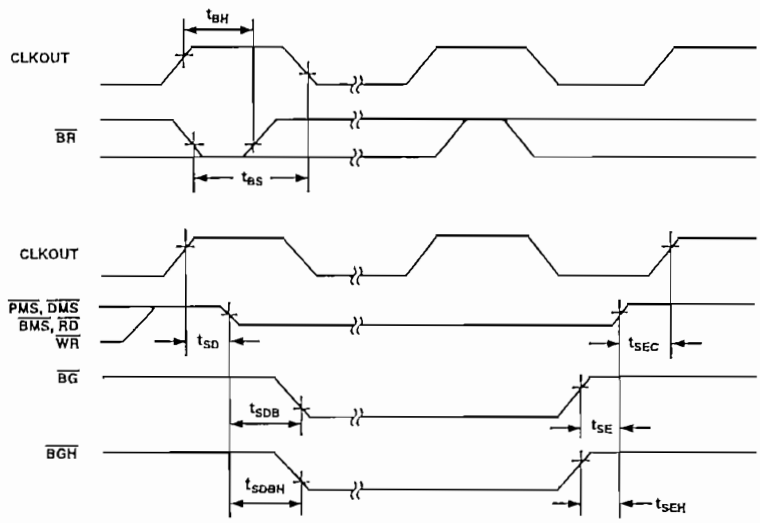


Figure 17. Bus Request–Bus Grant

Parameter		Min	Max	Unit
Memory Read				
Timing Requirements:				
t_{RDD}	$\overline{RD}$ Low to Data Valid		$0.5t_{CK} - 9 + w$	ns
t_{AA}	A0-A13, $\overline{xMS}$ to Data Valid		$0.75t_{CK} - 10.5 + w$	ns
t_{RDH}	Data Hold from $\overline{RD}$ High	0		ns
Switching Characteristics:				
t_{RP}	$\overline{RD}$ Pulse Width	$0.5t_{CK} - 5 + w$		ns
t_{CRD}	CLKOUT High to $\overline{RD}$ Low	$0.25t_{CK} - 5$	$0.25t_{CK} + 7$	ns
t_{ASR}	A0-A13, $\overline{xMS}$ Setup before $\overline{RD}$ Low	$0.25t_{CK} - 6$		ns
t_{RDA}	A0-A13, $\overline{xMS}$ Hold after $\overline{RD}$ Deasserted	$0.25t_{CK} - 3$		ns
t_{RWR}	$\overline{RD}$ High to $\overline{RD}$ or $\overline{WR}$ Low	$0.5t_{CK} - 5$		ns

w = wait states x t_{CK}
$\overline{xMS}$ = $\overline{PMS}$, $\overline{DMS}$, $\overline{CMS}$, $\overline{IOMS}$, $\overline{BMS}$.

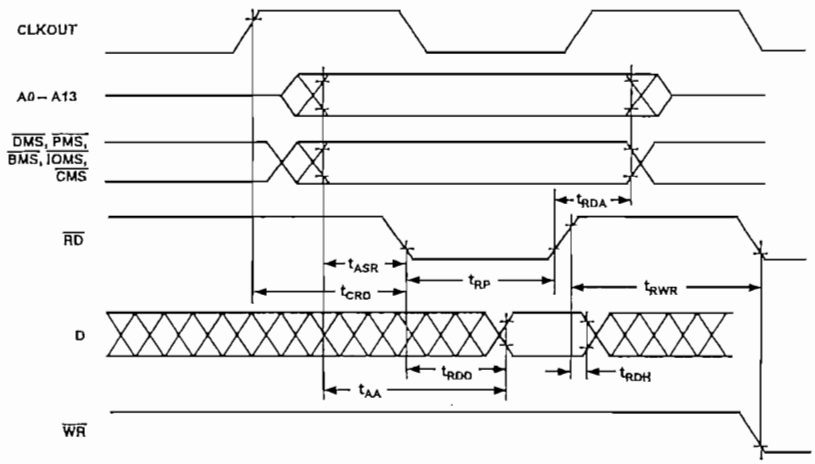


Figure 18. Memory Read

Parameter		Min	Max	Unit
Memory Write				
Switching Characteristics:				
t_{DW}	Data Setup before $\overline{WR}$ High	$0.5t_{CK} - 7 + w$		ns
t_{DH}	Data Hold after $\overline{WR}$ High	$0.25t_{CK} - 2$		ns
t_{WP}	$\overline{WR}$ Pulse Width	$0.5t_{CK} - 5 + w$		ns
t_{WDE}	$\overline{WR}$ Low to Data Enabled	0		ns
t_{ASW}	A0-A13, $\overline{xMS}$ Setup before $\overline{WR}$ Low	$0.25t_{CK} - 6$		ns
t_{DDR}	Data Disable before $\overline{WR}$ or $\overline{RD}$ Low	$0.25t_{CK} - 7$		ns
t_{CWR}	CLKOUT High to $\overline{WR}$ Low	$0.25t_{CK} - 5$	$0.25\ t_{CK} + 7$	ns
t_{AW}	A0-A13, $\overline{xMS}$, Setup before $\overline{WR}$ Deasserted	$0.75t_{CK} - 9 + w$		ns
t_{WRA}	A0-A13, $\overline{xMS}$ Hold after $\overline{WR}$ Deasserted	$0.25t_{CK} - 3$		ns
t_{WWR}	$\overline{WR}$ High to $\overline{RD}$ or $\overline{WR}$ Low	$0.5t_{CK} - 5$		ns

w = wait states x t_{CK}.
$\overline{xMS}$ = $\overline{PMS}$, $\overline{DMS}$, $\overline{CMS}$, $\overline{IOMS}$, $\overline{BMS}$.

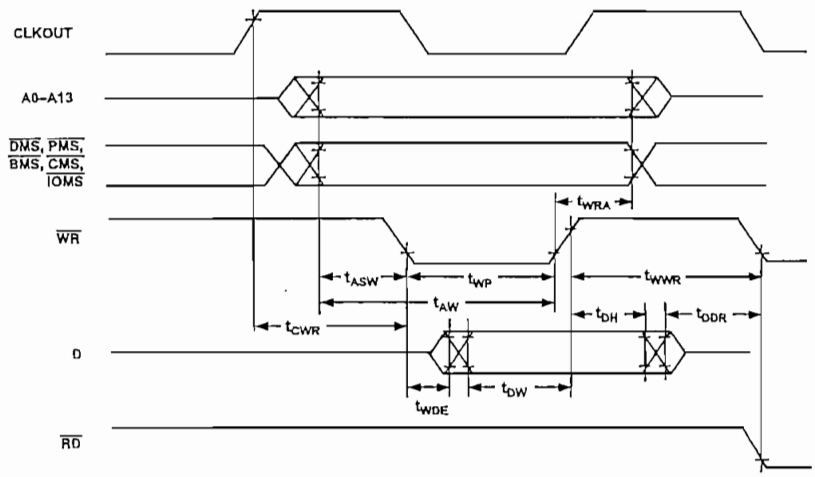


Figure 19. Memory Write

Parameter		Min	Max	Unit
Serial Ports				
Timing Requirements:				
t_{SCK}	SCLK Period	50		ns
t_{SCS}	DR/TFS/RFS Setup before SCLK Low	4		ns
t_{SCH}	DR/TFS/RFS Hold after SCLK Low	7		ns
t_{SCP}	$SCLK_{IN}$ Width	20		ns
Switching Characteristics:				
t_{CC}	CLKOUT High to $SCLK_{OUT}$	$0.25t_{CK}$	$0.25t_{CK} + 10$	ns
t_{SCDE}	SCLK High to DT Enable	0		ns
t_{SCDV}	SCLK High to DT Valid		15	ns
t_{RH}	TFS/RFS_{OUT} Hold after SCLK High	0		ns
t_{RD}	TFS/RFS_{OUT} Delay from SCLK High		15	ns
t_{SCDH}	DT Hold after SCLK High	0		ns
t_{TDE}	TFS (Alt) to DT Enable	0		ns
t_{TDV}	TFS (Alt) to DT Valid		14	ns
t_{SCDD}	SCLK High to DT Disable		15	ns
t_{RDV}	RFS (Multichannel, Frame Delay Zero) to DT Valid		15	ns

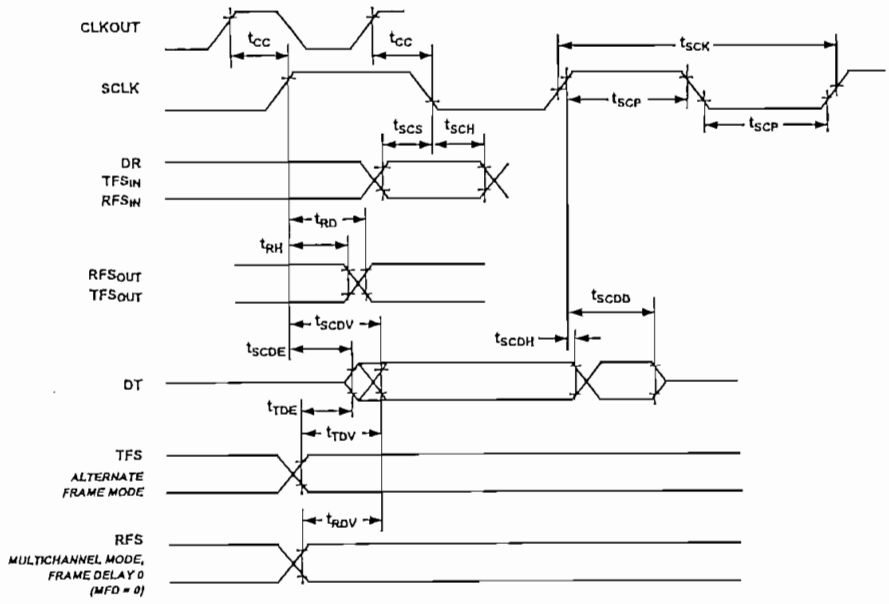


Figure 20. Serial Ports

Parameter		Min	Max	Unit
Serial Ports				
Timing Requirements:				
t_{SCK}	SCLK Period	50		ns
t_{SCS}	DR/TFS/RFS Setup before SCLK Low	4		ns
t_{SCH}	DR/TFS/RFS Hold after SCLK Low	7		ns
t_{SCP}	$SCLK_{IN}$ Width	20		ns
Switching Characteristics:				
t_{CC}	CLKOUT High to $SCLK_{OUT}$	$0.25t_{CK}$	$0.25t_{CK} + 10$	ns
t_{SCDE}	SCLK High to DT Enable	0		ns
t_{SCDV}	SCLK High to DT Valid		15	ns
t_{RH}	TFS/RFS_{OUT} Hold after SCLK High	0		ns
t_{RD}	TFS/RFS_{OUT} Delay from SCLK High		15	ns
t_{SCDH}	DT Hold after SCLK High	0		ns
t_{TDE}	TFS (Alt) to DT Enable	0		ns
t_{TDV}	TFS (Alt) to DT Valid		14	ns
t_{SCDD}	SCLK High to DT Disable		15	ns
t_{RDV}	RFS (Multichannel, Frame Delay Zero) to DT Valid		15	ns

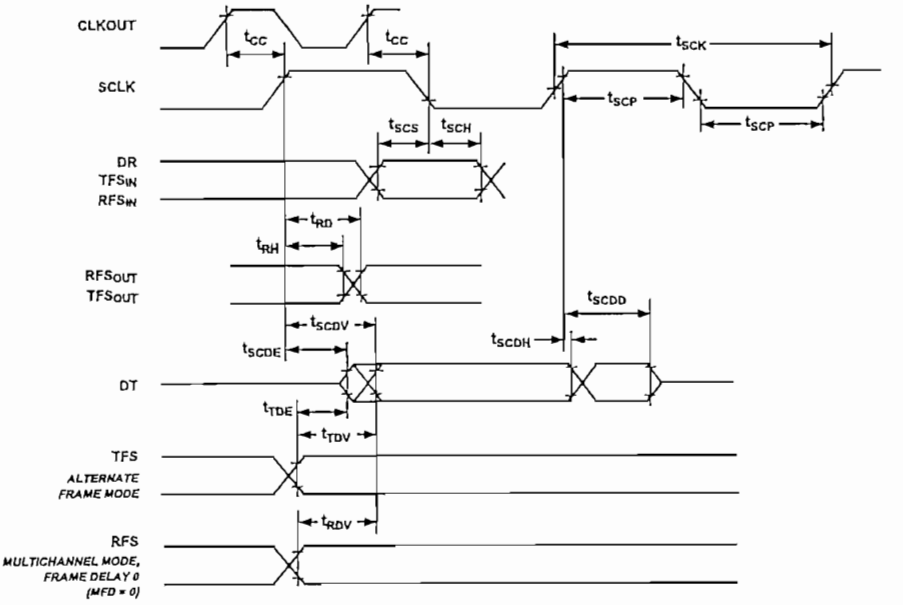


Figure 20. Serial Ports

Parameter		Min	Max	Unit
IDMA Write, Short Write Cycle				
Timing Requirements:				
t_{IKW}	$\overline{IACK}$ Low before Start of Write[1]	0		ns
t_{IWP}	Duration of Write[1, 2]	15		ns
t_{IDSU}	IAD15–0 Data Setup before End of Write[2, 3, 4]	5		ns
t_{IDH}	IAD15–0 Data Hold after End of Write[2, 3, 4]	2		ns
Switching Characteristics:				
t_{IKHW}	Start of Write to $\overline{IACK}$ High		15	ns

NOTES
[1]Start of Write = $\overline{IS}$ Low and $\overline{IWR}$ Low.
[2]End of Write = $\overline{IS}$ High or $\overline{IWR}$ High.
[3]If Write Pulse ends before $\overline{IACK}$ Low, use specifications t_{IDSU}, t_{IDH}.
[4]If Write Pulse ends after $\overline{IACK}$ Low, use specifications t_{IKSU}, t_{IKH}.

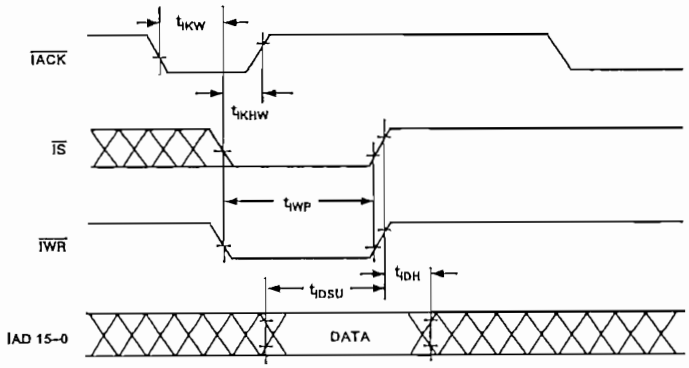


Figure 22. IDMA Write, Short Write Cycle

Parameter		Min	Max	Unit
IDMA Write, Long Write Cycle				
Timing Requirements:				
t_{IKW}	$\overline{IACK}$ Low before Start of Write[1]	0		ns
t_{IKSU}	IAD15–0 Data Setup before $\overline{IACK}$ Low[2, 3]	$0.5t_{CK} + 10$		ns
t_{IKH}	IAD15–0 Data Hold after $\overline{IACK}$ Low[2, 3]	2		ns
Switching Characteristics:				
t_{IKLW}	Start of Write to $\overline{IACK}$ Low[4]	$1.5t_{CK}$		ns
t_{IKHW}	Start of Write to $\overline{IACK}$ High		15	ns

NOTES
[1]Start of Write = $\overline{IS}$ Low and $\overline{IWR}$ Low.
[2]If Write Pulse ends before $\overline{IACK}$ Low, use specifications t_{IDSU}, t_{IDH}.
[3]If Write Pulse ends after $\overline{IACK}$ Low, use specifications t_{IKSU}, t_{IKH}.
[4]This is the earliest time for $\overline{IACK}$ Low from Start of Write. For IDMA Write cycle relationships, please refer to the User's Manual.

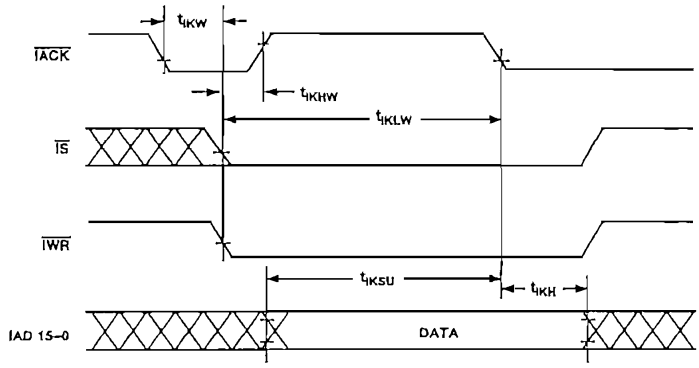


Figure 23. IDMA Write, Long Write Cycle

Parameter		Min	Max	Unit
IDMA Read, Long Read Cycle				
Timing Requirements:				
t_{IKR}	$\overline{IACK}$ Low before Start of Read[1]	0		ns
t_{IRP}	Duration of Read	15		ns
Switching Characteristics:				
t_{IKHR}	$\overline{IACK}$ High after Start of Read[1]		15	ns
t_{IKDS}	IAD15-0 Data Setup before IACK Low	$0.5t_{CK}$ -10		ns
t_{IKDH}	IAD15-0 Data Hold after End of Read[2]	0		ns
t_{IKDD}	IAD15-0 Data Disabled after End of Read[2]		12	ns
t_{IRDE}	IAD15-0 Previous Data Enabled after Start of Read	0		ns
t_{IRDV}	IAD15-0 Previous Data Valid after Start of Read		15	ns
t_{IRDH1}	IAD15-0 Previous Data Hold after Start of Read (DM/PM1)[3]	$2t_{CK}$ -5		ns
t_{IRDH2}	IAD15-0 Previous Data Hold after Start of Read (PM2)[4]	t_{CK}-5		ns

NOTES
[1]Start of Read = $\overline{IS}$ Low and $\overline{IRD}$ Low.
[2]End of Read = $\overline{IS}$ High or $\overline{IRD}$ High.
[3]DM read or first half of PM read.
[4]Second half of PM read.

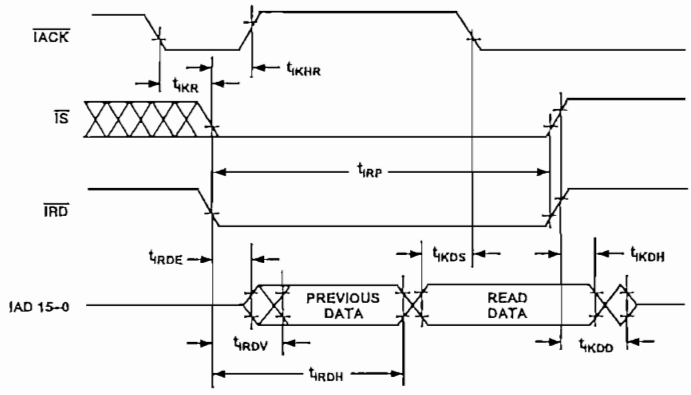


Figure 24. IDMA Read, Long Read Cycle

Parameter		Min	Max	Unit
IDMA Read, Short Read Cycle				
Timing Requirements:				
t_{IKR}	$\overline{IACK}$ Low before Start of Read[1]	0		ns
t_{IRP}	Duration of Read	15		ns
Switching Characteristics:				
t_{IKHR}	$\overline{IACK}$ High after Start of Read[1]		15	ns
t_{IKDH}	IAD15-0 Data Hold after End of Read[2]	0		ns
t_{IKDD}	IAD15-0 Data Disabled after End of Read[2]		12	ns
t_{IRDE}	IAD15-0 Previous Data Enabled after Start of Read	0		ns
t_{IRDV}	IAD15-0 Previous Data Valid after Start of Read		15	ns

NOTES
[1]Start of Read = $\overline{IS}$ Low and $\overline{IRD}$ Low.
[2]End of Read = $\overline{IS}$ High or $\overline{IRD}$ High.

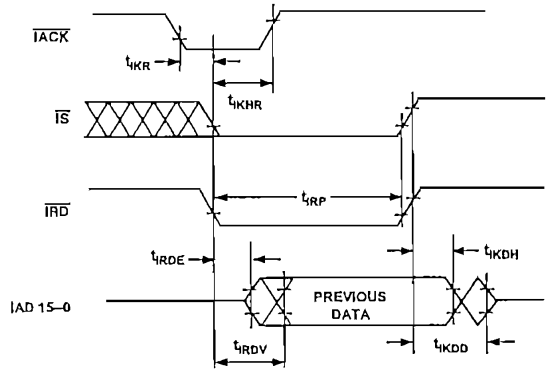
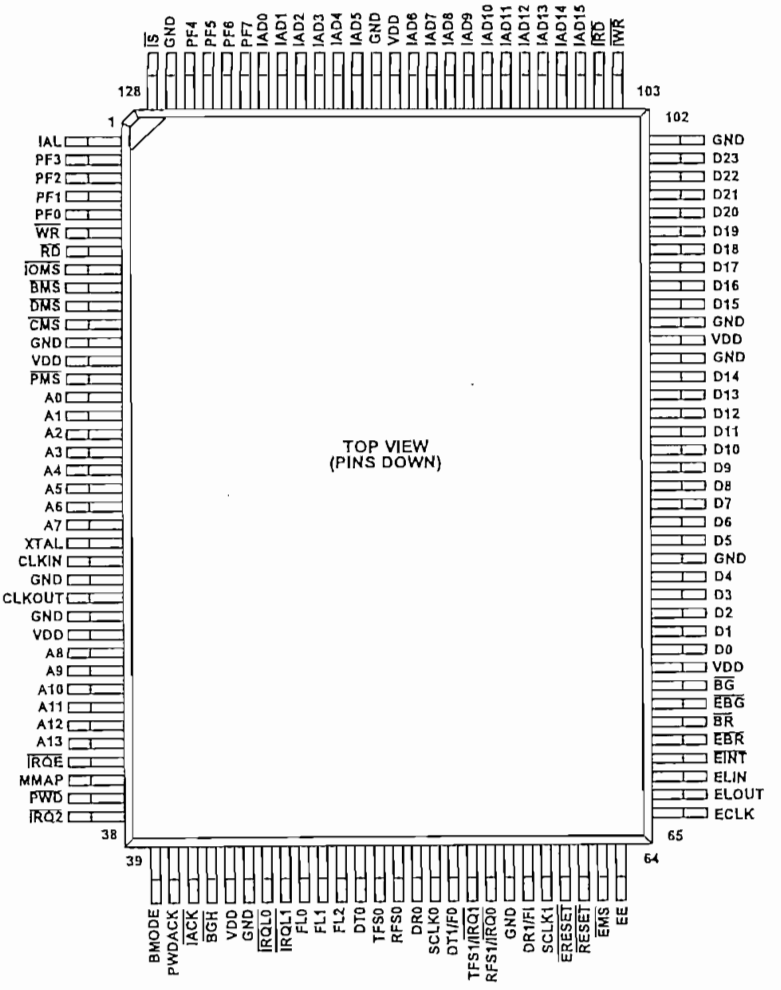


Figure 25. IDMA Read, Short Read Cycle

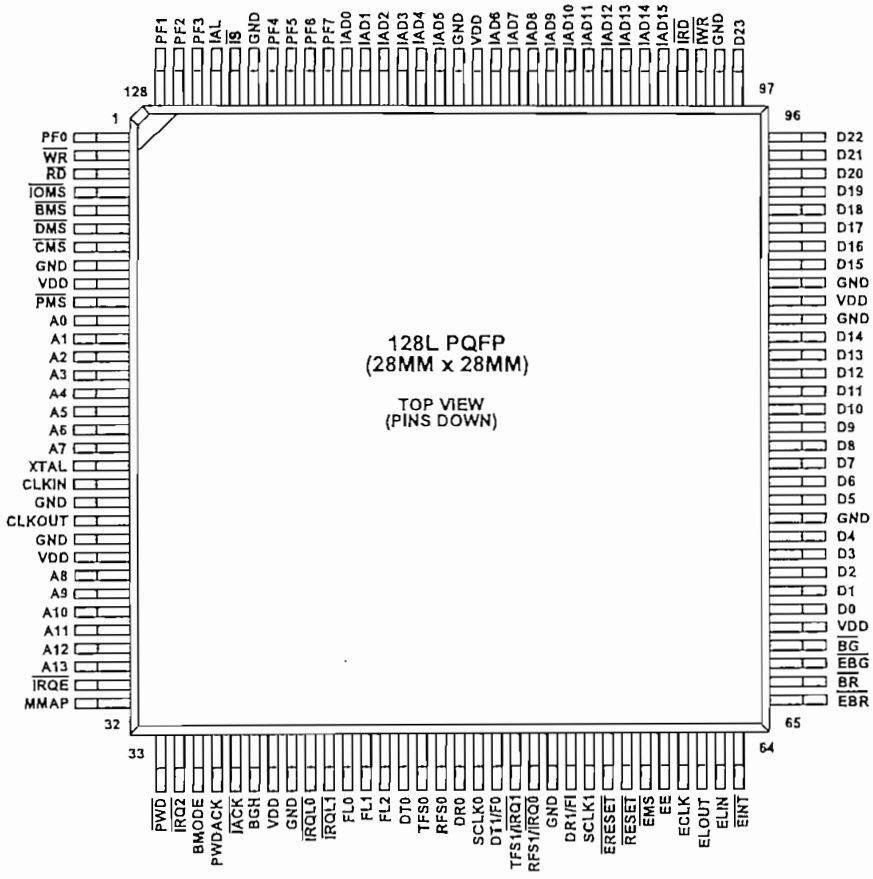
128-Lead TQFP Package Pinout



TQFP Pin Configurations

TQFP Number	Pin Name	TQFP Number	Pin Name	TQFP Number	Pin Name	TQFP Number	Pin Name
1	IAL	33	A12	65	ECLK	97	D19
2	PF3	34	A13	66	ELOUT	98	D20
3	PF2	35	IRQE	67	ELIN	99	D21
4	PF1	36	MMAP	68	EINT	100	D22
5	PF0	37	PWD	69	EBR	101	D23
6	WR	38	IRQ2	70	BR	102	GND
7	RD	39	BMODE	71	EBG	103	IWR
8	IOMS	40	PWDACK	72	BG	104	IRD
9	BMS	41	IACK	73	VDD	105	IAD15
10	DMS	42	BGH	74	D0	106	IAD14
11	CMS	43	VDD	75	D1	107	IAD13
12	GND	44	GND	76	D2	108	IAD12
13	VDD	45	IRQL0	77	D3	109	IAD11
14	PMS	46	IRQL1	78	D4	110	IAD10
15	A0	47	FL0	79	GND	111	IAD9
16	A1	48	FL1	80	D5	112	IAD8
17	A2	49	FL2	81	D6	113	IAD7
18	A3	50	DT0	82	D7	114	IAD6
19	A4	51	TFS0	83	D8	115	VDD
20	A5	52	RFS0	84	D9	116	GND
21	A6	53	DR0	85	D10	117	IAD5
22	A7	54	SCLK0	86	D11	118	IAD4
23	XTAL	55	DT1/F0	87	D12	119	IAD3
24	CLKIN	56	TFS1/IRQ1	88	D13	120	IAD2
25	GND	57	RFS1/IRQ0	89	D14	121	IAD1
26	CLKOUT	58	GND	90	GND	122	IAD0
27	GND	59	DR1/FI	91	VDD	123	PF7
28	VDD	60	SCLK1	92	GND	124	PF6
29	A8	61	ERESET	93	D15	125	PF5
30	A9	62	RESET	94	D16	126	PF4
31	A10	63	EMS	95	D17	127	GND
32	A11	64	EE	96	D18	128	IS

128-Lead PQFP Package Pinout



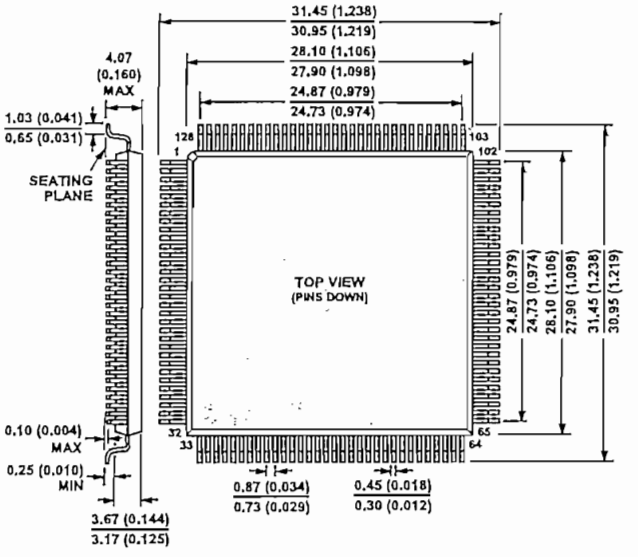
PQFP Pin Configurations

PQFP Number	Pin Name	PQFP Number	Pin Name	PQFP Number	Pin Name	PQFP Number	Pin Name
1	PF0	33	$\overline{PWD}$	65	$\overline{EBR}$	97	D23
2	$\overline{WR}$	34	$\overline{IRQ2}$	66	$\overline{BR}$	98	GND
3	$\overline{RD}$	35	BMODE	67	$\overline{EBG}$	99	$\overline{IWR}$
4	$\overline{IOMS}$	36	PWDACK	68	$\overline{BG}$	100	$\overline{IRD}$
5	$\overline{BMS}$	37	$\overline{IACK}$	69	VDD	101	IAD15
6	$\overline{DMS}$	38	$\overline{BGH}$	70	D0	102	IAD14
7	$\overline{CMS}$	39	VDD	71	D1	103	IAD13
8	GND	40	GND	72	D2	104	IAD12
9	VDD	41	$\overline{IRQL0}$	73	D3	105	IAD11
10	$\overline{PMS}$	42	$\overline{IRQL1}$	74	D4	106	IAD10
11	A0	43	FL0	75	GND	107	IAD9
12	A1	44	FL1	76	D5	108	IAD8
13	A2	45	FL2	77	D6	109	IAD7
14	A3	46	DT0	78	D7	110	IAD6
15	A4	47	TFS0	79	D8	111	VDD
16	A5	48	RFS0	80	D9	112	GND
17	A6	49	DR0	81	D10	113	IAD5
18	A7	50	SCLK0	82	D11	114	IAD4
19	XTAL	51	DT1/FO	83	D12	115	IAD3
20	CLKIN	52	TFS1/$\overline{IRQ1}$	84	D13	116	IAD2
21	GND	53	RFS1/$\overline{IRQ0}$	85	D14	117	IAD1
22	CLKOUT	54	GND	86	GND	118	IAD0
23	GND	55	DR1/FI	87	VDD	119	PF7
24	VDD	56	SCLK1	88	GND	120	PF6
25	A8	57	$\overline{ERESET}$	89	D15	121	PF5
26	A9	58	$\overline{RESET}$	90	D16	122	PF4
27	A10	59	$\overline{EMS}$	91	D17	123	GND
28	A11	60	EE	92	D18	124	$\overline{IS}$
29	A12	61	ECLK	93	D19	125	IAL
30	A13	62	ELOUT	94	D20	126	PF3
31	$\overline{IRQE}$	63	ELIN	95	D21	127	PF2
32	MMAP	64	$\overline{EINT}$	96	D22	128	PF1

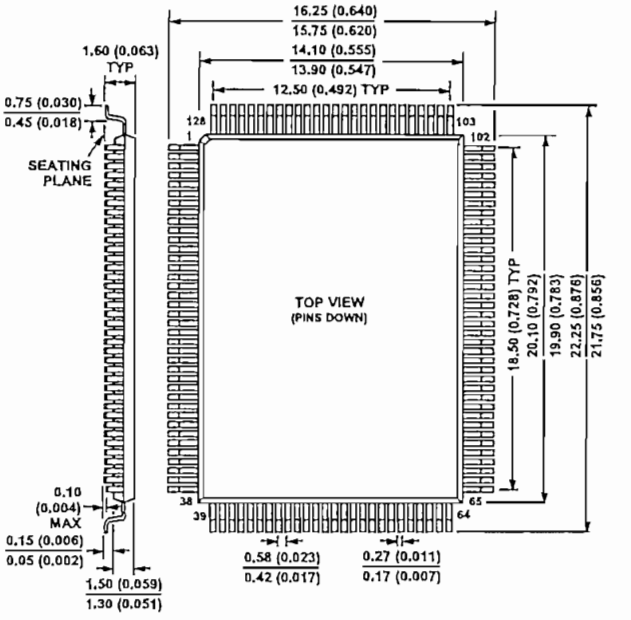
OUTLINE DIMENSIONS

Dimensions shown in mm and (inches).

128-Lead Metric Plastic Quad Flatpack (PQFP)
(S-128)



128-Lead Metric Thin Plastic Quad Flatpack (TQFP)
(ST-128)



ORDERING GUIDE

Part Number	Ambient Temperature Range	Instruction Rate (MHz)	Package Description	Package Option*
ADSP-2181KST-115	0°C to +70°C	28.8	128-Lead TQFP	ST-128
ADSP-2181BST-115	-40°C to +85°C	28.8	128-Lead TQFP	ST-128
ADSP-2181KS-115	0°C to +70°C	28.8	128-Lead PQFP	S-128
ADSP-2181BS-115	-40°C to +85°C	28.8	128-Lead PQFP	S-128
ADSP-2181KST-133	0°C to +70°C	33.3	128-Lead TQFP	ST-128
ADSP-2181BST-133	-40°C to +85°C	33.3	128-Lead TQFP	ST-128
ADSP-2181KS-133	0°C to +70°C	33.3	128-Lead PQFP	S-128
ADSP-2181BS-133	-40°C to +85°C	33.3	128-Lead PQFP	S-128
ADSP-2181KST-160	0°C to +70°C	40	128-Lead TQFP	ST-128
ADSP-2181BST-160	-40°C to +85°C	40	128-Lead TQFP	ST-128
ADSP-2181KS-160	0°C to +70°C	40	128-Lead PQFP	S-128
ADSP-2181BS-160	-40°C to +85°C	40	128-Lead PQFP	S-128

*S = Plastic Quad Flatpack (PQFP), ST = Plastic Thin Quad Flatpack (TQFP).

ANEXO 2

HOJA DE DATOS DEL CODEC AD1847

Serial-Port 16-Bit SoundPort Stereo Codec

AD1847

FEATURES

- Single-Chip Integrated ΣΔ Digital Audio Stereo Codec
- Supports the Microsoft Windows Sound System*
- Multiple Channels of Stereo Input
- Analog and Digital Signal Mixing
- Programmable Gain and Attenuation
- On-Chip Signal Filters
  - Digital Interpolation and Decimation
  - Analog Output Low-Pass
- Sample Rates from 5.5 kHz to 48 kHz
- 44-Lead PLCC and TQFP Packages
- Operation from +5 V Supplies
- Serial Digital Interface Compatible with ADSP-21xx Fixed-Point DSP

PRODUCT OVERVIEW

The AD1847 SoundPort® Stereo Codec integrates key audio data conversion and control functions into a single integrated circuit. The AD1847 is intended to provide a complete, low cost, single-chip solution for business, game audio and multimedia applications requiring operation from a single +5 V supply. It provides a serial interface for implementation on a computer motherboard, add-in or PCMCIA card. See Figure 1 for an example system diagram.

*Windows Sound System is a registered trademark of Microsoft Corp.
SoundPort is a registered trademark of Analog Devices, Inc.

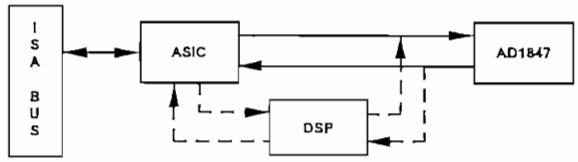


Figure 1. Example System Diagram

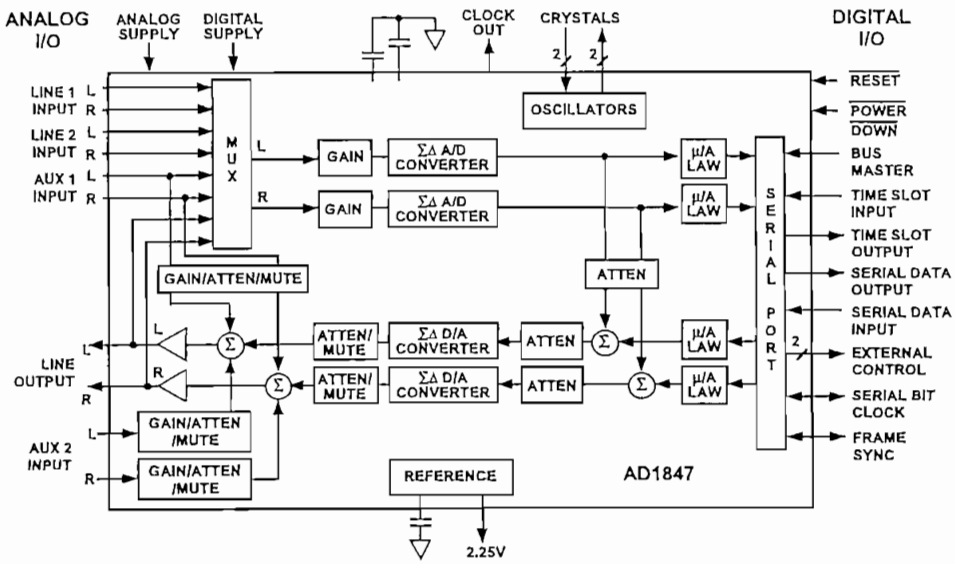
External circuit requirements are limited to a minimal number of low cost support components. Anti-imaging DAC output filters are incorporated on-chip. Dynamic range exceeds 70 dB over the 20 kHz audio band. Sample rates from 5.5 kHz to 48 kHz are supported from external crystals.

The Codec includes a stereo pair of ΣΔ analog-to-digital converters (ADCs) and a stereo pair of ΣΔ digital-to-analog converters (DACs). Inputs to the ADC can be selected from four stereo pairs of analog signals: line 1, line 2, auxiliary ("aux") line #1, and post-mixed DAC output. A software-controlled programmable gain stage allows independent gain for each channel going into the ADC. The ADCs' output can be digitally mixed with the DACs' input.

The pair of 16-bit outputs from the ADCs is available over a serial interface that also supports 16-bit digital input to the DACs and control/status information. The AD1847 can accept and generate 16-bit twos-complement PCM linear digital data, 8-bit unsigned magnitude PCM linear data, and 8-bit μ-law or A-law companded digital data.

(Continued on page 7)

FUNCTIONAL BLOCK DIAGRAM



REV. B

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 617/329-4700 Fax: 617/326-8703

AD1847–SPECIFICATIONS

STANDARD TEST CONDITIONS UNLESS OTHERWISE NOTED

Temperature	25	°C	*DAC Output Conditions*
Digital Supply (V_{DD})	5.0	V	0 dB Attenuation
Analog Supply (V_{CC})	5.0	V	Full-Scale Digital Inputs
Word Rate (F_S)	48	kHz	16-Bit Linear Mode
Input Signal	1007	Hz	No Output Load
Analog Output Passband	20	Hz to 20 kHz	Mute Off
FFT Size	4096		*ADC Input Conditions*
V_{IH}	2.4	V	0 dB Gain
V_{IL}	0.8	V	–3.0 dB Relative to Full Scale
V_{OH}	2.4	V	Line Input
V_{OL}	0.4	V	16-Bit Linear Mode

ANALOG INPUT

	Min	Typ	Max	Units
Full-Scale Input Voltage (RMS Values Assume Sine Wave Input)				
Line1, Line2, AUX1, AUX2		1		V rms
	2.54	2.8	3.10	V p-p
Input Impedance				
Line1, Line2, AUX1, AUX2†	10			kΩ
Input Capacitance†		15		pF

PROGRAMMABLE GAIN AMPLIFIER—ADC

	Min	Typ	Max	Units
Step Size (All Steps Tested, –30 dB Input)	1.10	1.5	1.90	dB
PGA Gain Range Span†	21.0		24.0	dB

AUXILIARY INPUT ANALOG AMPLIFIERS/ATTENUATORS

	Min	Typ	Max	Units
Step Size (+12 dB to –28.5 dB, Referenced to DAC Full Scale)	1.3	1.5	1.7	dB
(–30 dB to –34.5 dB, Referenced to DAC Full Scale)	1.1	1.5	1.9	dB
Input Gain/Attenuation Range Span†	45.5		47.5	dB
AUX Input Impedance†	10			kΩ

DIGITAL DECIMATION AND INTERPOLATION FILTERS†

	Min	Max	Units
Passband	0	$0.4 \times F_S$	Hz
Passband Ripple	–0.1	+0.1	dB
Transition Band	$0.4 \times F_S$	$0.6 \times F_S$	Hz
Stopband	$0.6 \times F_S$	∞	Hz
Stopband Rejection	74		dB
Group Delay		$30/F_S$	
Group Delay Variation Over Passband		0	µs

ANALOG-TO-DIGITAL CONVERTERS

	Min	Typ	Max	Units
Resolution		16		Bits
Dynamic Range (–60 dB Input, THD+N Referenced to Full Scale, A-Weighted)	70			dB
THD+N (Referenced to Full Scale)			0.040	%
			–68	dB
Signal-to-Intermodulation Distortion†		83		dB
ADC Crosstalk†				
Line Inputs (Input L, Ground R, Read R; Input R, Ground L, Read L)			–80	dB
Line1 to Line2 (Input Line1, Ground and Select Line2, Read Both Channels)			–80	dB
Line to AUX1			–80	dB
Line to AUX2			–80	dB
Line to DAC			–80	dB
Gain Error (Full-Scale Span Relative to V_{REFI})			±10	%
Interchannel Gain Mismatch (Difference of Gain Errors)			±0.2	dB
DC Offset			±55	LSB

DIGITAL-TO-ANALOG CONVERTERS

	Min	Typ	Max	Units
Resolution		16		Bits
Dynamic Range (–60 dB Input, THD+N Referenced to Full Scale, A-Weighted)	76			dB
THD+N (Referenced to Full Scale)			0.025	%
			–72	dB
Signal-to-Intermodulation Distortion†		86		dB
Gain Error (Full-Scale Span Relative to V_{REFI})			±10	%
Interchannel Gain Mismatch (Difference of Gain Errors)			±0.2	dB
DAC Crosstalk† (Input L, Zero R, Measure R_OUT; Input R, Zero L, Measure L_OUT)			–80	dB
Total Out-of-Band Energy† (Measured from 0.6 × F_S to 100 kHz)			–50	dB
Audible Out-of-Band Energy (Measured from 0.6 × F_S to 22 kHz, Tested at F_S = 5.5 kHz)			–55	dB

DAC ATTENUATOR

	Min	Typ	Max	Units
Step Size (0 dB to –22.5 dB) (Tested at Steps 0 dB, –19.5)	1.3	1.5	1.7	dB
Step Size (–24 dB to –94 dB)	1.0	1.5	2.0	dB
Output Attenuation Range Span†	–93		95	dB

DIGITAL MIX ATTENUATOR

	Min	Typ	Max	Units
Step Size (0 dB to –22.5 dB) (Tested at Steps 0 dB, –19.5)	1.3	1.5	1.7	dB
Step Size (–24 dB to –94 dB)	1.0	1.5	2.0	dB
Output Attenuation Range Span†	–93.5		95.5	dB

ANALOG OUTPUT

	Min	Typ	Max	Units
Full-Scale Line Output Voltage		0.707		V rms
V_{REFI} = 2.35*	1.80	2	2.20	V p-p
Line Output Impedance†			600	Ω
External Load Impedance	10			kΩ
Output Capacitance†			15	pF
External Load Capacitance			100	pF
V_{REF} (Clock Running)	2.00		2.50	V
V_{REF} Current Drive		100		µA
V_{REFI}		2.35		V
Mute Attenuation of 0 dB Fundamental† (LOUT)			–80	dB
Mute Click† (|Muted Output Minus Unmuted Midscale DAC Output|)			8	mV

*Full-scale line output voltage scales with V_{REF} (e.g., V_{OUT} (typ) = 2.0 V × (V_{REF}/2.35)).
†Guaranteed, Not Tested.

SYSTEM SPECIFICATIONS

	Min	Typ	Max	Units
System Frequency Response† (Line In to Line Out, 20 Hz to 20 kHz)		±0.3		dB
Differential Nonlinearity†			±1/2	Bit
Phase Linearity Deviation†.		1		Degrees

STATIC DIGITAL SPECIFICATIONS

	Min	Max	Units
High Level Input Voltage (V_{IH})			
Digital Inputs	2.0		V
XTAL1/2I	2.4		V
Low Level Input Voltage (V_{IL})		0.8	V
High Level Output Voltage (V_{OH}) I_{OH} = 1 mA	2.4	V_{DD}	V
Low Level Output Voltage (V_{OL}) I_{OL} = 4 mA		0.4	V
Input Leakage Current (GO/NOGO Tested)	-10	+10	µA
Output Leakage Current (GO/NOGO Tested)	-10	+10	µA

TIMING PARAMETERS (Guaranteed Over Operating Temperature Range)

	Min	Typ	Max	Units
Serial Frame Sync Period (t_I)		1/0.5 F_S		µs
Clock to Frame Sync [SDFS] Propagation Delay (t_{PD1})			20	ns
Data Input Setup Time (t_S)	15			ns
Data Input Hold Time (t_H)	15			ns
Clock to Output Data Valid (t_{DV})			25	ns
Clock to Output Three-State [High-Z] (t_{HZ})			20	ns
Clock to Time Slot Output [TSO] Propagation Delay (t_{PD2})			20	ns
$\overline{\text{RESET}}$ and $\overline{\text{PWRDOWN}}$ Lo Pulse Width (t_{RPWL})	100			ns

POWER SUPPLY

	Min	Max	Units
Power Supply Range – Digital & Analog	4.75	5.25	V
Power Supply Current – Operating (10 kΩ Line Out Load)		140	mA
Analog Supply Current – Operating (10 kΩ Line Out Load)		70	mA
Digital Supply Current – Operating (10 kΩ Line Out Load)		70	mA
Analog Power Supply Current – Power Down		400	µA
Digital Power Supply Current – Power Down		400	µA
Power Dissipation – Operating (Current × Nominal Supply)		750	mW
Power Dissipation – Power Down (Current × Nominal Supply)		4	mW
Power Supply Rejection (@ 1 kHz)†			
(At Both Analog and Digital Supply Pins, ADCs)	45		dB
(At Both Analog and Digital Supply Pins, DACs)	55		dB

CLOCK SPECIFICATIONS†

	Min	Max	Units
Input Clock Frequency		27	MHz
Recommended Clock Duty Cycle		±10	%
Initialization/Sample Rate Change Time			
16.9344 MHz Crystal Selected at Power-Up		171	ms
24.576 MHz Crystal Selected at Power-Up		171	ms
16.9344 MHz Crystal Selected Subsequently		6	ms
24.576 MHz Crystal Selected Subsequently		6	ms

†Guaranteed, not tested.
Specifications subject to change without notice.

ABSOLUTE MAXIMUM RATINGS*

	Min	Max	Units
Power Supplies			
Digital (V_{DD})	−0.3	6.0	V
Analog (V_{CC})	−0.3	6.0	V
Input Current			
(Except Supply Pins)		±10.0	mA
Analog Input Voltage (Signal Pins)	−0.3	(VA+) + 0.3	V
Digital Input Voltage (Signal Pins)	−0.3	(VD+) + 0.3	V
Ambient Temperature (Operating)	0	+70	°C
Storage Temperature	−65	+150	°C

*Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ORDERING GUIDE

Model	Temperature Range	Package Description	Package Option*
AD1847JP	0°C to +70°C	44-Lead PLCC	P-44A
AD1847JST	0°C to +70°C	44-Lead TQFP	ST-44

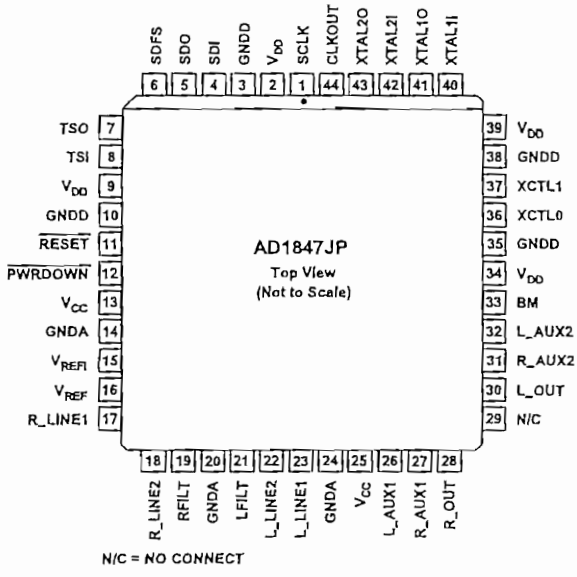
*P = PLCC; ST = TQFP.

CAUTION

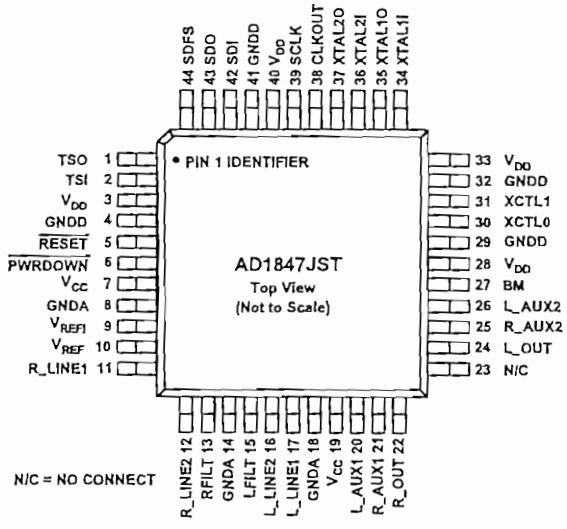
ESD (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000 V readily accumulate on the human body and test equipment and can discharge without detection. Although the AD1847 features proprietary ESD protection circuitry, permanent damage may occur on devices subjected to high energy electrostatic discharges. Therefore, proper ESD precautions are recommended to avoid performance degradation or loss of functionality.

PINOUTS

44-Lead PLCC



44-Lead TQFP



PIN DESCRIPTIONS

Parallel Interface

Pin Name	PLCC	TQFP	I/O	Description
SCLK	1	39	I/O	Serial Clock. SCLK is a bidirectional signal that supplies the clock as an output to the serial bus when the Bus Master (BM) pin is driven HI and accepts the clock as an input when the BM pin is driven LO. The serial clock output is fixed at 12.288 MHz when XTAL1 is selected, and 11.2896 MHz when XTAL2 is selected. SCLK runs continuously. An AD1847 should always be configured as the serial bus master unless it is a slave in a daisy-chained multiple codec system.
SDFS	6	44	I/O	Serial Data Frame Sync. SDFS is a bidirectional signal that supplies the frame synchronization signal as an output to the serial bus when the Bus Master (BM) pin is driven HI and accepts the frame synchronization signal as an input when the BM pin is driven LO. The SDFS frequency powers up at one half of the AD1847 sample rate (i.e., FRS bit = 0) with two samples per frame and can be programmed to match the sample rate (i.e., FRS bit = 1) with one sample per frame. An AD1847 should always be configured as the serial bus master unless it is a slave in a daisy-chained multiple codec system.
SDI	4	42	I	Serial Data Input. SDI is used by peripheral devices such as the host CPU or a DSP to supply control and playback data information to the AD1847. All control and playback transfers are 16 bits long, MSB first.
SDO	5	43	O	Serial Data Output. SDO is used to supply status/index readback and capture data information to peripheral devices such as the host CPU or a DSP. All status/index readback and capture data transfers are 16 bits long, MSB first. Three-state output driver.
$\overline{\text{RESET}}$	11	5	I	Reset. The $\overline{\text{RESET}}$ signal is active LO. The assertion of this signal will initialize the on-chip registers to their default values. See the "Control Register Definitions" section for a description of the contents of the control registers after $\overline{\text{RESET}}$ is deasserted.
$\overline{\text{PWRDOWN}}$	12	6	I	Powerdown. The $\overline{\text{PWRDOWN}}$ signal is active LO. The assertion of this signal will reset the on-chip control registers (identically to the $\overline{\text{RESET}}$ signal) and will also place the AD1847 in a low power consumption mode. V_{REF} and all analog circuitry are disabled.
BM	33	27	I	Bus Master. The assertion (HI) of this signal indicates that the AD1847 is the serial bus master. The AD1847 will then supply the serial clock (SCLK) and the frame sync (SDFS) signals for the serial bus. One and only one AD1847 should always be configured as the serial bus master. If BM is connected to logic LO, the AD1847 is configured as a bus slave, and will accept the SCLK and SDFS signals as inputs. An AD1847 should only be configured as a serial bus slave when an AD1847 serial bus master already exists, in daisy-chained multiple codec systems.
TSO	7	1	O	Time Slot Output. This signal is asserted HI by the AD1847 coincidentally with the LSB of the last time slot used by the AD1847. Used in daisy-chained multiple codec systems.
TSI	8	2	I	Time Slot Input. The assertion of this signal indicates that the AD1847 should immediately use the next three time slots (TSSEL = 1) or the next six time slots (TSSEL = 0) and then activate the TSO pin to enable the next device down the TDM chain. TSI should be driven LO when the AD1847 is the bus master or in single codec systems. Used in daisy-chained multiple codec systems.
CLKOUT	44	38	O	Clock Output. This signal is the buffered version of the crystal clock output and the frequency is dependent on which crystal is selected. This pin can be three-stated by driving the BM pin LO or by programming the CLKTS bit in the Pin Control Register. See the "Control Registers" section for more details. The CLKOUT frequency is 12.288 MHz when XTAL1 is selected and 16.9344 MHz when XTAL2 is selected.

Analog Signals

Pin Name	PLCC	TQFP	I/O	Description
L_LINE1	23	17	I	Left Line Input #1. Line level input for the #1 left channel.
R_LINE1	17	11	I	Right Line Input #1. Line level input for the #1 right channel.
L_LINE2	22	16	I	Left Line Input #2. Line level input for the #2 left channel.
R_LINE2	18	12	I	Right Line Input #2. Line level input for the #2 right channel.
L_AUX1	26	20	I	Left Auxiliary Input #1. Line level input for the AUX1 left channel.
R_AUX1	27	21	I	Right Auxiliary Input #1. Line level input for the AUX1 right channel.
L_AUX2	32	26	I	Left Auxiliary Input #2. Line level input for the AUX2 left channel.
R_AUX2	31	25	I	Right Auxiliary Input #2. Line level input for the AUX2 right channel.
L_OUT	30	24	O	Left Line Output. Line level output for the left channel.
R_OUT	28	22	O	Right Line Output. Line level output for the right channel.

Miscellaneous

Pin Name	PLCC	TQFP	I/O	Description
XTAL1I	40	34	I	24.576 MHz Crystal #1 Input.
XTAL1O	41	35	O	24.576 MHz Crystal #1 Output.
XTAL2I	42	36	I	16.9344 MHz Crystal #2 Input.
XTAL2O	43	37	O	16.9344 MHz Crystal #2 Output.
XCTL1:O	37 & 36	31 & 30	O	External Control. These TTL signals reflect the current status of register bits inside the AD1847. They can be used for signaling or to control external logic.
V_{REF}	16	10	O	Voltage Reference. Nominal 2.25 volt reference available externally as a voltage datum for dc-coupling and level-shifting. V_{REF} should not have any signal dependent load.
V_{REFI}	15	9	I	Voltage Reference Internal. Voltage reference filter point for external bypassing only.
L_FILT	21	15	I	Left Channel Filter Capacitor. This pin requires a 1.0 µF capacitor to analog ground for proper operation.
R_FILT	19	13	I	Right Channel Filter Capacitor. This pin requires a 1.0 µF capacitor to analog ground for proper operation.
NC	29	23		No Connect. Do not connect.

Power Supplies

Pin Name	PLCC	TQFP	I/O	Description
V_{CC}	13 & 25	7 & 19	I	Analog Supply Voltage (+5 V).
GNDA	14, 20, 24	8, 14, 18	I	Analog Ground.
V_{DD}	2, 9, 34, 39	40, 3, 28, 33	I	Digital Supply Voltage (+5 V).
GNDD	3, 10, 35, 38	41, 4, 29, 32	I	Digital Ground.

(Continued from page 1)

The ΣΔ DACs are preceded by a digital interpolation filter. An attenuator provides independent user volume control over each DAC channel. Nyquist images are removed from the DACs' analog stereo output by on-chip switched-capacitor and continuous-time filters. Two stereo pairs of auxiliary line-level inputs can also be mixed in the analog domain with the DAC output.

The AD1847 serial data interface uses a Time Division Multiplex (TDM) scheme that is compatible with DSP serial ports configured in Multi-Channel Mode with 32 16-bit time slots (i.e., SPORT0 on the ADSP-2101, ADSP-2115, etc.).

AUDIO FUNCTIONAL DESCRIPTION

This section overviews the functionality of the AD1847 and is intended as a general introduction to the capabilities of the device. As much as possible, detailed reference information has been placed in "Control Registers" and other sections. The user is not expected to refer repeatedly to this section.

Analog Inputs

The AD1847 SoundPort Stereo Codec accepts stereo line-level inputs. All inputs should be capacitively coupled (ac-coupled) to the AD1847. LINE1, LINE2, and AUX1, and post-mixed DAC output analog stereo signals are multiplexed to the internal programmable gain amplifier (PGA) stage.

The PGA following the input multiplexer allows independent selectable gains for each channel from 0 to 22.5 dB in +1.5 dB steps. The Codec can operate either in a global stereo mode or in a global mono mode with left-channel inputs appearing at both channel outputs.

Analog Mixing

AUX1 and AUX2 analog stereo signals can be mixed in the analog domain with the DAC output. Each channel of each auxiliary analog input can be independently gained/attenuated from +12 dB to –34.5 dB in –1.5 dB steps or completely muted. The post-mixed DAC output is available on L_OUT and R_OUT externally and as an input to the ADCs.

Even if the AD1847 is not playing back data from its DACs, the analog mix function can still be active.

Analog-to-Digital Datapath

The ΣΔ ADCs incorporate a proprietary fourth-order modulator. A single pole of passive filtering is all that is required for antialiasing the analog input because of the ADC's high 64 times oversampling ratio. The ADCs include digital decimation filters that low-pass filter the input to $0.4 \times F_S$. ("F_S" is the word rate or "sampling frequency.") ADC input overrange conditions will cause status bits to be set that can be read.

Digital-to-Analog Datapath

The ΣΔ DACs contain a programmable attenuator and a low-pass digital interpolation filter. The anti-imaging interpolation filter oversamples and digitally filters the higher frequency images. The attenuator allows independent control of each DAC channel from 0 dB to –94.5 dB in 1.5 dB steps plus full mute. The DACs' ΣΔ noise shapers also oversample and convert the signal to a single-bit stream. The DAC outputs are then filtered in the analog domain by a combination of switched-capacitor and continuous-time filters. These filters remove the very high frequency components of the DAC bitstream output. No external components are required.

Changes in DAC output attenuation take effect only on zero crossings of the digital signal, thereby eliminating "zipper" noise on playback. Each channel has its own independent zero-crossing detector and attenuator change control circuitry. A timer guarantees that requested volume changes will occur even in the absence of an input signal that changes sign. The time-out period is 8 milliseconds at a 48 kHz sampling rate and 48 milliseconds at an 8 kHz sampling rate. (Time-out [ms] ≈ $384/F_S$ [kHz]).

Digital Mixing

Stereo digital output from the ADCs can be mixed digitally with the input to the DACs. Digital output from the ADCs going out of the serial data port is unaffected by the digital mix. Along the digital mix datapath, the 16-bit linear output from the ADCs is attenuated by an amount specified with control bits. Both channels of the monitor data are attenuated by the same amount. (Note that internally the AD1847 always works with 16-bit PCM linear data, digital mixing included; format conversions take place at the input and output.)

Sixty-four steps of −1.5 dB attenuation are supported to −94.5 dB. The digital mix datapath can also be completely muted, preventing any mixing of the digital input with the digital output. Note that the level of the mixed signal is also a function of the input PGA settings, since they affect the ADCs' output.

The attenuated digital mix data is digitally summed with the DAC input data prior to the DACs' datapath attenuators. The digital sum of digital mix data and DAC input data is clipped at plus or minus full scale and does not wrap around. Because both stereo signals are mixed before the output attenuators, mix data is attenuated a second time by the DACs' datapath attenuators.

Analog Outputs

A stereo line-level output is available at external pins. Other output types such as headphone and speaker must be implemented in external circuitry. The stereo line-level outputs should be capacitively coupled (ac-coupled) to the external circuitry. Each channel of this output can be independently muted. When muted, the outputs will settle to a dc value near V_{REF}, the midscale reference voltage.

Digital Data Types

The AD1847 supports four global data types: 16-bit twos-complement linear PCM, 8-bit unsigned linear PCM, companded μ-law, and 8-bit companded A-law, as specified by control register bits. Eight-bit data is always left-justified in 16-bit fields; in other words, the MSBs of all data types are always aligned; in yet other words, full-scale representations in all four formats correspond to equivalent full-scale signals. The eight least significant bit positions of 8-bit data in 16-bit fields are ignored on digital input and zoned on digital output (i.e., truncated).

The 16-bit PCM data format is capable of representing 96 dB of dynamic range. Eight-bit PCM can represent 48 dB of dynamic range. Companded μ-law and A-law data formats use nonlinear coding with less precision for large-amplitude signals. The loss of precision is compensated for by an increase in dynamic range to 64 dB and 72 dB, respectively.

On input, 8-bit companded data is expanded to an internal linear representation, according to whether μ-law or A-law was specified in the Codec's internal registers. Note that when μ-law compressed data is expanded to a linear format, it requires 14 bits. A-law data expanded requires 13 bits.

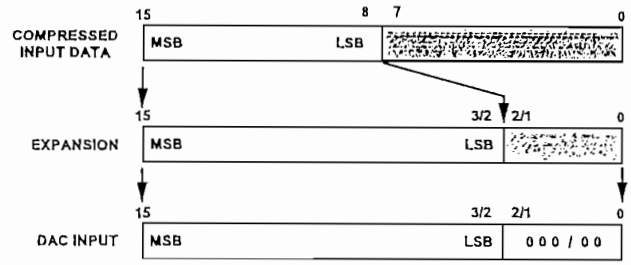


Figure 2. A-Law or μ-Law Expansion

When 8-bit companding is specified, the ADCs' linear output is compressed to the format specified.

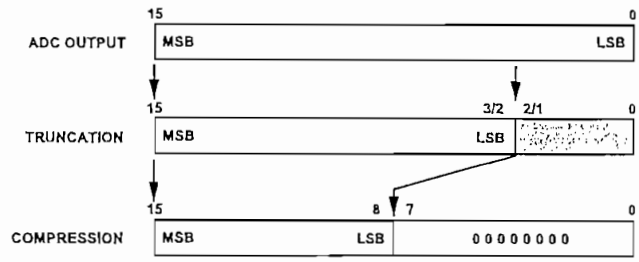


Figure 3. A-Law or μ-Law Compression

Note that all format conversions take place at input or output. Internally, the AD1847 always uses 16-bit linear PCM representations to maintain maximum precision.

Power Supplies and Voltage Reference

The AD1847 operates from +5 V power supplies. Independent analog and digital supplies are recommended for optimal performance though excellent results can be obtained in single-supply systems. A voltage reference is included on the Codec and its 2.25 V buffered output is available on an external pin (V_{REF}). The reference output can be used for biasing op amps used in dc coupling. The internal reference must be externally bypassed to analog ground at the V_{REFI} pin, and must not be used to bias external circuitry.

Clocks and Sample Rates

The AD1847 operates from two external crystals, XTAL1 and XTAL2. The two crystal inputs are provided to generate a wide range of sample rates. The oscillators for these crystals are on the AD1847, as is a multiplexer for selecting between them. They can be overdriven with external clocks by the user, if so desired. At a minimum, XTAL1 must be provided since it is selected as the reset default. If XTAL2 is not used, the XTAL2 input pin should be connected to ground. The recommended crystal frequencies are 16.9344 MHz and 24.576 MHz. From them, the following sample rates can be selected: 5.5125, 6.615, 8, 9.6, 11.025, 16, 18.9, 22.05, 27.42857, 32, 33.075, 37.8, 44.1, 48 kHz.

CONTROL REGISTERS

Control Register Mapping

The AD1847 has six 16-bit and thirteen 8-bit on-chip user-accessible control registers. Control information is sent to the AD1847 in the 16-bit Control Word. Status information is sent from the AD1847 in the 16-bit Status Word. Playback Data and Capture Data each have two 16-bit registers for the right and left channels. Additional 8-bit Index Registers are accessed via indirect addressing in the AD1847 Control Word. [Index Registers are reached with indirect addressing.] The contents of an indirect addressed Index Register may be readback by the host CPU or DSP (during the Status Word/Index Readback time slot) by setting the Read Request (RREQ) bit in the Control Word. Note that each 16-bit register is assigned its own time slot, so that the AD1847 always consumes six 16-bit time slots. Figure 4 shows the mapping of the Control Word, Status Word/Index Readback and Data registers to time slots when TSSEL = 0. TSSEL = 0 is used when the SDI and SDO pins are tied together (i.e., "1-wire" system). This configuration is efficient in terms of component interconnect (one bidirectional wire for serial data input and output), but inefficient in terms of time slot usage (six slots consumed on single bidirectional Time Division Multiplexed [TDM] serial bus). When TSSEL = 0, serial data input to the AD1847 occurs sequentially with serial data output from the AD1847 (i.e., Control Word, Left Playback and Right Playback data is received on the SDI pin, then the Status Word/Index Readback, Left Capture and Right Capture data is transmitted on the SDO pin).

Slot	Register Name (16-Bit)
0	Control Word Input
1	Left Playback Data Input
2	Right Playback Data Input
3	Status Word/Index Readback Output
4	Left Capture Data Output
5	Right Capture Data Output

Figure 4. Control Register Mapping with TSSEL = 0

Figure 5 shows the mapping of the Control Word, Status Word/Index Readback and Data registers to time slots when TSSEL = 1. Note that the six 16-bit registers "share" three time slots. TSSEL = 1 is used when the SDI and SDO pins are independent inputs and output (i.e., "2-wire" system). This configuration is inefficient in terms of component interconnect (two unidirectional wires for serial data input and output), but efficient in terms of time slot usage (three slots consumed on each of two unidirectional TDM serial buses). When TSSEL = 1, serial data input to the AD1847 occurs concurrently with serial data output from the AD1847 (i.e., Control Word reception on the SDI pin occurs simultaneously with Status Word/Index Readback transmission on the SDO pin).

Slot	Register Name (16-Bit)
0	Control Word Input
1	Left Playback Data Input
2	Right Playback Data Input
0	Status Word/Index Readback Output
1	Left Capture Data Output
2	Right Capture Data Output

Figure 5. Control Register Mapping with TSSEL = 1

An Index Register readback request to an invalid index address (11, 14 and 15) will return the contents of the Status Word. Attempts to write to an invalid index address (11, 14 and 15) will have no effect on the AD1847. As mentioned above, the RREQ bit of the Control Word is used to request Status Word output or Index Register readback output during either time slot 3 (TSSEL = 0) or time slot 0 (TSSEL = 1). RREQ is set for Index Register readback output, and reset for Status Word output. When Index Register readback is requested, the Index Readback bit format is the same as the Control Word bit format. All status bits are updated by the AD1847 before a new Control Word is received (i.e., at frame boundaries). Thus, if TSSEL = 0 and the Control Word written at slot 0 causes some status bits to change, the change will show up in the Status Word transmitted at slot 3 of the same sample.

Control Word (16-Bit)

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
CLOR	MCE	RREQ	res	IA3	IA2	IA1	IA0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

DATA7:0 Index Register Data. These bits are the data for the desired AD1847 Index Register referenced by the Index Address. Written by the host CPU or DSP to the AD1847.

IA3:0 Index Register Address. These bits define the indirect address of the desired AD1847 Index Register. Written by the host CPU or DSP to the AD1847.

RREQ Read Request. Setting this bit indicates that the current transfer is a request by the host CPU or DSP for readback of the contents of the indirect addressed Index Register. When this bit is set (RREQ = HI), the AD1847 will not transmit its Status Word in the following Status Word Index readback slot, but will instead transmit the data in the Index Register specified by the Index Address. Although the Index Readback is transmitted in the following Status Word/Index Readback time slot, the format of the Control Word is used (i.e., CLOR, MCE, RREQ and the Index Register Address in the most significant byte, and the readback Index Register Data in the least significant byte). When this bit is reset (RREQ = LO), the AD1847 will transmit its Status Word in the following Status Word Index Readback time slot.

A read request is serviced in the next available Index Readback time slot. If TSSEL = 0, the Index Register readback data is transmitted in slot 3 of the same sample. If TSSEL = 1, Index Register readback data is transmitted in slot 0 of the next sample. If TSSEL changes from 0 to 1, Index Register readback will occur twice, in slot 3 of the current sample, and slot 0 of the next. If TSSEL changes from 1 to 0, the last read request is lost.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

MCE Mode Change Enable. This bit must be set (MCE = HI) whenever protected control register bits of the AD1847 are changed. The Data Format register, the Miscellaneous Information register, and the ACAL bit of the Interface Configuration register can NOT be changed unless this bit is set. The DAC outputs will be muted when MCE is set. The user must mute the AUX1 and AUX2 channels when this bit is set (no audio activity should occur). Written by the host CPU or DSP to the AD1847. This bit is HI after reset.

CLOR Clear Overrange. When this bit is set (CLOR = HI), the overrange bits in the Status Word are updated every sample. When this bit is reset (CLOR = LO), the overrange bits in the Status Word will record the largest overrange value. The largest overrange value is sticky until the CLOR bit is set. Written by the host CPU or DSP to the AD1847. Since there can be up to 2 samples in the data pipeline, a change to CLOR may take up to 2 samples periods to take effect. This bit is HI after reset.

Immediately after reset, the contents of this register is: 1100 0000 0000 0000 (C000h).

Left/Right Playback/Capture Data (16-Bit)

The data formats for Left Playback, Right Playback, Left Capture and Right Capture are all identical.

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

DATA15:0 Data Bits. These registers contain the 16-bit, MSB first data for capture and playback. The host CPU or DSP reads the capture data from the AD1847. The host CPU or DSP writes the playback data to the AD1847. For 8-bit linear or 8-bit companded modes, only DATA15:8 contain valid data; DATA7:0 are ignored during capture, and are zeroed during playback. Mono mode plays back the same audio sample on both left and right channels. Mono capture only captures data from the left audio channel. See "Serial Data Format" Timing Diagram.

Immediately after reset, the content of these registers is: 0000 0000 0000 0000 (0000h).

Status Word (16-Bit)

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
res	res	RREQ	res	ID3	ID2	ID1	ID0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
res	res	ORR1	ORR0	ORL1	ORL0	ACI	INIT

INIT Initialization. This bit is an indication to the host that frame syncs will stop and the serial bus will be shut down. INIT is set HI on the last valid frame. It is reset LO for all other frames. Read by the host CPU or DSP from the AD1847.

The INIT bit is set HI on the last sample before the serial interface is inactivated. The only condition under which the INIT bit is set is when a different sample rate is programmed. If FRS = 0 (32 slots per frame, two samples per frame) and the sample rate is changed in the first sample of the 32 slot frame (i.e., during slots 0 through 15), the INIT bit will be set on the second sample of that frame (i.e., during slots 16 through 31). If FRS = 0 and the sample rate is changed in the second sample of the 32 slot frame, the INIT bit will be set on the second sample of the following frame.

ACI Autocalibrate In-Progress. This bit indicates that autocalibration is in progress or the Mode Change Enable (MCE) state has been recently exited. When exiting the MCE state with the ACAL bit set, the ACI bit will be set HI for 384 sample periods. When exiting the MCE state with the ACAL bit reset, the ACAL bit will be set HI for 128 sample periods, indicating that offset and filter values are being restored. Read by the host CPU or DSP from the AD1847.

0 Autocalibration not in progress

1 Autocalibration is in progress

ACI clear (i.e., reset or LO) should be recognized by first polling for a HI on the sample after the MCE bit is reset, and then polling for a LO. Note that it is important not to start polling until one sample after MCE is reset, because if MCE is set while ACI is HI, an ACI LO on the following sample will suggest a false clear of ACI.

ORL1:0 Overrange Left Detect. These bits indicate the overrange on the left input channel. Read by the host CPU or DSP from the AD1847.

0 Greater than –1.0 dB underrange

1 Between –1.0 dB and 0 dB underrange

2 Between 0 dB and 1.0 dB overrange

3 Greater than 1.0 dB overrange

ORR1:0 Overrange Right Detect. These bits indicate the overrange on the right input channel. Read by the host CPU or DSP from the AD1847.

0 Greater than –1.0 dB underrange

1 Between –1.0 dB and 0 dB underrange

2 Between 0 dB and 1.0 dB overrange

3 Greater than 1.0 dB overrange

ID3:0 AD1847 Revision ID. These four bits define the revision level of the AD1847. The first version of the AD1847 is designated ID = 0001. Read by the host CPU or DSP from the AD1847.

RREQ This bit is reset LO for the Status Word, echoing the RREQ state written by the host CPU or DSP in the previous Control Word. Read by the host CPU or DSP from the AD1847.

res Reserved for future expansion. All reserved bits read zero (LO).

Immediately after reset, the contents of this register is: 0000 0001 0000 0000 (0100h).

Index Readback (16-Bit)

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
CLOR	MCE	RREQ	res	IA3	IA2	IA1	IA0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

DATA7:0 Index Register Data. These bits are the readback data from the desired AD1847 Index Register referenced by the Index Address from the previous Control Word (with the RREQ bit set). Read by the host CPU or DSP from the AD1847.

IA3:0 Index Register Address. These bits echo the indirect address (written during the previous Control Word (with the RREQ bit set) of the desired AD1847 Index Register to be readback. Read by the host CPU or DSP from the AD1847.

RREQ Read Request. This bit is set HI for Index Readback, echoing the RREQ state written by the host CPU or DSP in the previous Control Word. Read by the host CPU or DSP from the AD1847.

res Reserved for future expansion. All reserved bits read zero (LO).

MCE Mode Change Enable. This bit echoes the MCE state written by the host CPU or DSP during the previous* Control Word (with the RREQ bit set). Read by the host CPU or DSP from the AD1847.

CLOR Clear Overrange. This bit echoes the CLOR state written by the host CPU or DSP during the previous Control Word (with the RREQ bit set). Read by the host CPU or DSP from the AD1847.

Immediately after reset, the contents of this register is: 1110 0000 0000 0000 (E000h).

Indirect Mapped Registers

Following in Figure 6 is a table defining the mapping of AD1847 8-bit Index Registers to Index Address. These registers are accessed by writing the appropriate 4-bit Index Address in the Control Word.

Index	Register Name
0	Left Input Control
1	Right Input Control
2	Left Aux #1 Input Control
3	Right Aux #1 Input Control
4	Left Aux #2 Input Control
5	Right Aux #2 Input Control
6	Left DAC Control
7	Right DAC Control
8	Data Format
9	Interface Configuration
10	Pin Control
11	Invalid Address
12	Miscellaneous Information
13	Digital Mix Control
14	Invalid Address
15	Invalid Address

Figure 6. Index Register Mapping

A detailed description of each of the Index Registers is given below.

Left Input Control Register (Index Address 0)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0000	LSS1	LSS0	res	res	LIG3	LIG2	LIG1	LIG0

LIG3:0 Left Input Gain Select. The least significant bit of this 16-level gain select represents +1.5 dB. Maximum gain is +22.5 dB.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

LSS1:0 Left Input Source Select. These bits select the input source for the left gain stage preceding the left ADC.

0 Left Line 1 Source Selected
1 Left Auxiliary 1 Source Selected
2 Left Line 2 Source Selected
3 Left Line 1 Post-Mixed Output Loopback Source Selected

This register's initial state after reset is: 0000 0000 (00h).

Right Input Control Register (Index Address 1)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0001	RSS1	RSS0	res	res	RIG3	RIG2	RIG1	RIG0

RIG3:0 Right Input Gain Select. The least significant bit of this 16-level gain select represents +1.5 dB. Maximum gain is +22.5 dB.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

RSS1:0 Right Input Source Select. These bits select the input source for the right gain stage preceding the right ADC.

0 Right Line 1 Source Selected
1 Right Auxiliary 1 Source Selected
2 Right Line 2 Source Selected
3 Right Line 1 Post-Mixed Output Loopback Source Selected

This register's initial state after reset is: 0000 0000 (00h).

Left Auxiliary #1 Input Control Register (Index Address 2)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0010	LMX1	res	res	LX1G4	LX1G3	LX1G2	LX1G1	LX1G0

LX1G4:0 Left Auxiliary Input #1 Gain Select. The least significant bit of this 32-level gain/attenuate select represents -1.5 dB. LX1G4:0 = 0 produces a +12 dB gain. LX1G4:0 = "01000" (8 decimal) produces 0 dB gain. Maximum attenuation is -34.5 dB. Gains referred to 2.0 V p-p full-scale output level.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

LMX1 Left Auxiliary #1 Mute. This bit, when set HI, will mute the left channel of the Auxiliary #1 input source. This bit is set HI after reset.

This register's initial state after reset is: 1000 0000 (80h).

Right Auxiliary #1 Input Control Register (Index Address 3)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0011	RMX1	res	res	RX1G4	RX1G3	RX1G2	RX1G1	RX1G0

RX1G4:0 Right Auxiliary Input #1 Gain Select. The least significant bit of this 32-level gain/attenuate select represents -1.5 dB. RX1G4:0 = 0 produces a +12 dB gain. RX1G4:0 = "01000" (8 decimal) produces 0 dB gain. Maximum attenuation is -34.5 dB. Gains referred to 2.0 V p-p full-scale output level.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

RMX1 Right Auxiliary #1 Mute. This bit, when set to HI, will mute the right channel of the Auxiliary #1 input source. This bit is set to HI after reset.

This register's initial state after reset is: 1000 0000 (80h).

Left Auxiliary #2 Input Control Register (Index Address 4)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0100	LMX2	res	res	LX2G4	LX2G3	LX2G2	LX2G1	LX2G0

LX2G4:0 Left Auxiliary #2 Gain Select. The least significant bit of this 32-level gain/attenuate select represents –1.5 dB. LX2G4:0 = 0 produces a +12 dB gain. LX2G4:0 = "01000" (8 decimal) produces 0 dB gain. Maximum attenuation is –34.5 dB. Gains referred to 2.0 V p-p full-scale output level.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

LMX2 Left Auxiliary #2 Mute. This bit, when set HI, will mute the left channel of the Auxiliary #2 input source. This bit is HI after reset.

This register's initial state after reset is: 1000 0000 (80h).

Right Auxiliary #2 Input Control Register (Index Address 5)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0101	RMX2	res	res	RX2G4	RX2G3	RX2G2	RX2G1	RX2G0

RX2G4:0 Right Auxiliary #2 Gain Select. The least significant bit of this 32-level gain/attenuate select represents –1.5 dB. RX2G4:0 = 0 produces a +12 dB gain. RX2G4:0 = "01000" (8 decimal) produces 0 dB gain. Maximum attenuation is –34.5 dB. Gains referred to 2.0 V p-p full-scale output level.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

RMX2 Right Auxiliary #2 Mute. This bit, when set HI, will mute the right channel of the Auxiliary #2 input source. This bit is HI after reset.

This register's initial state after reset is: 1000 0000 (80h).

Left DAC Control Register (Index Address 6)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0110	LDM	res	LDA5	LDA4	LDA3	LDA2	LDA1	LDA0

LDA5:0 Left DAC Attenuate Select. The least significant bit of this 64-level attenuate select represents –1.5 dB. LDA5:0 = 0 produces a 0 dB attenuation. Maximum attenuation is –94.5 dB.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

LDM Left DAC Mute. This bit, when set HI, will mute the left channel output. Auxiliary inputs are muted independently with the Left Auxiliary Input Control Registers. This bit is HI after reset.

This register's initial state after reset is: 1000 0000 (80h).

Right DAC Control Register (Index Address 7)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
0111	RDM	res	RDA5	RDA4	RDA3	RDA2	RDA1	RDA0

RDA5:0 Right DAC Attenuate Select. The least significant bit of this 64-level attenuate select represents –1.5 dB. RDA5:0 = 0 produces a 0 dB attenuation. Maximum attenuation must be at least –94.5 dB.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

RDM Right DAC Mute. This bit, when set HI, will mute the right DAC output. Auxiliary inputs are muted independently with the Right Auxiliary Input Control Registers. This bit is HI after reset.

This register's initial state after reset is: 1000 0000 (80h).

Data Format Register (Index Address 8)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1000	res	FMT	C/L	S/M	CFS2	CFS1	CFS0	CSL

The contents of this register can NOT be changed except when the AD1847 is in the Mode Change Enable (MCE) state (i.e., the MCE bit in the Control Word is HI). Write attempts to this register when the AD1847 is not in the MCE state will not be successful.

CSL Clock Source Select. This bit selects the clock source to be used for the audio sample rate.

0 XTAL1 (24.576 MHz)

1 XTAL2 (16.9344 MHz)

CFS2:0 Clock Frequency Divide Select. These bits select the audio sample rate frequency. The audio sample rate depends on which clock source is selected and the frequency of the clock source.

CFS2:0	Divide Factor	XTAL1 24.576 MHz	XTAL2 16.9344 MHz
0	3072	8.0 kHz	5.5125 kHz
1	1536	16.0 kHz	11.025 kHz
2	896	27.42857 kHz	18.9 kHz
3	768	32.0 kHz	22.05 kHz
4	448	Not Supported	37.8 kHz
5	384	Not Supported	44.1 kHz
6	512	48.0 kHz	33.075 kHz
7	2560	9.6 kHz	6.615 kHz

Note that the AD1847's internal oscillators can be overdriven by external clock sources at the crystal inputs. This is the configuration used by serial bus slave codecs in daisy-chained multiple codec systems. If an external clock source is applied, it will be divided down by the selected Divide Factor. The external clock need not be at the recommended crystal frequencies.

S/M Stereo/Mono Select. This bit determines how the audio data streams are formatted. Selecting stereo will result with alternating samples representing left and right audio channels. Mono playback plays the same audio sample on both channels. Mono capture only captures data from the left audio channel.

0 Mono

1 Stereo

C/L Companded/Linear Select. This bit selects between a linear digital representation of the audio signal or a nonlinear, companded format for all input and output data. The type of linear PCM or the type of companded format is defined by the FMT bits.

0 Linear PCM

1 Companded

FMT Format Select. This bit defines the format for all digital audio input and output based on the state of the C/L bit.

	Linear PCM (C/L = 0)	Companded (C/L = 1)
0	8-bit unsigned linear PCM	8-bit μ-law companded
1	16-bit signed linear PCM	8-bit A-law companded

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

This register's initial state after reset is: 0000 0000 (00h).

Interface Configuration Register (Index Address 9)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1001	res	res	res	res	ACAL	res	res	PEN

PEN Playback Enable. This bit will enable the playback of data in the format selected. PEN may be set and reset without setting the MCE bit.

0 Playback Disabled
1 Playback Enabled

ACAL Autocalibrate Enable. This bit determines whether the AD1847 performs an autocalibrate when exiting from the Mode Change Enable (MCE) state. If the ACAL bit is not set, the previous autocalibration values are used when returning from the Mode Change Enable (MCE) state and no autocalibration takes place. Autocalibration must be preformed after initial power-up for proper operation. This bit is HI after reset.

0 No autocalibration
1 Autocalibration allowed

NOTE: The ACAL bit can only be changed when the AD1847 is in the Mode Change Enable (MCE) state.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

This register's initial state after reset is: 0000 1000 (08h).

Pin Control Register (Index Address 10)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1010	XCTL1	XCTL0	CLKTS	res	res	res	res	res

CLKTS Clock Three-State. If the BM bit is HI, and the CLKTS bit is HI, then the CLKOUT pin will be three-stated. If the BM bit is HI, and the bit CLKTS is LO, then the CLKOUT pin is not three-stated. If the BM bit is LO, then the CLKOUT pin is always three-stated.

XCTL1:0 External Control. The state of these independent bits is reflected on the respective XCTL1 and XCTL0 pins of the AD1847.

0 TTL logic LO on XCTL1, XCTL0 pins
1 TTL logic HI on XCTL1, XCTL0 pins

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

This register's initial state after reset is: 0000 0000 (00h).

Invalid Address (Index Address 11)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1011	inval	inval	inval	inval	inval	inval	inval	inval

inval Writes to this index address are ignored. Index readback of this index address will return the Status Word.

Miscellaneous Information Register (Index Address 12)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1100	FRS	TSSEL	res	res	res	res	res	res

The Miscellaneous Information Register can only be changed when the AD1847 is in the Mode Change Enable (MCE) state. Changes to this register are updated at the next Serial Data Frame Sync (SDFS) boundary. If FRS is LO (i.e., 32 slots per frame), and either TSSEL or FRS change in the first sample of a frame, the change is not updated at the second sample of the same frame, but at the first sample of the next frame.

TSSEL Transmit Slot Select. This bit determines which TDM time slots the AD1847 should transmit on.
- 0 Transmit on time slots 3, 4 and 5. Used when SDI and SDO are tied together (i.e., "1-wire" system).
- 1 Transmit on slots 0, 1 and 2. Used when SDI and SDO are independent inputs and outputs (i.e., "2-wire" system).

FRS Frame Size. This bit selects the number of time slots per frame.
- 0 Selects 32 slots per frame (two samples per frame sync or frame sync at half the sample rate).
- 1 Selects 16 slots per frame (one sample per frame sync or frame sync at the sample rate).

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

This register's initial state after reset is: 0000 0000 (00h).

Digital Mix Control Register (Index Address 13)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1101	DMA5	DMA4	DMA3	DMA2	DMA1	DMA0	res	DME

DME Digital Mix Enable. This bit enables the digital mix of the ADCs' output with the DACs' input. When enabled, the data from the ADCs is digitally mixed with other data being delivered to the DACs (regardless of whether or not playback [PEN] is enabled, i.e., set). If there is a capture overrun, then the last sample captured before overrun will be used for the digital mix. If playback is enabled (PEN set) and there is a playback underrun, then a midscale zero will be added to the digital mix data.
- 0 Digital mix disabled (muted)
- 1 Digital mix enabled

DMA5:0 Digital Mix Attenuation. These bits determine the attenuation of the ADC output data mixed with the DAC input data. The least significant bit of this 64-level attenuate select represents –1.5 dB. Maximum attenuation is –94.5 dB.

res Reserved for future expansion. Write zeros (LO) to all reserved bits.

This register's initial state after reset is: 0000 0000 (00h).

Invalid Address (Index Address 14)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1110	inval	inval	inval	inval	inval	inval	inval	inval

inval Writes to this index address are ignored. Index readback of this index address will return the Status Word.

Invalid Address (Index Address 15)

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0
1111	inval	inval	inval	inval	inval	inval	inval	inval

inval Writes to this index address are ignored. Index readback of this index address will return the Status Word.

Serial Data Interface

The AD1847 serial data interface uses a Time Division Multiplex (TDM) scheme that is compatible with DSP serial ports configured in Multi-Channel Mode with either 32 or 16 16-bit time slots. An AD1847 is always the serial bus master, transmitting the serial clock (SCLK) and the serial data frame sync (SDFS). The AD1847 always receives control and playback data in time slots 0, 1 and 2. The AD1847 will transmit status or index register readback and capture data in time slots 0, 1 and 2 if TSSEL = 1, and will transmit status or index register readback and capture data in time slots 3, 4 and 5 if TSSEL = 0. The following table in Figure 7 shows an example of how the time slots might be assigned.

In this example design, which uses the ADSP-21xx DSP, each frame is divided into 32 time slots of 16-bits each (FRS = 0). Two audio samples are contained in the 32 time slots, with a single frame sync (SDFS) at the beginning of the frame. The ADSP-21xx serial port (SPORT0) supports 32 time slots. The format of the first 16 time slots (sample N) is the same as the format of the second 16 time slots (sample N+1). In this example, 24 time slots are used, as indicated below. Note that time slots 12 through 15 and 28 through 31 are unused in this example, and that Figure 7 presumes that TSSEL = 0 ("1-wire" system).

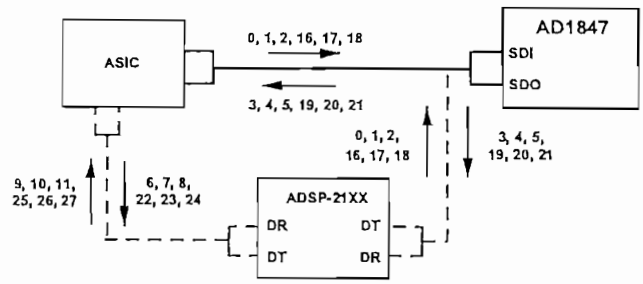
Slot Number	Source	Destination	Format
0, 16	ASIC	AD1847	AD1847 Control Word
1, 17			Left Playback Data
2, 18			Right Playback Data
3, 19	AD1847	ASIC	AD1847 Status Word/ Index Readback
4, 20			Left Capture Data
5, 21			Right Capture Data
0, 16	DSP	AD1847	AD1847 Control Word
1, 17			Left Playback Data
2, 18			Right Playback Data
3, 19	AD1847	DSP	AD1847 Status Word/ Index Readback
4, 20			Left Capture Data
5, 21			Right Capture Data
6, 22	ASIC	DSP	DSP Control
7, 23			Left Processed Playback Data
8, 24			Right Processed Playback Data
9, 25	DSP	ASIC	DSP Status
10, 26			Left Processed Capture Data
11, 27			Right Processed Capture Data

Figure 7. Time Slot Assignment Example

Note that in this "1-wire" system example, the Digital Signal Processor (DSP) and ISA Bus Interface ASIC (ASIC) use the same slots to communicate to the AD1847. This reduces the number of total time slots required and eliminates the need for the AD1847 to distinguish between DSP data and ASIC data. Also, in this example the ASIC and the DSP do not send data to the AD1847 at the same time, so separate slots are unnecessary.

The digital data in the serial interface is pipelined up to 2 samples deep. This pipelining is required to properly resolve the interface between the relatively fast fixed SCLK rate, and the relatively slow sample rates (and therefore frame sync rates) at which the AD1847 is capable of running. At low sample rates, two samples of data can be serviced in a fraction of a sample period. For example, at an 8 kHz sample rate, 32 time slots only consume 32 × 16 × (1/12.288 MHz) = 41.67 µs out of a 125 µs period. The two-deep data pipeline thus allows sample overrun (capture) and sample underrun (playback) to be avoided.

Figure 8 represents a logical view of the slot utilization between devices.



NOTE: DSP MUST HAVE TWO SERIAL PORTS

Figure 8. Time Slot Allocation Example

Note that this is a system specific 1-wire example. For non-DSP operation, the DSP is either not present or disabled. If the DSP is present, the ASIC configures the DSP through slot 6 (and slot 22) to three-state its outputs in time slots 0, 1 and 2 (and slots 16, 17 and 18). The ASIC can then enable its drivers for time slots 0, 1 and 2 (and slots 16, 17 and 18). For DSP operation, the ASIC three-states its outputs for time slots 0, 1 and 2 (and slots 16, 17 and 18) and enables the DSP drivers for slots 0, 1 and 2 (and slots 16, 17, and 18).

An application note is available from Analog Devices with additional information on interfacing to the AD1847 serial port. This application note can be obtained through your local Analog Devices representative, or downloaded from the DSP Bulletin Board Service at (617) 461-4258 (8 data bits, no parity, 1 stop bit, 300/1200/2400/4600 baud).

Control Word

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
CLOR	MCE	RREQ	res	IA3	IA2	IA1	IA0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

Left Playback Data

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

Right Playback Data

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

Status Word

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
res	res	RREQ	res	ID3	ID2	ID1	ID0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
res	res	ORR1	ORR0	ORL1	ORL0	ACI	INIT

Index Readback

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
CLOR	MCE	RREQ	res	IA3	IA2	IA1	IA0
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

Left Capture Data

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA 1	DATA0

Right Capture Data

Data 15	Data 14	Data 13	Data 12	Data 11	Data 10	Data 9	Data 8
DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Data 7	**Data 6**	**Data 5**	**Data 4**	**Data 3**	**Data 2**	**Data 1**	**Data 0**
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

IA3:0	Data 7	Data 6	Data 5	Data 4	Data 3	Data 2	Data 1	Data 0	Index
0000	LSS1	LSS0	res	res	LIG3	LIG2	LIG1	LIG0	0
0001	RSS1	RSS0	res	res	RIG3	RIG2	RIG1	RIG0	1
0010	LMX1	res	res	LX1G4	LX1G3	LX1G2	LX1G1	LX1G0	2
0011	RMX1	res	res	RX1G4	RX1G3	RX1G2	RX1G1	RX1G0	3
0100	LMX2	res	res	LX2G4	LX2G3	LX2G2	LX2G1	LX2G0	4
0101	RMX2	res	res	RX2G4	RX2G3	RX2G2	RX2G1	RX2G0	5
0110	LDM	res	LDA5	LDA4	LDA3	LDA2	LDA1	LDA0	6
0111	RDM	res	RDA5	RDA4	RDA3	RDA2	RDA1	RDA0	7
1000	res	FMT	C/L	S/M	CFS2	CFS1	CFS0	CSL	8
1001	res	res	res	res	ACAL	res	res	PEN	9
1010	XCTL1	XCTL0	CLKTS	res	res	res	res	res	10
1011	inval	inval	inval	inval	inval	inval	inval	inval	11
1100	FRS	TSSEL	res	res	res	res	res	res	12
1101	DMA5	DMA4	DMA3	DMA2	DMA1	DMA0	res	DME	13
1110	inval	inval	inval	inval	inval	inval	inval	inval	14
1111	inval	inval	inval	inval	inval	inval	inval	inval	15

Figure 9. Register Map Summary

Control Register Mapping Summary
A detailed map of the control register bit assignments is summarized for reference in Figure 9.

Daisy-Chained Multiple Codecs
Multiple AD1847s can be configured in a daisy-chain system with a single master Codec and one or more slave Codecs. Codecs in a daisy-chained configuration are synchronized at the sample level.

The master and slave AD1847s should be powered-up together. If this is not possible, the slave(s) should power-up before the master Codec, such that the slave(s) are ready when the master starts to drive the serial interface, and a serial data frame sync (SDFS) can synchronize the master and slave(s).

The sample rate for the master and slave(s) should be programmed together. If this is not possible, the slave(s) should be programmed before the master AD1847. A slave AD1847 enters a time-out period after a new sample rate has been selected. During this time-out period, a slave will ignore any activity on the SDFS signal (i.e., frame syncs). There is no software means to determine when a slave has exited from this time-out period and is ready to respond to frame syncs. However, as long as the AD1847 master is driving the serial interface, a frame sync will not occur before the slave Codec(s) are ready.

Note that the time slots for all slave AD1847s must be assigned to those slots which immediately follow the time slots consumed by the master AD1847 so that the TSO (Time Slot Output)/TSI (Time Slot Input) signaling operates properly. For example, in a 2-wire system with one master and one slave, the time slot assignment should be 0, 1, 2 (16, 17, 18) for the master AD1847, and 3, 4, 5 (19, 20, 21) for the slave AD1847.

Figure 10 illustrates the connection between master and slave(s) in a daisy-chained, multiple Codec system. Note that the TSI pin of the master Codec should be tied to digital ground. The XTAL1I pin of the slaves should be connected to digital ground, and XTAL1O pin should be left unconnected, while the XTAL2I pin should be connected to the CLKOUT pin of the AD1847 master, and the XTAL2O pin generates a driven version of the CLKOUT signal applied to the XTAL2I pin.

INITIALIZATION AND PROCEDURES

Reset and Power Down
A total reset of the AD1847 is defined as any event which requires both the digital and analog section of the AD1847 to return to a known and stable state. Total reset mode, as well as power down, occurs when the $\overline{\text{PWRDOWN}}$ pin of the AD1847 has been asserted low for minimum power consumption. When the $\overline{\text{PWRDOWN}}$ signal is deasserted, the AD1847 must be calibrated by setting the ACAL bit and exiting from the Mode Change Enable (MCE) state.

The reset occurs, and only resets the digital section of the AD1847, when the $\overline{\text{RESET}}$ pin of the AD1847 has been asserted LO to initialize all registers to known values. See the register definitions for the exact values initialized. The register reset defaults include TSSEL = 0 (1-wire system) and FRS = 0 (32 slots per frame). If the target application requires a 2-wire system design or 16 slots per frame, the AD1847 can be bootstrapped into these configurations.

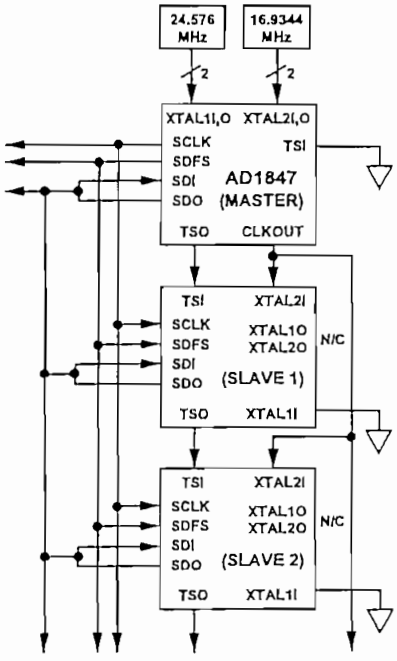


Figure 10a. One-Wire Daisy-Chained Codec Interconnect

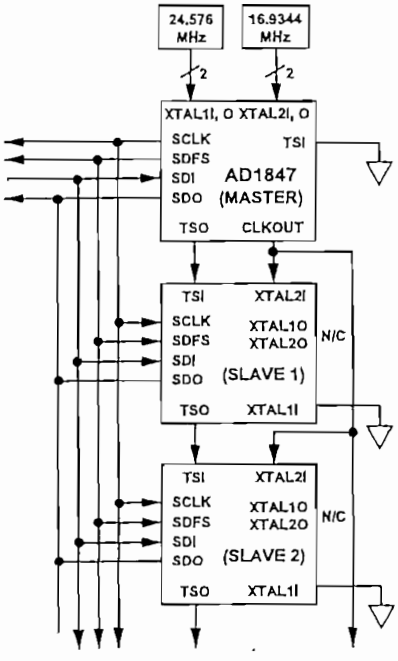


Figure 10b. Two-Wire Daisy-Chained Codec Interconnect

To bootstrap into TSSEL = 1 (i.e., 2-wire system design), the host CPU or DSP must transmit to the AD1847 in slot 0 a Control Word with the MCE bit set HI, IA3:0 = "1100" to address the Miscellaneous Information Index Register, and DATA7:0 = "X100 000" to set the TSSEL bit HI. To bootstrap into FRS = 1 (i.e., 16 slots per frame), the host CPU or DSP must transmit to the AD1847 in slot 0 a Control Word with the MCE bit set HI, IA3:0 = "1100" to address the Miscellaneous Information Index Register, and DATA7:0 = "1X00 0000" to set the FRS bit HI.

The host CPU or DSP must maintain the MCE bit set HI in slot 16, which is the Control Word of the second sample of the frame, so that the AD1847 does not initiate autocalibration prematurely. At the next frame sync, the AD1847 will be reconfigured.

The AD1847 must be reset after power up. When the $\overline{\text{RESET}}$ signal is deasserted, the AD1847 will autocalibrate when the MCE bit is reset LO (i.e., when exiting the Mode Change Enable state) only if the ACAL bit is set. If the ACAL bit is not set, the previous autocalibration values will be used.

The AD1847 will not function properly unless an autocalibration is performed after power up.

During power down, the serial port digital output pins and the analog output pins take the following states:

SCLK–LO if BM is HI (i.e., bus master), input pin if BM is LO (i.e., bus slave)
SDFS–LO if BM is HI, input pin if BM is LO
SDO–three-state
TSO–three-state
CLKOUT–LO if BM HI, three-state if BM is LO
V_{REF}–pulled to analog ground
L_OUT, R_ OUT– pulled to analog ground

Clock Connections and Clock Rates

When the AD1847 is configured as a bus slave (BM = LO), the XTAL1I pin should be connected to digital ground, and the XTAL2I pin should be tied to the CLKOUT of the AD1847 bus master. The XTAL1O and the XTAL2O pins should be left unconnected. When the AD1847 is configured as a bus master (BM = HI), the XTAL1I and the XTAL1O pin should be connected to a 24.576 MHz crystal, and the XTAL2I and XTAL2O pin should be connected to a 16.9344 MHz crystal.

When XTAL1 is selected (by resetting the CSL bit LO in the Data Format Register) as the clock source, the SCLK pin will generated a serial clock at 12.288 MHz (or one half of the crystal frequency applied at XTAL1), and the CLKOUT pin will also generate a clock output at 12.288 MHz when the AD1847 is in bus master mode (BM = HI). When XTAL2 is selected (by setting the CSL bit HI in the Data Format Register) as the clock source, the SCLK pin will generate a serial clock at 11.2896 MHz (or two thirds of the crystal frequency applied at XTAL2), and the CLKOUT pin will generate a clock output at 16.9344 MHz when the AD1847 is in bus master mode (BM = HI). The CLKOUT pin will be three-stated when the AD1847 is placed in bus slave mode (BM = LO).

When the selected frame size is 32 slots per frame (by resetting the FRS bit LO in the Miscellaneous Information Register), the SDFS pin will generate a serial data frame sync at the frequency of the selected sample rate divided by two, when the AD1847 is in bus master mode (BM = HI). When the selected frame size is 16 slots per frame (by setting the FRS bit HI in the Miscellaneous Information Register), the SDFS pin will generate a serial data frame sync at the frequency of the selected sample rate, when the AD1847 is in bus master mode (BM = HI).

When the AD1847 is in bus slave mode (BM = LO), the TSI pin should be connected to the TSO pin of the AD1847 master or slave which has been assigned to the preceding time slots. The signal on the TSO pin is essentially the signal received on the TSI pin, but delayed by 3 or 6 time slots from TSI (depending on the state of TSSEL). The frequency of the transitions on the TSI and TSO lines is equivalent to the frequency on the SDFS pin.

When the AD1847 is in bus master mode (BM = HI), the TSI pin should be connected to digital ground. The signal on the TSO pin is essentially the same as the signal output on the SDFS pin, but delayed by 3 or 6 time slots from SDFS (again, depending on the state of TSSEL).

Mode Change Enable State

The AD1847 must be in the Mode Change Enable (MCE) state before any changes to the ACAL bit of the Interface Configuration Register, the Data Format Register, or the Miscellaneous Information Register are allowed. Note that the MCE bit does not have to be reset LO in order for changes to take effect.

Digital Mix

Digital mix is enabled via the DME bit in the Digital Mix Control Register. The digital mix routes the digital data from the ADCs to the DACs. The mix can be digitally attenuated via bits also in the Digital Mix Control Register. The ADC data is summed with the DAC data supplied at the digital bus interface. When digital mix is enabled and the PEN bit is not set, ADC data is summed with zeros to produce the DAC output.

If the sum of the digital mix (ADC output and DAC input from the serial bus interface) is greater than full scale, the AD1847 will send a positive or negative full scale value to the DACs, whichever is appropriate (clipping).

Autocalibration

The AD1847 has the ability to calibrate its ADCs and DACs for greater accuracy by minimizing dc offsets. Autocalibration occurs whenever the AD1847 exits from the Mode Change Enable (MCE) state AND the ACAL bit in the Interface Configuration Register has been set.

The completion of the autocalibration sequence can be determined by polling the Autocalibration In-Progress (ACI) bit in the Status Word. This bit will be HI while the autocalibration is in progress and LO once autocalibration has completed. The autocalibration sequence will take at least 384 sample periods.

The autocalibration procedure is as follows:

1. Mute both left and right AUX1 and AUX2 inputs via the Left Auxiliary Input and Right Auxiliary Input Control Registers.
2. Place the AD1847 in the Mode Change Enable (MCE) state using the MCE bit of the AD1847 Control Word. Set the ACAL bit in the Interface Configuration Register.
3. Exit from the Mode Change Enable state by resetting the MCE bit.
4. Poll the ACI bit in the AD1847 Status Word for a HI (autocalibration in progress), then poll the ACI bit for a LO (autocalibration complete).
5. Unmute the AUX inputs, if used.

If ACAL is not set, the AD1847 is muted for 128 sample periods after resetting the MCE bit, and the ACI bit in the Status Word is set HI during this 128 sample periods. Autocalibration must be performed after power-up to ensure proper operation of the AD1847.

Exiting from the MCE state always causes ACI to go HI. If the ACAL bit is set when MCE state is exited, then the ACI bit will be HI for 384 sample periods. If the ACAL bit is reset when MCE is exited, then the ACI bit will be HI for 128 sample periods.

Changing Sample Rates

The internal states of the AD1847 are synchronized by the selected sample frequency defined in the Data Format Register. The changing of either the clock source or the clock frequency divide requires a special sequence for proper AD1847 operation.

1. Mute the outputs of the AD1847 and enter the Mode Change Enable (MCE) state by setting the MCE bit of the AD1847 Control Word.
2. During a single atomic or nondivisible write cycle, change the Clock Frequency Divide Select (CFS) and/or the Clock Source Select (CSL) bits of the Data Format Register to the desired values. CFS and CSL can be programmed in the same Control Word as MCE.
3. The INIT bit in the Status Word will be set HI at the last sample of the next frame to indicate that the serial port will be disabled for a timeout period.
4. The AD1847 requires a period of time to resynchronize its internal states to the newly selected clock. During this time, the AD1847 will be unable to respond at its serial interface port (i.e., no frame syncs will be generated). The time-out period is $2^{21} \times \text{SCLK} \approx 170$ ms after power-up, and ≈ 5 ms for subsequent changes of sample rate.
5. Exit the Mode Change Enable state by resetting the MCE bit. Upon exiting the MCE state, an autocalibration of duration 384 sample periods or an output mute of duration 128 sample periods occurs, depending on the state of the ACAL bit.
6. Poll the ACI bit in the AD1847 Status Word for a HI (indicating that autocalibration is in progress) then poll the ACI bit for a LO (indicating that autocalibration has completed). Once the ACI bit has been read back LO, normal operation of the Codec can resume.

The CSL and CFS bits cannot be changed unless the AD1847 is in the Mode Change Enable state (i.e., the MCE bit in the AD1847 Control Word is set). Attempts to change the contents of the Data Format Register without MCE set will result in the write cycle not being recognized (the bits will not be updated).

The MCE bit should not be reset until after the INIT bit in the AD1847 Status Word is detected HI. After the INIT bit is detected HI, the serial port is disabled. When the next frame sync arrives (after the time-out period), all internal clocks are stable and the serial port is ready for normal operation.

DATA FORMAT DEFINITIONS

There are four data formats supported by the AD1847: 16-bit signed, 8-bit unsigned, 8-bit companded μ-law, and 8-bit companded A-law. The AD1847 supports these four formats because each of them have found wide use in important applications.

16-Bit Signed Format

The 16-bit signed format (also called 16-bit twos-complement) is the standard method of representing 16-bit digital audio. This format yields 96 dB of dynamic range and is common in consumer compact disk audio players. This format uses the value – 32768 (8000h) to represent minimum analog amplitude while 32767 (7FFFh) represents maximum analog amplitude. Intermediate values are a linear interpolation between minimum and maximum amplitude values.

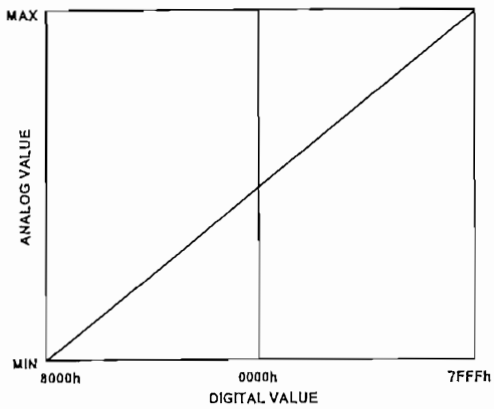


Figure 11. 16-Bit Signed Format

8-Bit Unsigned Format

The 8-bit unsigned format is commonly used in the personal computer industry. This format delivers 48 dB of dynamic range. The value 0 (00h) is used to represent minimum analog amplitude while 255 (FFh) is used to represent maximum analog amplitude. Intermediate values are a linear interpolation between minimum and maximum amplitude values. The least significant byte of the 16-bit internal data is truncated to create the 8-bit output samples.

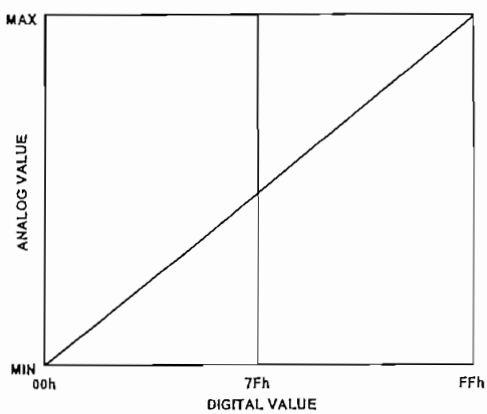


Figure 12. 8-Bit Unsigned Format

8-Bit Companded Formats

The 8-bit companded formats (μ-law and A-law) are used in the telecommunications industry. Both of these formats are used in ISDN communications and workstations; μ-law is the standard for the United States and Japan while A-law is used in Europe. Companded audio allows either 64 dB or 72 dB of dynamic range using only 8-bits per sample. This is accomplished using a nonlinear formula which assigns more digital codes to lower amplitude analog signals at the expense of resolution of higher amplitude signals. The μ-law format of the AD1847 conforms to the Bell System $\mu = 255$ companding law while the A-law format conforms to CCITT "A" law models. Figure 13 shows approximately how both the μ-law and A-law companding schemes behave. Refer to the standards mentioned above for an exact definition.

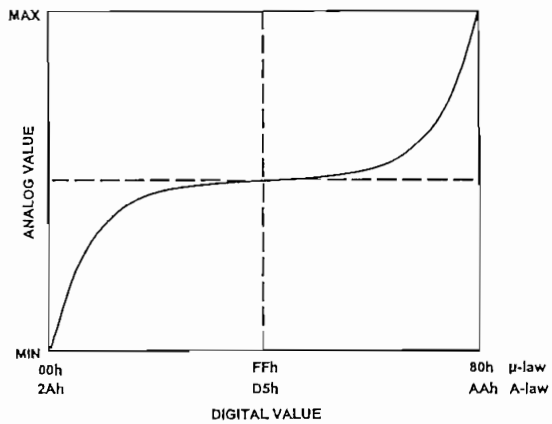


Figure 13. 8-Bit Companded Format

APPLICATIONS CIRCUITS

The AD1847 Stereo Codec has been designed to require a minimum of external circuitry. The recommended circuits are shown in Figures 14 through 22. Analog Devices estimates that the total cost of all the components shown in these Figures, including crystals, to be less than $3 in 10,000 quantities.

Industry-standard compact disc "line-levels" are 2 V_{rms} centered around analog ground. (For other audio equipment, "line level" is much more loosely defined.) The AD1847 SoundPort is a +5 V only powered device. Line level voltage swings for the AD1847 are defined to be 1 V_{rms} for a sine wave ADC input and 0.707 V_{rms} for a sine wave DAC output. Thus, 2 V_{rms} input analog signals must be attenuated and either centered around the reference voltage intermediate between 0 V and +5 V or ac-coupled. The V_{REF} pin will be at this intermediate voltage, nominally 2.25 V. It has limited drive but can be used as a voltage datum to an op amp input. Note, however, that dc-coupled inputs are not recommended, as they provide no performance benefits with the AD1847 architecture. Furthermore, dc offset differences between multiple dc-coupled inputs create the potential for "clicks" when changing the input mux selection.

Circuits for 2 V_{rms} line-level inputs and auxiliaries are shown in Figure 14 and Figure 15. Note that these are divide-by-two resistive dividers. The input resistor and 560 pF (1000 pF) capacitor provide the single-pole of antialias filtering required for the ADCs. If line-level inputs are already at the 1 V_{rms} levels expected by the AD1847, the resistors in parallel with the 560 pF (1000 pF) capacitors can be omitted. If the application does not route the AUX2 inputs to the ADCs, then no antialias filtering is required (only the 1 µF ac coupling capacitor).

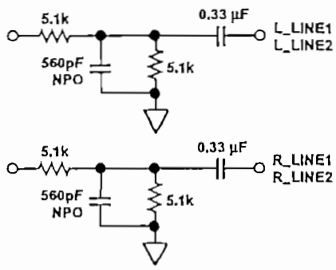


Figure 14. 2 V_{rms} Line-Level Input Circuit for Line Inputs

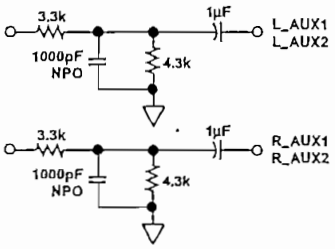


Figure 15. 2 V_{rms} Line-Level Input Circuit for AUX Inputs

Figure 16 illustrates one example of how an electret condenser microphone requiring phantom power could be connected to the AD1847. V_{REF} is shown buffered by an op amp; a transistor like a 2N4124 will also work well for this purpose. Note that if a battery-powered microphone is used, the buffer and R2s are not needed. The values of R1, R2, and C should be chosen in light of the mic characteristics and intended gain. Typical values for these might be R1 = 20 kΩ, R2 = 2 kΩ, and C = 220 pF.

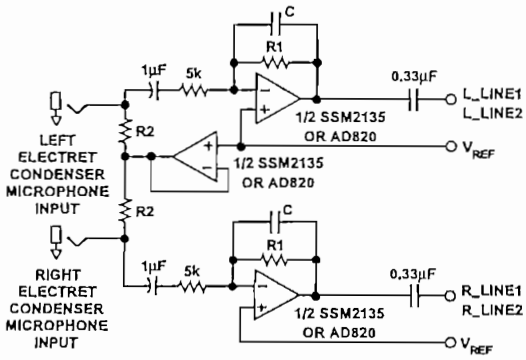


Figure 16. "Phantom-Powered" Microphone Input Circuit

Figure 17 shows ac-coupled line outputs. The resistors are used to center the output signals around analog ground. If dc-coupling is desired, V_{REF} could be used with op amps as mentioned previously.

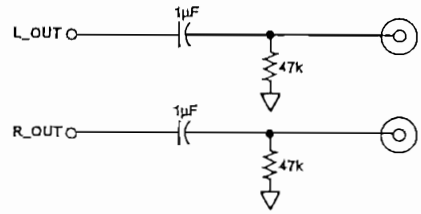


Figure 17. Line Output Connections

A circuit for headphone drive is illustrated in Figure 18. Drive is supplied by +5 V operational amps. The circuit shown ac couples the headphones to the line output.

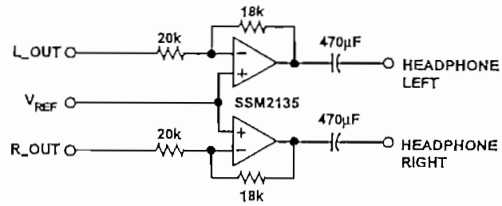


Figure 18. Headphone Drive Connections

Figure 19 illustrates reference bypassing. V_{REFI} should only be connected to its bypass capacitors.

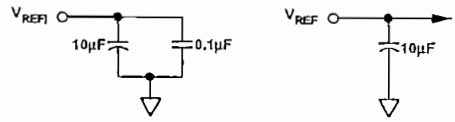


Figure 19. Voltage Reference Bypassing

Figure 20 illustrates signal-path filtering capacitors, L_FILT and R_FILT. The AD1847 must use 1.0 µF capacitors. The 1.0 µF capacitors required by the AD1847 can be of any type.

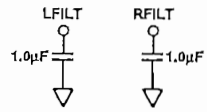


Figure 20. External Filter Capacitor Connections

The crystals shown in the crystal connection circuitry of Figure 21 should be fundamental-mode and parallel-tuned. Two sources for the exact crystals specified are Component Marketing Services in Massachusetts, U.S. at 617/762-4339 and Cardinal Components in New Jersey, U.S. at 201/746-0333. Note that using the exact data sheet frequencies is not required and that external clock sources can be used to overdrive the AD1847s internal oscillators. (See the description of the CFS2:0 control bits above.) If using an external clock source, apply it to the crystal input pins while leaving the crystal output pins unconnected. Attention should be paid to providing low-jitter external input clocks .

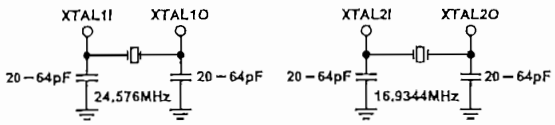


Figure 21. Crystal Connections

Analog Devices also recommends a pull-down resistor on the $\overline{\text{PWRDOWN}}$ signal.

Good, standard engineering practices should be applied for power-supply decoupling. Decoupling capacitors should be placed as close as possible to package pins. If a separate analog power supply is not available, the circuit shown in Figure 22 is recommended when using a single +5 V supply. Ferrite beads suffice for the inductors shown. This circuitry should be as close to the supply pins as is practical.

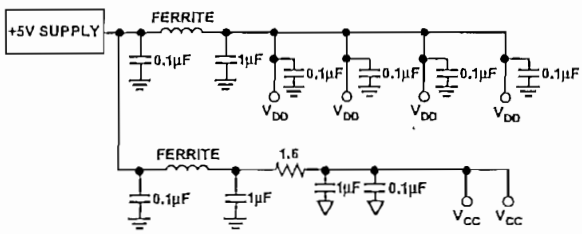


Figure 22. Recommended Power Supply Bypassing

Analog Devices recommends a split ground plane as shown in Figure 23. The analog plane and the digital plane are connected directly under the AD1847. Splitting the ground plane directly under the SoundPort Codec is optimal because analog pins will be located directly above the analog ground plane and digital pins will be located directly above the digital ground plane for the best isolation. The digital and analog grounds should be tied together in the vicinity of the AD1847. Other schemes may also yield satisfactory results. If the split ground plane recommended here is not possible, the AD1847 should be entirely over the analog ground plane with the ASIC and DSP over the digital plane.

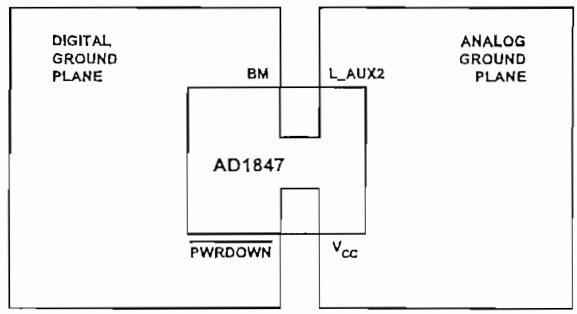


Figure 23. Recommended Ground Plane

FREQUENCY RESPONSE PLOTS

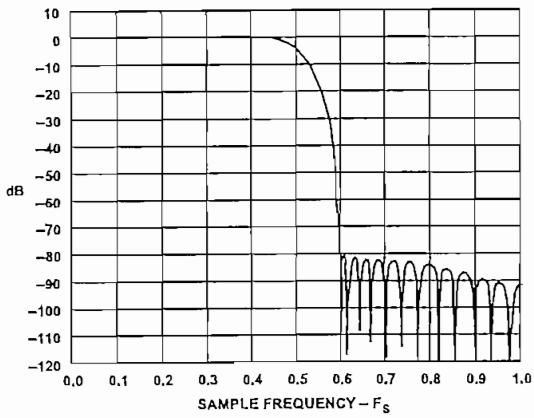


Figure 24. AD1847 Analog-to-Digital Frequency Response (Full-Scale Line-Level Inputs, 0 dB Gain)

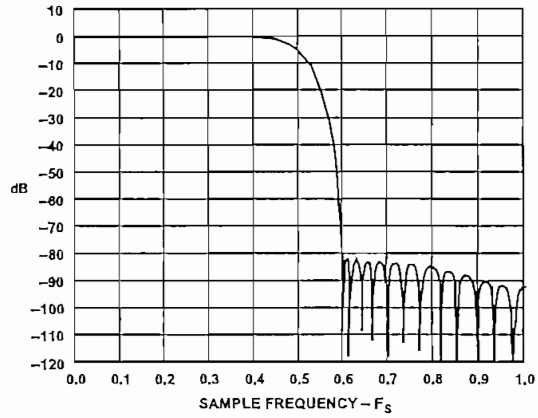


Figure 26. AD1847 Digital-to-Analog Frequency Response (Full-Scale Inputs, 0 dB Attenuation)

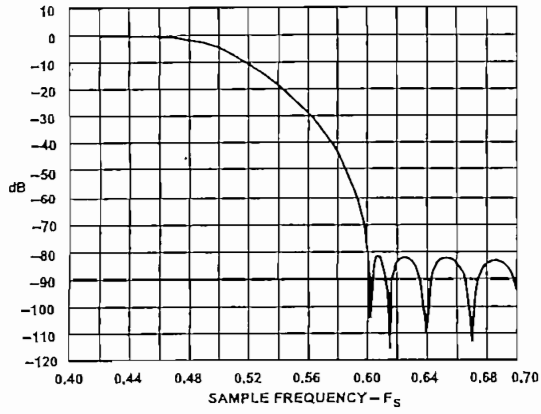


Figure 25. AD1847 Analog-to-Digital Frequency Response –Transition Band (Full-Scale Line-Level Inputs, 0 dB Gain)

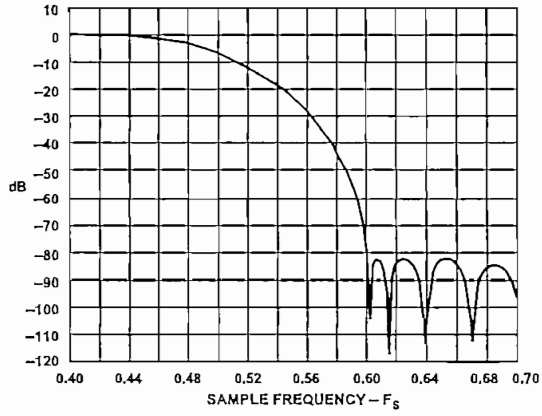


Figure 27. AD1847 Digital-to-Analog Frequency Response –Transition Band (Full-Scale Inputs, 0 dB Attenuation)

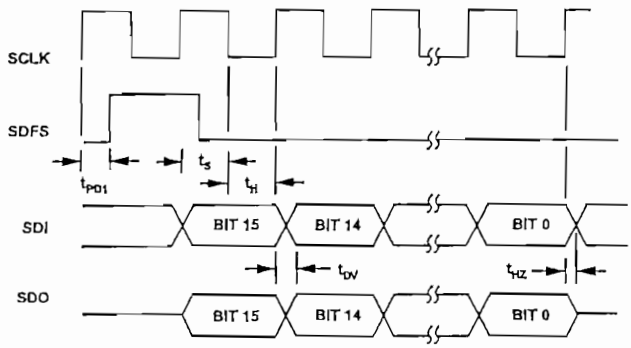


Figure 28. Time Slot Timing Diagram

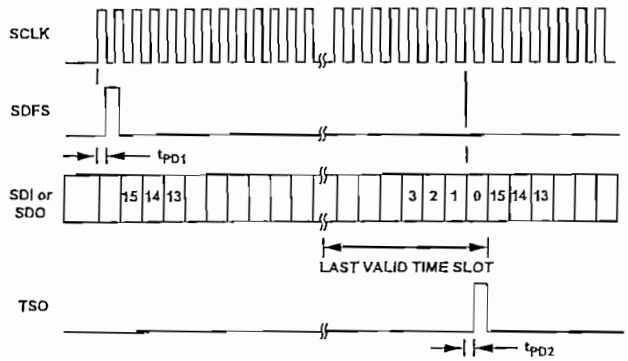


Figure 29. TSO Timing Diagram

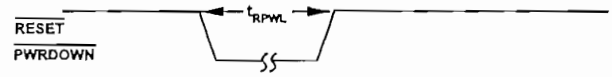


Figure 30. Reset and Power Down Timing Diagram

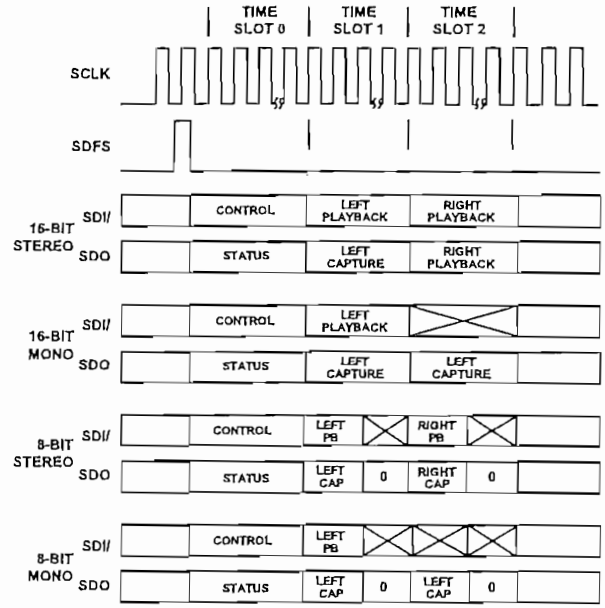


Figure 31. Serial Data Format, 2-Wire System (TSSEL = 1)

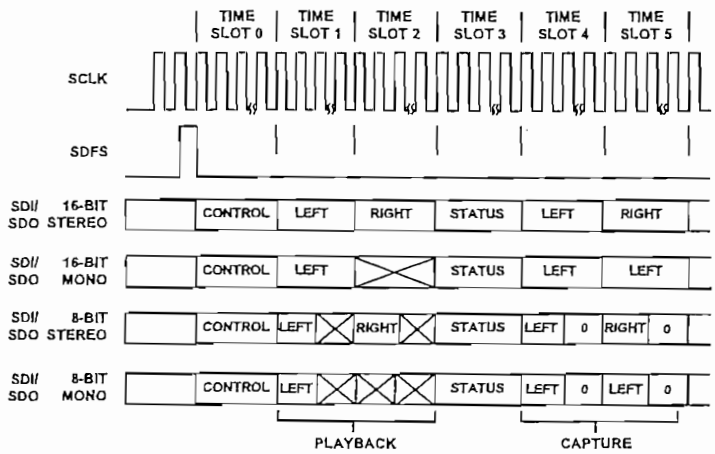
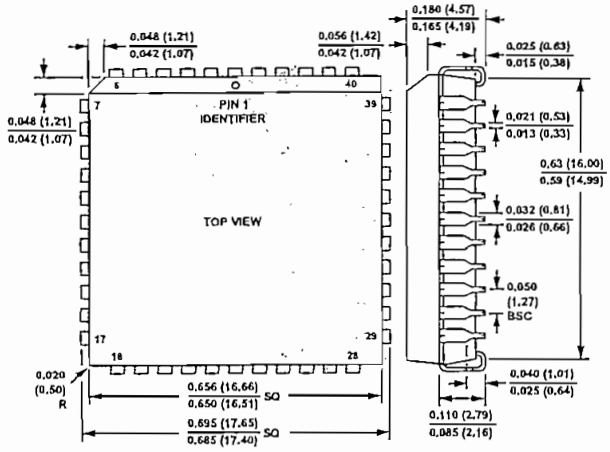


Figure 32. Serial Data Format, 1-Wire System (TSSEL = 0)

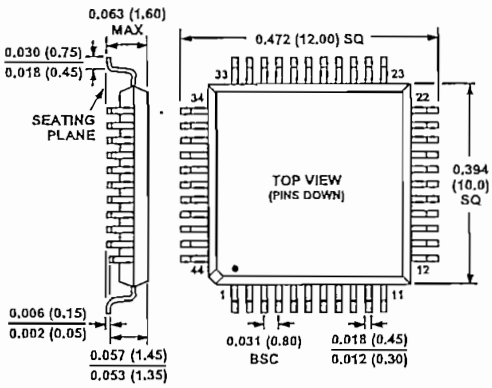
OUTLINE DIMENSIONS

Dimensions shown in inches and (mm).

44-Lead PLCC (P-44A)



44-Terminal Plastic Thin Quad Flatpack (TQFP) (ST-44)



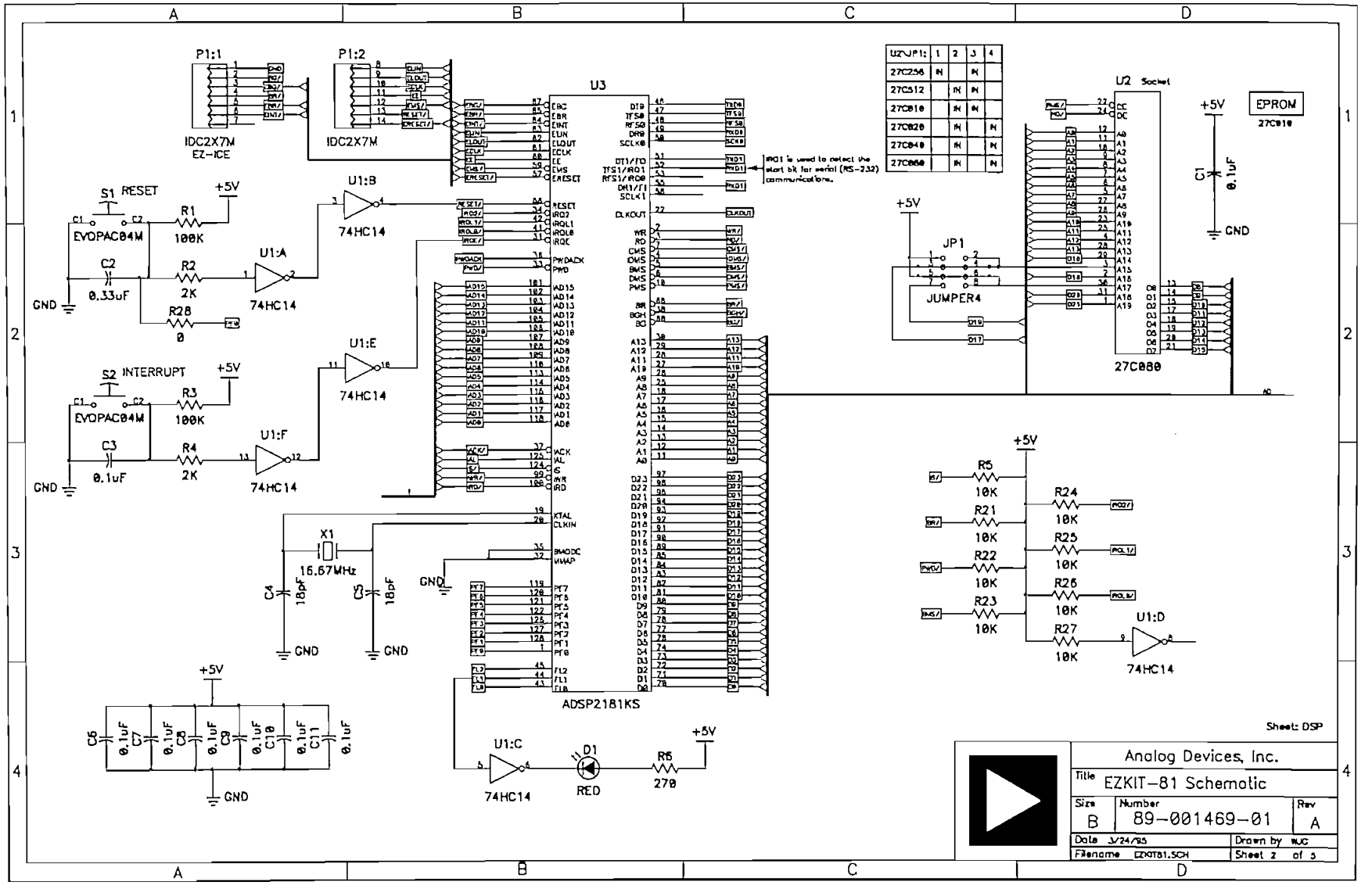
INDEX PAGE

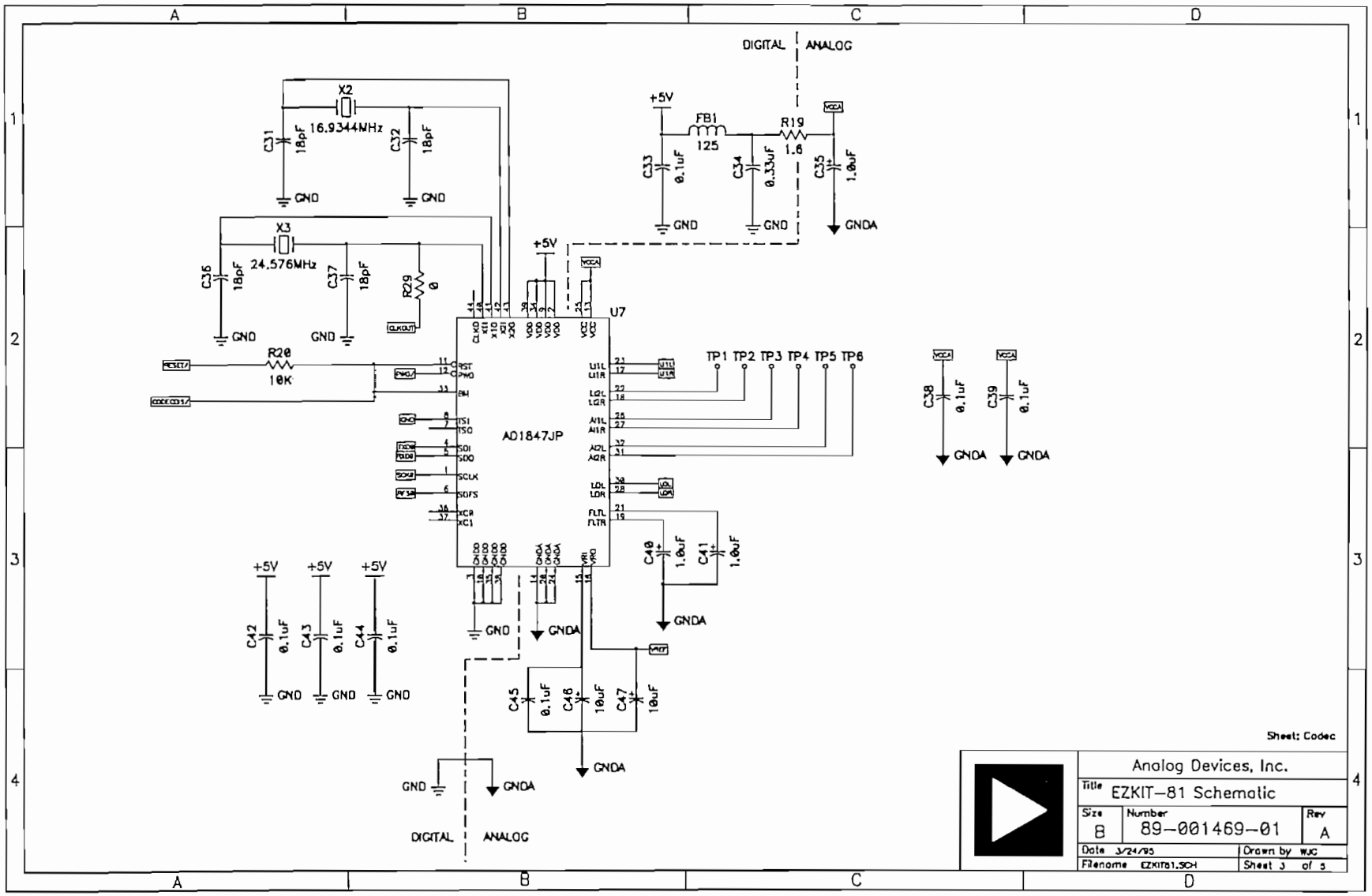
C1871b–2–8/96

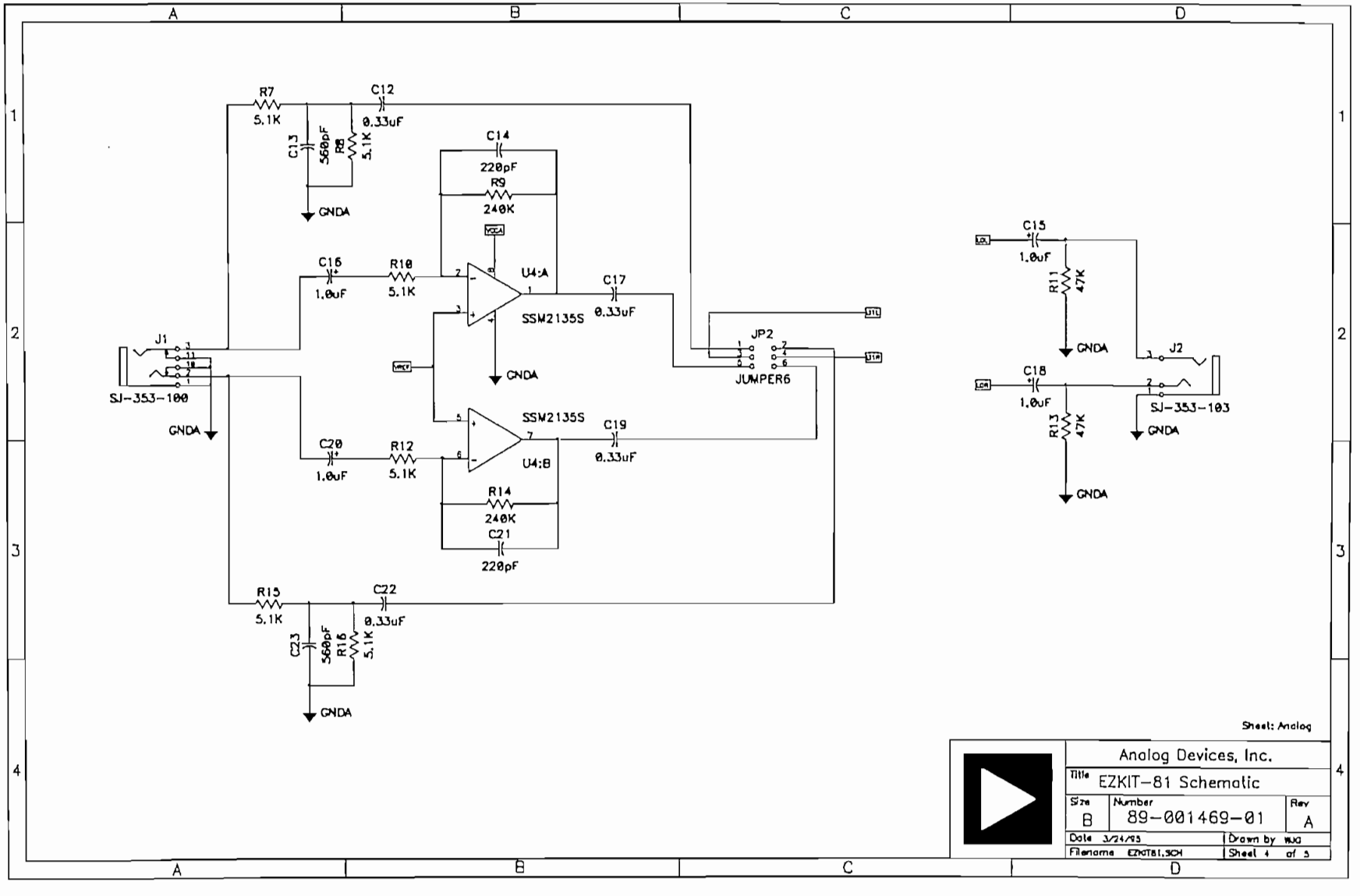
PRINTED IN U.S.A.

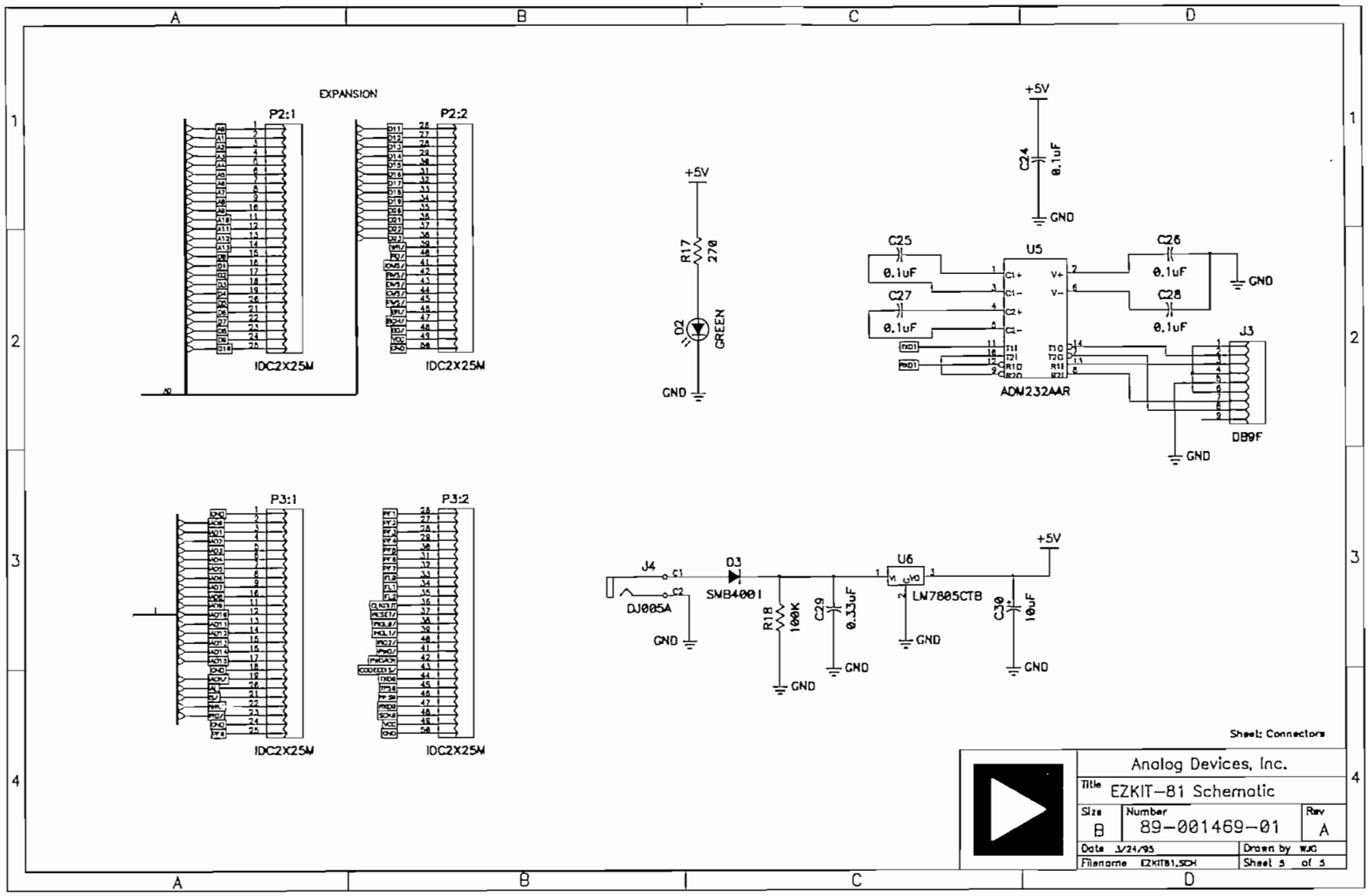
ANEXO 3

ESQUEMÁTICOS DE LA TARJETA EZ-KIT LITE


P1:1
IDC2X7M
EZ-ICE
P1:2
IDC2X7M
U3
ADSP2181KS
U2 Socket
27C080
EPROM
27C010
U2/JP1:
27C256
27C512
27C010
27C020
27C040
27C080
IRQ1 is used to detect the start bit for serial (RS-232) communications.
S1 RESET
EVQPAC04M
R1
100K
C2
0.33uF
R2
2K
R28
0
U1:A
U1:B
74HC14
U1:E
74HC14
S2 INTERRUPT
EVQPAC04M
R3
100K
C3
0.1uF
R4
2K
U1:F
74HC14
X1
16.67MHz
C4
18pF
C5
18pF
GND
+5V
C1
0.1uF
JP1
JUMPER4
R5
10K
R21
10K
R22
10K
R23
10K
R24
10K
R25
10K
R26
10K
R27
10K
U1:D
74HC14
C6
0.1uF
C7
0.1uF
C8
0.1uF
C9
0.1uF
C10
0.1uF
C11
0.1uF
U1:C
74HC14
D1
RED
R6
270
Sheet: DSP
Analog Devices, Inc.
Title
EZKIT-81 Schematic
Size
B
Number
89-001469-01
Rev
A
Date 3/24/95
Drawn by WJC
Filename EZKIT81.SCH
Sheet 2 of 5


DIGITAL
ANALOG
X2
16.9344MHz
C31 18pF
C32 18pF
GND
X3
24.576MHz
C36 18pF
C37 18pF
R29 0
CLKOUT
+5V
FB1 125
R19 1.6
C33 0.1uF
C34 0.33uF
C35 1.0uF
VCCA
GNDA
U7
AD1847JP
R20 10K
RESET/
CODECCS1/
TP1 TP2 TP3 TP4 TP5 TP6
C38 0.1uF
C39 0.1uF
C40 1.0uF
C41 1.0uF
C42 0.1uF
C43 0.1uF
C44 0.1uF
C45 0.1uF
C46 10uF
C47 10uF
Sheet: Codec
Analog Devices, Inc.
Title EZKIT-81 Schematic
Size B
Number 89-001469-01
Rev A
Date 3/24/95
Drawn by WJC
Filename EZKIT81.SCH
Sheet 3 of 5


R7
5.1K
C12
0.33uF
C13
560pF
R8
5.1K
GNDA
C14
220pF
R9
240K
VCCA
C16
1.0uF
R10
5.1K
U4:A
SSM2135S
C17
0.33uF
J1
SJ-353-100
GNDA
VREF
GNDA
JP2
JUMPER6
SSM2135S
C19
0.33uF
C20
1.0uF
R12
5.1K
U4:B
R14
240K
C21
220pF
R15
5.1K
C22
0.33uF
C23
560pF
R16
5.1K
GNDA
C15
1.0uF
R11
47K
GNDA
J2
C18
1.0uF
R13
47K
SJ-353-103
GNDA
GNDA
Sheet: Analog
Analog Devices, Inc.
Title
EZKIT-81 Schematic
Size
B
Number
89-001469-01
Rev
A
Date 3/24/95
Drawn by WJG
Filename EZKIT81.SCH
Sheet 4 of 5


EXPANSION
P2:1
P2:2
IDC2X25M
IDC2X25M
P3:1
P3:2
IDC2X25M
IDC2X25M
+5V
R17
270
D2
GREEN
GND
+5V
C24
0.1uF
GND
C25
0.1uF
C27
0.1uF
U5
C26
0.1uF
C28
0.1uF
GND
J3
ADM232AAR
DB9F
GND
J4
DJ005A
GND
D3
SMB4001
R18
100K
GND
C29
0.33uF
GND
U6
LM7805CTB
GND
+5V
C30
10uF
GND
Sheet: Connectors
Analog Devices, Inc.
Title
EZKIT-81 Schematic
Size
B
Number
89-001469-01
Rev
A
Date 1/24/95
Drawn by WJG
Filename EZKIT81.SCH
Sheet 5 of 5

ANEXO 4

CÓDIGO DEL PROGRAMA GENARBFOR.DSP

```
{*****************************************************************************************************
*                                    GENARBFOR. DSP
*
* Este programa toma las muestras de las formas de onda creadas y trasmitidas desde Labview
* hacia las localidades de memoria de datos 0x0000 a 0x1388 (4800 muestras máximo) del DSP.
*
* El Codec de audio AD1847 incluido en la tarjeta de desarrollo genera una trasmisión serial
* de los datos de entrada analógico convertidos a digital cada cierto periodo, este periodo
* es el periodo de muestreo, que se tiene que indicar en el modo de inicialización del codec
* mediante la trasmisión serial de ciertos registros de formato desde el DSP hacia el codec.
*
* El programa no lee ningún dato analógico convertido a digital desde el Codec pero aprovecha
* la interrupción generada dada por la frecuencia de muestreo, para enviar los datos de las
* muestras de las formas de onda desde el DSP hacia el Codec y así poder convertirlos de
* digital a analógico.
*
* La interrupción se genera cuando los datos del Codec están en memoria,
* mediante la propiedad de autobuferado del puerto serial del DSP. En la subrutina de
* atención a esta interrupción se toman las muestras de las formas de ondas trasmitidas
* que están en las direcciones 0x0000 a 0x1388 de memoria de datos  y se las envían
* hacia el AD1847.
*
***************************************************************************************************** }

.MODULE/RAM/ABS=0x0000   Proc_datos;


{********* ASIGNACION DE DIRECCIONES DE REGISTROS MAPEADOS EN MEMORIA ********}

.const  IDMA=                   0x3fe0;
.const  BDMA_BIAD=              0x3fe1;
.const  BDMA_BEAD=              0x3fe2;
.const  BDMA_BDMA_Ctrl=         0x3fe3;
.const  BDMA_BWCOUNT=           0x3fe4;
.const  PFDATA=                 0x3fe5;
.const  PFTYPE=                 0x3fe6;
.const  SPORT1_Autobuf=         0x3fef;
.const  SPORT1_RFSDIV=          0x3ff0;
.const  SPORT1_SCLKDIV=         0x3ff1;
.const  SPORT1_Control_Reg=     0x3ff2;
.const  SPORT0_Autobuf=         0x3ff3;
.const  SPORT0_RFSDIV=          0x3ff4;
.const  SPORT0_SCLKDIV=         0x3ff5;
.const  SPORT0_Control_Reg=             0x3ff6;
.const  SPORT0_TX_Channels0=            0x3ff7;
.const  SPORT0_TX_Channels1=            0x3ff8;
.const  SPORT0_RX_Channels0=            0x3ff9;
.const  SPORT0_RX_Channels1=            0x3ffa;
.const  TSCALE=                 0x3ffb;
.const  TCOUNT=                 0x3ffc;
.const  TPERIOD=                0x3ffd;
.const  DM_Wait_Reg=            0x3ffe;
.const  System_Control_Reg=     0x3fff;


{****************************** DECLARACION DEL BUFFER DE DATOS ******************************}

.var/dm/ram/SEG=mem_datos/circ      buf_rx[3];      {* declaración del buffer de recepción *,}
                                                    {* Status + L data + R data   * }
```

```
.var/dm/ram/SEG=mem_datos/circ     buf_tx[3];        {* declaración del buffer de trasmisión,*}
                                                       {* Cmd + L data + R data    * }
.var/dm/ram/SEG=mem_datos/circ     cod_inicial[13];        {* declaración de buffer de códigos de
                                                              inicialización del codec AD1847   * }
.var/dm/ram/SEG=mem_datos          band_estado;      {* declaración de una variable a ser usada
                                                       como bandera de estado *}
.var/dm/ram/SEG=mem_datos          band_retorno;     {* declaración de una variable a ser
                                                       usada como bandera de retorno  *}
.var/dm/ram/SEG=mem_datos      cont_muestras_gen;  {*declaración de una variable a ser usada
                                                        como un indicador del numero de
                                                        muestras generadas *}

.var/dm/ram/SEG=mem_datos      punt_muestras;  {* declaración de una variable a ser usada
                                                  como puntero de la localidad de la muestra*}

.var/dm/ram/SEG=mem_datos      cont_frec;        {* declaración de una variable a ser usada
                                                  como contador para baja frecuencia *}


.var/dm/ram/SEG=muestras  Forma_de_Onda[4800];  {* muestras de la forma  de onda  a ser
                                                  generadas *}


{***************************** INICIALIZACION DEL BUFFER DE DATOS ******************************}


{ * El DSP se comunica serial y sincrónicamente con el  AD1847 CODEC. El AD1847 trasmite y
  * recibe simultáneamente 3 palabras de 16 bits por cada muestra. En la siguiente sección se
  * declara 2 buffer circulares de datos, que ocupan 3 localidades cada uno, buf_rx es el buffer
  * para la recepción y buf_tx para la trasmisión. En cada muestra analógica recibida por el
  * AD1847 este envía una palabra de estado, los datos del canal derecho y los datos del canal
  * izquierdo hacia el buffer de recepción del DSP (buf_rx) y el DSP trasmite la palabra de control,
  * los datos del canal derecho y los datos del canal izquierdo a través de buffer de trasmisión
  * (buf_tx) hacia el  AD1847 para generar las muestras.
  *
  * El puerto serial del DSP esta configurado como multicanal y utiliza autobuferado para
  * comunicarse con el AD1847. Gracias a el autobuferado, el DSP recibe y trasmite un buffer de 3
  * palabras en forma automática. El DSP por medio de la interrupción dada por el Autobuferado,
  * sabe cuando a trasmitido y recibido completamente un buffer de 3 palabras  *}

.init buf_tx:   0xc000, 0x0000, 0x0000;        { * Inicializa el buffer de trasmision (buf_tx).
                                                 * Palabra de control = 0xc0000 (11000000 00000000)
                                                 * CLOR = 1, los bits de indicación de sobrerabgo en
                                                 * la palabra de estado son actualizados en cada
                                                 * muestra
                                                 * MCE = 1, habilita el cambio de modo (Modo
                                                 * Change Enable)
                                                 * Datos del canal izquierdo = 0x0000
                                                 * Datos del canal derechoo = 0x0000.                *}

{ * En esta sección se inicializa el buffer de comandos a enviar al AD1847 con el  arreglo
  * cod_inicial [ ] con los siguientes valores.
  * Estos valores a trasmitir es la configuración inicial del AD1847. los bits del 0 al 7 son los 8 bits
  *  de datos de los registros indexados (datos 0-7) . Los bits 8-11 indican
  * la direccion de los registros indexados (IA0-3)
  * El Bit 13 es el bit para readback  mode (RREQ).
  * Bit 14 es el bit MCE. El Bit 15 es CLOR.
```

```
 * El programa se configura para almacenar el buffer cod_inicial en el AD1847 automaticamente.
 *}

.init cod_inicial:   0xc002,   {* (11000000 00000010); setea CLOR , setea MCE , direc reg
                                * index = 0, dato = 0x2
                                * reg de control de la entrada del canal izquierdo
                                * b7-6: 0= Line 1
                                *        1=Entrada Auxiliar 1
                                *        2= Line2
                                *        3= Salida post-mezclada del DAC
                                * b5-4: res
                                * b3-0: Selección de ganancia  x 1.5 dB          *}

                     0xc102,   {* (11000001 00000010) setea CLOR , setea MCE , direc reg
                                * index = 1, dato = 0x2
                                * registro de entrada de canal derecho
                                * b7-6:  0= Line 1
                                *        1= Entrada Auxiliar 1
                                *        2= Line 2
                                *        3= sallida post-mezclada del DAC
                                * b5-4:  res
                                * b3-0:  Seleccion de ganancia  x 1.5 db          *}

                     0xc288,   {*  (11000010 10001000 ) setea CLOR, setea MCE set, direc reg
                                * index = 2, dato = 0x88
                                *  reg de cont del canal auxiliar 1 izquierdo
                                *  b7 :   1=left aux 1 mute (silenciado)
                                *  b6-5:   res
                                *  b4-0:   gain/atten x 1.5, 08= 0dB, 00= 12dB   *}

                     0xc388,   {*  (11000011 10001000) setea CLOR , setea MCE, direc reg
                                *   index = 3, data = 0x88
                                *  reg de cont del canal auxiliar 1 derecho
                                *  b7 :   1=right aux 1 mute (silenciado)
                                *  b6-5:   res
                                *  b4-0:   gain/atten x 1.5, 08= 0dB, 00= 12dB   *}
                     0xc488,   {*  (11000100 10001000) setea CLOR , setea MCE, direc reg
                                *  index = 4, dato = 0x88
                                *  reg de cont del canal auxiliar 2 izquierdo
                                *  b7 :   1=left aux 2 mute (silenciado)
                                *  b6-5:   res
                                *  b4-0:   gain/atten x 1.5, 08= 0dB, 00= 12dB   *}

                     0xc588,   {*  (11000101 10001000) setea CLOR,  setea MCE, direc reg
                                *  index = 5, dato=0x88
                                *  reg de cont del canal auxiliar 2 derecho
                                *  b7 :   1=right aux 2 mute (silenciado)
                                *  b6-5:   res
                                *  b4-0:   gain/atten x 1.5, 08= 0dB, 00= 12dB   *}

                     0xc680,   {*  (11000110 10000000) setea CLOR , setea MCE, direc reg
                                *  index = 6, dato = 0x80
                                *  reg de cont del DAC del canal izquierdo
                                *  b7 :   1=left DAC mute (silenciado)
                                *  b6 :   res
                                *  b5-0:   atenuación x 1.5 dB   *}

                     0xc780,   {* (11000111 10000000) setea CLOR , setea MCE, direc reg
                                * index = 7, dato = 0x80
```

```
                * reg de cont del DAC del canal derecho
                * b7 :  1 = right DAC mute (silenciado)
                * b6 :  res
                * b5-0:  atenuación x 1.5 dB    *}

0xc85c,         {* (11001000 01011100) setea CLOR , setea MCE, direc reg
                * index = 8, dato = 0x5c
                * reg de formato de datos
                * b7 :  res
                * b5-6:  0 = 8-bit unsigned linear PCM
                *       1 = 8-bit u-law companded
                *       2 = 16-bit signed linear PCM
                *       3 = 8-bit A-law companded
                * b4 :  0 = mono, 1 =  stereo
                * b0-3:  0 =  8.00000 Khz
                *       1 =  5.51250 Khz
                *       2 = 16.00000 Khz
                *       3 = 11.02500 Khz
                *       4 = 27.42857 Khz
                *       5 = 18.90000 Khz
                *       6 = 32.00000 Khz
                *       7 = 22.05000 Khz
                *       8 = ------------  .
                *       9 = 37.80000 Khz
                *       a = ------------  .
                *       b = 44.10000 Khz
                *       c = 48.00000 Khz
                *       d = 33.07500 Khz
                *       e =  9.60000 Khz
                *       f =  6.61500 Khz
                * (b0) :  0 = XTAL1 24.576 Mhz; 1 = XTAL2 16.9344 Mhz    *}

0xc909,         {* (11001001 00001001) setea CLOR, setea MCE, direc reg
                * index = 9, dato = 0x09
                * reg de configuración de la interfaz
                * b7-4:  res
                * b3 :  1 = autocalibración
                * b2-1:  res
                * b0 :  1 = playback habilitado    *}

0xca00,         {* (11001010 00000000) setea CLOR, setea MCE, direc reg
                index=10, dato = 0x00
                * reg de cont de pines
                * b7 :  estado lógico del pin XCTL1
                * b6 :  estado lógico del pin XCTL0
                * b5 :  master - 1 = tri-state CLKOUT
                *       slave  - x = tri-state CLKOUT
                * b4-0:  res                    *}


0xcc40,         {* (11001100 01000000) setea CLOR, setea MCE, direc reg
                * index=12, dato=0x40
                * reg de información miscelanea
                * b7 :  1 = 16 slots por trama
                *       0 = 32 slots por trama
                * b6 :  1 = sistema 2-wire
                *       0 =  sistema 1-wire
                * b5-0:  res                    *}
```

```
                0xcd00;    {*  (11001101 00000000 ) setea CLOR, setea MCE, dir reg
                            *  index=13, dato=0x00
                            *
                            *  b7-2:   atenuación x 1.5 dB
                            *  b1  :   res
                            *  b0  :   1 = mezcla digital habilitada *}


.init Forma_de_Onda:<Muestras.dat>;

{********************************** Tabla de Vectores de interrupciones ********************************}

{*  El momento que ocurre una interrupción, el flujo de programa salta hacia la tabla de vectores
 *  La localidad del vector tiene 4 localidades pm  subsiguientes para atender a la interrupción.
 *  Si la interrupción esta enmascarada no se produce el salto.   *}

        jump inicio;            {* address = 0x00: reset interrupt vector *}
         rti;
         rti;
         rti;

        rti;                    {* address = 0x04: IRQ2 interrupt vector*}
         rti;
         rti;
         rti;

        rti;                    {* address = 0x08: IRQL1 interrupt vector*}
         rti;
         rti;
         rti;

        rti;                    {* address = 0x0c: IRQL0 interrupt vector*}
         rti;
         rti;
         rti;

        ar = dm(band_estado);   {* address = 0x10: SPORT0 tx interrupt vector *}
                ar = pass ar;
                if eq rti;
         jump sig_inic;

        jump muest_ent;         {* address = 0x14: SPORT0 rx interrupt vector*}
         rti;
         rti;
         rti;

        jump irqe;              {* address = 0x18: IRQE interrupt vector  *}
         rti;           {* Esta es la interrupción que se genera al presionar el pulsante  *}
         rti;           {* en la tarjeta de desarrollo EZ-KIT LITE *}
         rti;

        rti;            {* address = 0x1C: BDMA interrupt vector *}
         rti;
         rti;
         rti;

        rti;            {* address = 0x20: SPORT1 tx or IRQ1 interrupt vector *}
         rti;
```

```
        rti;
        rti;

        rti;            {* address = 0x24: SPORT1 rx or IRQ0 interrupt vector *}
        rti;
        rti;
        rti;

        rti;            {* address = 0x28: timer interrupt vector *}
        rti;
        rti;
        rti;

        rti;            {* address = 0x2c: power down interrupt vector  *}
        rti;
        rti;
        rti;

{************************************** Inicialización del ADSP 2181 ********************************}

{************************************** Inicialización de DAGs ******************************************}

inicio:


        i0 = ^buf_rx;       {* seteo i0 = la direc de inicio del buffer rx    *}
        l0 = %buf_rx;       {* seteo l0 = la longitud del buffer rx           *}
        i1 = ^buf_tx;       {* seteo i1 = la direc de inicio del buffer tx    *}
        l1 = %buf_tx;       {* seteo l0 = la longitud del buffer tx           *}
        i3 = ^cod_inicial;  {* seteo i3 = la direc de inicio de cod inicialización  *}
        l3 = %cod_inicial;  {* seteo l3 = la longitud del buffer cod inicialización *}
        l5 = 0;             {* seteo l5=0 para uso de direccion lineal   *}
        m1 = 1;             {* seteo m1 = 1 para usarlo con direc circular   *}
        m0 = 0;             {* m0-3 puede usarse entre i0-3 como registros de modificación *}


{********************************** Configuración del Puerto Serial #0 ************************************}

{* El puerto serial se lo configura escribiendo en registros mapeados en memoria
 * SPORT 0 es configurado para usar autobuferado y en modo multicanal
 * usando una señal de reloj serial externa y sincronización de trama externo.
 * El puerto serial usa los DAGs para accesar a memoria de datos automáticamente.
 * En este programa, el  autobuferado usa a i0, i1, y m1. i0 apunta a el bufer buf_rx,
 * i1 apunta a buf_tx y m1 = 1 (registro que aumenta automáticamente en 1 a 1).
 * El modo Multicanal es habilitado y configurado para 32 canales. El AD1847 es tambien
 * configurado para 32 canales. En este modo cada canal es una palabra y esta asociada con
 * un slot de tiempo. Los 32 canales tienen 32 slots de tiempo. El puerto serial puede
 * trasmitir y/o recibir datos o los puede ignorar, llegando en un tercer estado para cierto
 * slot de tiempo. Una trama consiste de 32 canales. Una trama de sincronización empezará
 * la transferencia de los 32 canales. Se configura al puerto serial para recibir los canales
 * 0, 1, 2, 16, 17 y 18 para trasmisión y los canales 0, 1, 2, 16, 17, y 18 para recepción.
 * Los demás son ignorados. Tanto el reloj serial y la trama de sincronización son externas y
 * por lo tanto son señales que entran al DSP.        *}

        ax0 =0x287;
        dm (SPORT0_Autobuf) = ax0;      {*  00000010 10000111 = 0x287
                                         *  TBUF=1 y RBUF=1 Habilita Autobuferado para tx y rx
                                         *  RMREG=m1, RIREG=i0, TMREG=m1, TIREG=i1,
                                         *  habilitado CLKOUT , deshabilitado BIASRND  *}
```

```
        ax0 = 0;
        dm (SPORT0_RFSDIV) = ax0;  {* RFSDIV = SCLK Hz/RFS Hz - 1
                                    * se usa trama externa de sinc como ent ( RFS=input ) *}
        ax0 = 0;
        dm (SPORT0_SCLKDIV) = ax0;     {* SCLK = CLKOUT / (2  (SCLKDIV + 1) )
                                        * se usa reloj serial externo ( SCLK=input )       *}

        ax0 =0x860f;
        dm (SPORT0_Control_Reg) = ax0; {* 10000110 00001111 = 0x860f
                                        * SLEN= 16 bits, just derech, llenado de ceros MSBs,
                                        * INVRFS=0, INVTFS=0, IRFS=0, ITFS=1,
                                        * MFD=1, ISCLK=0, MCE=1                    *}

        ax0 =0x7;
        dm (SPORT0_TX_Channels0) = ax0; {* habilita canales 0, 1 y 2 para trasmitir      *}

        ax0 =0x7;
        dm (SPORT0_TX_Channels1) = ax0; {* habilita canales 16, 17 y 18 para trasmitir  *}

        ax0 =0x7;
        dm (SPORT0_RX_Channels0) = ax0; {*  habilita canales 0, 1 y 2 para recibir      *}

        ax0 =0x7;
        dm (SPORT0_RX_Channels1) = ax0; {*  habilita canales 16, 17 y 18 para recibir   *}


{******************************** Configuración del sistema y de memoria ******************************}

        ax0 = 0xfff;
        dm (DM_Wait_Reg) = ax0;                {*  Seteo IOWAIT a 7 estados de espera,
                                                *  sete0 DWAIT a 0 estados de espera         *}

        ax0 = 0x1000;                          {* 10000 00000000 PWAIT = 0, habilito SPORT0 *}
        dm (System_Control_Reg) = ax0;

        ifc = 0xff;                            {* limpio interrupciones pendientes     *}
        nop;
        icntl = 0;                             {* interrupciones externas se setean para ser
                                                * sensitivas a nivel , deshabilito
                                                * interrupciones anidadas     *}
        mstat = 0x40;                          {* habilito modo go *}


{************************************** Inicialización del CODEC AD1847 ***********************************}

        ax0 = 1;
        dm(band_estado) = ax0;         {* inicializo bandera de estado a 1    *}
        imask = b#0001000000;          {* desnmascar interrupción de trasmisión del SPORT0  *}
        ax0 = dm (i1, m1);             {* seteo de ax0 = al primer valor de buf_tx          *}
        tx0 = ax0;                     {* empieza la trasmisión con autobuferado
                                        * cuando se complete la trasmisión del bufer se genera
                                        * una interrupción              *}

{*  Se comprueba que el buffer cod_inicial se ha enviado enteramente al codec  *}

Test_inicial:                          { * este era check_init *}
        ax0 = dm (band_estado);        {* setea ax0 con la band de estado. Esta bandera se setea
                                        * en la rutina de atención a la interrupción
```

```
                                        * de trasmisión del SPORT0  *}

        af = pass ax0;          {* pasa el valor ax0 para que se seten las banderas de estado
                                {* de la ALU  *}
        if ne jump Test_inicial;        {* si el valor no es cero salta a Test_inicial  *}

{*  Una vez inicializado, se espera que el codec termine de autocalibrarse revisando el bit ACI
 *  del registro de estado del AD1847. Para asegurarse se realiza dos veces, la primera para
 *  asegurarse que el codec en el modo de autocalibración, y la otra para saber que
 *  el modo de autocalibración esta completo.                                    *}

        ay0 = 2;

Test_aci1:                      {* lazo que testea el bit ACI = 1 ( en autocalibración )     *}
        ax0 = dm (buf_rx);      {* lee la palabra de estado del AD1847                 *}
        ar = ax0 and ay0;       {* AND de 2 con la palabra de estado                 *}
        if eq jump Test_aci1;   {* Si el resultado de AND no es = 0 deja el lazo     *}

Test_aci2:                      {* lazo que testea el bit ACI = 0 ( auto calibración completa ) *}
        ax0 = dm (buf_rx);      {* espera para que el bit cambie  a cero                 *}
        ar = ax0 and ay0;
        if ne jump Test_aci2;
        idle;

{* Una vez que Autocalibracion esta completa se activan los canales DAC izquierdo y derecho
 * escribiendo a los respectivos registros indexados                                *}

        ay0 = 0xbf3f;           {* 10111111 00111111 deshabilito mute del DAC izquierdo *}
        ax0 = dm (cod_inicial + 6);
        ar = ax0 AND ay0;
        dm (buf_tx) = ar;
        idle;                           {* espera para que la trasmisión sea completa      *}
        ax0 = dm (cod_inicial + 7);     {* deshabilito mute del DAC derecho                *}
        ar = ax0 AND ay0;
        dm (buf_tx) = ar;
        idle;                           {* espera para que la trasmisión sea completa      *}

        ifc = 0xff;                     {* limpio cualquier interrupción pendiente         *}
        nop;


        ax0 = 0;
        dm(cont_muestras_gen) = ax0;            {* cuenta el número de muestras generadas   *}
        ax0 = 2;
        dm(punt_muestras) = ax0;        { * apuntador a la localidad dm(0x0002) que es la 1era
                                        muestra *}
        ax0 = 0;
        dm(band_retorno) = ax0;                 {* bandera para retornar al Monitor   *}
        ax0 = 0;
        dm(cont_frec) = ax0;


        imask = 0x30;                   {* desenmascaro rx0 and IRQE interrupciones        *}

{* Espera en modo IDLE por una interrupción. Cuando el programa retorna de una interrupción
 * Salta del  modo IDLE.                                                            *}
```

```
{************************** Inicialización de variables auxiliares y banderas de estado *****************}


Test_retorno:     idle;

                  ax0 = dm(band_retorno);      {* Si band_retorno = 1, retorna al prog Monitor   *}

                  ay0 = 1;
                  none = ax0 - ay0;
                  if eq rts;
                  jump Test_retorno;


{********************************** Rutinas de atención a Interrupciones ******************************}

{****************************** Rutina de servicio a la interrupción de recepción **********************}

{* esta subrutina de atención a recepción es generada por el CODEC e indica que ha transcurrido
*  e l periodo de muestreo. Aqui es donde se trasmiiran las myestras al Codec desde el DSP  *}

muest_ent:

        ena sec_reg;                    {* se usa el banco secundario de registros *}
        ax0 = dm(buf_rx + 1);           {* Se obtiene el dato del canal izquierdo      *}
        ay0 = dm(buf_rx + 2);           {* Se obtiene el dato del canal derecho        *}


        ax0 = dm(0x0);                  {* En la localidad dm(0) se encuentra la longitud
                                           de las muestras a generar enviada desde Labview. *}

        ay0 = dm(cont_muestras_gen);       {* cont_muestras_gen, indica el numero de
                                           muestras generadas.                  *}
        af = ax0 XOR ay0;
        ar = pass af;                   {* Si la longitud de muestras a generar es = a el numero
                                           de muestras generadas, Se vuelve a tomar la 1era muestra* }

        if eq jump sig_periodo;         {* Se revisa todas las localidades en que residen las muestras
                                           si a llegado a la ultima muestra, se debe iniciar otro periodo* }

        i2 = dm(punt_muestras);  {* se utiliza a I2 como puntero de las localidades de memoria
        I2 = 0;                  * en donde se encuentran las muestras *}
        m2 = 0;

        ax0 = dm(punt_muestras);  {* luego se aumenta en 1 a la localidad que indica la posición
        ar = ax0 + 1;             * de la muestra actual, para apuntar a la próxima muestra
        dm(punt_muestras) = ar;

        ax0 = dm(cont_muestras_gen);  {* luego se aumenta en 1 al contador de las muestras
        ar = ax0 +1;                  * generadas            *}
        dm(cont_muestras_gen) = ar;

        ax0 = dm(i2,m2);        {* se traslada el valor de la muestra hacia ay0 y axo *}
        ay0 = dm(i2,m2);

        jump enviar;
```

```
sig_periodo:
            ax0 = 1;    {* en el siguiente periodo se reinicializa a los punteros y contadores *}
            dm(1000) = ax0;
            ax0 = 0x0;                  {* el contador vuele a cero  *}
            dm(cont_muestras_gen) = ax0;
            ax0 = 2;
            dm(punt_muestras) = ax0;  {* se apunta hacia la loc de mem de la 1era muestra *}
            i2 = dm(punt_muestras);
            l2 = 0;
            m2 = 0;
            ax0 = dm(i2,m2);
            ay0 = dm(i2,m2);


enviar:

            dm (buf_tx + 1) = ax0;        {* Se envia el dato del canal izquierdo      *}
            dm (buf_tx + 2) = ay0;        {* Se envia el dato del canal derecho       *}


rti;


{************************ Rutina de servicio a la interrupción de trasmisión ******************************}

{* La rutina de servicio a la interrupción de trasmisión es usada para inicializar al codec AD1847.
 * I3 apunta al buffer cod_inicial . El AD1847 esta esperando una palabra de control,
 * y 2 palabras de datos desde el DSP. Aquí se carga el siguiente código de inicialización
 * en la primera localidad del buffer buf_tx , las 2 palabras de datos se han inicializado
 * a cero al comienzo del programa. Esta rutina chequea si el último dato ha sido trasmitido
 * Si la última palabra ha sido trasmitida, el AD1847 sale del modo de inicialización y
 * retorna de la interrupción *}

sig_inic:

        ena sec_reg;
        ax0 = dm (i3, m1);        {* se obtiene la palabra del código desde el buffer cod_inicial *}
        dm (buf_tx) = ax0;        {* Se ubica la palabra del código en la primera localidad del
                                    * buffer buf_tx ( transmite slot 0)    *}
        ax0 = i3;
        ay0 = ^cod_inicial;
        ar = ax0 - ay0;         {* testea por códigos adicionales            *}
        if gt rti;              {* sale de la rutina si hay mas códigos *}
        ax0 = 0xaf00;           {* caso contrario se setea la bandera de estado
                                  * y se remueve MCE si esta hecha la inicializacion   *}
        dm (buf_tx) = ax0;
        ax0 = 0;
        dm (band_estado) = ax0;   {* se pone en cero a la bandera de estado *}
        rti;

{********************************** Rutina de servicio a la interrupción IRQE ******************************}

{* La rutina de atención a IRQE retorna el control a el programa Monitor, seteando la bandera de
retorno *}

irqe:
```

```
        ax0 = 1;
        dm(band_retorno) = ax0;

        rti;                            {* Retorna de la interrupción *}

.endmod;
```

ANEXO 5

COMANDOS EMPLEADOS POR EL PROGRAMA MONITOR

EZ-KIT Lite Monitor Program 7

PROGRAM OVERVIEW

The Monitor Program resides in an on-board EPROM. It is automatically loaded into the on-chip program and data memories at reset. As soon as the monitor begins execution it performs a self test of DSP registers, on-chip memories, and a reset and initialization of AD1847 codec. It then waits for commands via the RS-232 serial communication port. A codec talk through routine is also running in an interrupt routine.

MONITOR FEATURES

This monitor has the following features.

- Power up self test
- AD1847 codec initialization and talk through
- 9600 bps UART emulation
- upload and download of both program and data memory contents via serial link

RESTRICTIONS

In order to facilitate the function of the monitor, certain restrictions are imposed on all user programs. These are hard restrictions because violation will interfere with the proper downloading of the user programs. These restrictions include the following.

- Reserved program memory 0x3800 to 0x3fff
  This is where the monitor program resides. The user program should not download into these locations. Any violation will destroy the monitor resulting in incomplete user program download.

- Reserved data memory 0x3e00 to 0x3fdf
  These are the monitor operating variables. User programs downloading into these locations will interfere with the monitor execution resulting in an incomplete user program download.

- Returning to the monitor
  After the download of your program is complete, the monitor software calls your code as a subroutine. Your code should end with an `RTS` instruction to properly return to the monitor.

  For user programs that do not return to the monitor there is no restriction imposed after the program has been downloaded. In other words, while the memory image file should not attempt to initialize the reserved memories, the user program is nevertheless free to use these memories after its execution.

CREATING YOUR OWN PROGRAMS TO BE USED WITH THE MONITOR

You can edit one of the example programs supplied or you can create your own program using the development tools included in your EZ-KIT Lite. See chapter 5 for more information. There are a few things to note when developing programs that you want to run with the monitor.

User programs may use all non-reserved program memory and non-reserved data memory. These are locations 0x0000 to 0x37ff for program memory and locations 0x0000 to 0x3dff for data memory.

- Program entry point
  User programs to be downloaded by the Host Program should have their entry point at the beginning of the first program memory kernel. This can be achieved by making the entry point the first instruction in the module and specifying that module first during linking. In most cases, your program will start at location 0x0000. You can use the assembler directive `.module/ram/abs=0` to specify this (see examples).

- Interrupt enable
  All interrupts are masked (by IMASK = 0.) before execution is transferred to the user programs. This is so that the user program can safely carry out initializations before it re-enables interrupts.

- Interrupt vectors
  The user program must initialize all interrupt vectors it is using. Interrupt IRQ1 can not be used in your program since it is dedicated to the RS232 communications. You can place "dummy" `RTI` instructions in your code at this interrupt vector (location 0x0020 through 0x0023).

5. download data memory content
   received: $DD[hi start][lo start][hi len][lo len][hi][lo]...
   sent: ![hi sum][lo sum]

   [hi start][lo start] is the starting memory location
   [hi len][lo len] is the number of words
   [hi][lo]... are the memory contents starting from the first location.
   [hi sum][lo sum] is a 16-bit checksum of the data calculated and sent by the monitor program.

6. download PM
   receivedt: $DP[hi start][lo start][hi len][lo len][hi][mi][lo]...
   sent: ![hi sum][mi sum][lo sum]

   [hi start][lo start] is the starting memory location
   [hi len][lo len] is the number of words
   [hi][mi][lo]... are the memory content starting from the first location.
   [hi sum][lo sum] is a 16-bit checksum of the data calculated and sent by the monitor program.
7. Begin user program execution
   received: $GO[hi address][loaddress]
   sent: ![hi address][loaddress]

   [hi address][loaddress] is the program entry point sent by the host to the monitor program. It is also the address sent by the monitor program back to the host for confirmation

DEBUGGING

The following techniques may be helpful in debugging user programs.

- SET FL1
- RESET FL1
- TOGGLE FL1

The FL1 LED is directly controlled by the DSP FL1 flag. These instructions may be placed anywhere in the users program. These may be used to provide a visual indication of the binary state of a variable, or whether a certain section is executing.

For non time-critical programs, it is possible to provide information by programming the FL1 to blink at different rates. For example the following code blinks the flag at a certain rate. Change the counter value for other rates.

```
      cntr = 80;
      do lp1 until ce;
        cntr = 50;
        do lp2 until ce;
          cntr = 12000;
          do lp3 until ce;
lp3:        nop;
lp2:      nop;
lp1:    toggle fl1;
```

DSP MEMORIES

The user programs may store historical data in user program memory or user data memory. This data may be retrieved after user programs have returned control back to the Monitor program (by RTS).

The upload program memory or the upload data memory commands of the host program may be used to retrieve this data.