12

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN

SISTEMA PARA ORIENTACION ESPACIAL DE SENSORES AEROPORTADOS EN EL ESPACIO GEOGRAFICO

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO MECANICO ELECTRICISTA

P R E S E N T A

MARCELO BASTIDA TAPIA

ASESOR: DR. RICARDO PERALTA Y FABI

CUAUTITLAN IZCALLI, EDO. DE MEXICO 2000

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN
UNIDAD DE LA ADMINISTRACION ESCOLAR
DEPARTAMENTO DE EXAMENES PROFESIONALES

UNIVERSIDAD NACIONAL AVENIDA II MEXICO

U. N. A. M.
FACULTAD DE ESTUDIOS SUPERIORES-CUAUTITLAN
DEPARTAMENTO DE EXAMENES

ASUNTO: VOTOS APROBATORIOS

DR. JUAN ANTONIO MONTARAZ CRESPO
DIRECTOR DE LA FES CUAUTITLAN
P R E S E N T E

ATN: Q. Ma. del Carmen García Mijares
Jefe del Departamento de Exámenes
Profesionales de la FES Cuautitlán

Con base en el art. 28 del Reglamento General de Exámenes, nos permitimos comunicar a usted que revisamos la TESIS:

"Sistema para Orientación Espacial de Sensores Aeroportados en el Espacio Geográfico"

que presenta el pasante: Marcelo Bastida Tapia
con número de cuenta: 9137737-5 para obtener el titulo de :
Ingeniero Mecánico Electricista

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el EXAMEN PROFESIONAL correspondiente, otorgamos nuestro VOTO APROBATORIO.

A T E N T A M E N T E
"POR MI RAZA HABLARA EL ESPIRITU"
Cuautitlán Izcalli, Méx. a 30 de Junio de 2000

PRESIDENTE	Ing. Ubaldo Ramírez Urizar	
VOCAL	Ing. Blanca de la Peña Valencia	
SECRETARIO	Dr. Ricardo Peralta y Fabi	
PRIMER SUPLENTE	Ing. Norma Reyes Cruz	
SEGUNDO SUPLENTE	Ing. Adriana Corina Sandoval García	

A Mis Padres
A mis Hermanos
Quienes siempre me han apoyado para seguir adelante.

A todas aquellas personas que contribuyeron a la realización de esta Tesis.

Índice

Índice.

Página.

Índice de Imágenes.

Página.

Imágenes

Gráficas

Introducción

Introducción.

El motivo de este trabajo.

Dotar de orientación espacial a sensores aeroportados que deben localizar, registrar y clasificar objetos en el espacio geográfico. Además, lograr desarrollos propios en instalación aeroespacial y con aplicaciones en otros campos.

Descripción del contenido.

En el capitulo 1 se describen brevemente los dispositivos utilizados en el desarrollo del sistema, como trabajan y cual es su función..

En el capitulo 2 se hace una reseña detallada de la integración del sistema, incluyendo algunos comentarios sobre mejoras o inconvenientes que se hicieron evidentes durante su desarrollo.

En el capitulo 3, se muestran los análisis realizados al sistema, que permitieron la evaluación del funcionamiento y desempeño, para mejor servir en su aplicación.

En el capitulo 4 se describe la forma en que el sistema puede ser utilizado y las posibles modificaciones que se pueden realizar, así como algunas mejoras que amplíen su uso previsto.

Al final se encuentran las conclusiones, recomendaciones y los apéndices, en los cuales se presenta la información necesaria para desarrollar y/o modificar este sistema, de acuerdo a las necesidades futuras.

Capítulo 1

Descripción y aplicación del equipo empleado

Capítulo 1.
Descripción y aplicación del equipo empleado.

Para el desarrollo del subsistema se requiere del uso de diferentes dispositivos, tales como el GPS (receptor para localizarse en el espacio geográfico), una brújula electrónica, un giróscopo vertical, varios acelerómetros, un microcontrolador y una microcomputadora o computadora personal (PC). En este capitulo se dará una breve descripción de cada uno de estos dispositivos así como la forma en la que funcionan.

GPS

El sistema GPS o Sistema de Posicionamiento Global, nació en los EUA y la URSS con el objetivo de mejorar anteriores sistemas militares de satélites de navegación, y es además utilizado en técnicas geodésicas en todo el mundo. De esta forma se pretendía conseguir una navegación en tiempo real, precisa y de forma continua en tierra, mar o aire (incluso en órbita), sin importar las condiciones meteorológicas y bajo un sistema unificado de cobertura global con precisiones de unos pocos metros en coordenadas geográficas.

Una vez funcionando el sistema, sus enormes aplicaciones no pasaron desapercibidas al sector científico, que aprovechando que se obtiene un posicionamiento en tiempo real de precisión, inicio su utilización y subsecuente divulgación entre la comunidad comercial.

El Sistema de posicionamiento global está constituido por tres segmentos claramente diferenciados:

Segmento espacial.

Comprende hasta hoy dos constelaciones de satélites denominadas NAVSTAR o GLONASS (nombre que designa al conjunto y tipo de los satélites utilizados por EU y Rusia, respectivamente). Cada constelación está formada por cerca de 24 Satélites, de los cuales cuando menos 4 están en línea de vista al mismo tiempo, a cualquier hora del día y desde cualquier punto de la superficie terrestre. En un caso, por ejemplo, los satélites se distribuyen en 6 órbitas circulares con una inclinación de 55° respecto al plano ecuatorial terrestre y 60° con respecto a las órbitas adyacentes, a una altitud aproximada de 20,200 Km., lo cual resulta en un periodo orbital cada 12 horas.

Segmento de control.

Está constituido por cinco estaciones de control repartidas alrededor del mundo y con coordenadas muy precisas. Todas ellas reciben continuamente las señales GPS con receptores de 2 frecuencias y provistos de osciladores de cesio. También se registran de forma precisa, otra serie de parámetros como presión y temperatura que afectan de manera importante a la propagación de la información que se recibe de los satélites.

Los datos se trasmiten a una estación principal para EU, situada en Colorado Springs, en donde se procesa la información, obteniendo de esta manera todas las posiciones de los satélites en sus órbitas (sus efemérides) y los estados de los relojes, de la más alta precisión que llevan cada uno de ellos, para que con posterioridad los mismos satélites radiodifundan dicha información a los usuarios; haciendo posible localizar, geográficamente al receptor con base en triangulación.

Segmento utilitario.

Está formado por todos los equipos utilizados para la recepción de las señales emitidas por los satélites, así como por el programa necesario para la comunicación del receptor con el ordenador y el procesado de la información para la obtención de los resultados.

Podemos considerar el "equipo GPS" compuesto por tres unidades principales: receptor, antena y accesorios.

La antena es el elemento al cual viene referido el posicionamiento, se conecta a través de un preamplificador al receptor, directamente o mediante cable. La función de la antena es la de conversión de energía electromagnética en corriente eléctrica que es conducida al receptor.

El receptor consta de una serie de elementos que se encargan de la recepción de las radiofrecuencias enviadas por los satélites. Además cuentan con diferentes canales para recibir simultáneamente la señal de varios satélites; un procesador interno con su correspondiente apoyo lógico; una unidad de memoria para el almacenamiento de información; teclado de control; pantalla de comunicación con el usuario, diferentes conectores para funciones varias y una fuente de alimentación, interna o externa.

Por último, también pueden emplearse accesorios como: trípodes, cables especiales, equipos de control meteorológico y diverso material auxiliar.

Obtención de la información.

Una vez estacionados en el punto requerido y con el equipo en funcionamiento, el receptor proporciona al operador, a través de la pantalla y con ayuda del teclado, una gran cantidad de información sobre la recepción que estamos realizando, tal como:

- Número y nombre de los satélites localizados.
- Satélites en seguimiento.
- Acimut de cada satélite en seguimiento.
- Elevación de cada satélite en seguimiento.
- La posición aproximada actual del receptor. (Longitud, latitud y altitud).
- Dirección y velocidad del movimiento, para navegación.
- Bondad de la geometría de observación

- Bondad de la medida que puede hacerse sobre cada satélite.
- Edad o antigüedad de la información ofrecida.
- Progreso de la observación: satélites que se pierden y captan, y número de observaciones realizadas a cada uno.
- Nombre y número de la sesión que damos a la estación de observación, así como la identificación del operador y notas varias.
- Registros meteorológicos y datos locales introducidos.
- Estado de la fuente de alimentación.

Estructura de la señal.

Cada satélite va provisto de un reloj-oscilador que provee una frecuencia fundamental de 10.23 MHz, sobre la que se estructura todo el conjunto de la señal radiodifundida por el satélite. El satélite emite información sobre dos portadoras, la primera es el resultado de multiplicar la fundamental por 154 (1,575.42 MHz) y se denomina L1. La segunda, utiliza un factor de 120 (1,227.60 MHz) y se denomina L2. El término "L" viene determinado porque los valores usados están en la banda L de radiofrecuencias que abarca desde 1 GHz a 2 GHz (1000 a 2000 MHz). El utilizar 2 frecuencias permite determinar por comparación el retardo ionosférico, difícilmente predecible por otros medios.

Sobre estas dos portadoras se envía una información modulada compuesta por dos códigos y un mensaje, generados también a partir de la frecuencia fundamental correspondiente. El primer código denominado C/A (course /adquisition) o S (standard), es una moduladora con la frecuencia fundamental dividida por 10 es decir 1.023 MHz. El segundo código llamado P (precise) modula directamente con la frecuencia fundamental de 10.23 MHz y por último el mensaje se envía con la bajísima frecuencia moduladora de 50 Hz.

Los códigos consisten en una secuencia de dígitos binarios o bits (ceros y unos). La modulación de las portadoras con éstos códigos genera un ruido electrónico que, en principio, no sigue ninguna ley y parece aleatorio, pero en realidad sus secuencias están establecidas mediante desarrollos polinómicos, este fenómeno se conoce con el término

ruido seudo-aleatorio (Pseudo Random Noise, PRN), y tiene la característica de que puede correlacionarse con una réplica generada por otro instrumento.

Cada uno de éstos códigos posee una configuración particular para cada satélite y constituye el denominado PRN característico, con el que se identifica los satélites del sistema GPS.

Sobre la portadora L1 se modulan los dos códigos vistos, el C/A y el P, además del mensaje correspondiente. En la L2 sólo se modula el mensaje y el código P.

El sistema que se utiliza en GPS para modular los códigos binarios se denominan Modulación Binaria por Cambio de Fase o Modulación Binaria Bifase.

El mensaje modulado sobre ambas portadoras tiene una duración de 12 min. y 30 s. Debido principalmente a su longitud y su baja velocidad de transmisión. La información que contiene viene referida a:

- Precisión y estado del satélite, ya que los satélites pueden encontrarse inoperantes.
- Antigüedad de la información y de las efemérides radiodifundidas.
- Almanaque y el estado de los relojes.
- Un modelo ionosférico, para el cálculo de los retardos.
- Información UTC (hora universal).
- Dos claves: - TLM, de telemetría, por si la órbita del satélite desde tierra.
  -HOW, que da acceso, para los usuarios autorizados, al código P.

Giróscopo Vertical

El giróscopo vertical es un instrumento que establece una referencia en relación a un sistema cartesiano. Generalmente, se utiliza para referirse a un eje en posición vertical, con respecto al plano de la superficie terrestre; por lo que normalmente coincide con el vector de gravedad, que une cualquier punto sobre la superficie con el centro teórico de la tierra.

Esa referencia es ampliamente utilizada en navegación en todos los medios conocidos: tierra, aire, mar, órbita, espacio interplanetario, y subsuelo. Asimismo, es una referencia esencial para establecer la orientación de un instrumento o sensor en el espacio geográfico: objetivo central de este trabajo.

Un giróscopo es un dispositivo electromecánico, donde el elemento esencial es un cilindro metálico o rotor, girando a alta velocidad angular sobre su eje. Para tener un giro libre, el rotor primero se encuentra pivoteando sobre una base, que a su vez se encuentra ortogonalmente sobre una segunda base, esto es la primer base gira en el eje 'x', la segunda en el eje 'y' y el rotor en el eje 'z'. Como se muestra en la figura 1.1, esto le permite al rotor girar libremente independientemente de la orientación del marco.

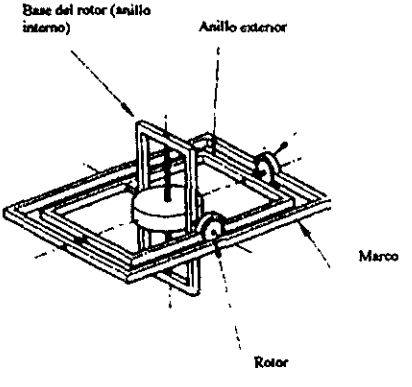


Figura 1.1. Esquema que muestra el giro libre del Giróscopo.

En operación, el giróscopo posee dos propiedades fundamentales importantes: la inercia giroscópica o rigidez y la precesión. Estas propiedades dependen del principio de que el momento angular de un cuerpo rotando en un eje dado permanece constante, a menos que alguna fuerza aplicada perturbe este estado.

Rigidez.

Esta propiedad resiste una fuerza cuando tiende a cambiar el plano de rotación del rotor de un giróscopo, y depende de tres factores: 1) la masa del rotor, 2) el momento angular del rotor, y 3) la distancia de la masa al eje de rotación.

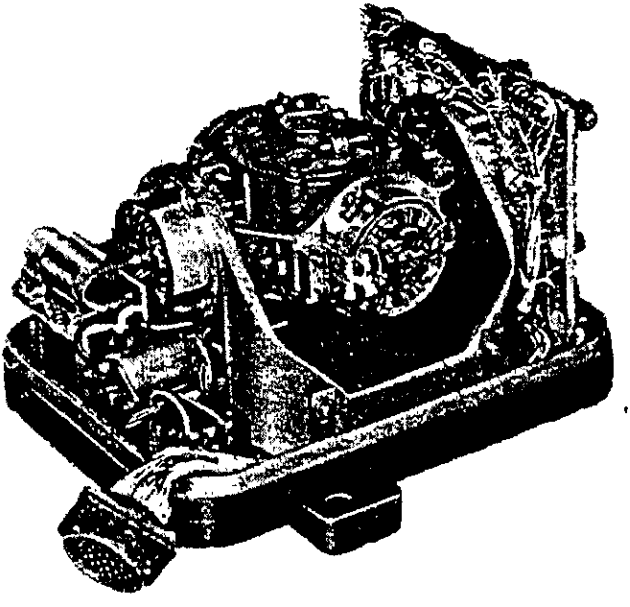


Figura 1.2 Estructura interna del Giróscopo.

Precesión.

La precesión es el cambio de velocidad angular en la dirección del plano de rotación bajo la influencia de la fuerza aplicada. El cambio de velocidad es proporcional a la intensidad de la fuerza aplicada, e inversamente proporcional al momento de inercia y momento angular del rotor. La fuerza resulta de la fricción de las monturas en pivote.

Acelerómetros

Los acelerómetros son dispositivos para de detectar movimiento lineal o curvilíneo y oscilatorio del objeto al cual están sujetos. Para que un acelerómetro detecte y genere datos útiles, debe estar correctamente fijado al objeto de prueba lo que requiere de que el montaje del acelerómetro sea rígido dentro del intervalo de frecuencias de trabajo.

Acelerómetros piezoresistivos.

Los materiales piezoresistivos tienen la característica de cambiar su resistencia bajo la influencia de presión física o de trabajo mecánico, así que si es sometida a solicitaciones mecánicas, esta resistencia interna cambiará y permanecerá fija hasta restablecer el estado original del material. Para detectar el cambio de resistencia, es necesaria una fuente de alimentación estable Los acelerómetros de banda utilizados en esta tesis son dispositivos pasivos.

La mayoría de los diseños del acelerómetro piezoresistivo utilizan el principio de viga libre, y están fabricados de silicio monocristalino como elemento sensor. Su marco de referencia como se dijo, se inmoviliza por fijación al objeto de prueba. El componente (piezoresistivo) de la resistencia variable se configura como puente de Wheatstone, en el cual la salida es un voltaje proporcional al cambio en resistencia, y por ello a las fuerzas asociadas

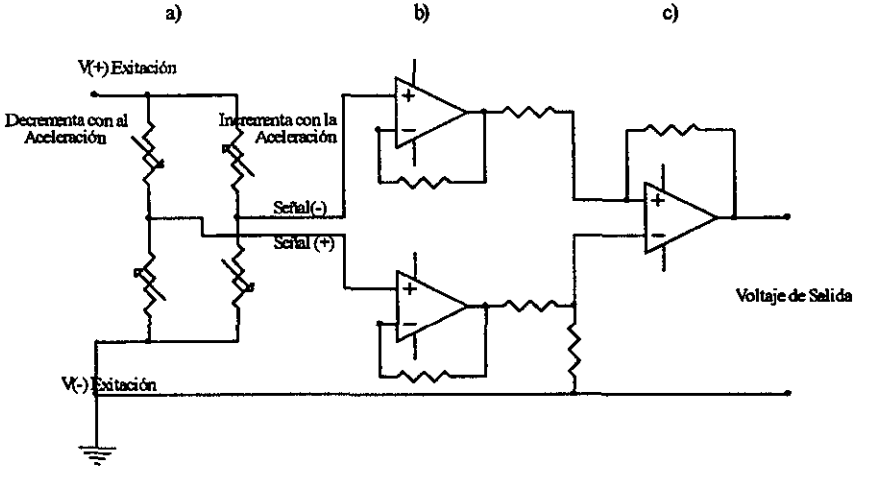


Figura 1.3. Circuito equivalente del acelerómetro piezoresistivo típico

La figura 1.3 muestra un ejemplo simplificado del circuito para la detección y amplificación de la temperatura usado por los acelerómetros de viga de silicio. El circuito se divide en tres partes principales:

a) El puente de Wheatstone, mide el cambio de tensión en la resistencia de la celda del acelerómetro y proporciona una diferencia correspondiente al voltaje de salida.
b) Los amplificadores actúan como almacén intermediario para reducir al mínimo la corriente trazada del puente y de amplificación de señales del puente, aumentando la diferencia del voltaje
c) El amplificador diferencial, mide la diferencia final del voltaje y entrega el voltaje de salida.

El circuito del puente de Wheatstone del acelerómetro piezoresistivo se puede utilizar también para negar la sensibilidad de la temperatura del sensor. Si los cuatro resistores en el puente se hacen del mismo material que la viga, los efectos de la sensibilidad de la temperatura deben cancelar del circuito

En este caso se utilizo un acelerómetro de silicio micromaquinado. El acelerómetro esta contenido en un módulo, junto con el elemento de detección (viga libre), preamplificador de la señal, compensador de temperatura, regulador de voltaje, referencia de voltaje y una salida lineal de ± 2 volts de corriente directa para una aceleración de ± 2g's. En la figura 1.4 se muestra un diagrama de bloques del modulo.

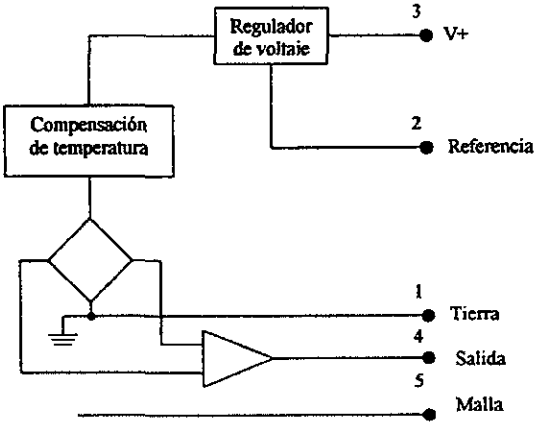


Figura 1.4. Diagrama de bloques del modulo del acelerómetro.

Brújula electrónica

La brújula del sistema es un sensor para aeronáutica comercial que utiliza como transductor una bobina sensible al campo magnético de la tierra, de modo que ofrece la orientación de la aeronave con respecto al Norte. La bobina se encuentra suspendida en un fluido dentro de un contenedor, cuyo propósito es amortiguar las vibraciones de vuelo y que junto con el arreglo mecánico que lo sujeta a la nave, permite mantener la referencia respecto al norte, aun en inclinaciones de ± 45° en los ejes de alabeo y cabeceo que son ortogonales respecto a la vertical.

El sensor emplea un magnetómetro toroidal con un núcleo flotando libremente en su centro, que esta compuesto de una bobina de acero inoxidable enrollado con cinta magnética. Una *señal de excitación de campo es aplicada al núcleo y el campo magnético terrestre* interactúa con este, produciendo un campo asimétrico de flujo en el núcleo. Este flujo cambiante es detectado por dos bobinas y la señal resultante se procesa para obtener la orientación.

El sensor incluye un sistema de compensación de desviación basado en una bobina de Helmhotz, que es utilizada para corregir las anomalías del campo magnético local. La corrección de la desviación generada de esta manera no es afectada por el al ángulo de inclinación, como lo son las brújulas convencionales y electrónicas. El campo magnético lineal generado por la bobina de Helmhotz se compensa cuando el núcleo está inclinado o nivelado.

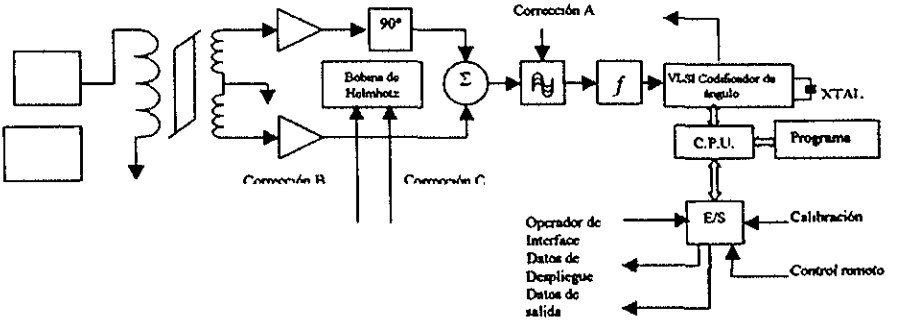


Figura 1.5. Diagrama a bloques de la brújula

El sistema que conforma la brújula se divide en tres partes:

- La circuiteria del magnetómetro, que incluye el manejo de la circuiteria estabilizadora y un núcleo con una bobina primaria y dos secundarias.

- El circuito de acondicionamiento de señal, que realiza la suma vectorial de las dos salidas en cuadratura. La señal resultante es filtrada en un pasa banda y alimenta a un circuito limitador para eliminar todos los efectos de variación de la intensidad del campo magnético.

- El codificador de ángulo y el microprocesador, el cual proporciona datos aislados, la corrección de la desviación y el formato de conversión de datos para transmisión serial o en otros formatos.

Microcontrolador MC68HC11F1

El microcontrolador usado es el MC68HC11F1, de Motorola, esta fabricado con tecnología HCMOS (Alta densidad Metal Oxido Semiconductor Complementaria). Es un MCU (Unidad microcontroladora) avanzado de 8 bits con funciones periféricas de alta complejidad dentro del mismo integrado. Contiene un ducto de datos no multiplexado con una velocidad nominal de 2 MHz (con un reloj de 8 MHz). Debido a la tecnología CMOS muestra una alta inmunidad al ruido , además de un bajo consumo de energía (50 mA).

El microcontrolador MC68HC11 es utilizado ampliamente en aplicaciones como por ejemplo: en sistemas de inyección electrónica de automóviles, equipos de monitoreo atmosférico, alarmas, equipos biomédicos, control de motores eléctricos, conmutadores telefónicos, tacómetros, velocímetros, etc.

Entre sus principales ventajas se encuentra el que pese ha ser un microcontrolador de 8 bits, cuenta con instrucciones para realizar algunas operaciones de 16 bits, como sumas y

divisiones, además de tener integrados varios sistemas que lo hacen ideal para realizar funciones de control. En la figura 1.6 se muestra el diagrama a bloques de este microcontrolador.

A continuación se mencionan las principales características de MCU, y en los capítulos siguientes se describe más a fondo acerca de aquellas que se usaron para el desarrollo de la *tarjeta de adquisición de datos, como parte de este trabajo.*

- Interfaz de comunicación serie (SCI) mejorada, con formato registro sin retorno a cero.

  Este sistema SCI puede ser usado para conectar una terminal CRT o una computadora personal al MCU, o diversos MCUs distribuidos a grandes distancias pueden usar sus subsistemas SCI para formar una red de comunicación serie.

  El SCI utiliza un formato estándar de no retorno a cero (NRZ) con una variación en la velocidad de transmisión (baud rate) derivada del circuito del cristal del reloj. La interfaz se realiza mediante las terminales del puerto D. D0 es usada para la recepción de datos (RxD) y D1 para la transmisión (TxD).

- Convertidor Analógico-Digital (A/D) de 8 bits.

  Convierte voltajes de 0 V a 5 V, con 255 valores, esto es 0V= 00 y 5V= FF

- Encapsulado de 68 terminales.

- La figura 1.7 nos muestra el encapsulado con el numero y nombre de cada una de las terminales.

- Interfaz periférica serie (SPI).

  La interfaz periférica serial (SPI) es uno de los dos subsistemas periféricos incluidos en el MCU. Es usado principalmente para permitir al MCU comunicarse con dispositivos periféricos y también puede interprocesar comunicaciones dentro de un sistema maestro múltiple, y ser configurado como un dispositivo maestro o como dispositivo que forme parte de un subsistema mayor.

- 512 bytes de EEPROM.

  Esta memoria al ser eléctricamente borrable y programable, se puede manejar desde programa y guarda el programa principal con el cual se programa el MCU.

- Mecanismo de protección de bloques de memoria para EEPROM y CONFIG.
- Ducto de datos expandido no multiplexado.
- Modos "STOP y WAIT" de ahorro de consumo de energía.
- Capacidad de direccionamiento de memoria de 64 K.
- Circuito acumulador de pulsos de 8 bits.
- 1024 bytes (1K) de RAM estática.
- Instrucciones de brinco y prueba de bits
- Circuito de interrupción de tiempo real.
- Cuatro "chip selects" programables
- Sistema vigilante de operación correcta del computador (COP).

Microcomputadora.

La computadora utilizada se basa en un microprocesador Intel 386, es una computadora AT compatible, que incorpora la capacidad de una computadora convencional en un sistema de 3.3 kilogramos de peso. Tiene una velocidad de reloj de 20 MHz, y conmuta a 4 MHz cuando no se utiliza, ahorrando energía Sus principales características son la presencia de

una pantalla de cristal líquido con retro-iluminación, que nos permite la clara observación de los parámetros en diferentes condiciones de iluminación, a las que se somete durante los vuelos de prueba. Además requiere un bajo consumo de potencia, en comparación con otros tipos de monitor.

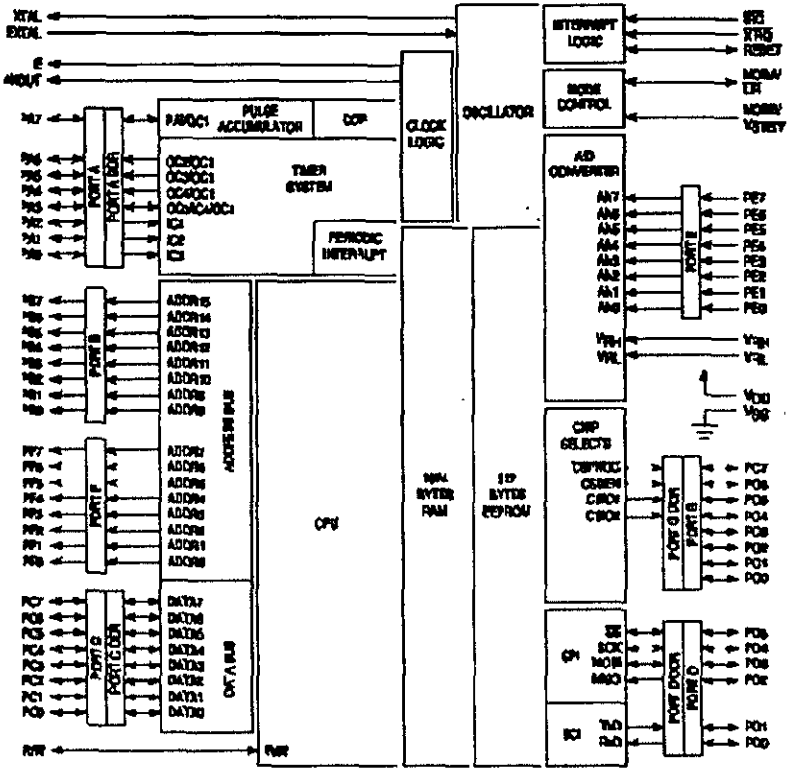


Figura 1.6. Diagrama a bloques del MCU MC68HC11

El sistema cuenta con dos canales en serie RS232 de tal manera que en uno de ellos esta conectado el receptor GPS y en el otro la comunicación con el sistema del

microprocesador. La computadora funciona con baterías propias de Níquel-Cadmio, durante periodos que alcanzan hasta dos horas. Esta equipada con un disco duro de 60 MB y uno flexible de 1.44 MB, y cuenta con un compartimiento para expansión de memoria de tipo DRAM, y una serie de indicadores que describen el estado de funcionamiento en la pantalla de las baterías, la velocidad del procesador; tiene incluido el teclado. Por ser una computadora portátil su diseño incluye tolerancia a vibraciones, como las presentes normalmente en vuelo.

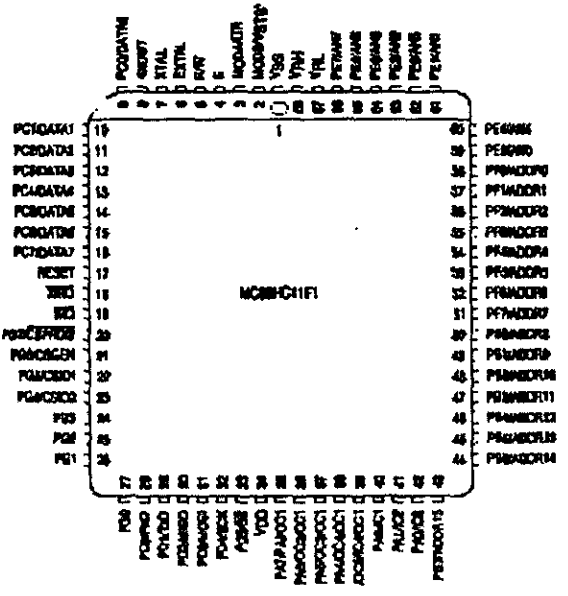


Figura 1.7. Asignación de patas para el encapsulado del MC68HC11F1.

Capítulo 2

Sistema Basado en el Microcontrolador

Capítulo 2.
Desarrollo del sistema basado en el microcontrolador.

Este capitulo trata sobre como se integran los dispositivos a la tarjeta receptora de datos provenientes de los sensores analógicos; por lo que en primer lugar se tienen que digitalizar para que el microcontrolador pueda leerlos y procesarlos.

Conversión de las señales analógicas en digitales.

El convertidor utilizado es el AD574A, con un voltaje de 0 a 10 V, lo que proporciona una salida de 12 bits que va de h000 a hFFF, o 4096 valores de voltaje.

La salida de los dispositivos no es mayor a 5 V, esto ocasiona que al convertirlos en binario sólo se puedan tener 2048 valores, desperdiciando la mitad de la capacidad del convertidor, por lo tanto lo primero que se hace es amplificar estas señales para aprovechar al máximo la resolución del convertidor.

Amplificación de las señales.

La primera parte de la tarjeta se basa en una serie de amplificadores para incrementar el valor del voltaje de las señales, estos amplificadores son también no inversores.

Circuito de la brújula.

La brújula proporciona una salida lineal de 0.1 V a 5.1, para manejar un intervalo de 0 a 10 V se conecto un amplificador operacional sumador no inversor. Por medio de un divisor de tensión se suma a la entrada una señal de –0.1 V, y la señal es amplificada al doble, con esto obtenemos la señal deseada de 0 a 10 V.

De acuerdo a las mediciones realizadas, para una orientación de 0° respecto al Norte el voltaje de salida de la brújula es de 0.1 V y para una orientación de 359.9° respecto al norte un voltaje de salida de 5.1 V. De acuerdo a esto tenemos que cada variación de 0.1° equivale a 13.89 mV., pero al pasar a través del convertidor analógico digital, el valor de 5.1 V equivale a una salida con un valor decimal de 2047, lo cual para la resolución de la brújula no era adecuado, por lo tanto se agrego un amplificador operacional entre la salida de la brújula y el convertidor de manera que la señal a la salida sea de amplitud suficiente para que proporcione una salida en valor decimal de 0 a 3599, para un valor de 8.7866 V.

El diseño se realizó con base en un amplificador sumador, que elimina la deriva generada por la brújula, de manera que el voltaje a la salida del amplificador varíe de 0.0 V a 8.7866 V, esto se logro agregando en la entrada del sumador un voltaje negativo de 0.1 V y así amplificar solo la señal de entrada que se encuentra en el intervalo de 0.0 V a 5.0 V. El circuito completo se encuentra en la figura 2.1.

Este arreglo nos permite que la señal que llega al convertidor, tenga una mayor resolución a la salida del mismo. El diodo zener a la salida del operacional se encuentra limitando el voltaje, para evitar que sea mayor al que puede recibir el convertidor.

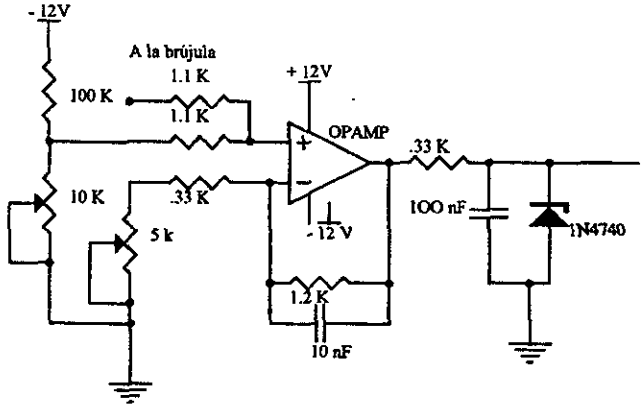


Figura 2.1. Circuito de amplificación para la señal de la brújula.

Giróscopo.

Al igual que a la brújula, al giróscopo se le acondicionaron amplificadores operacionales, uno para cada salida (alabeo y cabeceo).

El giróscopo también entrega una salida lineal que dependiendo de la inclinación varia según el ángulo del giróscopo; de esta manera cuando se obtiene una señal de 2.4 V en reposo (0° de inclinación), se amplifica ésta hasta tener 5 V, para así permitir una salida en valor decimal de 2048, cuando el giróscopo se encuentra la posición horizontal (0° alabeo, 0° cabeceo) y de acuerdo al movimiento en una dirección u otra, este valor aumenta o disminuye, traduciéndose en una lectura directa para la PC. Para la elaboración de este amplificador se tomo como base el de la brújula, pero sólo se tiene la entrada del giróscopo en la entrada no inversora del operacional, en este caso se utilizó un TL081 (un circuito dual) para así combinar los dos amplificadores en un integrado de 8 patas.

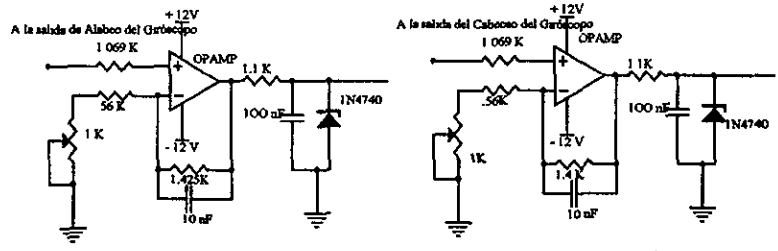


Figura 2. 2. Amplificadores usados para el giróscopo.

Acelerómetros.

Los acelerómetros usados son los piezoresistivos que responden al intervalo de 0 a 1 KHz, y proporcionan una salida analógica de 2.488 V en aceleración de 0 g, con variaciones de ± 2g's de acuerdo a la orientación del dispositivo.

Se acondicionó la señal que proporciona cada acelerómetro a manera de tener 5 V en 0 g, 1 V para –2g y 9 V para 2g, como se muestra en la figura 3.3. Estos son los circuitos que amplifican la señal de cada acelerómetro.

Multiplexaje de las señales.

Una vez amplificadas las señales, se digitizan, para lograr esto con un sólo convertidor, se requiere de un multiplexor para que cada una de las señales sea digitizada por separado. En este caso como las señales son analógicas, se requiere de un multiplexor analógico, que no afecte a la señal de entrada, y proporcione la señal de salida.

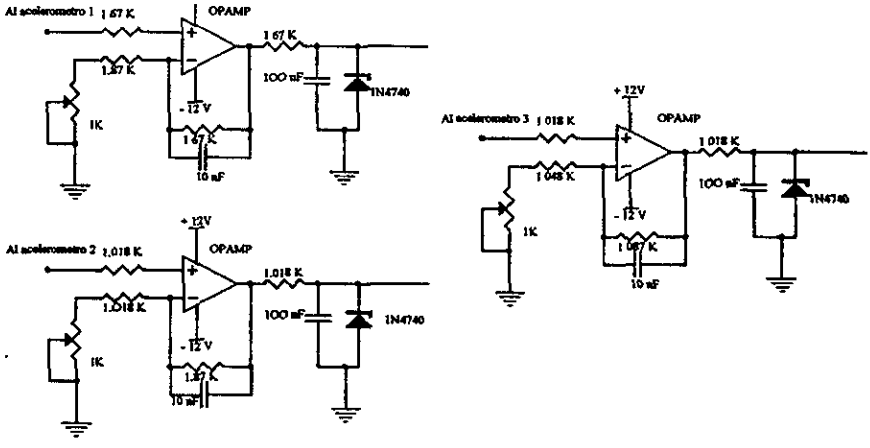


Figura 2.3. Circuitos amplificadores para los acelerómetros.

Se utiliza un multiplexor analógico de ocho entradas y una salida, lo que permite manejar los seis dispositivos que requieren convertir sus señales analógicas en digitales. El multiplexor no requiere de circuiteria externa, sólo de una alimentación de 12 V, conectándose directamente las salidas de los operacionales a las entradas del multiplexor, y de la salida de este a la entrada de 0 a 10 V del convertidor.

Conversión de las señales analógicas en digitales.

El convertidor analógico digital usado es el AD574A, que es un convertidor con salidas en paralelo y resolución de 12 bits, proporcionando una salida de 0 a 4096 valores según sea la intensidad del voltaje.

Para operar se requiere alimentar con +5, +12 y –12 V, así como una circuiteria externa que le permita convertir en el intervalo de 0 a 10 V, ya que este convertidor funciona en diferentes intervalos de V como son –5 a 5 V, -10 a 10 V, 0 a 20 V y 0 a 10 V los cuales

son de interés. Además requiere de un reloj para realizar la conversión, ya que cuando la pata 3 se encuentra en alto, toma la señal analógica que se encuentra en ese momento y al cambio a bajo realiza la conversión. El circuito de reloj se implementó con un multivibrador 555, ya que la frecuencia del AD574 es baja, pues tiene que ser menor a 26 KHz.

En el arreglo externo se cuenta con dos potenciómetros que regulan el balance y la ganancia del convertidor. La señal digital (salida paralela de 12 bits) se encuentra conectada a dos puertos del microcontrolador. La figura 2.4 muestra la conexión y circuiteria empleados en el ADC.

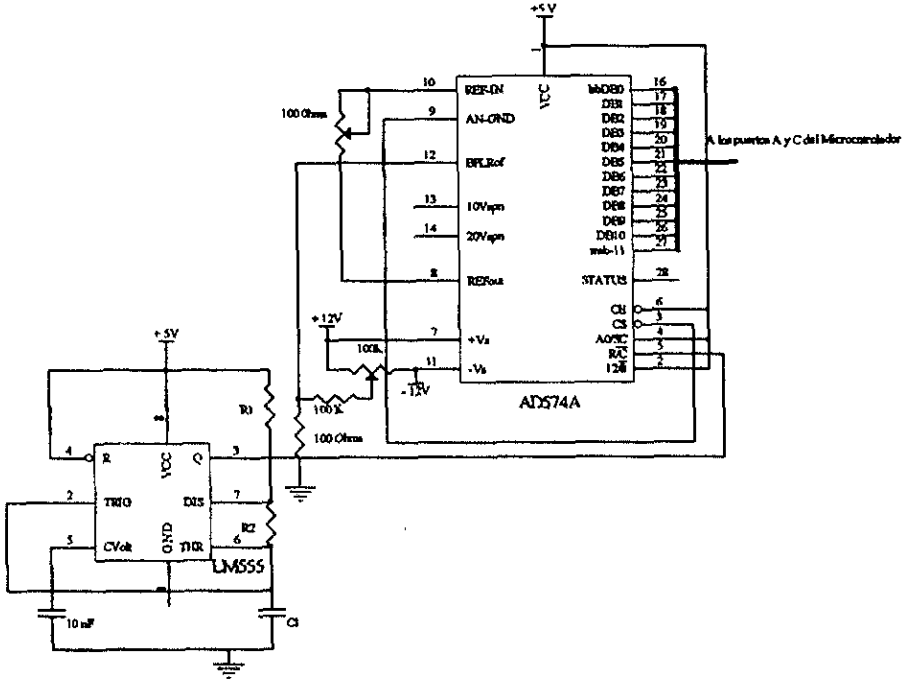


Figura 2.4 Convertidor Analógico Digital con circuito de reloj.

Procesamiento de las señales.

La ultima sección de la tarjeta esta formada por la circuiteria necesaria para que el microcontrolador pueda funcionar y conectarse a una PC.

Para que el microcontrolador pueda operar y comunicarse con la PC, requiere de ciertos circuitos adicionales para esa tarea. Los circuitos mínimos que necesita para su operación son: un circuito de reloj, un circuito que proporcione un bajo voltaje para el restaurador, una interfaz RS232 para la comunicación serial.

Circuito de reloj.

El reloj, que determina la velocidad a la que se ejecutan las instrucciones, es en este caso de 8MHz, conectándose en las terminales XTAL y EXTAL del encapsulado, estas terminales habilitan la interfaz para un cristal o circuito de reloj compatible con CMOS, para controlar la circuiteria interna generadora del pulso del tiempo.

Este reloj se conforma de un cristal conectado en paralelo a las terminales XTAL y EXTAL, así como una resistencia de 100 MΩ y dos capacitores conectados a tierra como se muestra en la figura 2.6.

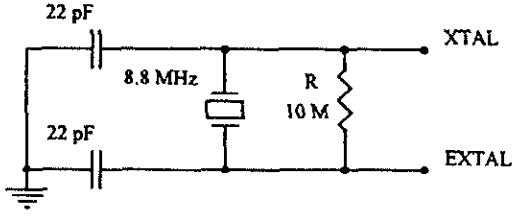


Figura 2.6. Conexión común del cristal.

Circuito de comunicación.

Para comunicar el microcontrolador con la PC, se emplea el CI Max232, que posibilita una comunicación serial, y se conecta a las terminales del puerto D. Dichas terminales RxD (PD0)y TxD (PD1), se habilitan para la transmisión y recepción de datos. En el Apéndice B, se muestra el programa que maneja las configuraciones para habilitar las salidas del puerto D.

El arreglo de este CI proporciona una interfaz serie RS232 (DB9) común en las computadoras personales. En la figura 2.7, se muestra el circuito de dicha interfaz.

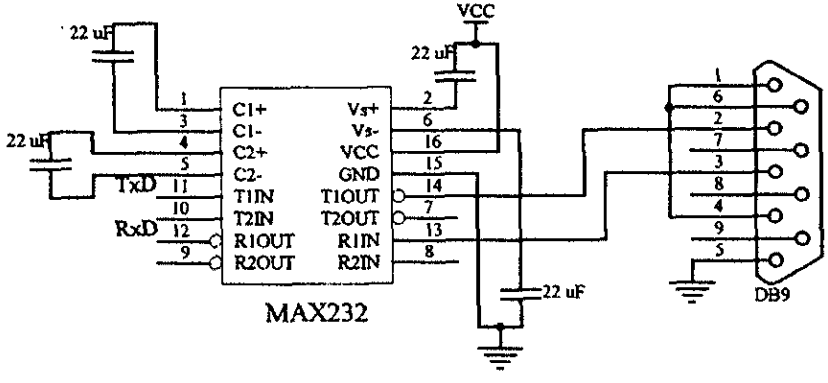


Figura 2.7. Interfaz de comunicación serial (RS232)

Microcontrolador.

Una vez que todos los circuitos que requiere el microcontrolador para operar se encuentran armados, se conectan en cada uno de las patas correspondientes. Al microcontrolador, a demás del reloj y la interfaz de comunicación serie se agrega un circuito para poder aplicar un bajo voltaje en la pata de restituir el funcionamiento, por medio del cual se inicia el microcontrolador, así como algunos filtros para eliminar armónicas y voltaje de Vcc y Vss a las patas 2 y 3 para indicarle el modo de operación.

Para nuestro propósito, el microcontrolador opera en modo simple, por lo que se coloca Vcc (5V) a la pata 2 y Vss (0V) a la pata 3. Con estos voltajes, el microcontrolador inicia la ejecución del programa previamente cargado en EEPROM. Al momento de presionar el botón de restituir la tarjeta, este programa selecciona cada dispositivo que va a entrar al ADC según sean requeridos; así como comunicar al microcontrolador con la PC y enviar los datos obtenidos de los dispositivos para su procesamiento y muestra en pantalla: los detalles se verán más adelante.

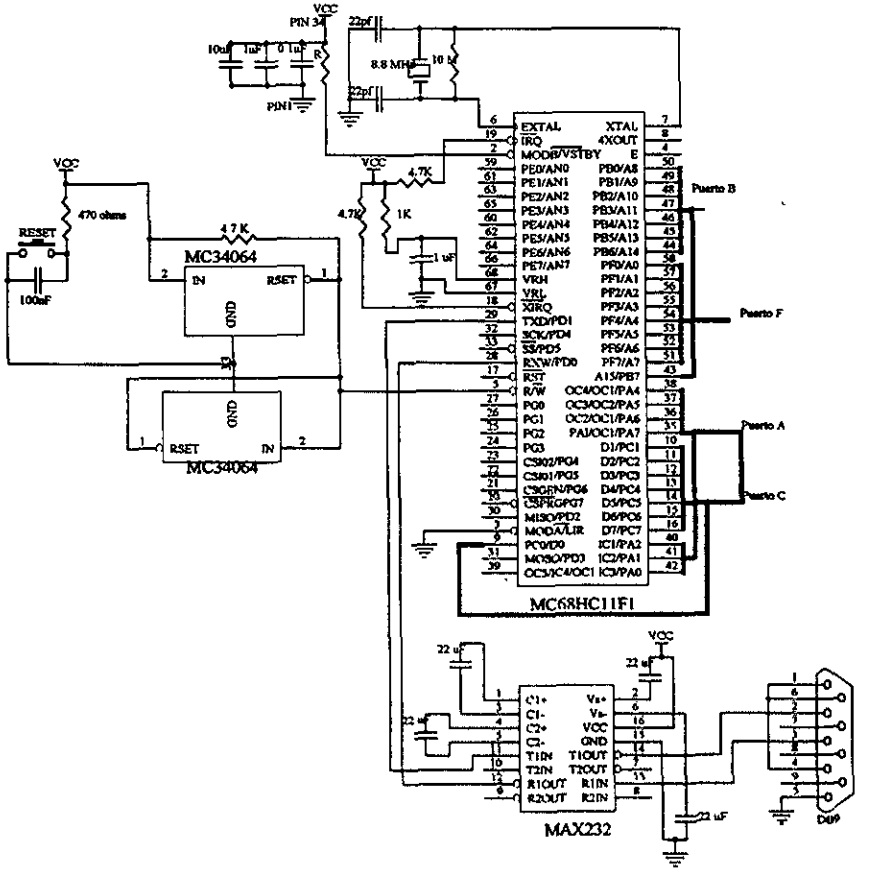


Figura 2.8. Microcontrolador con circuitería para operación en modo simple.

Capítulo 3

Evaluación del Sistema

Capítulo 3.
Evaluación del Sistema.

En este capítulo se verá el funcionamiento de cada uno de los dispositivos creados, primero por separado y después en conjunto para así demostrar el cumplimiento de los objetivos del trabajo.

Para ello primero se diseñaron cada uno de los amplificadores necesarios y se evaluaron en forma individual. Una vez evaluados y al verificar un funcionamiento correcto, se procedió a la evaluación por partes:

- Sistema de adquisición de datos.
- Digitalización de las señales.
- Comunicación con la PC.
- Finalmente se evalúo el sistema completo.

La información se obtuvo básicamente por medio de gráficas que describen el comportamiento del parámetro a evaluar en función del voltaje o algún otro parámetro.

Sistema de adquisición de datos.

Brújula electrónica.

Una vez que se terminaron los circuitos necesarios para la amplificación de la señal, se procedió a realizar las pruebas necesarias para evaluar su desempeño. Se conecto la brújula a la entrada y se oriento al Norte para tener una salida de 0.0 V, al ubicarla en esta posición se detecto una pequeña inestabilidad (±0.2°) debido a que falta una rejilla de compensación

en la brújula (la cual se extravió antes de que se realizara este proyecto), por lo que no pudo ser corregida. Cuando la estabilidad fue máxima, por medio de los potenciómetros de calibración del circuito amplificador se establecieron los valores a la entrada y salida del amplificador de acuerdo a como se muestran en las graficas 3.1 y 3.2. Esta prueba se realiza lejos de materiales ferrosos, y campos magnéticos artificiales, con el fin de minimizar el error de lectura.

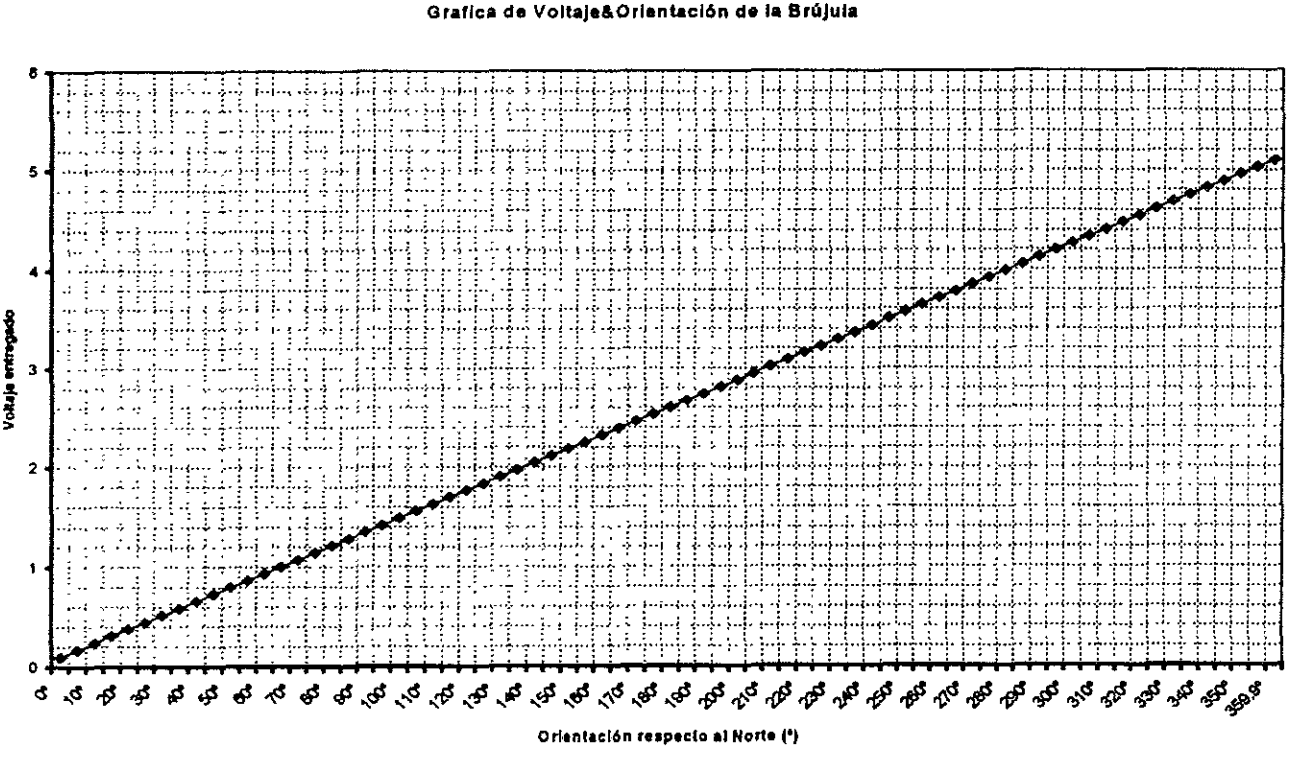
Giróscopo Vertical.

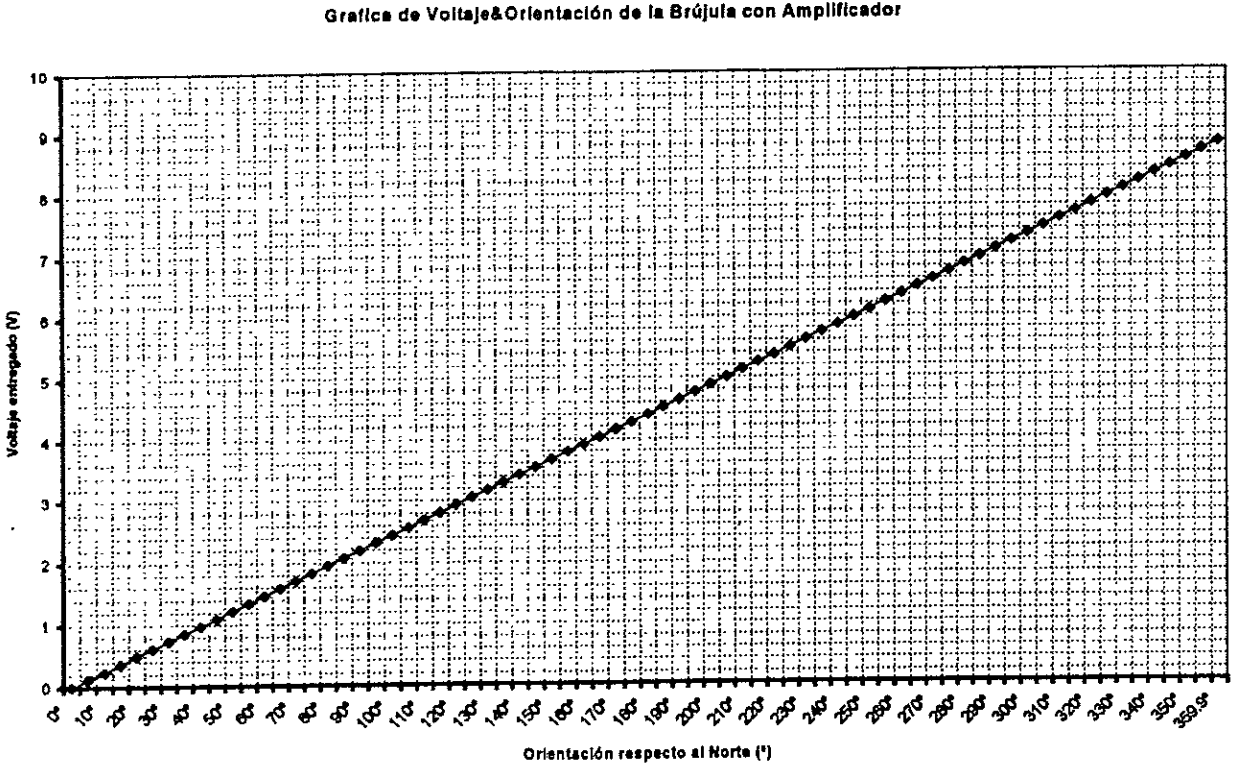
El giróscopo maneja dos salidas analógicas, una para el alabeo y otra para el cabeceo, dicha salida varía en amplitud de voltaje según la orientación. En las graficas 3.3 y 3.4 se puede apreciar como las variaciones de voltaje proporcionadas por las salidas del aparato no tienen una gran variación (± 1 V), por lo cual es indispensable amplificar la salida para tener una mejor resolución.

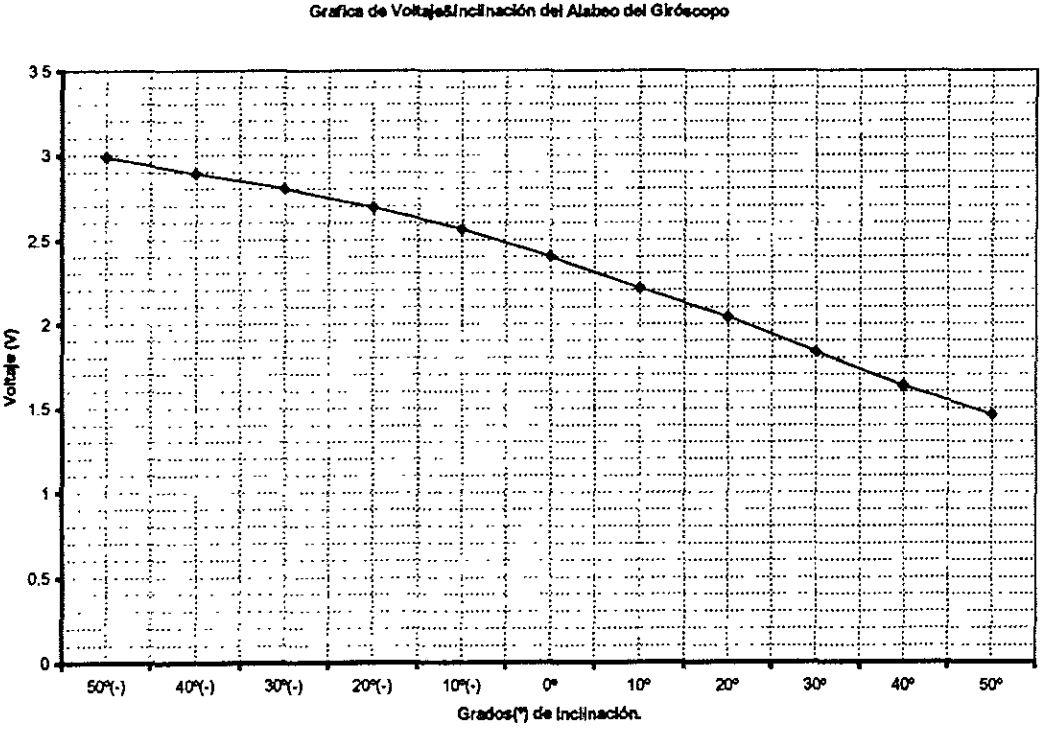
Debido a que el giróscopo consume mucha corriente (10 Amp.), se utilizaron unos optoacopladores para realizar una conexión segura entre este y los amplificadores, ya que el circuito del microcontrolador requiere de muy poca corriente y podría dañarse.

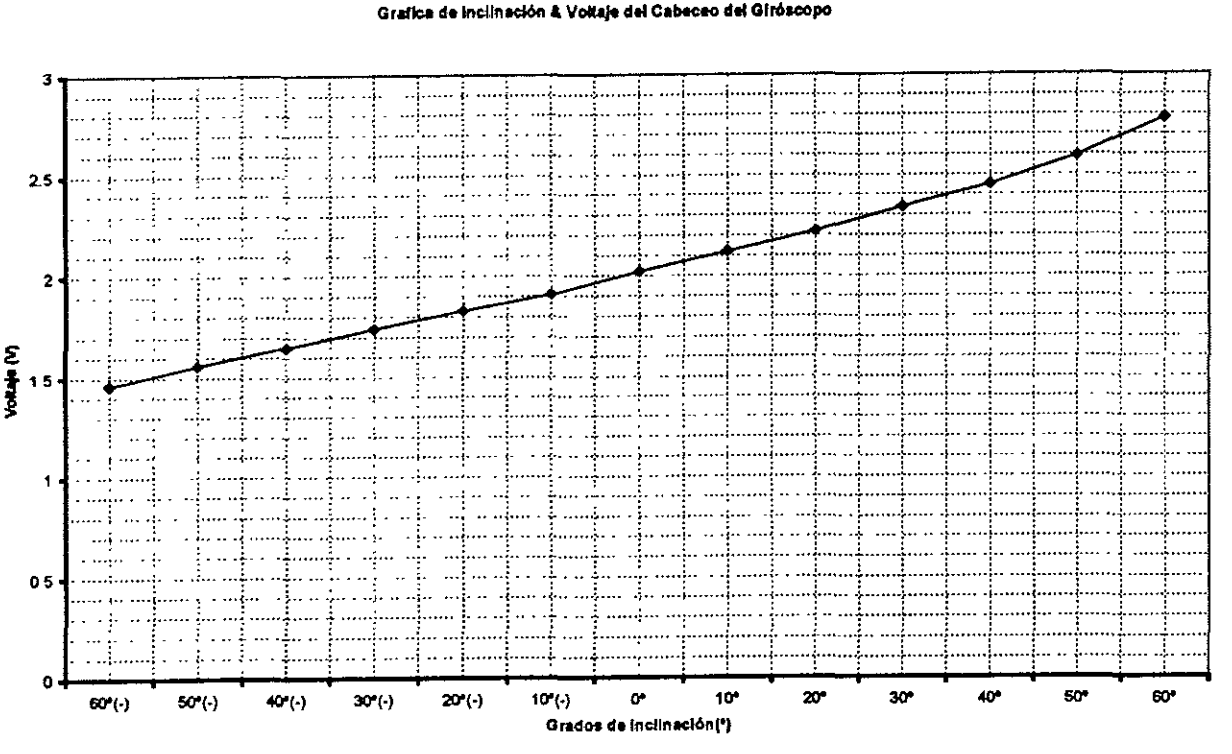
Acelerómetros.

Para el correcto uso de los acelerómetros se ajustaron a 5 Vcd para una aceleración de 0 g's, de tal manera que para –2 g's se cuente con +1 Vcd y para una aceleración de 2 g's se tenga una salida de 9 V cd, respectivamente.


Grafica de Voltaje&Orientación de la Brújula
Voltaje entregado
0
1
2
3
4
5
6
0°
10°
20°
30°
40°
50°
60°
70°
80°
90°
100°
110°
120°
130°
140°
150°
160°
170°
180°
190°
200°
210°
220°
230°
240°
250°
260°
270°
280°
290°
300°
310°
320°
330°
340°
350°
359.9°
Orientación respecto al Norte (°)


Grafica de Voltaje&Orientación de la Brújula con Amplificador
Voltaje entregado (V)
0
1
2
3
4
5
6
7
8
9
10
Orientación respecto al Norte (°)
0°
10°
20°
30°
40°
50°
60°
70°
80°
90°
100°
110°
120°
130°
140°
150°
160°
170°
180°
190°
200°
210°
220°
230°
240°
250°
260°
270°
280°
290°
300°
310°
320°
330°
340°
350°
359.9°


Grafica de Voltaje&Inclinación del Alabeo del Giróscopo
Voltaje (V)
3.5
3
2.5
2
1.5
1
0.5
0
50°(-)
40°(-)
30°(-)
20°(-)
10°(-)
0°
10°
20°
30°
40°
50°
Grados(°) de inclinación.


Grafica de inclinación & Voltaje del Cabeceo del Giróscopo
Voltaje (V)
3
2.5
2
1.5
1
0.5
0
60°(-)
50°(-)
40°(-)
30°(-)
20°(-)
10°(-)
0°
10°
20°
30°
40°
50°
60°
Grados de inclinación(°)

Digitalización de las señales

Para que el microcontrolador pueda leer los datos es necesario que las señales analógicas sean convertidas en digitales, para este propósito se diseño el circuito del convertidor analógico-digital. Para la evaluación de su desempeño, y aprovechar al máximo la velocidad de conversión, se diseño el reloj (multivibrador estable) en una frecuencia de 10 KHz, velocidad que se encuentra cercana a la máxima que puede aceptar el convertidor (26 KHz). Una vez que el reloj está en la frecuencia necesaria, se conectó a una fuente variable con un voltaje de 0.0 sin compensación, se varía la fuente como se indica en la tabla 3.5 desde 0.0 V hasta 10.0 V para así obtener los diferentes valores del dato en forma hexadecimal, necesaria para el microcontrolador (000 para 0 V y FFF para 10 V).

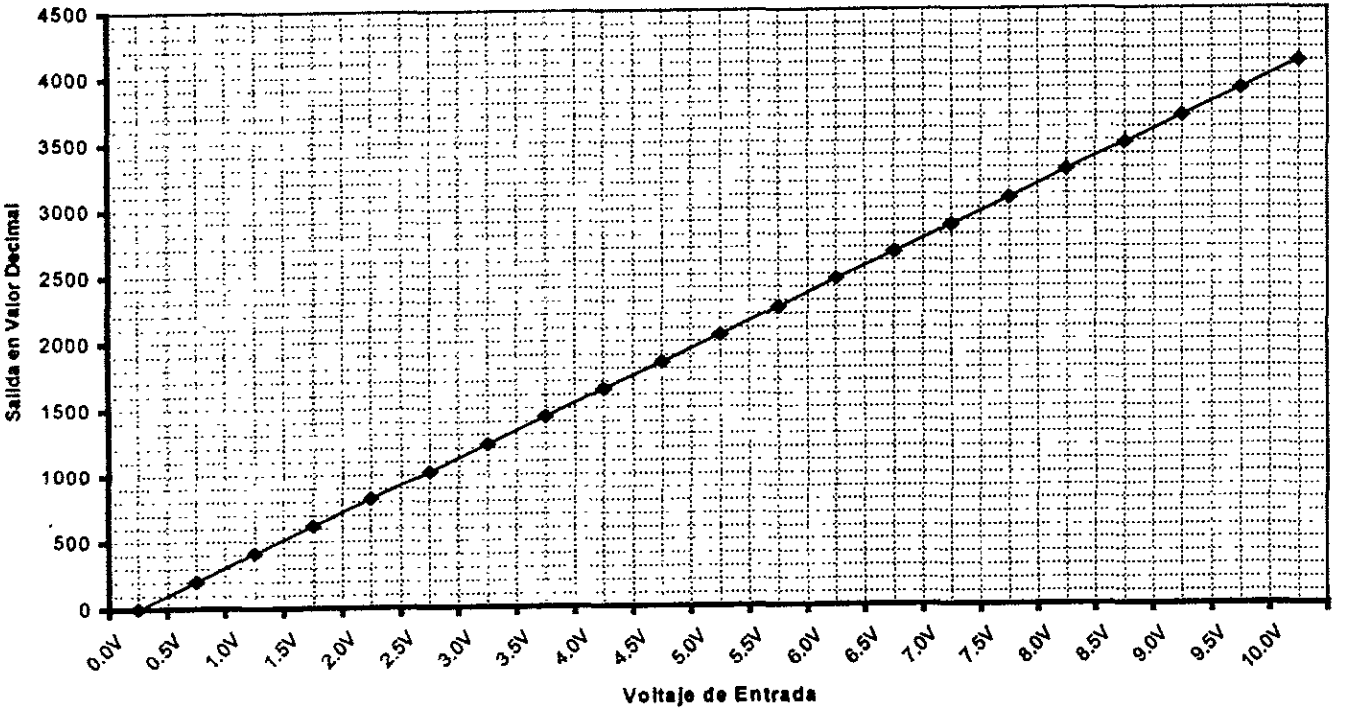
El circuito del convertidor cuenta con dos potenciómetros para la calibración del mismo, uno encargado de eliminar el balance o compensación que pueda encontrar a la entrada la señal analógica, y el otro de la precisión de la conversión; por medio de éstos se llegó a obtener una conversión lineal en relación a la señal de entrada.

Sistema del Microcontrolador y comunicación con la PC

El microcontrolador se destina como interfaz entre el sistema de navegación y la PC, por lo que es necesario establecer una comunicación entre ambos, *vía* RS232 o *puerto serial con* conector DB9 (ambos comunes en las PC)

Para evaluar el microcontrolador, se diseño un circuito formado por el microcontrolador, un circuito para restaurar, un reloj y un circuito de comunicación formado por el C.I. MAX232 y el conector DB9.

Grafica del Convertidor Analógico/Digital



Las primeras pruebas se basaron en el programa Avsim11 que es un sistema de simulación del desempeño del microcontrolador; las pruebas se centraron en la lectura de datos por medio de un puerto, algún proceso con ese dato (como una suma o una división) y revisar la respuesta en otro puerto. Listo el programa, se grabo en la EEPROM (memoria de sólo lectura eléctricamente borrable-programable) y se procedió a evaluar si los puertos se encontraban en buen estado y si el microcontrolador podría explotarlos para los objetivos del trabajo. Con a esta prueba, se encontró que un primer microcontrolador venía con un puerto dañado, por lo que se sustituyó.

Durante las pruebas anteriores, el microcontrolador se encontraba conectado a la PC en modo de prueba, y las instrucciones se proporcionaban por medio del programa de evaluación; que en este caso es el programa PUMA, que indica donde inicia el programa y el tiempo de ejecución, ya sea instrucción por instrucción o todo el programa Una vez operando correctamente el microcontrolador, se procedió a probarlo independientemente del programa de evaluación, se instaló el programa en una nueva dirección de memoria, asignada al momento de programar el microcontrolador y que debe especificarse en el programa. Para grabar el microcontrolador se utilizó el programa Pcbug11, el cual se carga con el microcontrolador funcionando en modo de prueba. Una vez cargado el programa, se colocó el microcontrolador en modo simple y con el botón de rehabilitar el microcontrolador empezó a ejecutar el programa; leía un dato por el puerto especificado y por medio de unos diodos luminosos (leds), era posible observar el resultado en el otro puerto, comprobando así que el microcontrolador puede operar sin la necesidad del programa de evaluación y del cable de conexión a la PC.

La prueba final, consistió en conectar el microcontrolador a la PC, sin el programa de evaluación, pero con un programa externo, que en este caso consistía en dos programas, uno cargado en el microcontrolador y otro en la PC. El programa de la PC, envía un dato a través del puerto serial, al recibir dicho dato, el microcontrolador le da salida en uno de sus puertos paralelos por medio de diodos luminosos, y a su vez, manda otro dato a la PC, para comprobar que la comunicación entre ambos se realiza sin problemas como resultado de varias pruebas, donde se varia el tiempo entre el envió y la recepción de los datos.

Sistema Completo

Para evaluar el desempeño del sistema, se procedió a analizar las diferentes componentes del mismo, ya conectadas entre sí, esto es desde la adquisición de los datos proporcionados por los dispositivos, hasta su visualización en la pantalla de la PC.

Se conectaron los seis dispositivos a sus respectivos amplificadores, comprobándose su respuesta ante variaciones en el valor de la señal. Para el ingreso de las señales al convertidor, se incorporó un multiplexor analógico, para que sólo recibiera la señal del dispositivo requerido por la PC, para esto la PC manda un dato, que indica al controlador que está lista para leer el dato de un determinado dispositivo, el microcontrolador selecciona este dispositivo a través del multiplexor y así convierte la señal analógica del dispositivo deseado en un dato digital que puede ser interpretado por la PC.

Figura 3 1. Sistema Completo

El dato de salida del convertidor es almacenado en el acumulador D del microcontrolador para enviarse a la PC. El programa del microcontrolador separa en dos el dato para enviarlo de manera asíncrona en dos bytes, a una velocidad de 9600 baudios. Ya con los datos en la memoria temporal de la PC, el programa se encarga de unir los bytes para formar el dato original, y transformarlo en un dato coherente para la persona observando la pantalla de la PC.

Al inicio de está tesis, se hace referencia a otro dispositivo el cual no se ha tratado aún, el GPS, el cual se integra en la PC, y no al sistema del microcontrolador. El programa que se encuentra cargado en la PC, es el encargado de leer los datos que llegan a través de los puertos seriales, en el puerto 1 se reciben los datos del GPS, que llegan en cadenas, cada cadena es precedida por un encabezado, que identifica la cadena. En estas cadenas viene toda la información que manda el GPS, como son las coordenadas; latitud y longitud, la altitud, el error estimado, los satélites visibles, la hora GMT, la fecha, etc, para el propósito de este trabajo, interesan, las coordenadas Latitud y Longitud, la altitud, así como la fecha y hora.

Al efectuar unas pruebas (de laboratorio, porque no se pudieron realizar de campo debido a suspensión de actividades en la UNAM por 10 meses), con el sistema completo: la brújula, el giróscopo, los acelerómetros y el GPS, el sistema de orientación nos proporcionó las coordenadas del laboratorio, la orientación en los tres ejes: nuestra orientación respecto al norte por medio de la brújula, y las inclinaciones en los ejes 'x' y 'y', con el giróscopo.

La forma en que se muestran los datos, se encuentra en la figura 3.2.

Análisis de Resultados y Corrección de Fallas.

El sistema integrado cumple con el objetivo inicial, muestra los datos para la orientación espacial de un vehiculo autónomo en una posición dada. Muestra la orientación en coordenadas en latitud, longitud y altitud, con una precisión de ± 15.0m (dado que es un GPS comercial), la orientación con respecto al norte esta dada con una precisión de 0.1°,

que es la proporcionada por la brújula, y en el giróscopo se pueden detectar inclinaciones hasta de ± 60°, con resolución de ± 0.25°.

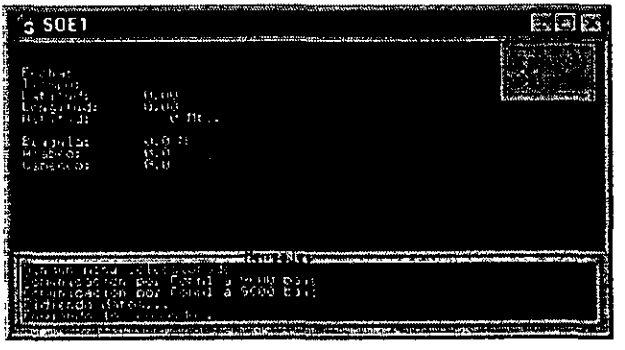


Figura 3.2. Presentación del Programa en la Pantalla.

Las principales imprecisiones se encuentran en la brújula, ya que por la falta de una rejilla, da incertidumbre de ± 0.2°, pues la falta de estas rejillas permite el paso de señales parásitas (ruido), que alteran la posición de la brújula.

Capítulo 4

Manejo y Mejoras del Aparato

Capítulo 4.
Manejo y Mejoras del Sistema.

El manejo.

El sistema de orientación puede ser usado en cualquier computadora personal que tenga dos puertos seriales, y que sea 486 o mejor. Sin embargo para un desempeño optimo se recomienda que se use una computadora Pentium a 200 MHz, con sistema operativo Windows 95, o más reciente.

El sistema sólo requiere de los programas incluidos en el Apéndice B, para su funcionamiento.

Instrucciones Específicas

El sistema, en la carátula frontal tiene el interruptor de encendido y un botón de rehabilitar; en caso de que no responda al encendido, basta con oprimir el botón y funcionará de inmediato. Así mismo, en la parte posterior se encuentran cinco entradas, una con siete patas para la brújula, una con cuatro para el giróscopo y tres con cinco para los acelerómetros.

En el caso de la brújula y los acelerómetros, la alimentación es proporcionada por el sistema, ya que no requieren de fuentes dedicadas para su funcionamiento. El giróscopo requiere de una alimentación de +27 V con un amperaje de 10 A. Por lo cual necesita de una fuente externa para funcionar. En la entrada para el giróscopo tenemos una alimentación de + 12 V, que se requieren para el acoplamiento óptico necesario entre el sistema y el giróscopo y sin el cual el nivel de ruido es inadecuado.

Como ya se menciono, para que el sistema nos de una referencia de donde estamos, es necesario que la PC tenga dos puertos seriales DB9, en el puerto serial 1, se coloca el cable del GPS, en el puerto serial 2 se coloca el cable del sistema integrado para esta tesis. Ya que se encuentran conectados ambos dispositivos, se encienden y se ejecuta el programa.

El programa desarrollado, es el que se encuentra en el Apéndice B, Una vez instalados los dispositivos, al ejecutarse el programa, aparece una pantalla como la de la figura 4.1., con tres opciones, estas son:

- Archivos>
- SOE>
- GPS>

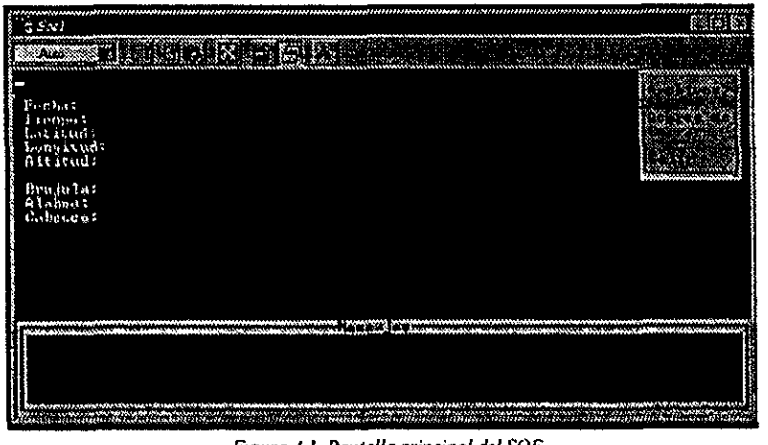


Figura 4.1. Pantalla principal del SOE.

En la sección de "Archivo", solo proporciona información acerca del programa.

Al seleccionar el submenú "SOE", los datos que provienen del sistema del microprocesador, se muestran automáticamente y en este caso son los datos correspondientes a la orientación en los tres ejes: brújula (eje z), alabeo (eje y) y cabeceo (eje x)

En la sección "SOE", se encuentra el submenú:

- FIN

En el Menú GPS, se encuentran los submenús:

- Inicializa
- Mensajes
- Datos
- Reiniciar
- Fin

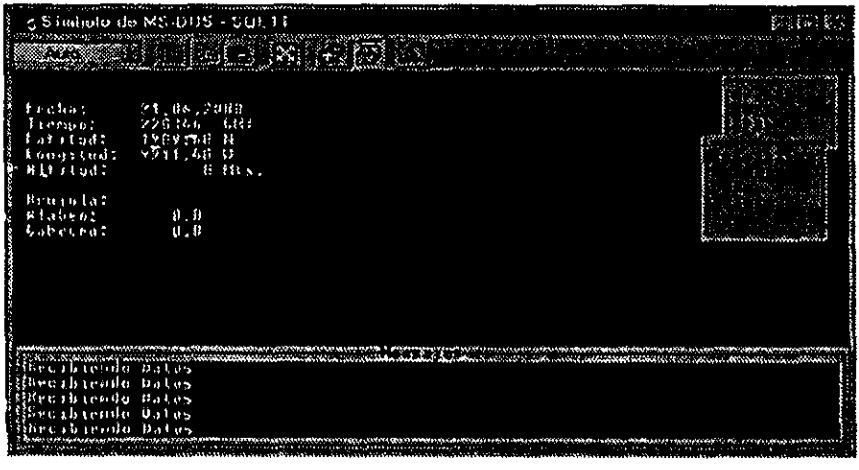


Figura 4.2. Pantalla mostrando submenu GPS.

Para que el GPS inicie es necesario seguir los siguientes pasos:

- Seleccionar Inicializa y oprimir <enter>.
- Seleccionar Mensajes, oprimir <enter> y en la parte inferior en azul aparece "mensaje", se escribe GPGGA.
- Y por ultimo seleccionar el submenú Datos y Oprimir <enter>.

El GPS tarda en ubicarse en el espacio geográfico aproximadamente 20 minutos, dependiendo de interferencias y bloqueo de señales por edificaciones.

Mejoras

Para una mejor resolución y una disminución del error en el GPS, es necesario que la integración de los acelerómetros sea más precisa, esto se puede resolver con una mejora en el programa, específicamente en la rutina de la adquisición de datos del sistema, ya que de acuerdo al procesador que tenga la maquina, la velocidad entre una lectura y otra varia ligeramente, lo que ocasiona que la distancia "estimada", no sea correcta en más de una lectura, así como reducir este tiempo lo más que se puede para tener una aproximación más exacta del movimiento del vehículo, ya que los acelerómetros trabajan en función de la aceleración, y de acuerdo a:

$$v(t) = \int_0^t a dt$$

$$v(t) = at + c$$

donde c es la velocidad inicial v_0.

Y sabiendo:

$$x = \int_0^t v(t) dt$$

o que:

$$x = \int_0^t at + v_0$$

obtenemos,

$$x = x_0 + v_{x0}t + \tfrac{1}{2} a_x t^2$$

donde el desplazamiento x depende directamente del tiempo en que se tomen las muestras de los acelerómetros, es por esto que si las muestras son tomadas en un lapso de tiempo

menor, se tendrá un mejor cálculo de las distancias recorridas en esos intervalos, que si se toman en intervalos de tiempo mayores, como obliga la PC a disposición del proyecto.

Como se menciono al inicio de este trabajo, el sistema forma parte de un proyecto el cual tiene previsto ubicar desde el aire zonas de estudio en el espacio geográfico, para lo cual es recomendable que se agreguen, previo a vuelos de práctica imágenes de la zona de exploración y en estas imágenes se marquen todas las lecturas que proporcione el sistema para un mejor aprovechamiento del mismo; motivo de una tesis paralela y complementaria a este trabajo.

Conclusiones.

El equipo descrito en este trabajo cumple con las funciones para las que fue diseñado, construido y probado, como se muestra en las pruebas realizadas en el capítulo 3 y las imágenes de la pantalla en el capítulo 4. Se construyó un sistema económico, con la adaptación y uso de aparatos con los cuales se contaba en el Laboratorio, como son la brújula, el giróscopo, los acelerómetros y el GPS. Integrándolos de tal forma que en conjunto funcionen como un solo sistema que preste ayuda en la investigación científica de diversas zonas geográficas, por medio de tomas aéreas, con cámaras y otros sensores.

Dado el aprovechamiento de una microcomputadora más reciente (Pentium 400MHz), la velocidad del procesamiento de los datos se incremento considerablemente respecto a la 386, lo cual posibilita que sean tomadas más muestras en un menor tiempo, ayudando así a que el sistema tenga una mayor sensibilidad, por lo que es recomendable que al momento de integrar los dispositivos en una aeronave, se lleve consigo una microcomputadora portátil de una velocidad de 200MHz con procesador Pentium MMX o compatible.

La sencillez del diseño permite su uso aún sin el equipo de computo, lo cual lo vuelve más versátil, y aun más económico, ya que la microcomputadora se integra porque a futuro se desea emplear imágenes, en las cuales se registren las coordenadas y datos proporcionados por el sistema, para una ambientación más realista de los investigadores a bordo de la aeronave.

El aspirante está convencido de que este sistema, al ser económico, haga posible aprovechar más los recursos dedicados a la investigación de los diversos fenómenos del territorio nacional y que con la información registrada en esta tesis sea aplicado y mejorado en pro de la labor científica de la teledetección.

Apéndice A.

Recomendaciones

Las recomendaciones que se presentan para una explotación adecuada del sistema son:

- El sistema del microprocesador debe de estar lo más cerca posible de los sensores y dentro de contenedor metálico, como medio de aislamiento térmico y para evitar interferencias electromagnéticas, provenientes del vehículo; aéreo o terrestre.

- Para la conducción de señales es deseable contar con cables no mayores a 5 m de largo, blindados y aterrizados en uno de sus extremos.

- Es recomendable, con base a las consideraciones de operación, que la ubicación más adecuada para los sensores sea la siguiente:

- La brújula opuesta al motor.

- Los acelerómetros y el giróscopo en el centro de masa de la aeronave, tomando referencia de los tres ejes cartesianos.

Las recomendaciones para mejorar el sistema son:

- Corregir el programa en lenguaje C de tal manera que el tiempo entre la lectura de los acelerómetros sea lo más rápida posible, probablemente en el intervalo de los μ segundos.

- Buscar en el mercado optoacopladores de corriente alterna (ca) para el acoplamiento óptico entre el sistema del microprocesador y el giróscopo, ya que los instalados se encuentran obsoletos y uno de ellos no funciona con suficiente fiabilidad.

- En este caso se empleó el sistema con base en una microcomputadora, ya que el propósito es que a futuro estos datos se registren en una base de datos con imágenes; debido a ello es necesario que los resultados puedan ser observados en una pantalla que permita el agregar imágenes. Pero si sólo se requiere del sistema de orientación, es posible usarlo sin la microcomputadora, en el puerto de comunicación RS232 del micro se puede conectar el GPS y mostrar los datos a través de una pantalla de cristal liquido (exhibidor alfanumérico LCD tipo AND491) que se conecta en el puerto B del microprocesador. Esto abre la posibilidad a más aplicaciones.

Apéndice B

Programas Usados.

A continuación se presentan los listados de los programas usados en el sistema desarrollado. El primer programa está en lenguaje ensamblador y es el programa que se encuentra cargado en la EEPROM del microcontrolador MC68HC11. El segundo programa se encuentra en lenguaje C++ y es el encargado de mostrar en la pantalla de la PC los datos recibidos tanto del sistema como del GPS.

- Programa en Lenguaje ensamblador

```
*****************************************************************
**********************programa de comunicacion*******************
*****************************************************************

porta   equ   $1000
ddra    equ   $1001
portc   equ   $1006
ddrc    equ   $1007
portd   equ   $1008
ddrd    equ   $1009
portf   equ   $1005
baud    equ   $102b
sccr1   equ   $102c
sccr2   equ   $102d
scsr    equ   $102e
scdr    equ   $102f
RDRF    equ   $20
TDRE    equ   $80
OR      equ   $08
FE      equ   $02

        org   $fe00
        lds   #$03ff
*****************************************************************
**********;configuracion de puertos y transmision xerial**********
*****************************************************************
        ldaa  #$00    ;se configura el puerto a como entrada
        staa  ddra
        staa  ddrc    ;se configura el puerto c como entrada
        ldaa  #$02    ;se configura para que bit 2 sea entrada en modo spi
        staa  ddrd
        ldaa  #$30    ;se habilita velocidad de 9600 baudios
        staa  baud
        ldaa  #$00
        staa  sccr1   ;se configura para tranamision y recepcion de 8 bits
```

```
    ldaa   #$2c   ;habilitar int. por rx e int  generales, se encienden rx y tx
    staa   scr2



*****************************************************************************
**********************;programa principal*********************************
*****************************************************************************
********************** salida de datos a la pc*****************************
*****************************************************************************
inicio ldab   #$01   ;se habilita la pc
     ldaa   scsr   ;leemos bandera de int. y se limpia
     bpl    inicio ; espero
     stab   scdr   ;mandamos dato del acumulador b
     jsr    retar
dato   ldaa   scsr   ;leemos bandera de int. y se limpia
     anda   #RDRF
     beq    dato
     ldab   scdr   ;leemos el dato que envia la pc
     cmpb   #$10   ;compara con 10
     beq    bruju1 ;brinca a bruju1
     cmpb   #$11   ;comparamos con 11
     beq    bruju2 ;brinca a bruju2
     cmpb   #$12   ;compara con 12
     beq    gala1  ;brinca a gala1
     cmpb   #$13   ;comparamos con 13
     boq    gala2  ;brinca a gala2
     cmpb   #$14   ;compara con 14
     beq    gcab1  ;brinca a gcab1
     cmpb   #$15   ;comparamos con 15
     beq    gcab2  ;brinca a gcab2
     cmpb   #$16   ;compara con 16
     beq    acelx1  ;brinca a acelx1
     cmpb   #$17   ;comparamos con 17
     beq    acelx2 ;brinca a acelx2
     cmpb   #$1a   ;compara con 1a
     beq    acely1 ;brinca a acely11
     cmpb   #$1b   ;comparamos con 1b
     beq    acely2 ;brinca a acely2
     cmpb   #$18   ;compara con 18
     beq    acelz1 ;brinca a acelz1
     cmpb   #$19   ;comparamos con 19
     beq    acelz2 ;brinca a acelz2
     jmp    dato
 bruju1 ldab   #$07   ;habilito lectura de la brujula
     stab   portf
     jsr    lect1  ;leemos parte alta
     jmp    dato
 bruju2 ldab   #$07   ;habilito lectura de la brujula
     stab   portf
     jsr    lect2  ;leemos parte baja
     jmp    dato
 gala1  ldab   #$06   ;habilito lectura del alabeo del giroscopo
     stab   portf
     jsr    lect1  ;leemos parte alta
     jmp    dato
 gala2  ldab   #$06   ;habilito lectura del alabeo
     stab   portf
     jsr    lect2  ;leemos parte baja
     jmp    dato
 gcab1  ldab   #$05   ;habilito lectura del cabeceo del giroscopo
     stab   portf
     jsr    lect1  ;leemos parte alta
     jmp    dato
 gcab2  ldab   #$05   ;habilito lectura del cabeceo
     stab   portf
     jsr    lect2  ;leemos parte baja
     jmp    dato
 acelx1  ldab   #$02   ;habilito lectura del acelerometro x
```

```
      stab   portf
      jsr    lect1  ;leemos parte alta
      jmp    dato
acelx2 ldab  #$02   ;habilito lectura del eje x
      stab   portf
      jsr    lect2  ;leemos parte baja
      jmp    dato
acely1 ldab  #$00   ;habilito lectura del acelerometro eje y
      stab   portf
      jsr    lect1  ;leemos parte alta
      jmp    dato
acely2 ldab  #$00   ;habilito lectura del eje y
      stab   portf
      jsr    lect2  ;leemos parte baja
      jmp    dato
acelz1 ldab  #$04   ;habilito lectura del acelerometro eje z
      stab   portf
      jsr    lect1  ;leemos parte alta
      jmp    dato
acelz2 ldab  #$04   ;habilito lectura del eje z
      stab   portf
      jsr    lect2  ;leemos parte baja
      jmp    dato

**********Cargamos los datos de los puertos***************
lect1  ldaa   scsr   ;leemos bandera de int. y se limpia
      bpl    lect1  ; espero
      ldaa   portc  ; cargo un dato "x"
      staa   scdr   ; mando parte alta del contador por el serial
      jsr    retar
      rts
lect2  ldaa   scsr   ;leemos bandera de int. y se limpia
      bpl    lect2  ; espero
      ldab   porta
      stab   scdr   ;mando parte baja
      jsr    retar
      rts
retar  ldx    #$0100
retard dex
      bne    retard
      rts
*************************************************************************
**********tabla de vectores de interrupcion (modo simple)*********
*************************************************************************
      org    $ffd6  ; direccion del vector de interrupcion de Rx
      fcb    $fc    ; direccion de la rutina para recibir el dato(DAT)
      fcb    $2f
      org    $fffe  ; vector de reset en modo simple
      fcb    $fc    ; inicio de eeprom (programa principal)
      fcb    $00
      end
```

- Programa en Lenguaje C.

```
/* Programa de Control del GPS */
#include <dos.h>
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <bios.h>
#include <string.h>
#include <io.h>
#include "xcom.h"
#define GPGGA          0x01        /* Mensajes del GPS */
#define PMGLB          0x02
#define GPZDA          0x03
#define PMGLG          0x04
#define PMGLF          0x05
```

```
#define VidNormal                   0x71        /* Colores de caracteres en pantalla */
#define VidInverso                  0x72
#define VidNormal1                  0x07
#define VidNormal2                  0xa1
#define VidNormal3                  0x17
#define ColorMensaje                0x71
#define DirMemVid (char far *)      0xB8000000;         /* Dir memoria de video */
#define ESC                         27          /* Tecla Escape */
#define Max_Frame                   5                   /* Maximo nivel de submenus */
#define Modo_Off                    0
#define Modo_Soe                    1
#define Modo_Gps                    2
#define BYTE

/* Modulo de inicialización y Reestructuración del ambiente de la PC" */
void ambiente(), restaura();

/* Modulo de información */
void ayuda(), activamenu(), dibujaborde(), despliegamenu(), submenu(), cierramenu();
void respuesta(), salir(), vtr(), ejecuta(), pulldown(), salvavideo(), restauravideo();
int make_menu(), esta_en(), validacion();

/* Modulo de presentación de información en pantalla */
void datosdelgps(), pantalla(), write_string(), write_char(), cls(), borracurser();
void clsmensajes(), bordemensajes(), datosapantalla(), mensajes();

/* Modulo de comunicación con el GPS */
void inicializacom1(), com1reset(), gps();
void revisamensajegps(), gpggs(), pmglb(), gazda(), pmglg(), pmglf();
int checagps(), tipomensajegps();
void generachecksum(), tx_gps(), al_gps(), leecomandogps(), inicializagps();
void pidedatosgps(), borragps(), generachecksumbx();

/* Modulo de comunicacion con el SOE */
void  soe(), inicializacom2(), datosbrujula(), datosgiroscopo(), acel();
void borde(int inix, int iniy, int finx, int finy, int atributo);

/* Modulo de maneja de archivos */
void almacena(), cerrar();

/* Variables Generales */
int dir_puerto, bparametro;
int num_menu, opcion, nueva_funcion, mensajesx, mensajesy;          /* De los menus */
float brujula, alabeo, cabeceo;
struct registro {                                                   /* Formato de datos */
                    float brujula, latitud, longitud, alabeo, cabeceo;
                    long altitud, tiempo,
                    int error;
                    } datos;
int modo_operacion;
unsigned cual_gps, dato_gps, datos_gps;                 /* Comunicación con el GPS */
char cad_gps1[100], cad_gps2[100], mensaje_gps[100];
char caracter_gps[1];
int posicioncaracter=0;
struct menu_frame {
        int startx, endx, starty, endy;
        unsigned char *p;
        char **menu;
        char *llaves;
        int borde, contador;
        int activo;
} frame[Max_Frame],i;
char *Principal[] = {                                               /* Menú Principal */
                                "Archivos>", "SOE>    ", "GPS>    "};
char *Archivos[] = {
                                "Acerca de", "Cerrar  ", "      ","Salir   "};
char *Soe[] = {
                            "Fin    "};
char *Gps[] = {
                        "Inicializa", "Mensajes ", "Datos    ", "Reiniciar ", "Fin      "};
```

```
/* Programa principal */

main()
{
        ambiente();                                          /* Establece las condiciones iniciales */
        for(;;){                                             /* Ciclo de espera para salida */
                while(!kbhit());                             /* Espera a que presione una tecla */
                activamenu();                                /* Activa el menu */
                switch(modo_operacion){
                        case Modo_Off: if(nueva_funcion) ejecuta();      /* Ejecuta funcion */
                                break;
                        case Modo_Soe: soe();
                                break;
                        case Modo_Gps: gps();                /* Modo GPS */
                                break;
                        default:;
                }
}                                                            /* fin de main() */


/* Condiciones  iniciales */
void ambiente()
{
        outportb(0x0021,0x88);                               /* Inicializa controlador de
interrupciones */
        pantalla();                                  /* Despliega pantalla inicial */
        bordemensajes();                             /* Dibuja el borde de la ventana de mensajes */
        make_menu(0, Principal, "fvg",3,0,68,1);             /* Crea los marcos de menu */
        make_menu(1, Archivos,"acvs", 4,2,68,1);
        make_menu(3, Soe,"f",1,3,63,1);
        make_menu(4, Gps,"lmdrf",5,4,66,1);
        /* inicializa variables globales */
        modo_operacion = Modo_Off;
        nueva_funcion = 0;                           /* Si elige una nueva función */
        num_menu = 0;                                /* Nº de menu desplegado */
        opcion = 0;                                  /* Opcion del menu desplegado */
        mensajesx = 19;                              /* Posicion x,y de la ventana de mensajes */
        mensajesy = 1;
        cual_gps = 0x01;                             /* Cual cadena se esta utilizando RX de GPS */
        dato_gps = 0x00;                             /* Si existe un nuevo dato GPS */
        datos_gps = 0x00;                            /* Si se recibio el bloque completo de datos */
        posicioncaracter = 0;                                /* Apunta cadena que utiliza para RX
GPS */
        cad_gps1[0] = 0x00;                          /* Mensajes recibidos de GPS */
        cad_gps2[0] = 0x00;
        mensaje_gps[0] = 0x00;
        caracter_gps[1] = 0x00;
        /* Despliega el menu principal */
        salvavideo(num_menu);                                /* Salva la porción de pantalla a
utilizar */
        frame[num_menu].activo = 1,                          /* Activa menu 1 */
        dibujaborde(num_menu);
        despliegamenu(num_menu);
        return;                                      /* Fin de ambiente() */
}
/* Restaura el estado que existia antes del programa */
void restaura()
{
        outportb(0x0021,0xB8);                       /* Restaura controlador de interrupciones */
        return;
}

/* Despliega pantalla principal */
void pantalla()
{
        char *pantalla[]= {                          /* Mensajes a pantalla principal */
                "Fecha:","Tiempo:", "Latitud:", "Longitud:","Altitud:",
                        "Brujula:","Alabeo:", "Cabeceo:");
```

```
        cls();                                          /* Borra pantalla */
        write_string(2,1,pantalla[0],VidNormal1);                       /* Mensajes a pantalla */
        write_string(3,1,pantalla[1],VidNormal1);
        write_string(4,1,pantalla[2],VidNormal1);
        write_string(5,1,pantalla[3],VidNormal1);
        write_string(6,1,pantalla[4],VidNormal1);
        write_string(8,1,pantalla[5],VidNormal1);
        write_string(9,1,pantalla[6],VidNormal1);
        write_string(10,1,pantalla[7],VidNormal1);
        return;
}

/* Despliega una cadena con acceso directo a memoria de video */
void write_string(ren, col, p, atributo)
int ren, col;                                           /* Renglon y columna de inicio */
char *p;                                                /* Apunta al inicio de una cadena */
int atributo;                                   /* Color del caracter a desplegar */
{
        register int i;
        char far *v;

        v = DirMemVid;                                  /* Dir memoria de video */
        v += (ren*160) + col*2;                         /* Calcula dirección */
        for(i=col;*p; i++){
                *v++ = *p++;                            /* Escribe el caracter */
                *v++ = atributo;                        /* Especifica atributo */
        }
        return;
}

/* Despliega un acaracter con acceso directo a memoria  de video */
void write_char(x,y,ch,atributo)
int x, y;                                       /* Posicion en pantalla */
char ch;                                        /* Caracter a mostrar en pantalla */
int atributo;                                   /* Color del caracter */
{
        register int i;
        char far *v;
        v = DirMemVid;                                  /* Dir memoria de video */
        v += (x*160) + y*2;                             /* Calcula dirección */
        *v++ = ch;                                      /* Escribe el caracter */
        *v = atributo;                                  /* Especifica atributo */
        return;
}

/* Borra la pantalla */
void cls()
{
        union REGS r;
        r.h.ah=6;
        r.h.al=0;
        r.h.ch=0;
        r.h.cl=0;
        r.h.dh=24;
        r.h.dl=79;
        r.h.bh=7;
        int86(0x10, &r, &r);
        return;
}

/* Borra la ventana de mensajes */
void clsmensajes()
{
        char far *v;
        int i, j;
        v = DirMemVid;                                  /* Dir de memoria de video */
        v += 19*160;                                    /* Calcula desplazamiento */
        for(i=19; i<=34; i++)
                for(j=0; j<=89; j++){
                *v++ = ' ';                             /* Borra caracter */
```

```
                *v++ = ColorMensaje;                    /* Especifica atributo */
        }
        return;
}

/* Borra el cursor */
void borracursor()
{
        union REGS r;
        r.x.cx=0x2000;
        r.h.ah=0x01;
        int86(0x10, &r, &r);
        return;
}

/* Crea menu y asingna memoria para soportarlo */
int make_menu(num,menu,llaves,contador,x,y,borde)
int num;
char *menu[], *llaves;
int contador, x, y, borde;
{
        register int i, len;
        int endx, endy, eleccion;
        unsigned char *p;

                                                /* Evita errores */
        if(num>Max_Frame) {                     /* Demasiados menus */
                printf("\Demasiados Menus\n");
                exit(0);                        /* Suspende la ejecición del programa */
        }
        if(x>24||x<0||y>79||y<0){               /* Error en las coordenadas */
                printf("\nError en las coordenadas del menu");
                exit(0);
        }
        len = 0;                                /* Error en el tamaño del menú */
        for(i=0;i<contador;i++)
                if(strlen(menu[i])>len) len = strlen(menu[i]);
                endy=len+1+y;
                endx=contador+1+x;
                if((endx+1 > 34)||(endy+1 > 80)){
                        printf("\n El menu no cabe\n");
                        clrscr();
                        exit(0);
                }

                                                /* Asigna suficiente memoria para soportarlo */
        p = (unsigned char *)malloc(2*(endx-x+1)*(endy-y+1)+10);
        if(!p) exit(1);                         /* Aborta en caso de error */
        frame[num].startx=x,frame[num].endx=endx;       /* Construye el marco del menu */
        frame[num].starty=y;frame[num].endy=endy;
        frame[num].p=p;
        frame[num].menu=(char **)menu;
        frame[num].borde=borde;
        frame[num].llaves=llaves;
        frame[num].contador=contador;
        frame[num].activo=0;
        return(1);
}

/* Activa menu o responde a la tecla presionada */
void activamenu()
{
        switch(frame[0].activo){
                case 1: respuesta();                    /* Responde a la tecla presionada */
                        break;
                case 0: num_menu = 0;                   /* Activa el menu */
                        opcion = 0;
                        pulldown(0);
                        bioskey(0);                     /* Elimina la tecla presionada */
        }
```

```
        return;
}

/* Control de menus, respuesta a la tecla presionada */
void respuesta()
{
        union llave {                                   /* Tecla presionada */
                char ch[2];
                int i;
        } c;
        int llave_elegida, temporal;                    /* Si es una tecla especial */

        c.i = bioskey(0);                               /* Lee la tecla */
        write_string(frame[num_menu].startx+1+opcion,           /* Video normal */
                frame[num_menu].starty+1,frame[num_menu].menu[opcion],VidInverso);
        if(c.ch[0]){                                    /* Si es tecla normal */
                llave_elegida = esta_en(frame[num_menu].llaves,
                        tolower(c.ch[0]));                      /* Checa si es letra especial */
                        temporal = opcion;
                        opcion = llave_elegida-1;
                        if (llave_elegida && validacion()){             /* Si es valida la nueva opción */
                        write_string(frame[num_menu].startx+1+opcion,           /* Video inverso la nueva opcion */
                frame[num_menu].starty+1,frame[num_menu] menu[opcion],VidNormal);
                        submenu();)                             /* Checa si existe submenu */
                else opcion = temporal;                         /* Restaura opcion anterior */

                switch(c.ch[0]){                                /* Checa si es enter o barra espaciadora */
                        case 'r':
                                write_string(frame[num_menu].startx+1+opcion,       /* Video inverso la nueva
opcion */
                                frame[num_menu].starty+1,frame[num_menu].menu[opcion],VidNormal);
                                submenu();                              /* Checa si existe submenu */
                                break;
                        case ' '                                /* Checa validacion de la opción */
                                do ++opcion; while(!validacion());
                                break;
                        case ESC:                               /* Cierra el menu abierto */
                                cierramenu();
                                if(!frame[num_menu].activo) return;
                                break;)
        }
        else {
                 switch(c.ch[1]){                               /* Checa si se presiono una flecha */
                        case 72:
                                do --opcion; while(!validacion());                      /* Validacion de opcion */
                                break,
                        case 80:
                                do ++opcion; while(!validacion());                      /* Validacion de opcion */
                                break;)
        }
        write_string(frame[num_menu].startx+1+opcion,           /* Video inverso la nueva opcion */
        frame[num_menu] starty+1,frame[num_menu].menu[opcion],VidNormal);
        return;
}

/* Activa el menu correspondiente */
void pulldown(num)
int num;                                        /* Numero de menu a activar */
{
        if(!frame[num].activo){                         /* Activa el menu */
                salvavideo(num);                        /* Salva la parte de video */
                frame[num].activo = 1;
                dibujaborde(num),                       /* Dibuja el borde del menu */
                despliegamenu(num);                     /* Despliega el menu */
        }
        return,
}

/* Dibuja el borde del menu */
void dibujaborde(num)
```

```
int num;                                        /* Número de menú */
{
        register int i;
        char far *v, far *t;
        v = DirMemVid;                                  /* Dir memoria video */
        t = v;
        for(i=frame[num].startx+1;i<frame[num].endx;i++){
                v +=((i*160) + frame[num].starty*2;             /* calcula la dirección */
                *v++ = 179;                                     /* Escribe caracter */
                *v = VidInverso;                                /* Especifica atributo */
                v = t;
                v += (i*160) + frame[num].endy*2;               /* calcula la dirección */
                *v++ = 179;                                     /* Escribe caracter */
                *v = VidInverso;                                /* Especifica atributo */
                v = t;
        }

        for(i=frame[num].starty+1;i<frame[num].endy,i++){
                v +=(frame[num].startx*160) + i*2;              /* calcula la dirección */
                *v++ = 196;                                     /* Escribe caracter */
                *v = VidInverso;                                /* Especifica atributo */
                v = t;
                v += (frame[num].endx*160) + i*2;               /* calcula la dirección */
                *v++ = 196;                                     /* Escribe caracter */
                *v = VidInverso;                                /* Especifica atributo */
                v = t;
        }                                       /* Caracteres 191,192,217,218 */
        write_char(frame[num].startx,frame[num].starty,215,VidInverso);
        write_char(frame[num].startx,frame[num].endy,191,VidInverso);
        write_char(frame[num].endx,frame[num].starty,192,VidInverso);
        write_char(frame[num].endx,frame[num].endy,214,VidInverso);
        return;
}

/* Dibuja el borde de la ventana de mensajes */
void bordemensajes()
{
        register int i;
        char far *v, far *t;
        v = DirMemVid;
        t = v;
        for(i=18;i<=24;i++){
                v += (i*160);                                   /* Calcula la dirección */
                *v++ = 179;                                     /* Escibe caracter */
                *v = ColorMensaje;                              /* Especifica atributo */
                v = t;
                v += (i*160)+158;                               /* Calcula la dirección */
                *v++ = 179;                                     /* Escibe caracter */
                *v = ColorMensaje;                              /* Especifica atributo */
                v = t;
        }

        for(i=0;i<=79;i++){
                v += (2880) + i*2;                              /* Calcula la dirección */
                *v++ = 196;                                     /* Escibe caracter */
                *v = ColorMensaje;                              /* Especifica atributo */
                v = t;
                v += (3840) + i*2;                              /* Calcula la dirección */
                *v++ = 196;                                     /* Escibe caracter */
                *v = ColorMensaje;                              /* Especifica atributo */
                v = t;
        }

                                                /* Caracteres 191,192,217,218 */
        write_char(18,0,218,ColorMensaje);
        write_char(18,79,191,ColorMensaje);
        write_char(24,0,192,ColorMensaje);
        write_char(24,79,217,ColorMensaje);
        write_string(18,35,"Mensajes",ColorMensaje);
        return;
```

```
}

/* Despliega el menu */
void despliegamenu(num)
int num;                                    /* Numero de menu */
{
        register int i, x;
        char **m;

        x = frame[num].startx+1;                    /* Rengion en pantalla */
        m = frame[num].menu;                        /* Opciones del menú */
        for(i=0;i<frame[num].contador;i++,x++)              /* Video normal las opciones */
                write_string(x,frame[num].starty+1,m[i],VidInverso);

        write_string(frame[num].startx+1+opcion,frame[num].starty+1,
                m[opcion],VidNormal);                       /* Video inverso la opcion */
        return;
}

/* Checa si es una letra especial del menu */
int este_en(s,c)
char *s, c;                                 /* s->letras especiales, c->letra presionada */
{
        register int i;
        for(i=0;*s;i++) if(*s++ == c) return(i+1);
        return(0);
}

/* Salva preción de video utilizado por menu */
void salvavideo(num)
int num;                                    /* Numero de menu */
{
        register int i,j;
        char *buf_ptr;
        char far *v, far *t;

        buf_ptr = frame[num].p;                     /* Dir donde se almacenará la porcion de video */
        v = DirMemVid;                              /* Dir memoria de video */
        for(i=frame[num].starty;i<frame[num].endy+1;i++)
                for(j=frame[num].startx;j<frame[num].endx+1;j++){
                        t = (v+(i*160) + j*2);              /* Dir de la porcion de video */
                        *buf_ptr++ = *t++;                  /* Salvar el caracter de la porcion de video */
                        *buf_ptr++ = *t;                    /* Salva su atributo */
                        *(t-1)=' ';                         /* Limpia la porcion de video */
                        *t = VidNormal;
                }
                return;
}

/* Restaura la porcion de video utilizado por el menu */
void restauravideo(num)
int num;                                    /* Numero de menú */
{
        register int i,j;
        char *buf_ptr;
        char far *v, far *t;

        buf_ptr = frame[num].p;                     /* Dir de respaldo de la porcion de video */
        v = DirMemVid;                              /* Dir memoria de video */
        t = v;
        for(i=frame[num].starty;i<frame[num].endy+1,i++)
                for(j=frame[num].startx;j<frame[num].endx+1;j++){
                        v = t;
                        v += (i*160) + j*2;                 /* Calcula la dirección del renglon y columna */
                        *v++ = *buf_ptr++;                  /* Restaura caracter de la porcion de video */
                        *v = *buf_ptr++;                    /* Restaura atributo */
                }
                frame[num].activo=0;                    /* Desactiva menú */
                return;
}
```

```
/* Cierra el menu correspondiente */
void cierramenu()
{
        if(frame[num_menu].activo){
                restauravideo(num_menu);                    /* Restaura la porcion de video utilizada */
                switch(num_menu){
                        case 0: break;
                        case 1: num_menu = 0;                   /* Menú activo principal */
                                opcion = 0;
                                break;
                        case 3: num_menu = 0;                   /* Menú activo GPS */
                                opcion = 1;
                                break;
                        case 4: num_menu = 0;
                                opcion = 2;
                                break;
                }
        }
        return;
}

/* Llamada al submenú */
void submenu()
{
        switch(num_menu){
                case 0: switch(opcion){
                        case 0: num_menu = opcion+1;
                                if(modo_operacion != Modo_Off) opcion = 0;
                                pulldown(num_menu);                 /* Activa submenu Archivos */
                                break;
                        case 1: modo_operacion = Modo_Soe;
                                num_menu = 3;
                                opcion = 0;
                                pulldown(num_menu);
                                break;

                        case 2: modo_operacion = Modo_Gps;
                                num_menu = 4;
                                opcion = 0;
                                pulldown(num_menu),                 /* Activa submenu GPS */
                                break;}
                        break;

                default: nueva_funcion = 1;}
        return;
}

/* Validacion de la nueva opción del menu activo*/
int validacion()
{
        if(opcion==frame[num_menu].contador) opcion=0;              /* Filtro de tope de la opcion */
        if(opcion<0) opcion = frame[num_menu].contador-1;

        switch(num_menu){
                case 0: switch(modo_operacion){
                                case Modo_Off: break;               /* Todas validas */
                                case Modo_Soe: if(opcion == 2) return(0);
                                        break;
                                case Modo_Gps: if(opcion==1) return(0);     /* Opcion no valida */
                                        break;
                        }
                        break;
                case 1: switch(modo_operacion){
                                case Modo_Off: if(opcion<3) return(0);      /* Opciones no validas */
                                        break;
                                case Modo_Gps: if(opcion==2) return(0),     /*Opcion no valida */
                                        break;
                                default:;}
                default: return(1),
```

```
        }
        return(1);
}

/* Ejecuta la funcion elegida */
void ejecuta()
{

        switch(num_menu){
                case 0: break;                          /* No realiza tareas */
                case 1: switch(opcion){
                                                case 0: ayuda();                        /* Salir del programa */
                                                        break;
                                                case 1: cerrar();                       /* Salir del programa */
                                                        break;
                                                case 2: ayuda();
                                                        break;
                                                case 3: salir();                                        /* Salir del
programa */
                                                        break;
                                                default:;}
                                break;
                case 3: switch(opcion){
                                                case 0: modo_operacion = Modo_Off;
                                                        cierramenu(3);
                                                        break;
                                                default:; }
                                break;
                case 4: switch(opcion){
                                                case 0: inicializagps();                /* Comunicacion con el GPS */
                                                        break;
                                                case 1: al_gps();                       /* TX mensajes al GPS */
                                                        break;
                                                case 2: pidedatosgps();                 /* Datos utilizados */
                                                        break;
                                                case 3: borragps();                     /* Borra la memoria del GPS */
                                                        break;
                                                case 4: modo_operacion = Modo_Off;
                                                        cierramenu(4);
                                                        break;
                                                default : break;}
                                                        break;
                        default:;}
        nueva_funcion = 0;
        return;
}

/* Salida del programa */
void salir()
{
        cerrar();                               /* Cierra el archivo */
        restaura();                             /* Restaura estado de PC */
        cls();                                  /* Borra la pantalla */
        clrscr();
        exit(0);                                                /* Aborta el programa */
}

/* Maneja la ventana de mensajes */
void mensajes(mensaje,atributo)
char *mensaje;                                                  /* mensaje a la ventana */
int atributo;                                   /* Color */
{
        char far *v, far *t, far *z;
        int i, j;

        if(mensajesx == 24){                            /* Recorre los renglones de la ventana */
                z = DirMemVid;                                  /* Direccion memoria de video */
                for(i=20;i<=23;i++){
                        t = z +((i-1)*160)+2;                   /* Dir renglon anterior */
                        v = z + (i*160) +2;                     /* Dir renglon */
```

```
                    for(j=1;j<=78;j++){
                        *t++ = *v;                          /* Renglon anterior = renglon */
                        *t++ = *(v+1);                      /* Atributo */
                        *v++ = ' ';                         /* Borra renglon */
                        *v++ = VidNormal3;                  /* Atributo */
                    }
                }
                mensajesx=23;                               /* Ultimo renglon */
        }
        write_string(mensajesx++,mensajesy,mensaje,atributo);   /* Mensaje */
        return;
}

/* Opcion de cerrar archivos */
void cerrar()
{
                mensajes("Ningun menu seleccionado                              ", VidNormal3);
                return;
}

/* Crea una ventana de mensaje */
void ayuda()
{

        mensajes("SOE version 1.1                                       ", VidNormal2);
        mensajes("Programa que muestra la orientacion de una aeronave                   ", VidNormal2);
        mensajes("Realizado por:                                        ", VidNormal2);
        mensajes("          Marcelo Bastida Tapia                       ", VidNormal2);
        mensajes("Julio de 1999                                         ", VidNormal2);
        delay(1000);
        return;

}

/* Fijar los parametros de comunicaciones */
void parcom(long velocidad,int paridad,int bitdatos,int parada)
{
unsigned int divisor;
unsigned char lab,mab;
BYTE bparametro;
divisor=115200L/velocidad;
mab=divisor>>8;
lab=(divisor<<8)>>8;

outportb(REG_FDATOS,FDATOS_DLAB);                       /* Habilita acceso al registro */
outportb(dir_puerto,lab);                               /* lab del divisor */
outportb(dir_puerto+1,mab);                             /* mab del divisor */

bparametro=bitdatos-5;

 if (parada==2)
        bparametro|=FDATOS_PARADA;

 if (paridad!=PARIDAD_NO)
        bparametro|=FDATOS_PARIDAD_SN;

 if (paridad==PARIDAD_PAR)
        bparametro|=FDATOS_PARIDAD_TIPO;

 outportb(REG_FDATOS,bparametro);
}

/* Inicializa el puerto para el SOE */
void inicializacom2()
{
        delay(1);
        dir_puerto= 0x2f8;                      /* Comm2 */
        parcom(9600,PARIDAD_NO,8,1);            /* Velocidad de trans  9600, sin paridad, 8 bits de datos, 1 bit
de parada */
        delay(1);
```

```
        outportb(0x2f9, 0x01);                          /* Interrupción por RX habilitada */
        mensajes("Comunicacion por Comm2 a 9600 baus                    ", VidNormal3);
        return;
}

/* inicializa el puerto para el GPS */
void inicializacom1()
{
        delay(1);
        dir_puerto= 0x3f8;                      /* Comm1 */
        parcom(9600,PARIDAD_NO,8,1);                    /* Velocidad de trans  9600, sin paridad, 8 bits de datos, 1 bit
de parada */
        delay(1);
        outportb(0x3f9, 0x01);                  /* Interrupción por RX habilitada */
        mensajes("Comunicacion por Comm1 a 9600 baus                    ", VidNormal3);
        return;
}

/* Reestablece Comm1 */
void com1reset()
{
        delay(1);
        dir_puerto= 0x3f8;                      /* Comm1 */
        parcom(9600,PARIDAD_NO,8,1);
        delay(1);
        outportb(0x3f9, 0x00);                  /* Interrupción por RX deshabilitada */
        mensajes("Comunicacion por Comm1 a 9600 baus                    ", VidNormal3);
        return;
}

/* Reestablece Comm2 */
void com2reset()
{
        delay(1);
        dir_puerto= 0x2f8;                      /* Comm2 */
        parcom(9600,PARIDAD_NO,8,1);
        delay(1);
        outportb(0x2f9, 0x00);                  /* Interrupción por RX deshabilitada */
        mensajes("Comunicacion por Comm2 a 9600 baus                    ", VidNormal3);
        return;
}

/* Muestra los datos en pantalla */
void datosapantalla()
{
        int i;
        gotoxy(12,5);
        printf("%8.2f",datos.latitud);
        gotoxy(11,6);
        printf("%9.2f",datos.longitud);
        gotoxy(13,7);
        printf(" %5d Mts.", datos.altitud);
        gotoxy(16,9);
        printf("%3.1f",brujula);
        gotoxy(16,10);
        printf("%3.1f",alabeo);
        gotoxy(16,11);
        printf("%3.1f",cabeceo);
return;
}

/* Modo SOE condiciones iniciales */
void soe()
{
        inicializacom2();                                               /* Inicializa Com1 comunicacion SOE
*/
        while (modo_operacion == Modo_Soe){             /* Ciclo de espera para salida */
                        datosbrujula();
                        datosgiroscopo();
                if(kbhit()){  activamenu();             /* Atiende al usuario */
```

```
                if(nueva_funcion) ejecuta();}
        }
        com2reset();
        cerrar();
        return;
}

/* Recibiendo los datos de la brújula */
void datosbrujula()
{
        int dato, dato1;
        float data,

        outportb(0x2f8,0x10);                                   /* Solicito parte alta de la brújula */
  delay(10);
  dato=inportb(0x2f8);            /* Lee lo que hay en el puerto COM2 */
  outportb(0x2f8,0x11);                                         /* Solicito parte baja de la brújula */
  delay(10);
  dato1=inportb(0x2f8);           /* Lee lo que hay en el puerto COM2 */
  data=(dato*256)+dato1;
  brujula=data/10;
        if ( brujula==0 )                                       /* Mando el dato a pantalla */
        {
                gotoxy(16,9);
                printf("0.0 N",brujula);                        /* se pone en pantalla */
        }
  if ( brujula> 0 && brujula < 90 )
        {
                gotoxy(16,9);
                printf("%3.1f NW",brujula);                     /* se pone en pantalla */
        }
  if ( brujula==90 )
        {
                gotoxy(16,9);
                printf("0.0 W",brujula);                        /* se pone en pantalla */
        }
  if ( brujula > 90 && brujula < 180 )
        {
                brujula=180-brujula;
                gotoxy(16,9);
                printf("%3.1f SW",brujula);                     /* se pone en pantalla */
        }
  if ( brujula==180 )
        {
                gotoxy(16,9);
                printf("0.0 S",brujula);                        /* se pone en pantalla */
        }
  if ( brujula > 180 && brujula < 270 )
        {
                brujula=brujula-180;
                gotoxy(16,9);
                printf("%3.1f SE",brujula);                     /* se pone en pantalla */
        }
  if ( brujula==270 )
        {
                gotoxy(16,9);
                printf("0.0 E",brujula);                        /* se pone en pantalla */
        }
  if ( brujula > 270 && brujula < 360 )
        {
                brujula=360-brujula;
                gotoxy(16,9);
                printf("%3.1f NE",brujula);                     /* se pone en pantalla */
        }
        return;
}

/* Datos del SOE, parte giroscopo */
        void datosgiroscopo()
{
```

```
int dato2, dato3, dato4, dato5;

        outportb(0x2f8,0x12);                   /* Pide parte alta del alabeo */
        delay(10);
        dato2=inportb(0x2f8);                   /* Lee lo que hay en el puerto COM2 */
        outportb(0x2f8,0x13);                   /* Pide parte baja del alabeo */
        delay(10);
        dato3=inportb(0x2f8),                   /* Lee lo que hay en el puerto COM2 */
        outportb(0x2f8,0x14);                   /* Pide parte alta del cabeceo */
        delay(10);
        dato4=inportb(0x2f8);                   /* Lee lo que hay en el puerto COM2 */
        outportb(0x2f8,0x15);                   /* Pide parte baja del cabeceo */
        delay(10);
        dato5=inportb(0x2f8);                   /* Lee lo que hay en el puerto COM2 */
        alabeo=(dato2*256)+dato3;
        cabeceo=(dato4*256)+dato5;
        if ( alabeo>=2550 )
        {
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf("Inclinación mayor a -50 Grados");;                      /* se pone en
pantalla */
        }
    if ( alabeo > 2465 && alabeo < 2550 )
        {
                alabeo=-(((alabeo-2465)/85)+40);
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf("%3.1f Grados", alabeo);                 /* se pone en pantalla */
        }
        if ( alabeo==2465 )
        {
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf(" -40 Grados");;                         /* se pone en pantalla */
        }
     if ( alabeo > 2388 && alabeo < 2465 )
        {
                alabeo=-(((alabeo-2388)/77)+30);
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf("%3.1f Grados", alabeo);                 /* se pone en pantalla */
        }
        if ( alabeo==2388 )
        {
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf(" -30 Grados");.                         /* se pone en pantalla */
        }
     if ( alabeo > 2295 && alabeo < 2388 )
        {
                alabeo=-(((alabeo-2295)/93)+20);
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf("%3 1f Grados", alabeo);                 /* se pone en pantalla */
        }
        if ( alabeo==2295 )
        {
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf("-20 Grados");;                          /* se pone en pantalla */
        }
     if ( alabeo > 2184 && alabeo < 2295 )
        {
                alabeo=-(((alabeo-2184)/11)+10);
                gotoxy(16,10);                                                  /*
Datos a la pantalla */
                printf("%3.1f Grados", alabeo),                 /* se pone en pantalla */
        }
        if ( alabeo==2184 )
```

```
                {
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("-10 Grados");;                          /* se pone en pantalla */
                }
        if ( alabeo > 2047 && alabeo < 2184 )
                {
                        alabeo=-(((alabeo-2047)/137));
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("%3.1f Grados", alabeo),                 /* se pone en pantalla */
                }
                if ( alabeo==2047 )
                {
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("0 Grados");;                            /* se pone en pantalla */
                }
        if ( alabeo > 1885 && alabeo < 2047 )
                {
                        alabeo=((alabeo-1885)/162);
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", alabeo);                /* se pone en pantalla
*/
                }
                if ( alabeo==1885 )
                {
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("10 Grados");;                           /* se pone en pantalla */
                }
        if ( alabeo > 1740 && alabeo < 1885 )
                {
                        alabeo=((alabeo-1740)/145)+10;
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", alabeo);                /* se pone en pantalla
*/
                }
                if ( alabeo==1740 )
                {
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("20 Grados");;                           /* se pone en pantalla */
                }
        if ( alabeo > 1561 && alabeo < 1740 )
                {
                        alabeo=((alabeo-1561)/179)+20,
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", alabeo);                /* se pone en pantalla
*/
                }
                if ( alabeo==1561 )
                {
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("30 Grados");;                           /* se pone en pantalla */
                }
        if ( alabeo > 1390 && alabeo < 1561 )
                {
                        alabeo=((alabeo-1390)/171)+30;
                        gotoxy(16,10);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", alabeo),                /* se pone en pantalla
*/
                }
                if ( alabeo==1390 )
                {
```

```
                    gotoxy(16,10);                                          /*
Datos a la pantalla */
                    printf("40 Grados");                    /* se pone en pantalla */
                }
            if ( alabeo > 1245 && alabeo < 1390 )
                {
                    alabeo=((alabeo-1245)/145)+40;
                    gotoxy(16,10);                                          /*
Datos a la pantalla */
                    printf("%3.1f       Grados", alabeo);           /* se pone en pantalla
*/
                }
                if ( alabeo<=1245 )
                {
                    gotoxy(16,10);                                          /*
Datos a la pantalla */
                    printf("Inclinación mayor a 50 Grados");            /* se pone en
pantalla */
                }
                /* Datos del giroscopo (cabeceo) */

                if ( cabeceo<=1480 )
                {
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("Inclinación menor a -60 Grados");           /* se pone en
pantalla */
                }
            if ( cabeceo > 1480 && cabeceo < 1581 )
                {
                    cabeceo=-(((cabeceo-1480)/1101)+50);
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("%3.1f       Grados", cabeceo);          /* se pone en
pantalla */
                }
                if ( cabeceo==1581 )
                {
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("-50 Grados");               /* se pone en pantalla */
                }
            if ( cabeceo > 1581 && cabeceo < 1672 )
                {
                    cabeceo=-(((cabeceo-1581)/91)+40);
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("%3.1f       Grados", cabeceo);          /* se pone en
pantalla */
                }
                if ( cabeceo==1672 )
                {
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("-40 Grados");               /* se pone en pantalla */
                }
            if ( cabeceo > 1672 && cabeceo < 1764 )
                {
                    cabeceo=-(((cabeceo-1672)/92)+30);
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("%3.1f       Grados", cabeceo);          /* se pone en
pantalla */
                }
                if ( cabeceo==1764 )
                {
                    gotoxy(16,11);                                          /*
Datos a la pantalla */
                    printf("-30 Grados");               /* se pone en pantalla */
                }
```

```
        if ( cabeceo > 1764 && cabeceo < 1855 )
                {
                        cabeceo=-(((cabeceo-1754)/91)+20);
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", cabeceo);               /* se pone en
pantalla */
                }
                if ( cabeceo==1855 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("-20 Grados");;                                  /* se pone en pantalla */
                }
        if ( cabeceo > 1855 && cabeceo < 1936 )
                {
                        cabeceo=-(((cabeceo-1855)/81)+10);
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", cabeceo);               /* se pone en
pantalla */
                }
                if ( cabeceo==1936 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("-10 Grados");;                                  /* se pone en pantalla */
                }
        if ( cabeceo > 1936 && cabeceo < 2047 )
                {
                        cabeceo=-((cabeceo-1936)/111);
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", cabeceo);               /* se pone en
pantalla */
                }
                if ( cabeceo==2047 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("0 Grados");;                                    /* se pone en pantalla */
                }
        if ( cabeceo > 2047 && cabeceo < 2149 )
                {
                        cabeceo=((cabeceo-2047)/102);
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", cabeceo);               /* se pone en
pantalla */
                }
                if ( cabeceo==2149 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("10 Grados");;                                   /* se pone en pantalla */
                }
        if ( cabeceo > 2149 && cabeceo < 2250 )
                {
                        cabeceo=((cabeceo-2149)/101)+10;
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f          Grados", cabeceo);               /* se pone en
pantalla */
                }
                if ( cabeceo==2250 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("20 Grados");;                                   /* se pone en pantalla */
                }
```

```
        if ( cabeceo > 2250 && cabeceo < 2372 )
                {
                        cabeceo=((cabeceo-2250)/122)+20;
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f       Grados", cabeceo),                  /* se pone en
pantalla */
                }
                if ( cabeceo==2372 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("30 Grados");;                                   /* se pone en pantalla */
                }
        if ( cabeceo > 2370 && cabeceo < 2483 )
                {
                        cabeceo=((cabeceo-2370)/111)+30;
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f       Grados", cabeceo),                  /* se pone en
pantalla */
                }
                if ( cabeceo==2483 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf(" 40 Grados");,                                  /* se pone en pantalla */
                }
                if ( cabeceo > 2483 && cabeceo < 2625 )
                {
                        cabeceo=((cabeceo-2483)/142)+40;
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f       Grados", cabeceo);                  /* se pone en
pantalla */
                }
                if ( cabeceo==2625 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf(" 50 Grados");;                                  /* se pone en pantalla */
                }
                if ( cabeceo > 2625 && cabeceo < 2818 )
                {
                        cabeceo=((cabeceo-2625)/193)+50;
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf("%3.1f       Grados", cabeceo);                  /* se pone en
pantalla */
                }
                if ( cabeceo>=2818 )
                {
                        gotoxy(16,11);                                          /*
Datos a la pantalla */
                        printf(" Inclinación mayor a 60 Grados");;              /* se pone en
pantalla */
                }


        return;
}

/* Recepción de datos acelerometros */
void acel()
{
        int dato, dato1, dato2, dato3, dato4, dato5, dato10, dato11, dato12, dato13, dato14, dato15;
        int acelx, acelx1, acelx3, acely, acely1, acely3;
        int acelz, acelz1, acelz3;
  outportb(REG_TX,0x16);
```

```
delay(10);
dato=inportb(REG_RX);                    /* Lee lo que hay en el puerto COM2 */
outportb(REG_TX,0x17);
delay(10);
dato1=inportb(REG_RX);                   /* Lee lo que hay en el puerto COM2 */
acebx=(dato*256)+dato1;
        outportb(REG_TX,0x1a);
delay(10);
dato2=inportb(REG_RX);                   /* Lee lo que hay en el puerto COM2 */
outportb(REG_TX,0x1b);
delay(10);
dato3=inportb(REG_RX);                   /* Lee lo que hay en el puerto COM2 */
aceby=(dato2*256)+dato3;
        outportb(REG_TX,0x18);
delay(10);
dato4=inportb(REG_RX);                   /* Lee lo que hay en el puerto COM2 */
outportb(REG_TX,0x19);
delay(10);
dato5=inportb(REG_RX);                   /* Lee lo que hay en el puerto COM2 */
acebz=(dato4*256)+dato5;
delay(100);
outportb(REG_TX,0x16);
delay(10);
dato10=inportb(REG_RX);                  /* Lee lo que hay en el puerto COM2 */
outportb(REG_TX,0x17);
delay(10);
dato11=inportb(REG_RX);                  /* Lee lo que hay en el puerto COM2 */
acebx1=(dato10*256)+dato11;
        outportb(REG_TX,0x1a);
delay(10);
dato12=inportb(REG_RX);                  /* Lee lo que hay en el puerto COM2 */
outportb(REG_TX,0x1b);
delay(10);
dato13=inportb(REG_RX);                  /* Lee lo que hay en el puerto COM2 */
aceby1=(dato12*256)+dato13;
        outportb(REG_TX,0x18);
delay(10);
dato14=inportb(REG_RX);                  /* Lee lo que hay en el puerto COM2 */
outportb(REG_TX,0x19);
delay(10);
dato15=inportb(REG_RX);                  /* Lee lo que hay en el puerto COM2 */
acebz1=(dato14*256)+dato15;

        if ( acebx == acebx1 )
        {
        distanciax=((.5*(acebx1-30309)/66.67)*(.1))+ distanciax;
        }
 else
        {
        acebx3=(acebx1-aceba)/66.67;
        distanciax=(.5*acebx3*(.01))+distancia;
        }
 if ( aceby == aceby1 )
        {
        distanciay=((.5*(aceby1-2020)/66.67)*(.1))+ distanciay;
        }
 else
        {
        aceby3=(aceby1-aceby)/66.67;
        distanciay=(.5*aceby3*(.01))+distanciay;
        }
        if ( acebz == acebz1 )
        {
        distanciaz=((.5*(acebz1-2020)/66.67)*(.1))+ distanciaz;
        }
 else
        {
        acebz3=(acebz1-acebz)/66.67;
        distanciaz=(.5*acebz3*(.01))+distanciaz;
        }
```

```
}

/* Modo Gps condiciones iniciales */
void gps()
{
        int i;
        void interrupt rx_gps();                        /* Servicio de interrupcion del GPS */
        void interrupt (*oldcom1)();
        inicializacom1();                               /* Inicializa Com1 comunicacion GPS */
        oldcom1 = getvect(0x0c);
        setvect(0x0c, rx_gps);                          /* Nuevo servicio de interrupcion */

                                                        /* Inicializa las variables */
        datos_gps = 0x00;
        dato_gps = 0x00;
        cual_gps = 0x01;
        *cad_gps1 = 0x00;
        *cad_gps2 = 0x00;
        posicioncaracter = 0;
        outportb(0x21, 0x20);                           /* Habilita IRQ4=Com1=GPS */
        datosdelgps();
        while (modo_operacion == Modo_Gps){             /* Ciclo de espera para salida */
                almacena();
                if(kbhit()){   activamenu();            /* Atiende al usuario */
                if(nueva_funcion) ejecuta();}
        }
        outportb(0x21,0xB8);
                /* Restaura controlador de int */
        setvect(0x0c, oldcom1);
                /* Restaura vector de int */
        cerrar();
        com1reset();
        return;
}

/* Servicio de interrupción para RX de datos del GPS */
void interrupt rx_gps()
{
        disable();                                                      /* Sin interrupciones Hardware */
        outportb(0x20, 0x64);                           /* Borra IRQ4 */
        if(cual_gps){
                cad_gps1[posicioncaracter]= inportb(0x3f8);     /* Lee caracter enviado por GPS */
                cad_gps1[posicioncaracter+1] = 0x00;            /* Caracter de fin de cadena */
                if(cad_gps1[posicioncaracter] == 0x0A){                 /* Caracter de fin de cadena */
                        posicioncaracter=0;                     /* Actualiza apuntador */
                        cual_gps = 0x00;                        /* Conmuta cadena */
                        date_gps = 0x01;}                       /* Bandera de mensaje */
                else posicioncaracter++;}
        else
        {
                cad_gps2[posicioncaracter] = inportb(0x3f8);
                cad_gps2[posicioncaracter+1] = 0x00;            /* Caracter de fin de cadena */
                if(cad_gps2[posicioncaracter] == 0x0A){         /* Caracter de fin de cadena */
                        posicioncaracter=0;                     /* Actualiza apuntador */
                        cual_gps = 0x01;                        /* Conmuta cadena */
                        date_gps = 0x01;}                       /* Bandera de mensaje */
                else posicioncaracter++;}
        enable();
}


/* Recibiendo los datos del GPS */
void datosdelgps()
{
        final cadena, reg;
        disable ();
        outportb(0x20, 0x64);                           /* Borra IRQ4 */
        if(cual_gps){
```

```
                reg=inportb(0x3f8);
                /* Checamos si caracter= '$' */
                if (reg==0x24){
                do {
                cadena = inportb(0x3f8);
                cad_gps1[posicioncaracter]=cadena;                /* Lee caracter enviado por GPS */
                cad_gps1[posicioncaracter+1] = 0x00;              /* Caracter de fin de cadena */
                posicioncaracter++;
                delay(35);
                                /* Tiempo de espera */
                } while (cadena != 0x0A);                       /* Caracter de fin de cadena */
                        posicioncaracter=0;                     /* Actualiza apuntador */
                        cual_gps = 0x00,                        /* Conmuta cadena */
                        dato_gps = 0x01;                        /* Bandera de mensaje */
        }}
        else
        {
                reg=inportb(0x3f8);
                /* Checamos si caracter= '$' */
                if (reg==0x24){
                do {
                cadena= inportb(0x3f8);
                cad_gps2[posicioncaracter] = cadena;                                        /* Lee caracter
enviado por GPS */
                cad_gps2[posicioncaracter+1] = 0x00,               /* Caracter de fin de cadena */
                posicioncaracter++;
                delay(35);
                                /* Tiempo de espera */
                } while (cadena != 0x0A);                       /* Caracter de fin de cadena */
                        posicioncaracter=0,                     /* Actualiza apuntador */
                        cual_gps = 0x01;                        /* Conmuta cadena */
                        date_gps = 0x01;                        /* Bandera de mensaje */
        }}
        enable();
}

/* Almacena en disco la informacion del gps */
void almacena()
{
        datosdelgps();
                        /* Leemos mensaje */
        if(date_gps){
                if(cual_gps){  strcpy(mensaje_gps, cad_gps2);          /* Actualiza mensaje del GPS */
                                                        revisamensajegps();             /* Revisa el mensaje */
                                                        cad_gps2[0] = 0x00;             /* Restaura cadena para
otro mensaje */
                                                        }
                else {      strcpy(mensaje_gps, cad_gps1);          /* Actualiza mensaje del GPS */
                                                        revisamensajegps();             /* Revisa el mensaje */
                                                        cad_gps1[0] = 0x00;             /* Restaura cadena para
otro mensaje */
                }
                date_gps=0x00;}
        datosapantalla();
        return;
}

/* Revisa el mensaje recibido del GPS */
void revisamensajegps()
{
        if(checa_gps(mensaje_gps)==0){
                switch(tipomensajegps(mensaje_gps)){
                case GPGGA:    gpgga(mensaje_gps);
                        break;
                case PMGLB:    pmglb(mensaje_gps);
                        break;
                case GPZDA:    gpzda(mensaje_gps);
                        break;
                default:       mensajes(mensaje_gps, VidNormal3);
                        break;}
```

```
        }
        return;
}

/* Revisa si contiene error el mensaje del GPS */
int checa_gps()
{
        char sumcheck1[2], *apunta, asterisco='*';
        int x = 1;                                      /* Si error a menos que otra cosa */
        generachecksum(sumcheck1);                      /* Genera el caracter de erro para compararlo */
        sumcheck1[2] = 0x00;                            /* Fin de cadena */
        apunta = strchr(mensaje_gps, asterisco);        /* Copia el caracter de comprobación de error */
        if(apunta)                                      /* Si existe y son iguales */
                if((sumcheck1[0] == *(apunta+1))&&(sumcheck1[1]==*(apunta+2))) x=0;
                return(x);
}
/* Genera caracteres de chek sum, protocolo de comunicación con GPS */
void generachecksum(sumcheck2)
char *sumcheck2;
{
        unsigned chk_gps;
        int temp=2;
        chk_gps = *(mensaje_gps+1);                     /* A partir del primer caracter */
        for(temp=2;*(mensaje_gps+temp)!='*';temp++)     /* Calcula cheksum */
                if(*(mensaje_gps+temp)!= 0x00) chk_gps = (chk_gps ^ *(mensaje_gps+temp));
                *(sumcheck2+1) = (chk_gps & 0x0F);      /* Transforma el valor a ASCII */
                *sumcheck2 = (chk_gps >> 4);
                if(*sumcheck2 < 0x0A) *sumcheck2 += 0x30;
                                      else    *sumcheck2 += 0x37;
                if(*(sumcheck2+1) < 0x0A) *(sumcheck2+1) += 0x30;
                                      else    *(sumcheck2+1) += 0x37;
                *(sumcheck2+2) = 0x00;                  /* Fin de cadena */
                return;
}

/* Revisa el tipo de mensaje del GPS */
int tipomensajegps()
{
        if(strstr(mensaje_gps,"$GPG")==mensaje_gps) return(GPGGA);
        if(strstr(mensaje_gps,"$PMG")==mensaje_gps) return(PMGLB);
        if(strstr(mensaje_gps,"$GPZ")==mensaje_gps) return(GPZDA);
        return(0);
}

/* Obtiene parametros del mensaje $GPGGA del GPS */
void gpgga()
{
        char *latitud, *longitud;
        char *signo;
        mensajes("Recibiendo Datos                                    ", VidNormal);
        latitud = malloc(7);
        longitud = malloc(8);
        signo = malloc(1);
        strncpy(latitud,(mensaje_gps+14),7);            /* Obtiene latitud */
        *(latitud+7) = 0x00;                            /* Fin de cadena */
        strncpy(signo,(mensaje_gps+22),1);              /* Obtiene signo */
        *(signo+1) = 0x00;                              /* Fin de cadena */
        datos.latitud = atof(latitud);                  /* Convierte a numerico */
        if(*signo == 'S'){
                gotoxy(12,5);
                printf("%8.2f S",datos.latitud);}
        gotoxy(12,5);
        printf("%8.2f N",datos.latitud);

        strncpy(longitud,(mensaje_gps+24),8);           /* Obtiene longitud */
        *(longitud+8) = 0x00;                           /* Fin de cadena */
        strncpy(signo,(mensaje_gps+33),1);              /* Obtiene signo */
        *(signo+1) = 0x00;                              /* Fin de cadena */
        datos.longitud= atof(longitud);                 /* Convierte a numerico */
        if(*signo == 'W') {
```

```
                gotoxy(11,6);
                printf("%9.2f W",datos.longitud);}
        gotoxy(11,6);
        printf("%9.2f E",datos.longitud);

        free(latitud);                          /* Libera memoria */
        free(longitud);
        free(signo);
        return;
}

/* Obtiene parametros del mensaje $PMGLB del GPS */
void pmglb()
{
        char *temp;
        temp = malloc(6);                       /* Obtiene altitud */
        strncpy(temp, (mensaje_gps+10),6);
        *(temp+6)= 0x00;
        datos.altitud = atoi(temp);
        gotoxy(13,7);
        printf(" %6d Mts.", datos.altitud);
        free(temp);
        return;
}

/* Obtiene parametros del mensaje $GPZDA del GPS */
void gpzda()
{
        char *tiempo, *fecha;
        tiempo = malloc(6);
        /* Obtiene la hora */
        fecha = malloc(10);
        /* obtiene la fecha */
        strncpy(tiempo, (mensaje_gps+7),6);
        *(tiempo+6)= 0x00;
        gotoxy(13,4);
        printf("%s  GMT",tiempo);
        strncpy(fecha, (mensaje_gps+14),10);
        *(fecha+10)= 0x00;
        gotoxy(13,3);
        printf("%s ",fecha);
        free(tiempo);
        free(fecha);
        return;
}


/* Transmite comandos al GPS */
void tx_gps(gps_mensaje)
char *gps_mensaje;                      /* Mensaje al GPS */
{
        unsigned  temp;
        while(*gps_mensaje != 0x00){
                temp=inportb(0x3fd);                    /* Lee estado de Comm1 */
                if(temp & 0x40) outportb(0x3f8,*gps_mensaje++);         /* Transmite */
        }
        return,
}

/* Lee Mensaje para TX al GPS */
void al_gps()
{
        char *gps_mensaje, *sumcheck, *temp;
        gps_mensaje = malloc(100);                      /* Asigna memoria */
        sumcheck = malloc(3);                           /* Caracter de verificación de error */
        *gps_mensaje= 0x00;                             /* Fin de cadena */
        *sumcheck = 0x00;
        mensajes("Comando:                                      ", VidNormal3),
        leecomandogps(gps_mensaje),                     /* Obtiene mensaje */
```

```
        if(*gps_mensaje==0x00){ mensajes("\nCancelado                                    ", VidNormal3);
/* Ningun mensaje al GPS */
        return;}

        generachecksumtx(gps_mensaje, sumcheck);          /* Genera el caracter check sum */
        temp = gps_mensaje;
        while(*++temp != 0x00);                           /* Puntero al final del mensaje */
        *temp++ = *sumcheck++;                            /* Agrega los de verificación de error */
        *temp++ = *sumcheck++;
        *temp++ = 0x0D;                                   /* Agrega los de fin de mensaje */
        *temp++ = 0x0A;
        *temp = 0x00;                                     /* Fin de Cadena */
        tx_gps(gps_mensaje);                              /* Transmite mensaje al GPS */
        mensajes("Transmitido el comando.                          ", VidNormal3);
        free(gps_mensaje);                                /* Libera memoria */
        free(sumcheck);
        return;
}

/* Lee comando para TX al GPS */
void leecomandogps(nombre)
char *nombre,                                     /* Mensaje al GPS */
{
        union tecla{   char ch[2];
        int i;} c;                                /* Tecla presionada */
        int j;
        c.ch[0] = 1;
        for(j=0;c.ch[0]!='\r';j++){               /* Sale al return */
                c.i = bioskey(0);                 /* Lee tecla */
                switch(c.ch[0]){                  /* Tipo de tecla */
                        case '\r':nombre[j] = 0x00;          /* Si enter fin de cadena */
                                break;
                        case '\b':if(j--){                   /* Si tecla <- */
                                                             write_char(mensajesx-1, 10+j, 0x20, VidNormal3);
                                                             }-;                    /* Borra anterior */
                                }
                                break;
                        case ESC, c.ch[0] = '\r';            /* Enter para salir */
                                nombre[0] = 0x00;            /* Cadena vacia */
                                break;
                        default: nombre[j] = c.ch[0];        /* Almacena caracter */
                                nombre[j+1] = 0x00;          /* Fin de cadena */
                                write_char(mensajesx-1, 10+j,c.ch[0], VidNormal3);
                                break;
                }
        }
        if(j>100) nombre[100] = 0x00;             /* Filtro de longitus de comando */
        return;
}

/* Inicializa el gps */
void inicializagps()
{
        char *gps_mensaje, *sumcheck, *temp,
        mensajes("Inicializando GPS...                    ", VidNormal3);
        gps_mensaje = malloc(70);                 /* Asigna memoria */
        sumcheck = malloc(3),
        temp = malloc(3);
        *gps_mensaje = 0x00;                      /* Mensaje GPS */
        *sumcheck = 0x00;                         /* Caracteres de verificacion de error */
        *temp =0x0D;                              /* Caracteres de fin de mensaje */
        *(temp+1) = 0x0A;
        *(temp+2) = 0x00;

        strcpy(gps_mensaje,"$PMGLN,00,1*"),               /* Mensaje Master Reset */
        generachecksumtx(gps_mensaje, sumcheck);          /* Genera el caracter check sum */
        strcat(gps_mensaje, sumcheck);                    /* Concatena check sum */
        strcat(gps_mensaje, temp),                        /* Concatena fin de mensaje */
        tx_gps(gps_mensaje),                              /* Transmite el mensaje al GPS */
        delay(100);                               /* Necesario para TX despues de un reset */
```

```
        strcpy(gps_mensaje,"$PMGLI,00,F02,2,A,*");          /* Pide información de satélites */
        generachecksumbx(gps_mensaje, sumcheck);            /* Genera el caracter check sum */
        strcat(gps_mensaje, sumcheck);                      /* Concatena check sum */
        strcat(gps_mensaje, temp);                          /* Concatena fin de mensaje */
        tx_gps(gps_mensaje);                                /* Transmite el mensaje al GPS */
        delay(1);                                           /* Necesario para TX despues de un reset */

        free(gps_mensaje);                                  /* libera memoria */
        free(sumcheck);
        free(temp);
        mensajes("GPS inicializado.                                    ", VidNormal3);
        return;
}

/* Pide los datos principales al GPS */
void pidedatosgps()
{
        char *gps_mensaje, *sumcheck, *temp;
        mensajes("Pidiendo datos...                                    ", VidNormal3);
        gps_mensaje = malloc(70);                           /* Asigna memoria y obtiene el mensaje */
        sumcheck = malloc(3);
        temp = malloc(3);
        *gps_mensaje = 0x00;                                /* Mensaje a ser TX al GPS */
        *sumcheck = 0x00;                                   /* Verificación de error */
        *temp = 0x0D;                                       /* Caracteres de fin de mensaje */
        *(temp+1) = 0x0A;
        *(temp+2) = 0x00;

        strcpy(gps_mensaje,"$PMGLD,00,3*");                 /* Modo 3D/Auto */
        generachecksumbx(gps_mensaje, sumcheck);            /* genera caracter check sum */
        strcat(gps_mensaje, sumcheck);                      /* Concatena check sum */
        strcat(gps_mensaje, temp);                          /* Concatena fin de mensaje */
        tx_gps(gps_mensaje);                                /* Transmite el mensaje al GPS */
        delay(1);
        strcpy(gps_mensaje,"$PMGLI,00,B00,2,A,*");          /* Latitud, longitud */
        generachecksumbx(gps_mensaje, sumcheck);            /* genera caracter check sum */
        strcat(gps_mensaje, sumcheck);                      /* Concatena check sum */
        strcat(gps_mensaje, temp);                          /* Concatena fin de mensaje */
        tx_gps(gps_mensaje);                                /* Transmite el mensaje al GPS */
        delay(1);
        strcpy(gps_mensaje,"$PMGLI,00,B02,2,A,*");          /* Altitud */
        generachecksumbx(gps_mensaje, sumcheck);            /* genera caracter check sum */
        strcat(gps_mensaje, sumcheck);                      /* Concatena check sum */
        strcat(gps_mensaje, temp);                          /* Concatena fin de mensaje */
        tx_gps(gps_mensaje);                                /* Transmite el mensaje al GPS */
        delay(1);
        strcpy(gps_mensaje,"$PMGLI,00,A00,2,A,*");          /* Tiempo */
        generachecksumbx(gps_mensaje, sumcheck);            /* genera caracter check sum */
        strcat(gps_mensaje, sumcheck);                      /* Concatena check sum */
        strcat(gps_mensaje, temp);                          /* Concatena fin de mensaje */
        tx_gps(gps_mensaje);                                /* Transmite el mensaje al GPS */
        delay(1);

        free(gps_mensaje);                                  /* Libera memoria */
        free(sumcheck);
        free(temp);
        mensajes("Enviando los comandos.                               ", VidNormal3);
        return;
}

/* Borra la memoria del GPS */
void borragps()
{
        char *gps_mensaje, *sumcheck, *temp;
        mensajes("Borrando memoria del GPS..                           ", VidNormal3);
        gps_mensaje = malloc(70);                           /* Asigna memoria */
        sumcheck = malloc(3);
        temp = malloc(3);
        *gps_mensaje = 0x00,                                /* TX al GPS */
```

```
        *sumcheck = 0x00;                                   /* Verificación de error */
        *temp = 0x0D;                                      /* Fin de mensaje */
        *(temp+1) = 0x0A;
        *(temp+2) = 0x00;
        strcpy(gps_mensaje,"$PMGLN,00,1*");                        /* Master Reset */
        generachecksumtx(gps_mensaje, sumcheck);                    /* genera caracter check sum */
        strcat(gps_mensaje, sumcheck);                          /* Concatena check sum */
        strcat(gps_mensaje, temp);                             /* Concatena fin de mensaje */
        tx_gps(gps_mensaje);                                  /* Transmite el mensaje al GPS */
        delay(100);
        free(gps_mensaje);                                   /* Libera memoria */
        free(sumcheck);
        free(temp);
        mensajes("Memoria Borrada del GPS.                                    ", VidNormal3);
        return;
}

/* Genera caracteres de check sum, protocolo de comunicación con GPS TX */
void generachecksumtx(txpalabra,sumcheck2)
char *txpalabra;                                  /* Mensaje tipo GPS */
char *sumcheck2;                                   /* Caracter de verificación de error */
{
        unsigned chk_gps;
        int temp=2;

        chk_gps = *(txpalabra+1);                                  /* A partir del primer cacater */
        for(temp=2;*(txpalabra+temp)!= '*';temp++)                      /* Calcula sumcheck */
                if(*(txpalabra+temp)!= 0x00) chk_gps = (chk_gps ^*(txpalabra+temp));
        *(sumcheck2+1) = (chk_gps & 0x0F);                          /* Convierte a ASCII */
        *sumcheck2 = (chk_gps >> 4);
        if(*sumcheck2 < 0x0A) *sumcheck2 += 0x30;
                              else    *sumcheck2 += 0x37;
        if(*(sumcheck2+1) < 0x0A) *(sumcheck2+1) += 0x30;
                                  else    *(sumcheck2+1) += 0x37;
        *(sumcheck2+2) = 0x00;                              /* Fin de cadena */
        return;
}
```

Apéndice C.

Negativos de los circuitos.

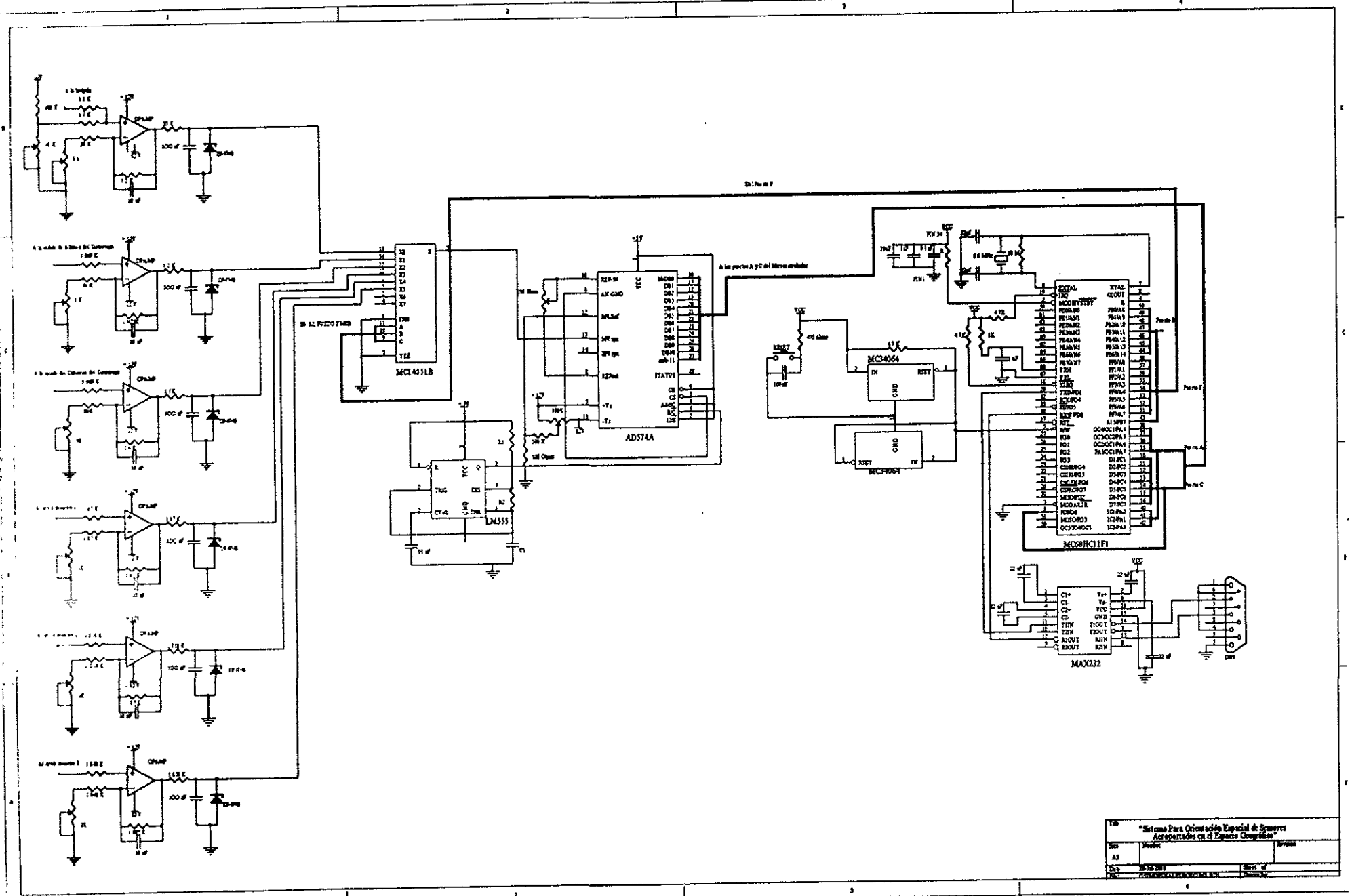
En este apéndice se presentan los negativos de los circuitos impresos, así como el diagrama eléctrico y los componentes que lo integran.

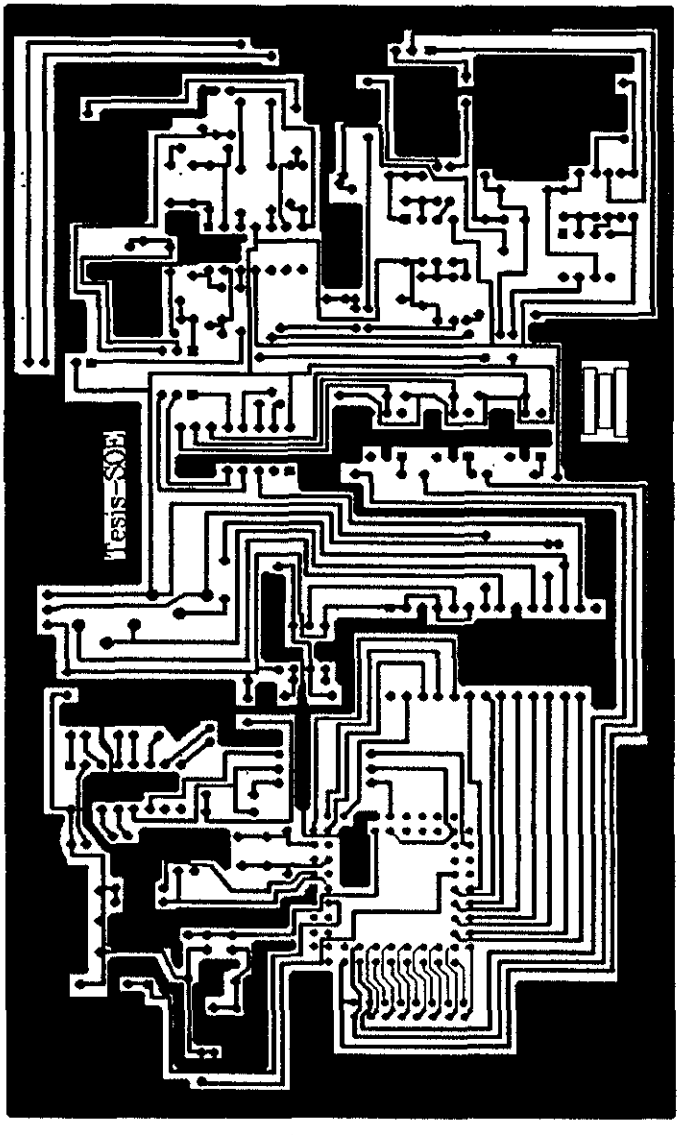
Carátulas de la caja que contiene el Circuito.

En estas imágenes se muestran, las carátulas frontales y posteriores del Sistema.

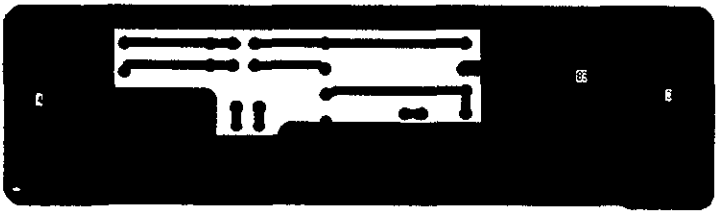
En la carátula principal (Frontal), se encuentra la entrada de la alimentación, la cual requiere de +12 V c.d., -12V c.d. y Tierra, así como el interruptor de encendido y un botón de reset para reiniciar el sistema en el caso que así se requiera.

En la cara posterior, se encuentran las entradas de los diferentes dispositivos que se usan en el sistema.

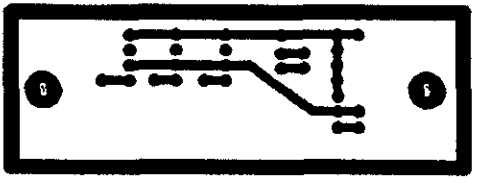

MC1451B
AD574A
LM555
MC34064
MC34064
MC68HC11F1
MAX232
"Sistema Para Orientación Espacial de Aparatos Aeroportados en el Espacio Geográfico"



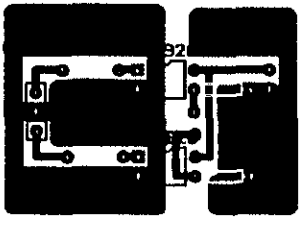
Vista Inferior del Circuito.



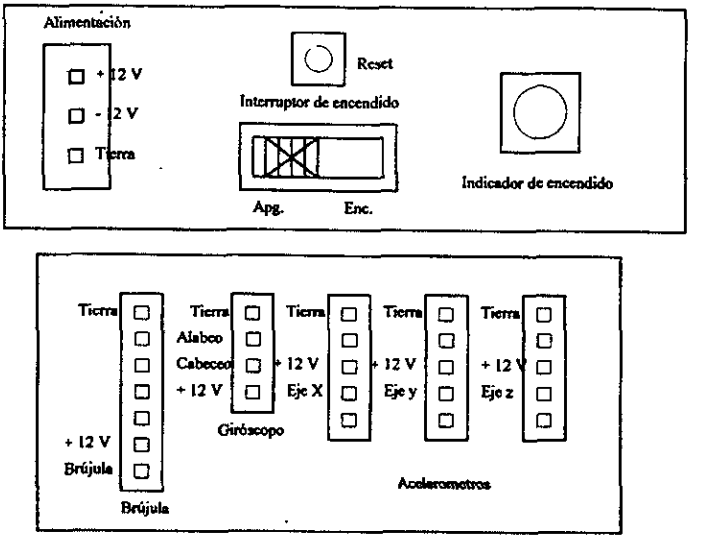
Carátula Frontal



Conectores de los componentes



Sistema de acoplo óptico para el giroscopo.



Carátulas Frontal y Posterior, con especificaciones.

Apéndice D.

Especificaciones de los componentes.

En el presente apéndice, anexo las hojas de especificaciones técnicas de los componentes que considero son lo de mayor relevancia para el desempeño optimo del sistema realizado.

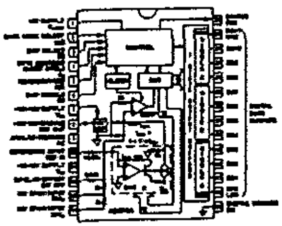
ANALOG DEVICES

Complete 12-Bit A/D Converter

AD574A*

FEATURES
- Complete 12-Bit A/D Converter with Reference and Clock
- 8- and 16-Bit Microprocessor Bus Interface
- Guaranteed Linearity Over Temperature
  - 0°C to +70°C – AD574AJ, K, L
  - –55°C to +125°C – AD574AS, T, U
- No Missing Codes Over Temperature
- 35 µs Maximum Conversion Time
- Buried Zener Reference for Long-Term Stability and Low Gain T.C. 10 ppm/°C max AD574AL; 12.5 ppm/°C max AD574AU
- Ceramic DIP, Plastic DIP or PLCC Package
- Available in Higher Speed, Pinout-Compatible Versions (15 µs AD674B, 8 µs AD774B; 10 µs Sample-Hold AD1674)
- Available in Versions Compliant with MIL-STD-883 and JAN QPL

BLOCK DIAGRAM AND PIN CONFIGURATION



PRODUCT DESCRIPTION

The AD574A is a complete 12-bit successive-approximation analog-to-digital converter with 3-state output buffer circuitry for direct interface to an 8- or 16-bit microprocessor bus. A high precision voltage reference and clock are included on-chip, and the circuit guarantees full-rated performance without external circuitry or clock signals.

The AD574A design is implemented using Analog Devices' Bipolar/I²L process, and integrates all analog and digital functions on one chip. Offset, linearity and scaling errors are minimized by active laser-trimming of thin-film resistors at the wafer stage. The voltage reference uses an implanted buried Zener for low noise and low drift. On the digital side, I²L logic is used for the successive-approximation register, control circuitry and 3-state output buffers.

The AD574A is available in six different grades. The AD574AJ, K, and L grades are specified for operation over the 0°C to +70°C temperature range. The AD574AS, T, and U are specified for the –55°C to +125°C range. All grades are available in a 28-pin hermetically-sealed ceramic DIP. Also, the J, K, and L grades are available in a 28-pin plastic DIP and PLCC, and the J and K grades are available in ceramic LCC.

The S, T, and U grades in ceramic DIP or LCC are available with optional processing to MIL-STD-883C Class B; the T and U grades are available as JAN QPL. The Analog Devices' Military Products Databook should be consulted for details on /883B testing of the AD574A.

PRODUCT HIGHLIGHTS

1. The AD574A interfaces to most 8- or 16-bit microprocessors. Multiple-mode three-state output buffers connect directly to the data bus while the read and convert commands are taken from the control bus. The 12 bits of output data can be read either as one 12-bit word or as two 8-bit bytes (one with 8 data bits, the other with 4 data bits and 4 trailing zeros).
2. The precision, laser-trimmed scaling and bipolar offset resistors provide four calibrated ranges: 0 volts to +10 volts and 0 volts to +20 volts unipolar, –5 volts to +5 volts and –10 volts to +10 volts bipolar. Typical bipolar offset and full-scale calibration errors of ±0.1% can be trimmed to zero with one external component each.
3. The internal buried Zener reference is trimmed to 10.00 volts with 0.2% maximum error and 15 ppm/°C typical T.C. The reference is available externally and can drive up to 1.5 mA beyond the requirements of the reference and bipolar offset resistors.
4. AD674B (15 µs) and AD774B (8 µs) provide higher speed, pin compatibility; AD1674 (10 µs) includes on-chip Sample-Hold Amplifier (SHA).

REV. B

AD574A—SPECIFICATIONS (@ +25°C with V_{CC} = +15 V or +12 V, V_{LOGIC} = +5 V, V_{EE} = –15 V or –12 V unless otherwise noted)

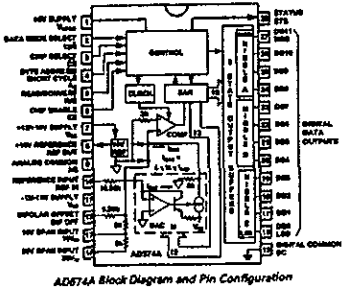
Model	AD574AJ Min	Typ	Max	AD574AK Min	Typ	Max	AD574AL Min	Typ	Max	Units
RESOLUTION			12			12			12	Bits
LINEARITY ERROR @ +25°C			±1			±1/2			±1/2	LSB
T_{min} to T_{max}			±1			±1/2			±1/2	LSB
DIFFERENTIAL LINEARITY ERROR (Minimum Resolution for Which No Missing Codes are Guaranteed) T_{min} to T_{max}	11			12			12			Bits
UNIPOLAR OFFSET (Adjustable to Zero)			±2			±1			±1	LSB
BIPOLAR OFFSET (Adjustable to Zero)			±4			±4			±2	LSB
FULL-SCALE CALIBRATION ERROR (With Fixed 50 Ω Resistor from REF OUT to REF IN) (Adjustable to Zero)			0.25			0.25			0.125	% of FS
TEMPERATURE RANGE	0		+70	0		+70	0		+70	°C
TEMPERATURE COEFFICIENTS (Using Internal Reference) T_{min} to T_{max}										
Unipolar Offset			±2 (10)			±1 (5)			±1 (5)	LSB (ppm/°C)
Bipolar Offset			±2 (10)			±1 (5)			±1 (5)	LSB (ppm/°C)
Full-Scale Calibration			±9 (50)			±5 (27)			±2 (10)	LSB (ppm/°C)
POWER SUPPLY REJECTION Max Change in Full-Scale Calibration										
V_{CC} = 15 V ± 1.5 V or 12 V ± 0.6 V			±2			±1			±1	LSB
V_{LOGIC} = 5 V ± 0.5 V			±1/2			±1/2			±1/2	LSB
V_{EE} = –15 V ± 1.5 V or –12 V ± 0.6 V			±2			±1			±1	LSB
ANALOG INPUT Input Ranges Bipolar	–5		+5	–5		+5	–5		+5	Volts
	–10		+10	–10		+10	–10		+10	Volts
Unipolar	0		+10	0		+10	0		+10	Volts
	0		+20	0		+20	0		+20	Volts
Input Impedance 10 Volt Span	3	5	7	3	5	7	3	5	7	kΩ
20 Volt Span	6	10	14	6	10	14	6	10	14	kΩ
DIGITAL CHARACTERISTICS[1] (T_{min}–T_{max}) Inputs[2] (CE, $\overline{CS}$, R/$\overline{C}$, A_0) Logic "1" Voltage	+2.0		+5.5	+2.0		+5.5	+2.0		+5.5	Volts
Logic "0" Voltage	–0.5		+0.8	–0.5		+0.8	–0.5		+0.8	Volts
Current	–20		+20	–20		+20	–20		+20	µA
Capacitance		5			5			5		pF
Output (DB11–DB0, STS) Logic "1" Voltage (I_{SOURCE} ≤ 500 µA)	+2.4			+2.4			+2.4			Volts
Logic "0" Voltage (I_{SINK} ≤ 1.6 mA)			+0.4			+0.4			+0.4	Volts
Leakage (DB11–DB0, High-Z State)	–20		+20	–20		+20	–20		+20	µA
Capacitance		5			5			5		pF
POWER SUPPLIES Operating Range[3] V_{LOGIC}	+4.5		+5.5	+4.5		+5.5	+4.5		+5.5	Volts
V_{CC}	+11.4		+16.5	+11.4		+16.5	+11.4		+16.5	Volts
V_{EE}	–11.4		–16.5	–11.4		–16.5	–11.4		–16.5	Volts
Operating Current I_{LOGIC}		30	40		30	40		30	40	mA
I_{CC}		2	5		2	5		2	5	mA
I_{EE}		18	30		18	30		18	30	mA
POWER DISSIPATION		390	725		390	725		390	725	mW
INTERNAL REFERENCE VOLTAGE	9.98	10.0	10.02	9.98	10.0	10.02	9.98	10.0	10.02	Volts
Output Current (Available for External Loads)[3] (External Load Should not Change During Conversion)			1.5			1.5			1.5	mA
PACKAGE OPTIONS[4] Ceramic (D-28)		AD574AJD			AD574AKD			AD574ALD		
Plastic (N-28)		AD574AJN			AD574AKN			AD574ALN		
PLCC (P-28A)		AD574AJP			AD574AKP					
LCC (E-28A)		AD574AJE			AD574AKE					

NOTES
[1] Detailed Timing Specifications appear in the Timing Section.
[2] 12/$\overline{8}$ Input is not TTL-compatible and must be hard wired to V_{LOGIC} or Digital Common.
[3] The reference should be buffered for operation on ±12 V supplies.
[4] D = Ceramic DIP; N = Plastic DIP; P = Plastic Leaded Chip Carrier.
Specifications subject to change without notice.

–2– REV. B

	AD574AJ/K			AD574AS/T			AD574AU			Units
	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Model										
RESOLUTION		12			12			12		Bits
LINEARITY ERROR @ +25°C T_{MIN} to T_{MAX}			±1/2 ±1			±1/2 ±1			±1/2 ±1	LSB LSB
DIFFERENTIAL LINEARITY ERROR Guaranteed Monotonic for Whole Re- Missing Codes are Guaranteed T_{MIN} to T_{MAX}	11		±2 ±2	11		±1 ±1	10		±2 ±2	Bits LSB
UNIPOLAR OFFSET (Adjustable to Zero)			±4			±4			±8	LSB
BIPOLAR OFFSET (Adjustable to Zero)			±4			±4			±8	LSB
FULL-SCALE CALIBRATION ERROR (With Fixed 50 Ω Resistor from REF OUT to REF IN) (Adjustable to Zero)			±0.3%			±0.3%			±0.3%	% of FS
TEMPERATURE RANGE	-55		+125	-55		+125	-25		+85	°C
TEMPERATURE COEFFICIENTS Using Internal Reference (T_{MIN} to T_{MAX}) Unipolar Offset Bipolar Offset Full-Scale Calibration			±5 (±9) ±4 (±8) ±20 (±36)			±1 (±1.5) ±1 (±1.5) ±4 (±6.5)			±1 (±1.5) ±1 (±1.5) ±4 (±6.5)	LSB typ/max LSB typ/max LSB typ/max
POWER SUPPLY REJECTION Max Change in Full-Scale Calibration V_{CC} = +13.5 to +16.5 V or +11.4 to +12.6 V V_{EE} = -16.5 V to -13.5 V V_{LOGIC} = +4.5 V to +5.5 V or -11.4 V to -12.6 V			±1 ±1/2 ±1			±1 ±1/2 ±1			±1 ±1/2 ±1	LSB LSB LSB
ANALOG INPUT Input Ranges Unipolar Bipolar Impedance	0 -5 -10 0 0 0		+5 +10 +10 +20 +20 +20	0 -5 -10 0 0 0		+5 +10 +10 +20 +20 +20	0 -5 -10 0 0 0		+5 +10 +10 +20 +20 +20	V V V V V V kΩ
Input Impedance 10 V/20 V Span 5 V/10 V Span	3 1 2	5 2.5 5	7 3.5 14	3 1 2	5 2.5 5	7 3.5 14	3 1 2	5 2.5 5	7 3.5 14	kΩ kΩ kΩ
DIGITAL CHARACTERISTICS (T_{MIN} to T_{MAX}) Inputs (CE, $\overline{CS}$, R/$\overline{C}$, A0) Logic "1" Voltage Logic "0" Voltage Current Capacitance Outputs (DB11-DB0, STS) Logic "1" Voltage (Source, I = 500 µA) Logic "0" Voltage (Sink, I = 1.6 mA) Leakage (DB11-DB0, High-Z State) Capacitance	+2.0 -0.5 -30 +2.4 -30		+5.5 +0.8 +30 5 +0.4 +30 5	+2.0 -0.5 -30 +2.4 -30		+5.5 +0.8 +30 5 +0.4 +30 5	+2.0 -0.5 -30 +2.4 -30		+5.5 +0.8 +30 5 +0.4 +30 5	V V µA pF V V µA pF
POWER SUPPLIES Operating Range V_{CC} V_{EE} V_{LOGIC} Operating Current I_{CC} I_{EE} I_{LOGIC}	+4.5 +13.5 -11.4		+5.5 +16.5 -16.5 30 1 35	+4.5 +13.5 -11.4		+5.5 +16.5 -16.5 30 1 35	+4.5 +13.5 -11.4		+5.5 +16.5 -16.5 30 1 35	V V V mA mA mA
POWER DISSIPATION		390	725		390	775		390	775	mW
INTERNAL REFERENCE VOLTAGE Source Current Available for External Loads Allowed Load Shunt on Change During Conversion	9.98	10.0	10.02 1.5	9.98	10.0	10.02 1.5	9.95	10.0	10.05 1.5	V mA
PACKAGE OPTION Ceramic (D-28)	AD574AJD			AD574ASD			AD574AUD			

NOTES
Electrical Timing Specifications appear at the Timing Section.
Logic inputs are TTL-compatible and must be held either at V_{LOGIC} or Digital Common.
The reference should be buffered for operation on ±10 V supplies.
With Contrast DIP.
Specifications subject to change without notice.



AD574A Block Diagram and Pin Configuration

ABSOLUTE MAXIMUM RATINGS[1]
(Specifications apply to all grades, except where noted)

V_{CC} to Digital Common	0 V to +16.5 V
V_{EE} to Digital Common	0 V to -16.5 V
V_{LOGIC} to Digital Common	0 V to +7 V
Analog Common to Digital Common	±1 V
Control Inputs (CE, $\overline{CS}$, A_0 12/$\overline{8}$, R/$\overline{C}$) to Digital Common	-0.5 V to V_{LOGIC} + 0.5 V
Analog Inputs (REF IN, BIP OFF, 10 V_{IN}) to Analog Common	V_{EE} to V_{CC}
20 V_{IN} to Analog Common	±24 V
REF OUT	Indefinite Short to Common Momentary Short to V_{CC}
Chip Temperature	175°C
Power Dissipation	825 mW
Lead Temperature (Soldering, 10 sec)	+300°C
Storage Temperature (Ceramic)	-65°C to +150°C
(Plastic)	-25°C to +100°C

[1]Stresses above 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 sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

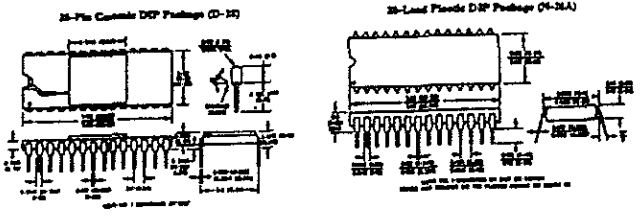
ORDERING GUIDE

Model[1]	Temperature Range	Linearity Error Max (T_{MIN} to T_{MAX})	Resolution No Missing Codes (T_{MIN} to T_{MAX})	Max Full Scale T.C. (ppm/°C)
AD574AJ(X)	0°C to +70°C	±1 LSB	11 Bits	50.0
AD574AK(X)	0°C to +70°C	±1/2 LSB	12 Bits	10.0
AD574AL(X)	0°C to +70°C	±1/2 LSB	12 Bits	50.0
AD574AS(X)[2]	-55°C to +125°C	±1 LSB	11 Bits	25.0
AD574AT(X)[2]	-55°C to +125°C	±1 LSB	12 Bits	25.0
AD574AU(X)[2]	-55°C to +125°C	±1 LSB	12 Bits	12.5

NOTES
[1]X = Package designator. Available package are: D (D-28) for all grades. E (E-28A) for J and K grades and /883B processed S, T and U grades. N (N-28) for J, K, and L grades. P (P-28A) for PLCC in J, K grades. Based in AD574AJD is K grade in plastic DIP.
[2]For details on grade and package offerings screened in accordance with MIL-STD-883, refer to Analog Devices Military Products Databook.

AD574A

OUTLINE DIMENSIONS



MAXIM

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220–MAX249

General Description

he MAX220–MAX249 family of line drivers/receivers is tended for all EIA/TIA-232E and V.28/V.24 communica- ons interfaces, particularly applications where ±12V is t available.

hese parts are especially useful in battery-powered sys- ems since their low-power shutdown mode reduces ower dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use external components and are recommended for appli- ations where printed circuit board space is critical.

Applications

Portable Computers
Low-Power Modems
Interface Translation
Battery-Powered RS-232 Systems
Multi-Drop RS-232 Networks

Features

Superior to Bipolar

- Operates from Single +5V Power Supply (+5V and +1.2V—MAX231/MAX239)
- Low-Power Receive Mode in Shutdown (MAX223/MAX242)
- Meet All EIA/TIA-232E and V.28 Specifications
- Multiple Drivers and Receivers
- 3-State Driver and Receiver Outputs
- Open-Line Detection (MAX243)

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220CJD	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.
Contact factory for dice specifications.

Selection Table

Part	Power Supply (V)	No. of RS-232 Drivers/Rcvrs	No. of RS-232 Ext. Caps	Shutdown & Three-State (Yes/No)	SHDN & Three-State Inputs	No. Active in SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	No/No	No	0	120	Low-power shutdown
MAX222	+5	2/2	4	No/Yes	No	0	200	MAX241 and succ
MAX223 (MAX213)	+5	4/5	4	Yes/Yes	Yes	0	120	Available in SO
MAX225	+5	5/5	0	Yes/Yes	Yes	0	120	5 drivers with shutdown
MAX230 (MAX200)	+5	5/5	4	No/Yes	No	0	120	Standard +5V or battery supplies
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	4	No/No	No	0	120	same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	No/No	No	0	120	Industry standard
MAX232A (MAX202)	+5	2/2	4	No/No	No	0	200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	No/No	No	0	120	No external caps
MAX233A (MAX203)	+5	2/2	0	No/No	No	0	200	No external caps, slew rate limited
MAX234	+5	4/5	4	No/No	No	0	120	Replaces 1488
MAX235 (MAX185)	+5	5/5	0	Yes/No	No	0	120	No external caps
MAX236 (MAX186)	+5	4/5	4	Yes/No	Yes	0	120	Shutdown, three-state
MAX237 (MAX187)	+5	5/5	4	No/No	No	0	120	Complements IBM PC serial port
MAX238 (MAX188)	+5	4/5	4	No/No	No	0	120	Replaces 1488 and 1489
MAX239 (MAX189)	+5 and +7.5 to +13.2	4/5	4	No/Yes	Yes	0	120	Standard +5V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	Yes/No	Yes	0	120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	Yes/No	Yes	0	120	Complete IBM PC serial port
MAX242	+5	2/2	4	Yes/Yes	Yes	5	120	Separate shutdown and enable
MAX243	+5	2/2	4	No/No	No	0	120	Open-line detection simplifies testing
MAX244	+5	8/10	4	No/No	No	0	120	High slew rate
MAX245	+5	8/10	4	No/No	No	0	120	High slew rate, int. caps, new shutdown modes
MAX246	+5	8/10	4	Yes/No	Yes	0	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	4	Yes/No	Yes	0	120	High slew rate, int. caps, new operating modes
MAX248	+5	8/8	4	Yes/Yes	Yes	0	120	High slew rate, selective half-chip enables
MAX249	+5	6/10	4	Yes/No	Yes	0	120	Available in quad flatpack package

MAX220–MAX249

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (Vcc) ..... -0.3V to +6V
Input Voltages
TIN ..... -0.3V to (Vcc + 0.3V)
RIN (Except MAX220) ..... ±30V
RIN (MAX220) ..... ±25V
TOUT (Except MAX220) (Note 1) ..... ±15V
TOUT (MAX220) ..... ±13.2V
Output Voltages
TOUT ..... ±15V
ROUT ..... -0.3V to (Vcc + 0.3V)
Driver/Receiver Output Short Circuited to GND ..... Continuous
Continuous Power Dissipation (T_A = +70°C)
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C) 440mW
18-Pin Narrow SO (derate 8.70mW/°C above +70°C) 696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) 800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C) 640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) 800mW
18-Pin CERDIP (derate 10.53mW/°C above +70°C) 842mW
Operating Temperature Ranges
MAX2_ _AC_ _ / MAX2_ _C_ _ ..... 0°C to +70°C
MAX2_ _AE_ _ / MAX2_ _E_ _ ..... -40°C to +85°C
MAX2_ _AM_ _ / MAX2_ _M_ _ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Note 1: Input voltage measured with T_{OUT} in high-impedance state, $\overline{SHDN}$ or Vcc = 0V
Note 2: For the MAX220, V+ and V- can have a maximum magnitude of 7V, but their absolute difference cannot exceed 13V.

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

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

(Vcc = +5V ±10%, C1–C4 = 0.1µF, MAX220, C1 = 0.047µF, C2–C4 = 0.33µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	All except MAX220, normal operation			5	40	µA
	SHDN = 0V, MAX222/242, shutdown, MAX220			±0.01	±1	
Output Leakage Current	Vcc = 5.5V, SHDN = 0V, VOUT = ±15V, MAX222/242			±0.01	±10	µA
	Vcc = SHDN = 0V, VOUT = ±15V			±0.01	±10	
Data Rate	All except MAX220, normal operation			200	116	kbits/ sec
	MAX220			22	20	
Transmitter Output Resistance	Vcc = V+ = V- = 0V, VOUT = ±2V		300	10M		Ω
Output Short-Circuit Current	VOUT = 0V			±7	±22	mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	Vcc = 5V	All except MAX243 $R2_{IN}$	0.8	1.3		V
		MAX243 $R2_{IN}$ (Note 2)	-3			
RS-232 Input Threshold High	Vcc = 5V	All except MAX243 $R2_{IN}$		1.8	2.4	V
		MAX243 $R2_{IN}$ (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	All except MAX243, Vcc = 5V, no hysteresis in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	IOUT = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	IOUT = -1.0mA		3.5	Vcc - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing VOUT = GND		-2	-10		mA
	Sinking VOUT = Vcc		10	30		

MAXIM

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220–MAX249

General Description

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where ±12V is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers
Low Power Modems
Interface Translation
Battery-Powered RS-232 Systems
Multi-Drop RS-232 Networks

Features

Superior to Bipolar

- Operates from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)
- Low-Power Receive Mode in Shutdown (MAX223/MAX242)
- Meet All EIA/TIA-232E and V.28 Specifications
- Multiple Drivers and Receivers
- 3-State Driver and Receiver Outputs
- Open-Line Detection (MAX243)

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220CJD	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering information continued at end of data sheet.
Contact factory for dice specifications.

Selection Table

	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Standard Cap Values (µF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Pin/Package
	+5	2/2	4	0.1/1.0	No	—	120	Ultra-low-power, industry-standard pinout
232	+5	2/2	4	0.1	Yes	—	200	Low-power shutdown
MAX233 (MAX213)	+5	5/5	4	1.0/1.0	Yes	✓	120	MAX241 and receivers active in shutdown
MAX235	+5	5/5	0	—	Yes	—	120	Available in SO
MAX236 (MAX206)	+5	5/5	4	1.0/1.0	Yes	—	120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0/1.0	No	—	120	Standard +5V- to +12V- or battery-supplied dual RS-232 transmitters/receivers
MAX232 (MAX205)	+5	2/2	4	1.0/1.0	No	—	120 [64]	Industry standard
MAX232A	+5	2/2	4	0.1	No	—	200	Higher data rates, small caps
MAX223 (MAX225)	+5	4/3	4	—	Yes	—	120	No external caps
MAX224	+5	4/3	4	—	Yes	—	120	No external caps, high slew rate
MAX230 (MAX200)	+5	5/3	4	1.0/1.0	No	—	120	Replaces 1488
MAX234 (MAX232)	+5	2/2	4	—	No	—	120	No external caps
MAX242 (MAX202)	+5	2/2	4	1.0/1.0	Yes	—	120	Shutdown, three state
MAX243 (MAX211)	+5	4/5	4	1.0/1.0	Yes	—	120	Complements IBM PC serial port
MAX244 (MAX208)	+5	4/4	4	1.0/1.0	No	—	120	Replaces 1488 and 1489
MAX245 (MAX213)	+5 and +7.5 to +13.2	4/5	8	1.0/1.0	No	—	120	Standard +5V- to +12V- or battery-supplied single-package solution for IBM PC serial port
MAX246	+5	4/5	4	1.0	Yes	—	120	DIP or flatpack package
MAX241 (MAX213)	+5	4/5	4	1.0/1.0	Yes	—	120	Complete IBM PC serial port
MAX222	+5	2/2	4	0.1	Yes	✓	200	Separate shutdown and enable
MAX223	+5	3/5	4	0.1	Yes	✓	200	Open-line detection simplifies cabling
MAX225	+5	5/5	0	1.0	Yes	—	120	High slew rate
MAX226	+5	8/10	0	—	Yes	✓	120	High slew rate, int. caps, two shutdown modes
MAX227	+5	8/10	0	—	Yes	✓	120	High slew rate, int. caps, three shutdown modes
MAX228	+5	6/10	0	—	Yes	✓	120	High slew rate, int. caps, new operating modes
MAX229	+5	4/5	0	—	Yes	✓	120	High slew rate, selective half-chip enables
MAX232	+5	2/2	4	1.0	No	—	120	Available in quad flatpack package
MAX249	+5	6/10	4	1.0	Yes	✓	120	

MAX220–MAX249

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (VCC) ..... -0.3V to +6V
Input Voltages
TIN ..... -0.3V to (VCC + 0.3V)
RIN (Except MAX220) ..... ±30V
RIN (MAX220) ..... ±25V
TOUT (Except MAX220) (Note 1) ..... ±15V
TOUT (MAX220) ..... ±13.2V
Output Voltages
TOUT ..... ±15V
ROUT ..... -0.3V to (VCC + 0.3V)
Driver/Receiver Output Short Circuited to GND ..... Continuous
Continuous Power Dissipation (TA = +70°C)
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C) 440mW
16-Pin Narrow SO (derate 8.70mW/°C above +70°C) 696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) 800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C) 640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) 800mW
18-Pin CERDIP (derate 10.53mW/°C above +70°C) 842mW
Operating Temperature Ranges
MAX2_ _AC_ _, MAX2_ _C_ _ ..... 0°C to +70°C
MAX2_ _AE_ _, MAX2_ _E_ _ ..... -40°C to +85°C
MAX2_ _AM_ _, MAX2_ _M_ _ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Note 1: Input voltage measured with TOUT in high-impedance state, SHDN or VCC = 0V.
Note 2: For the MAX220, V+ and V- can have a maximum magnitude of 7V, but their absolute difference cannot exceed 13V.

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

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

(VCC = +5V ±10%, C1–C4 = 0.1µF, MAX220, C1 = 0.047µF, C2–C4 = 0.33µF, TA = TMIN to TMAX, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	All except MAX220, normal operation			5	40	µA
	SHDN = 0V, MAX222/242, shutdown, MAX220			±0.01	±1	
Output Leakage Current	VCC = 5.5V, SHDN = 0V, VOUT = ±15V, MAX222/242			±0.01	±10	µA
	VCC = SHDN = 0V, VOUT = ±15V			±0.01	±10	
Data Rate	All except MAX220, normal operation			200	116	kbits/ sec
	MAX220			22	20	
Transmitter Output Resistance	VCC = V+ = V- = 0V, VOUT = ±2V		300	10M		Ω
Output Short-Circuit Current	VOUT = 0V			±7	±22	mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	VCC = 5V	All except MAX243 R2IN	0.8	1.3		V
		MAX243 R2IN (Note 2)	-3			
RS-232 Input Threshold High	VCC = 5V	All except MAX243 R2IN		1.8	2.4	V
		MAX243 R2IN (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	All except MAX243, VCC = 5V, no hysteresis in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	IOUT = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	IOUT = -1.0mA		3.5	VCC - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing VOUT = GND		-2	-10		mA
	Sinking VOUT = VCC		10	30		

MOTOROLA
SEMICONDUCTOR TECHNICAL DATA

Analog Multiplexers/Demultiplexers

MC14051B
MC14052B
MC14053B


L SUFFIX
CERAMIC
CASE 620


P SUFFIX
PLASTIC
CASE 648


D SUFFIX
SOIC
CASE 751B

ORDERING INFORMATION

MC14XXXBCP	Plastic
MC14XXXBCL	Ceramic
MC14XXXBD	SOIC

T_A = – 55° to 125°C for all packages

The MC14051B, MC14052B, and MC14053B analog multiplexers are digitally–controlled analog switches. The MC14051B effectively implements an SPST solid state switch, the MC14052B a DPST, and the MC14053B a Triple SPDT. All three devices feature low ON impedance and very low OFF leakage current. Control of analog signals up to the complete supply voltage range can be achieved.

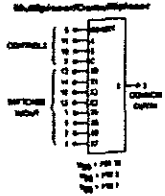
- Triple Diode Protection on Control Inputs
- Switch Function is Break Before Make
- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- Analog Voltage Range ($V_{DD} - V_{EE}$) = 3.0 to 18 V
  Note: V_{EE} must be ≤ V_{SS}
- Linearized Transfer Characteristics
- Low–noise – 12 nV/√Cycle, f ≥ 1.8 kHz Typical
- Pin–for–Pin Replacement for CD4051, CD4052, and CD4053
- For SPDT Switch, See MC14551B
- For Lower R_{ON}, Use the HC4051, HC4052, or HC4053 High–Speed CMOS Devices

MAXIMUM RATINGS*

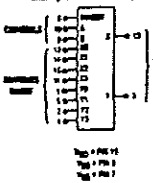
Symbol	Parameter	Value	Unit
V_{DD}	DC Supply Voltage (Referenced to V_{SS})	– 0.5 to + 18.0	V
V_{in}, V_{out}	Input or Output Voltage (DC or Transient) (Referenced to V_{SS} for Control Inputs and V_{EE} for Switch I/O)	– 0.5 to V_{DD} + 0.5	V
I_{in}	Input Current (DC or Transient), per Control Pin	± 10	mA
I_{out}	Switch Through Current	± 25	mA
P_D	Power Dissipation, per Package†	500	mW
T_{stg}	Storage Temperature	– 65 to + 150	°C
T_L	Lead Temperature (8–Second Soldering)	260	°C

* Maximum Ratings are those values beyond which damage to the device may occur.
† Temperature Derating: "P and D/DW" Packages: – 7.0 mW/°C From 65°C To 125°C
Ceramic "L" Packages: – 12 mW/°C From 100°C To 125°C

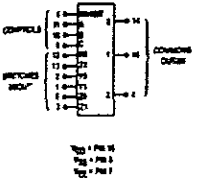
MC14051B
8–Channel Analog Multiplexer/Demultiplexer



MC14052B
Dual 4–Channel Analog Multiplexer/Demultiplexer



MC14053B
Triple 2–Channel Analog Multiplexer/Demultiplexer



Note: Control inputs referenced to V_{SS}. Analog inputs and outputs reference to V_{EE}. V_{EE} must be ≤ V_{SS}.

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 V, V_{EE} = Gnd, T_A = 25°C, unless otherwise noted.)

Characteristics	Symbol	LM324 Min	LM324 Typ	LM324 Max	LM324A Min	LM324A Typ	LM324A Max	LM224 Min	LM224 Typ	LM224 Max	LM2902 Min	LM2902 Typ	LM2902 Max	LM2902V Min	LM2902V Typ	LM2902V Max	Unit
Input Offset Voltage V_{CC} = 5.0 V to 30 V (26 V for LM2902, V), V_{ICR} = 0 V to V_{CC} – 1.7 V, V_O = 1.4 V, R_S = 0 Ω: T_A = 25°C; T_A = T_{high}[1]; T_A = T_{low}[1]	V_{IO}	– – –	2.0 – 7.0	5.0 7.0 –	– – –	2.0 – –	3.0 5.0 –	– – –	2.0 – –	7.0 9.0 9.0	– – –	2.0 – –	7.0 10 15	– – –	2.0 – –	7.0 13 15	mV
Average Temperature Coefficient of Input Offset Voltage T_A = T_{high} to T_{low}[1]	$\Delta V_{IO}/\Delta T$	–	7.0	–	–	7.0	30	–	7.0	–	–	7.0	–	–	7.0	–	μV/°C
Input Offset Current T_A = T_{high} to T_{low}[1]	I_{IO}	– –	3.0 –	30 100	– –	3.0 –	30 75	– –	3.0 –	50 150	– –	3.0 –	50 200	– –	3.0 –	50 200	nA
Average Temperature Coefficient of Input Offset Current T_A = T_{high} to T_{low}[1]	$\Delta I_{IO}/\Delta T$	–	10	–	–	10	300	–	10	–	–	10	–	–	10	–	pA/°C
Input Bias Current T_A = T_{high} to T_{low}[1]	I_{IB}	– –	–40 –	–150 –300	– –	–45 –	–100 –200	– –	–40 –	–250 –500	– –	–40 –	–250 –500	– –	–40 –	–250 –500	nA
Input Common Mode Voltage Range[2]: V_{CC} = 30 V (26 V for LM2902, V); V_{CC} = 30 V (26 V for LM2902, V), T_A = T_{high} to T_{low}	V_{ICR}	0 0	–	28.3 26	0 0	–	28.3 26	0 0	–	28.3 26	0 0	–	24.3 24	0 0	–	24.3 24	V
Differential Input Voltage Range	V_{IDR}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	V
Large Signal Open Loop Voltage Gain R_L = 2.0 kΩ, V_{CC} = 15 V, for Large V_O Swing, T_A = T_{high} to T_{low}[1]	A_{VOL}	50 25	100 –	–	25 15	100 –	–	25 15	100 –	–	25 15	100 –	–	25 15	100 –	–	V/mV
Channel Separation 10 kHz ≤ f ≤ 20 kHz, Input Referenced	CS	–	–120	–	–	–120	–	–	–120	–	–	–120	–	–	–120	–	dB
Common Mode Rejection, R_S ≤ 10 kΩ	CMR	70	85	–	65	70	–	65	70	–	50	70	–	50	70	–	dB
Power Supply Rejection	PSR	65	100	–	65	100	–	65	100	–	50	100	–	50	100	–	dB
Output Voltage – High Limit (T_A = T_{high} to T_{low})[1]: V_{CC} = 5.0 V, R_L = 2.0 kΩ, T_A = 25°C; V_{CC} = 30 V (26 V for LM2902, V), R_L = 2.0 kΩ; V_{CC} = 30 V (26 V for LM2902, V), R_L = 10 kΩ	V_{OH}	3.3 26 27	3.5 – 28	– – –	3.3 26 27	3.5 – 28	– – –	3.3 26 27	3.5 – 28	– – –	3.3 22 23	3.5 – 24	– – –	3.3 22 23	3.5 – 24	– – –	V

NOTES: 1. T_{low} = –25°C for LM224
= 0°C for LM324, A
= –40°C for LM2902
= –40°C for LM2902V
T_{high} = +85°C for LM224
= +70°C for LM324, A
= +105°C for LM2902
= +125°C for LM2902V

2. The input common mode voltage or either input signal voltage should not be allowed to go negative by more than 0.3 V. The upper end of the common mode voltage range is V_{CC} – 1.7 V.

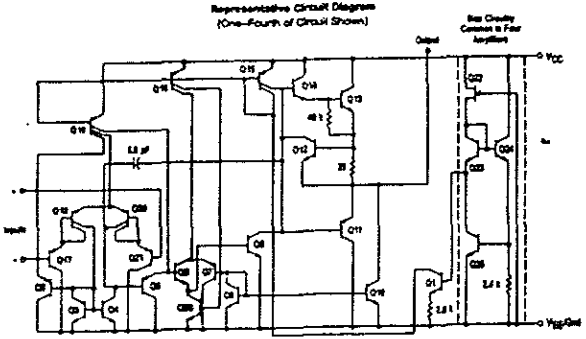
LM324, LM324A, LM224, LM2902, LM2902V

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 V, V_{EE} = Gnd, T_A = 25°C, unless otherwise noted.)

Characteristic	Symbol	LM224 Min	LM224 Typ	LM224 Max	LM324A Min	LM324A Typ	LM324A Max	LM324 Min	LM324 Typ	LM324 Max	LM2902 Min	LM2902 Typ	LM2902 Max	LM2902V Min	LM2902V Typ	LM2902V Max	Unit
Output Voltage – Low Level, V_{CC} = 5.0 V, R_L = 10 kΩ, T_A = T_{high} to T_{low}[1]	V_{OL}	–	5.0	20	–	5.0	20	–	5.0	20	–	5.0	100	–	5.0	100	mV
Output Source Current V_{in+} = +1.0 V, V_{in-} = 0 V, V_O = 2.0 V, T_A = 25°C T_A = T_{high} to T_{low}[1]	I_O +	20 10	40 30	–	20 10	40 30	–	20 10	40 30	–	20 10	40 30	–	20 10	40 30	–	mA
Output Sink Current V_{in-} = +1.0 V, V_{in+} = 0 V, V_O = 2.0 V, T_A = 25°C T_A = T_{high} to T_{low}[1] V_{in-} = +1.0 V, V_{in+} = 0 V, V_O = 200 mV, T_A = 25°C	I_O –	10 5.0 12	20 8.0 50	–	10 5.0 12	20 8.0 50	–	10 5.0 12	20 8.0 50	–	10 5.0 12	20 8.0 50	–	10 5.0 12	20 8.0 50	–	mA µA
Output Short Circuit to Ground[2]	I_{SC}	–	40	60	–	40	60	–	40	60	–	40	60	–	40	60	mA
Power Supply Current T_A = T_{high} to T_{low}, V_{CC} = 30 V (for LM2902, V), V_O = 0 V, R_L = ∞ V_{CC} = 5.0 V, R_L = ∞	I_{CC}	– –	– –	3.0 1.2	– –	1.4 0.7	3.0 1.2	– –	– –	3.0 1.2	– –	– –	3.0 1.2	– –	– –	3.0 1.2	mA

NOTES: 1. T_{low} = −40°C for LM224 / 0°C for LM324, A / −40°C for LM2902 / −40°C for LM2902V
T_{high} = +85°C for LM224 / +70°C for LM324, A / +105°C for LM2902 / +125°C for LM2902V
2. The input common mode voltage or differential input voltage should not be allowed to go negative by more than 0.3 V. The upper end of the common mode voltage range is V_{CC} −1.5 V.

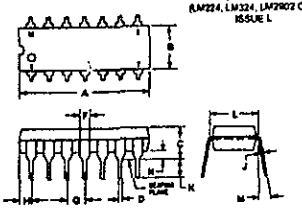
Representative Circuit Diagram
(One-Fourth of Circuit Shown)



LM324, LM324A, LM224, LM2902, LM2902V

OUTLINE DIMENSIONS

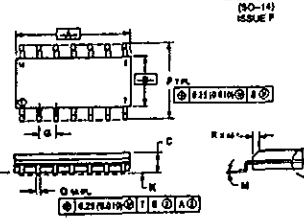
N SUFFIX
PLASTIC PACKAGE
CASE 646–06
(LM224, LM324, LM2902 Only)
ISSUE L



NOTES:
1. LEADS WITHIN 0.13 (0.005) RADIUS OF TRUE POSITION AT SEATING PLANE AT MAXIMUM MATERIAL CONDITION.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
4. ROUNDED CORNERS OPTIONAL.

DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	0.715	0.770	18.16	19.56
B	0.240	0.260	6.10	6.60
C	0.145	0.185	3.69	4.69
D	0.015	0.021	0.38	0.53
F	0.040	0.070	1.02	1.78
G	0.100 BSC		2.54 BSC	
H	0.052	0.095	1.32	2.41
J	0.008	0.015	0.20	0.38
K	0.115	0.135	2.92	3.43
L	0.300 BSC		7.62 BSC	
M	0°	10°	0°	10°
N	0.015	0.039	0.38	1.01

D SUFFIX
PLASTIC PACKAGE
CASE 751A–03
(SO–14)
ISSUE F



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSION A AND B DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.
5. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.127 (0.005) TOTAL IN EXCESS OF THE D DIMENSION AT MAXIMUM MATERIAL CONDITION.

DIM	MILLIMETERS MIN	MILLIMETERS MAX	INCHES MIN	INCHES MAX
A	8.55	8.75	0.337	0.344
B	3.80	4.00	0.150	0.157
C	1.35	1.75	0.054	0.068
D	0.35	0.49	0.014	0.019
F	0.40	1.25	0.016	0.049
G	1.27 BSC		0.050 BSC	
J	0.18	0.25	0.007	0.009
K	0.10	0.25	0.004	0.009
M	0°	7°	0°	7°
P	5.80	6.20	0.228	0.244
R	0.25	0.50	0.010	0.019

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution;
P.O. Box 20912; Phoenix, Arizona 85036. 1-800-441-2447 or 602-303-5454

EMAIL: RMFAX0@email.sps.mot.com – TOUCHTONE 602-244-6609
INTERNET: http://Design-NET.com

JAPAN: Nippon Motorola Ltd., Tatsumi-SPD-JL DC, 6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan. 03-81-3521-8315

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd., 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298

February 2000

National Semiconductor

LM555
Timer

General Description

The LM555 is a highly stable device for generating accurate time delays or oscillation. Additional terminals are provided for triggering or resetting if desired. In the time delay mode of operation, the time is precisely controlled by one external resistor and capacitor. For astable operation as an oscillator, the free running frequency and duty cycle are accurately controlled with two external resistors and one capacitor. The circuit may be triggered and reset on falling waveforms, and the output circuit can source or sink up to 200mA or drive TTL circuits.

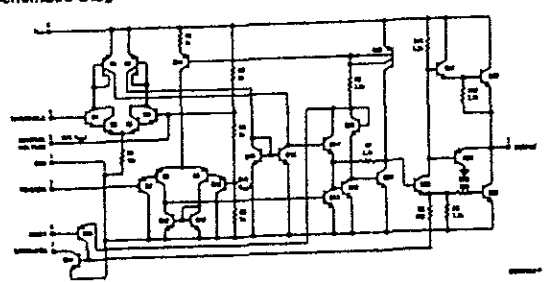
Features

- Direct replacement for SE555/NE555
- Timing from microseconds through hours
- Operates in both astable and monostable modes
- Adjustable duty cycle
- Output can source or sink 200 mA
- Output and supply TTL compatible
- Temperature stability better than 0.005% per °C
- Normally on and normally off output
- Available in 8-pin MSOP package

Applications

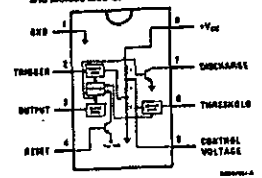
- Precision timing
- Pulse generation
- Sequential timing
- Time delay generation
- Pulse width modulation
- Pulse position modulation
- Linear ramp generator

Schematic Diagram



Connection Diagram

Dual-In-Line, Small Outline and Molded Mini Small Outline Packages



Top View

Ordering Information

Package	Part Number	Package Marking	Media Transport	NSC Drawing
8-Pin SOIC	LM555CM	LM555CM	Rails	M08A
	LM555CMX	LM555CM	2.5k Units Tape and Reel	
8-Pin MSOP	LM555CMM	Z55	1k Units Tape and Reel	MUA08A
	LM555CMMX	Z55	3.5k Units Tape and Reel	
8-Pin MDIP	LM555CN	LM555CN	Rails	N08E

Absolute Maximum Ratings (Note 3)

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

Supply Voltage	+18V
Power Dissipation (Note 3)	
LM555CM, LM555CN	1180 mW
LM555CMM	613 mW
Operating Temperature Ranges	
LM555C	0°C to +70°C
Storage Temperature Range	−65°C to +150°C

Soldering Information

Dual-In-Line Package	
Soldering (10 Seconds)	260°C
Small Outline Packages	
(SOIC and MSOP)	
Vapor Phase (60 Seconds)	215°C
Infrared (15 Seconds)	220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

Electrical Characteristics (Notes 1, 2)

(T_A = 25°C, V_{CC} = +5V to +15V, unless otherwise specified)

Parameter	Conditions	Limits LM555C Min	Typ	Max	Units
Supply Voltage		4.5		16	V
Supply Current	V_{CC} = 5V, R_L = ∞		3	6	mA
	V_{CC} = 15V, R_L = ∞ (Low State) (Note 4)		10	15	
Timing Error, Monostable					
Initial Accuracy			1		%
Drift with Temperature	R_A = R_B = 1k to 100kΩ, C = 0.1µF, (Note 5)		50		ppm/°C
Accuracy over Temperature			1.5		%
Drift with Supply			0.1		%/V
Timing Error, Astable					
Initial Accuracy			2.25		%
Drift with Temperature	R_A, R_B = 1k to 100kΩ, C = 0.1µF, (Note 5)		150		ppm/°C
Accuracy over Temperature			3.0		%
Drift with Supply			0.30		%/V
Threshold Voltage			0.667		x V_{CC}
Trigger Voltage	V_{CC} = 15V		5		V
	V_{CC} = 5V		1.67		V
Trigger Current			0.5	0.9	µA
Reset Voltage		0.4	0.5	1	V
Reset Current			0.1	0.4	mA
Threshold Current	(Note 6)		0.1	0.25	µA
Control Voltage Level	V_{CC} = 15V	9	10	11	V
	V_{CC} = 5V	2.6	3.33	4	
Pin 7 Leakage Output High			1	100	nA
Pin 7 Sat (Note 7)					
Output Low	V_{CC} = 15V, I_7 = 15mA		180		mV
Output Low	V_{CC} = 4.5V, I_7 = 4.5mA		80	200	mV

Electrical Characteristics (Notes 1, 2) (Continued)

(T_A = 25°C, V_{CC} = +5V to +15V, unless otherwise specified)

Parameter	Conditions	Limits LM555C Min	Typ	Max	Units
Output Voltage Drop (Low)	V_{CC} = 15V				
	I_{SINK} = 10mA		0.1	0.25	V
	I_{SINK} = 50mA		0.4	0.75	V
	I_{SINK} = 100mA		2	2.5	V
	I_{SINK} = 200mA		2.5		V
	V_{CC} = 5V				
	I_{SINK} = 8mA				V
	I_{SINK} = 5mA		0.25	0.35	V
Output Voltage Drop (High)	I_{SOURCE} = 200mA, V_{CC} = 15V		12.5		V
	I_{SOURCE} = 100mA, V_{CC} = 15V	12.75	13.3		V
	V_{CC} = 5V	2.75	3.3		V
Rise Time of Output			100		ns
Fall Time of Output			100		ns

Note 1: All voltages are measured with respect to the ground pin, unless otherwise specified.

Note 2: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. Operating Ratings indicate conditions for which the device is intended to be functional, but do not guarantee specific performance limits. Electrical Characteristics state DC and AC electrical specifications under particular test conditions which guarantee specific performance limits. This assumes that the device is within the Operating Ratings. Specifications are not guaranteed for parameters where no limit is given, however, the typical value is a good indication of device performance.

Note 3: For operating at elevated temperatures the device must be derated above 25°C based on a +150°C maximum junction temperature and a thermal resistance of 106°C/W (DIP), 178°C/W (SO-8), and 204°C/W (MSOP) junction to ambient.

Note 4: Supply current when output high typically 1 mA less at V_{CC} = 5V.

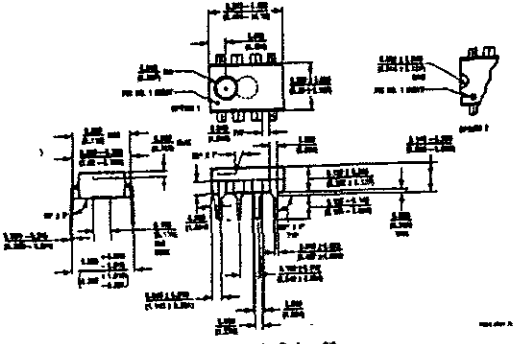
Note 5: Tested at V_{CC} = 5V and V_{CC} = 15V.

Note 6: This will determine the maximum value of R_A + R_B for 15V operation. The maximum total (R_A + R_B) is 20MΩ.

Note 7: No protection against excessive pin 7 current is necessary providing the package dissipation rating will not be exceeded.

Note 8: Refer to RETS555X drawing of military LM555H and LM555J versions for specifications.

Physical Dimensions inches (millimeters) unless otherwise noted (Continued)



Molded Dual-In-Line Package (M)
NS Package Number M08E

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

MOTOROLA

Order this document by LM324/D

LM324, LM324A, LM224, LM2902, LM2902V

QUAD DIFFERENTIAL INPUT OPERATIONAL AMPLIFIERS

SEMICONDUCTOR TECHNICAL DATA

Quad Low Power Operational Amplifiers

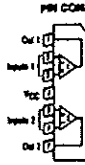
The LM324 series are low-cost, quad operational amplifiers with true differential inputs. They have several distinct advantages over standard operational amplifier types in single supply applications. The quad amplifier can operate at supply voltages as low as 3.0 V or as high as 32 V with quiescent currents about one-fifth of those associated with the MC1741 (on a per amplifier basis). The common mode input range includes the negative supply, thereby eliminating the necessity for external biasing components in many applications. The output voltage range also includes the negative power supply voltage.

- Short Circuited Protected Outputs
- True Differential Input Stage
- Single Supply Operation: 3.0 V to 32 V
- Low Input Bias Currents: 100 nA Maximum (LM324A)
- Four Amplifiers Per Package
- Internally Compensated
- Common Mode Range Extends to Negative Supply
- Industry Standard Pinouts
- ESD Clamps on the Inputs Increase Ruggedness without Affecting Device Operation

N SUFFIX
PLASTIC PACKAGE
CASE 646
(LM224, LM324, LM2902 Only)

D SUFFIX
PLASTIC PACKAGE
CASE 751A
(SO-14)

PIN CONNECTIONS



(Top View)

ORDERING INFORMATION

Device	Operating Temperature Range	Package
LM2902D	T_A = −40° to +105°C	SO-14
LM2902N		Plastic DIP
LM2902V-D	T_A = −40° to +125°C	SO-14
LM2902V-N		Plastic DIP
LM224D	T_A = −25° to +85°C	SO-14
LM224N		Plastic DIP
LM324AD	T_A = 0° to +70°C	SO-14
LM324AN		Plastic DIP
LM324D		SO-14
LM324N		Plastic DIP

MAXIMUM RATINGS (T_A = +25°C, unless otherwise noted.)

Rating	Symbol	LM324, LM324A, LM224	LM2902, LM2902V	Unit
Power Supply Voltages Single Supply Split Supplies	V_{CC} V_{CC}, V_{EE}	32 ±16	26 ±13	Vdc
Input Differential Voltage Range (See Note 1)	V_{IDR}	±32	±26	Vdc
Input Common Mode Voltage Range	V_{ICR}	−0.3 to 32	−0.3 to 26	Vdc
Output Short Circuit Duration	t_{SC}	Continuous		
Junction Temperature	T_J	150		°C
Storage Temperature Range	T_{stg}	−65 to +150		°C
Operating Ambient Temperature Range	T_A	−25 to +85 0 to +70	−40 to +105 −40 to +125	°C

NOTE: 1. Split Power Supplies.

ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	V_{DD}	Test Conditions	−55°C Min	−55°C Max	25°C Min	25°C Typ	25°C Max	125°C Min	125°C Max	Unit
SUPPLY REQUIREMENTS (Voltages Referenced to V_{EE})											
Power Supply Voltage Range	V_{DD}	—	V_{DD} = 3.0 x V_{SS} ≥ V_{EE}	3.0	18	3.0	—	18	3.0	18	V
Quiescent Current Per Package	I_{DD}	5.0 10 15	Control Inputs: V_{IN} = V_{SS} or V_{DD}, Switch I/O: V_{EE} ≤ $V_{I/O}$ ≤ V_{DD}, and ΔV_{switch} ≤ 500 mV**	— — —	5.0 10 20	— — —	0.005 0.010 0.015	5.0 10 20	— — —	150 300 600	µA
Total Supply Current (Dynamic Plus Quiescent, Per Package	$I_{D(AV)}$	5.0 10 15	T_A = 25°C only (The channel component, V_{IN} = V_{out})/R_{on}, is not included.)	Typical			(0.07 µA/kHz) f + I_{DD} (0.20 µA/kHz) f + I_{DD} (0.36 µA/kHz) f + I_{DD}				µA
CONTROL INPUTS — INHIBIT, A, B, C (Voltages Referenced to V_{SS})											
Low-Level Input Voltage	V_{IL}	5.0 10 15	R_{on} = per spec, I_{off} = per spec	— — —	1.5 3.0 4.0	— — —	2.25 4.50 6.75	1.5 3.0 4.0	— — —	1.5 3.0 4.0	V
High-Level Input Voltage	V_{IH}	5.0 10 15	R_{on} = per spec, I_{off} = per spec	3.5 7.0 11	— — —	3.5 7.0 11	2.75 5.50 8.25	— — —	3.5 7.0 11	— — —	V
Input Leakage Current	I_{IN}	15	V_{IN} = 0 or V_{DD}	—	±0.1	—	±0.00001	±0.1	—	1.0	µA
Input Capacitance	C_{IN}	—		—	—	—	5.0	7.5	—	—	pF
SWITCHES INOUT AND COMMONS OUT/IN — X, Y, Z (Voltages Referenced to V_{EE})											
Recommended Peak-to-Peak Voltage Into or Out of the Switch	$V_{I/O}$	—	Channel On or Off	0	V_{DD}	0	—	V_{DD}	0	V_{DD}	Vpp
Recommended Static or Dynamic Voltage Across the Switch** (Figure 8)	ΔV_{switch}	—	Channel On	0	600	0	—	600	0	300	mV
Output Offset Voltage	V_{OO}	—	V_{IN} = 0 V, No Load	—	—	—	10	—	—	—	µV
ON Resistance	R_{on}	5.0 10 15	ΔV_{switch} ≤ 500 mV**, V_{IN} = V_{IL} or V_{IH} (Control), and V_{IN} = 0 to V_{DD} (Switch)	— — —	800 400 220	— — —	260 120 80	1050 500 280	— — —	1200 520 300	Ω
ΔON Resistance Between Any Two Channels in the Same Package	ΔR_{on}	5.0 10 15		— — —	70 50 45	— — —	25 10 10	70 50 45	— — —	135 95 65	Ω
Off-Channel Leakage Current (Figure 10)	I_{off}	15	V_{IN} = V_{IL} or V_{IH} (Control) Channel to Channel or Any One Channel	—	±100	—	±0.05	±100	—	±1000	nA
Capacitance, Switch I/O	$C_{I/O}$	—	Inhibit = V_{DD}	—	—	—	10	—	—	—	pF
Capacitance, Common O/I	C_{OI}	—	Inhibit = V_{DD} (MC14051B) (MC14052B) (MC14053B)	— — —	— — —	— — —	80 33 17	— — —	— — —	— — —	pF
Capacitance, Feedthrough (Channel Off)	C_{IO}	— —	Pins Not Adjacent Pins Adjacent	— —	— —	— —	0.15 0.47	— —	— —	— —	pF

#Data labeled "Typ" is not to be used for design purposes, but is intended as an indication of the IC's potential performance.
* For voltage drops across the switch (ΔV_{switch}) > 600 mV (> 300 mV at high temperature), excessive V_{DD} current may be drawn, i.e. the current out of the switch may contain both V_{DD} and switch input components. The reliability of the device will be unaffected unless the Maximum Ratings are exceeded. (See first page of this data sheet.)

Rev 3
1

ELECTRICAL CHARACTERISTICS (C_L = 50 pF, T_A = 25°C) (V_{SS} = V_{SS} unless otherwise indicated)

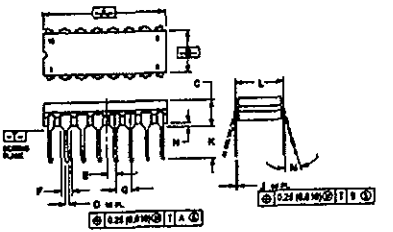
Characteristic	Symbol	V_{DD} – V_{SS} Vdc	Typ # All Types	Max	Unit
Propagation Delay Times (Figure 6)					ns
Switch Input to Switch Output (R_L = 10 kΩ)					
MC14551					
t_{PLH}, t_{PHL} = (8.17 ns/pF) C_L + 38.5 ns	t_{PLH}, t_{PHL}	5.0	35	80	
t_{PLH}, t_{PHL} = (3.65 ns/pF) C_L + 16 ns		10	15	40	
t_{PLH}, t_{PHL} = (2.65 ns/pF) C_L + 5.0 ns		15	12	30	
MC14552		5.0	75	175	ns
t_{PLH}, t_{PHL} = (8.17 ns/pF) C_L + 21.5 ns		10	32	80	
t_{PLH}, t_{PHL} = (3.65 ns/pF) C_L + 8.5 ns		15	25	60	
t_{PLH}, t_{PHL} = (2.65 ns/pF) C_L + 7.0 ns					
MC14553		5.0	75	150	ns
t_{PLH}, t_{PHL} = (8.17 ns/pF) C_L + 55.5 ns		10	30	75	
t_{PLH}, t_{PHL} = (3.65 ns/pF) C_L + 42 ns		15	23	55	
t_{PLH}, t_{PHL} = (2.65 ns/pF) C_L + 3.0 ns					
Inhibit to Output (R_L = 10 kΩ, V_{EE} = V_{SS})	t_{PHZ}, t_{PLZ}, t_{PZH}, t_{PZL}				ns
Output "1" or "0" to High Impedance or High Impedance to "1" or "0" Level					
MC14551B		5.0	300	700	
		10	170	340	
		15	130	260	
MC14052B		5.0	200	500	ns
		10	105	210	
		15	75	150	
MC14053B		5.0	275	550	ns
		10	145	290	
		15	110	225	
Control Input to Output (R_L = 10 kΩ, V_{EE} = V_{SS})	t_{PLH}, t_{PHL}				ns
MC14551B		5.0	350	720	
		10	160	330	
		15	120	240	
MC14052B		5.0	325	650	ns
		10	150	300	
		15	90	180	
MC14053B		5.0	300	600	ns
		10	135	270	
		15	90	180	
Sinewave Harmonic Distortion (R_L = 10kΩ, f = 1 kHz, V_{in} = 5 V_{pp})	—	10	0.02	—	%
Bandwidth (Figure 7) (R_L = 1 kΩ, V_{in} = ½ (V_{DD}–V_{EE}) p-p, C_L = 50pF @ Log (V_{out}/V_{in}) = –3 dB)	BW	10	17	—	MHz
Off Channel Feedthrough Attenuation (Figure 7) (R_L = 1kΩ, V_{in} = ½ (V_{DD} – V_{EE}) p-p, f_{in} = 4.5 MHz — MC14551B, f_{in} = 30 MHz — MC14052B, f_{in} = 55 MHz — MC14053B)	—	10	– 50	—	dB
Channel Separation (Figure 8) (R_L = 1 kΩ, V_{in} = ½ (V_{DD}–V_{EE}) p-p, f_{in} = 2.5 MHz)	—	10	– 50	—	dB
Crosstalk, Control Input to Common Out (Figure 9) (R_L = 1 kΩ, R_{in} = 10 kΩ, Control Input t_{TLH} = t_{THL} = 20 ns, square V_{EE} to V_{DD})	—	10	70	—	mV

* The formulas given are for the typical characteristics only at 25°C.

Data labeled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

OUTLINE DIMENSIONS

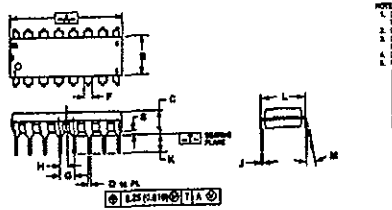
L SUFFIX
CERAMIC DIP PACKAGE
CASE 620–10
ISSUE V



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIMENSION F MAY NARROW TO 0.76 (0.030) WHERE THE LEAD ENTERS THE CERAMIC BODY.

DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	1.550	1.585	39.37	40.26
B	0.240	0.295	6.10	7.49
C	—	0.200	—	5.08
D	0.015	0.020	0.38	0.50
E	0.050 BSC		1.27 BSC	
F	0.055	0.065	1.40	1.65
G	0.100 BSC		2.54 BSC	
H	0.008	0.015	0.21	0.38
J	0.045	0.060	1.14	1.52
K	0.125	0.175	3.18	4.44
L	0.300 BSC		7.62 BSC	
M	0°	15°	0°	15°
N	0.020	0.040	0.51	1.01

P SUFFIX
PLASTIC DIP PACKAGE
CASE 648–08
ISSUE R



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
4. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
5. ROUNDED CORNERS OPTIONAL.

DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	1.740	1.770	44.20	44.96
B	0.250	0.270	6.35	6.85
C	0.145	0.175	3.69	4.44
D	0.015	0.021	0.38	0.53
F	0.040	0.70	1.02	1.77
G	0.100 BSC		2.54 BSC	
H	0.050 BSC		1.27 BSC	
J	0.008	0.015	0.21	0.38
K	0.110	0.130	2.80	3.30
L	0.300 BSC		7.62 BSC	
M	0°	10°	0°	10°
N	0.020	0.040	0.51	1.01

National Semiconductor

May 1986

LM741
Operational Amplifier

General Description

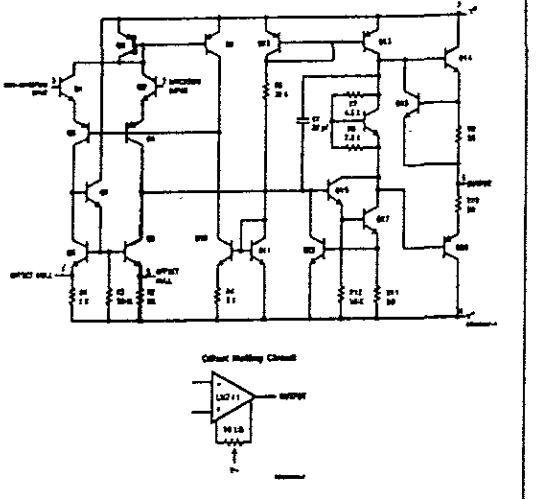
The LM741 series are general purpose operational amplifiers which feature improved performance over industry standards like the LM709. They are direct, plug-in replacements for the 709C, LM201, MC1439 and 748 in most applications.

The amplifiers offer many features which make their application nearly foolproof: overload protection on the input and output, no latch-up when the common mode range is exceeded, as well as freedom from oscillations.

The LM741C/LM741E are identical to the LM741/LM741A except that the LM741C/LM741E have their performance guaranteed over a 0°C to +70°C temperature range, instead of −55°C to +125°C.

LM741 Operational Amplifier

Schematic Diagram



www.national.com

Absolute Maximum Ratings (Note 1)

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

(Note 6)	LM741A	LM741E	LM741	LM741C
Supply Voltage	±22V	±22V	±22V	±18V
Power Dissipation (Note 2)	500 mW	500 mW	500 mW	500 mW
Differential Input Voltage	±30V	±30V	±30V	±30V
Input Voltage (Note 3)	±15V	±15V	±15V	±15V
Output Short Circuit Duration	Continuous	Continuous	Continuous	Continuous
Operating Temperature Range	−55°C to +125°C	0°C to +70°C	−55°C to +125°C	0°C to +70°C
Storage Temperature Range	−65°C to +150°C	−65°C to +150°C	−65°C to +150°C	−65°C to +150°C
Junction Temperature	150°C	100°C	150°C	100°C
Soldering Information				
H-Package (10 seconds)	260°C	260°C	260°C	260°C
J- or H-Package (10 seconds)	300°C	300°C	300°C	300°C
M-Package				
Vapor Phase (60 seconds)	215°C	215°C	215°C	215°C
Infrared (15 seconds)	215°C	215°C	215°C	215°C
See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.				
ESD Tolerance (Note 7)	400V	400V	400V	400V

Electrical Characteristics (Note 4)

Parameter	Conditions	LM741A/LM741E Min	LM741A/LM741E Typ	LM741A/LM741E Max	LM741 Min	LM741 Typ	LM741 Max	LM741C Min	LM741C Typ	LM741C Max	Units
Input Offset Voltage	$T_A = 25°C$					1.0	5.0		2.0	6.0	mV
	$R_S \le 10\ k\Omega$		0.8	3.0							mV
	$R_S \le 50\Omega$										
	$T_{AMIN} \le T_A \le T_{AMAX}$			4.0							mV
	$R_S \le 50\Omega$						6.0			7.5	mV
	$R_S \le 10\ k\Omega$										
Average Input Offset Voltage Drift				15							μV/°C
Input Offset Voltage Adjustment Range	$T_A = 25°C, V_S = \pm 20V$	±10				±15			±15		mV
Input Offset Current	$T_A = 25°C$		3.0	30		20	200		20	200	nA
	$T_{AMIN} \le T_A \le T_{AMAX}$			70		85	500			300	nA
Average Input Offset Current Drift				0.5							nA/°C
Input Bias Current	$T_A = 25°C$		30	80		80	500		80	500	nA
	$T_{AMIN} \le T_A \le T_{AMAX}$			0.210			1.5			0.8	uA
Input Resistance	$T_A = 25°C, V_S = \pm 20V$	1.0	6.0		0.3	2.0		0.3	2.0		MΩ
	$T_{AMIN} \le T_A \le T_{AMAX}$	0.5									MΩ
	$V_S = \pm 20V$										
Input Voltage Range	$T_A = 25°C$							±12	±13		V
	$T_{AMIN} \le T_A \le T_{AMAX}$				±12	±13					V

www.national.com 2

Electrical Characteristics (Note 4) (Continued)

Parameter	Conditions	LM741A/LM741E			LM741			LM741C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Large Signal Voltage Gain	$T_A = 25°C$, $R_L \geq 2\ k\Omega$ $V_S = \pm 20V$, $V_O = \pm 15V$ $V_S = \pm 15V$, $V_O = \pm 10V$	50			50	200		20	200		V/mV V/mV
	$T_{AMIN} \leq T_A \leq T_{AMAX}$ $R_L \geq 2\ k\Omega$ $V_S = \pm 20V$, $V_O = \pm 15V$ $V_S = \pm 15V$, $V_O = \pm 10V$ $V_S = \pm 5V$, $V_O = \pm 2V$	32 10			25			15			V/mV V/mV V/mV
Output Voltage Swing	$V_S = \pm 20V$ $R_L \geq 10\ k\Omega$ $R_L \geq 2\ k\Omega$	±16 ±15									V V
	$V_S = \pm 15V$ $R_L \geq 10\ k\Omega$ $R_L \geq 2\ k\Omega$				±12 ±10	±14 ±13		±12 ±10	±14 ±13		V V
Output Short Circuit Current	$T_A = 25°C$	10	25	35		25			25		mA
	$T_{AMIN} \leq T_A \leq T_{AMAX}$	10		40							mA
Common-Mode Rejection Ratio	$T_{AMIN} \leq T_A \leq T_{AMAX}$ $R_S \leq 10\ k\Omega$, $V_{CM} = \pm 12V$ $R_S \leq 50\Omega$, $V_{CM} = \pm 12V$	80	95		70	90		70	90		dB dB
Supply Voltage Rejection Ratio	$T_{AMIN} \leq T_A \leq T_{AMAX}$ $V_S = \pm 20V$ to $V_S = \pm 5V$ $R_S \leq 50\Omega$ $R_S \leq 10\ k\Omega$	86	96		77	96		77	96		dB dB
Transient Response Rise Time Overshoot	$T_A = 25°C$, Unity Gain		0.25 6.0	0.8 20		0.3 5			0.3 5		μs %
Bandwidth (Note 8)	$T_A = 25°C$	0.437	1.5								MHz
Slew Rate	$T_A = 25°C$, Unity Gain	0.3	0.7			0.5			0.5		V/μs
Supply Current	$T_A = 25°C$					1.7	2.8		1.7	2.8	mA
Power Consumption	$T_A = 25°C$ $V_S = \pm 20V$ $V_S = \pm 15V$		80	150		90	85		90	85	mW mW
LM741A	$V_S = \pm 20V$ $T_A = T_{AMIN}$ $T_A = T_{AMAX}$			165 135							mW mW
LM741E	$V_S = \pm 20V$ $T_A = T_{AMIN}$ $T_A = T_{AMAX}$			150 130							mW mW
LM741	$V_S = \pm 15V$ $T_A = T_{AMIN}$ $T_A = T_{AMAX}$					60 45	100 75				mW mW

Note 4: "Absolute Maximum Ratings" indicate limits beyond which damage to the device may occur. Operating Ratings indicate conditions for which the device is intended, but no guarantee exists for performance limits.

Electrical Characteristics (Note 4) (Continued)

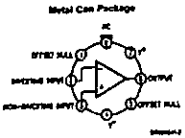
Note 3: For operation at elevated temperatures, these devices must be derated based on thermal resistance, and T_j max. (listed under "Absolute Maximum Ratings"). $T_j = T_A + \theta_{jA} P_D$.

Thermal Resistance	Cerdip (J)	DIP (N)	H08 (H)	SO-8 (M)
θ_{jA} (Junction to Ambient)	100°C/W	100 C/W	170°C/W	195°C/W
θ_{jC} (Junction to Case)	n/a	n/a	25°C/W	n/a

Note 2: For supply voltages less than ±15V, the absolute maximum input voltage is equal to the supply voltage.
Note 4: Unless otherwise specified, these specifications apply for V_S = ±15V, −55°C ≤ T_A ≤ +125°C (LM741/LM741A). For the LM741C/LM741E, these specifications are limited to 0°C ≤ T_A ≤ +70°C.
Note 5: Calculated value from: (6mV) × 0.25%/Year (Typical).
Note 6: For military specifications see RETS741X for LM741 and RETS741AX for LM741A.
Note 7: Human body model, 1.5 kΩ in series with 100 pF.

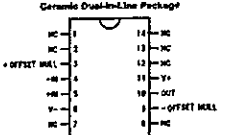
Connection Diagram

Metal Can Package



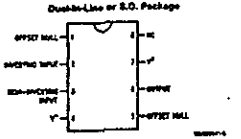
Note 9: LM741H is available per JM38510/10101
Order Number LM741H, LM741HRMQB (Note 9), LM741AH/883 or LM741CH
See NS Package Number H08C

Ceramic Dual-In-Line Package



Note 9: also available per JM38510/10101
Note 10: also available per JM38510/10102
Order Number LM741J-14/883 (Note 9), LM741AJ-14/883 (Note 10)
See NS Package Number J14A

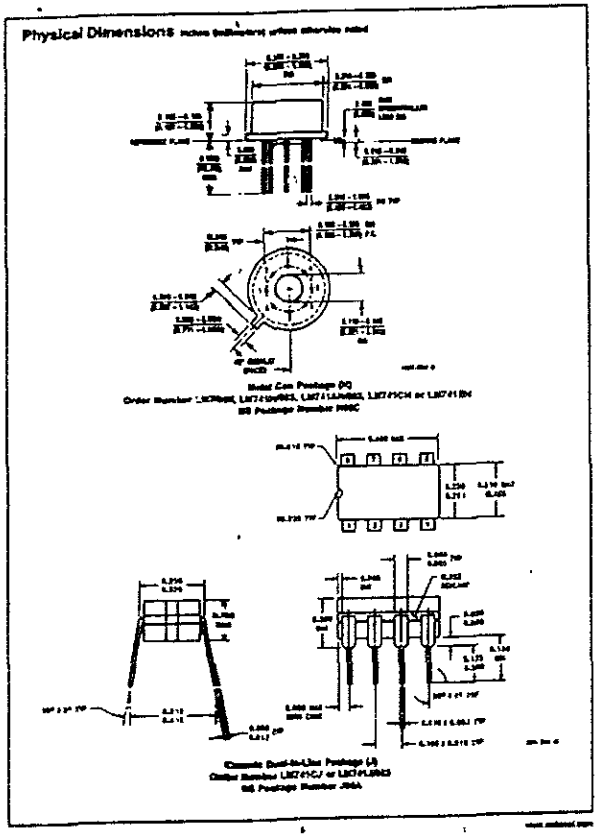
Dual-In-Line or S.O. Package



Order Number LM741J, LM741J/883, LM741CM, LM741CN or LM741EN
See NS Package Number J08A, M08A or N08E

Ceramic Flatpak

Order Number LM741W/883
See NS Package Number W10A


Physical Dimensions

Order this document by TL081/D

MOTOROLA

JFET Input Operational Amplifiers

TL081C,AC
TL082C,AC
TL084C,AC

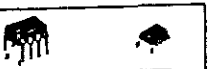
JFET INPUT
OPERATIONAL AMPLIFIERS

SEMICONDUCTOR
TECHNICAL DATA

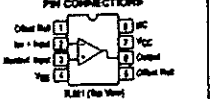
These low-cost JFET input operational amplifiers combine two state-of-the-art linear technologies on a single monolithic integrated circuit. Each internally compensated operational amplifier has well matched high voltage JFET input devices for low input offset voltage. The BIFET technology provides wide bandwidths and fast slewrates with low input bias currents, input offset currents, and supply currents.

These devices are available in single, dual and quad operational amplifiers which are pin-compatible with the industry standard MC1741, MC1458, and the MC34001/LM324 bipolar products.

- Input Offset Voltage Options of 5.0 and and 15 mV Max
- Low Input Bias Current: 30 pA
- Low Input Offset Current: 5.0 pA
- Wide Gain Bandwidth: 4.0 MHz
- High Slew Rate: 13 V/μs
- Low Supply Current: 1.4 mA per Amplifier
- High Input Impedance: $10^{12}\ \Omega$



PIN CONNECTIONS



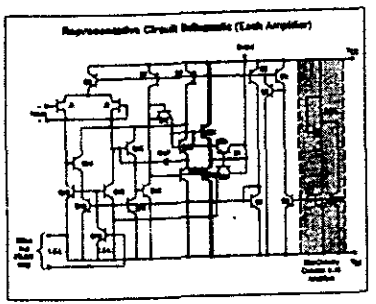
TL081 (Top View)

TL082 (Top View)

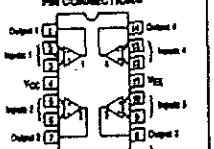
ORDERING INFORMATION

Op Amp Function	Device	Operating Temperature Range	Package
Single	TL081CD	T_A = 0° to +70°C	SO–8
	TL081ACP		Plastic DIP
Dual	TL082CD	T_A = 0° to +70°C	SO–8
	TL082ACP		Plastic DIP
Quad	TL084CN, ACN	T_A = 0° to +70°C	Plastic DIP

Representative Circuit Schematic (Each Amplifier)



PIN CONNECTIONS



TL084 (Top View)

Rev 1

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC} V_{EE}	18 −18	V
Differential Input Voltage	V_{ID}	±30	V
Input Voltage Range (Note 1)	V_{IDR}	±15	V
Output Short Circuit Duration (Note 2)	t_{SC}	Continuous	
Power Dissipation Plastic Package (N, P) Derate above T_A = +47°C	P_D $1/\theta_{JA}$	680 10	mW mW/°C
Operating Ambient Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

NOTES: 1. The magnitude of the input voltage must not exceed the magnitude of the supply voltage or 15 V, whichever is less.
2. The output may be shorted to ground or either supply. Temperature and/or supply voltages must be limited to ensure that power dissipation ratings are not exceeded.
3. ESD data available upon request.

ELECTRICAL CHARACTERISTICS (V_{CC} = 15 V, V_{EE} = −15 V, T_A = T_{low} to T_{high} (Note 1))

Characteristics	Symbol	Min	Typ	Max	Unit
Input Offset Voltage (R_S ≤ 10 k, V_{CM} = 0) TL081C, TL082C TL084C TL08_AC	V_{IO}	– – –	– – –	 20 20 7.5	mV
Input Offset Current (V_{CM} = 0) (Note 2) TL08_C TL08_AC	I_{IO}	– –	– –	5.0 3.0	nA
Input Bias Current (V_{CM} = 0) (Note 2) TL08_C TL08_AC	I_{IB}	– –	– –	10 7.0	nA
Large–Signal Voltage Gain (V_O = ±10 V, R_L ≥ 2.0 k) TL08_C TL08_AC	A_{VOL}	15 25	– –	– –	V/mV
Output Voltage Swing (Peak–to–Peak) (R_L ≥ 10 k) (R_L ≥ 2.0 k)	V_O	24 20	– –	– –	V

NOTES: 1. T_{low} = 0°C for TL081AC,C TL082AC,C TL084AC,C; T_{high} = 70°C for TL081AC TL082AC,C TL084AC,C
2. Input Bias currents of JFET input op amps approximately double for every 10°C rise in Junction Temperature as shown in Figure 3. To maintain junction temperature as close to ambient temperature as possible, pulse techniques must be used during testing.

Figure 1. Unity Gain Voltage Follower

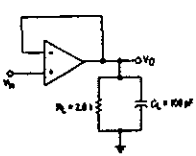
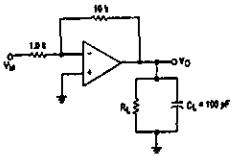


Figure 2. Inverting Gain of 10 Amplifier



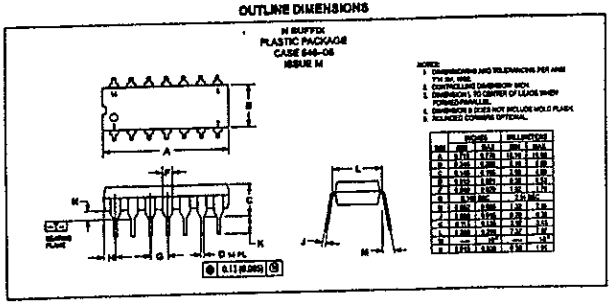
ELECTRICAL CHARACTERISTICS (V_{CC} = 15 V, V_{EE} = −15 V, T_A = 25°C, unless otherwise noted.)

Characteristics	Symbol	Min	Typ	Max	Unit
Input Offset Voltage (R_S ≤ 10 k, V_{CM} = 0) TL081C, TL082C TL084C TL08_AC	V_{IO}	– – –	5.0 5.0 2.5	15 15 6.5	mV
Average Temperature Coefficient of Input Offset Voltage R_S = 50 Ω, T_A = T_{low} to T_{high} (Note 1)	$\Delta V_{IO}/\Delta T$	–	10	–	μV/°C
Input Offset Current (V_{CM} = 0) (Note 2) TL08_C TL08_AC	I_{IO}	– –	5.0 5.0	200 100	pA
Input Bias Current (V_{CM} = 0) (Note 2) TL08_C TL08_AC	I_{IB}	– –	30 20	400 200	pA
Input Resistance	r_i	–	10^{12}	–	Ω
Common Mode Input Voltage Range TL08_C TL08_AC	V_{ICR}	±10 ±11	+15, −12 +15, −12	– –	V
Large Signal Voltage Gain (V_O = ±10 V, R_L ≥ 2.0 k) TL08_C TL08_AC	A_{VOL}	25 50	100 100	– –	V/mV
Output Voltage Swing (Peak-to-Peak) (R_L = 10 k)	V_O	24	28	–	V
Common Mode Rejection Ratio (R_S ≤ 10 k) TL08_C TL08_AC	CMRR	70 80	100 100	– –	dB
Supply Voltage Rejection Ratio (R_S ≤ 10 k) TL08_C TL08_AC	PSRR	70 80	100 100	– –	dB
Supply Current (Each Amplifier)	I_D	–	1.4	2.8	mA
Unity Gain Bandwidth	BW	–	4.0	–	MHz
Slew Rate (See Figure 1) V_{in} = 10 V, R_L = 2.0 k, C_L = 100 pF	SR	–	13	–	V/μs
Rise Time (See Figure 1)	t_r	–	0.1	–	μs
Overshoot (V_{in} = 20 mV, R_L = 2.0 k, C_L = 100 pF)	OS	–	10	–	%
Equivalent Input Noise Voltage R_S = 100 Ω, f = 1000 Hz	e_n	–	25	–	nV/√Hz
Channel Separation A_V = 100	CS	–	120	–	dB

NOTES: 1. T_{low} = 0°C for TL081AC,C T_{high} = +70°C for TL081AC,C
TL082AC,C TL082AC,C
TL084AC,C TL084AC,C

2. Input Bias currents of JFET input op amps approximately double for every 10°C rise in Junction Temperature as shown in Figure 3. To maintain junction temperature as close to ambient temperature as possible, pulse techniques must be used during testing.

OUTLINE DIMENSIONS



Order this document by LM324/D

MOTOROLA

LM324, LM324A, LM224, LM2902, LM2902V

Quad Low Power Operational Amplifiers

QUAD DIFFERENTIAL INPUT OPERATIONAL AMPLIFIERS

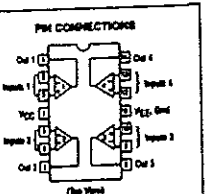
SEMICONDUCTOR TECHNICAL DATA

N SUFFIX
PLASTIC PACKAGE
CASE 646
(LM324, LM324A, LM2902 Only)

D SUFFIX
PLASTIC PACKAGE
CASE 751A
(SO–14)

The LM324 series are low–cost, quad operational amplifiers with true differential inputs. They have several distinct advantages over standard operational amplifier types in single supply applications. The quad amplifier can operate at supply voltages as low as 3.0 V or as high as 32 V with quiescent currents about one–fifth of those associated with the MC1741 (on a per amplifier basis). The common mode input range includes the negative supply, thereby eliminating the necessity for external biasing components in many applications. The output voltage range also includes the negative power supply voltage.

- Short Circuited Protected Outputs
- True Differential Input Stage
- Single Supply Operation: 3.0 V to 32 V
- Low Input Bias Currents: 100 nA Maximum (LM324A)
- Four Amplifiers Per Package
- Internally Compensated
- Common Mode Range Extends to Negative Supply
- Industry Standard Pinouts
- ESD Clamps on the Inputs Increase Ruggedness without Affecting Device Operation



MAXIMUM RATINGS (T_A = +25°C, unless otherwise noted.)

Rating	Symbol	LM324 LM324A, LM324A	LM2902, LM2902V	Unit
Power Supply Voltages Single Supply Split Supplies	V_{CC} V_{CC}, V_{EE}	32 ±16	26 ±13	Vdc
Input Differential Voltage Range (See Note 1)	V_{IDR}	±32	±26	Vdc
Input Common Mode Voltage Range	V_{ICR}	–0.3 to 32	–0.3 to 26	Vdc
Output Short Circuit Duration	tSC	Continuous		
Junction Temperature	T_J	150		°C
Storage Temperature Range	T_{stg}	–65 to +150		°C
Operating Ambient Temperature Range	T_A	–25 to +85 0 to +70	–40 to +105 –40 to +125	°C

NOTE: 1. Split Power Supplies.

ORDERING INFORMATION

Device	Operating Temperature Range	Package
LM2902D	T_A = –40° to +105°C	SO–14
LM2902N		Plastic DIP
LM2902VD	T_A = –40° to +125°C	SO–14
LM2902VN		Plastic DIP
LM224D	T_A = –25° to +85°C	SO–14
LM224N		Plastic DIP
LM324AD	T_A = 0° to +70°C	SO–14
LM324AN		Plastic DIP
LM324D		SO–14
LM324N		Plastic DIP

Rev 3

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 V, V_{EE} = Gnd, T_A = 25°C, unless otherwise noted.)

Characteristics	Symbol	LM224 Min	LM224 Typ	LM224 Max	LM324A Min	LM324A Typ	LM324A Max	LM324 Min	LM324 Typ	LM324 Max	LM2902 Min	LM2902 Typ	LM2902 Max	LM2902V Min	LM2902V Typ	LM2902V Max	Unit
Input Offset Voltage V_{CC} = 5.0 V to 30 V (26 V for LM2902, V); V_{ICR} = 0 V to V_{CC} –1.7 V, V_O = 1.4 V, R_S = 0 Ω T_A = 25°C T_A = T_{high}[1] T_A = T_{low}[1]	V_{IO}	– – –	2.0 – –	5.0 7.0 7.0	– – –	2.0 – –	3.0 5.0 5.0	– – –	2.0 – –	7.0 9.0 9.0	– – –	2.0 – –	7.0 10 10	– – –	2.0 – –	7.0 13 10	mV
Average Temperature Coefficient of Input Offset Voltage T_A = T_{low} to T_{high}[1]	$\Delta V_{IO}/\Delta T$	–	7.0	–	–	7.0	30	–	7.0	–	–	7.0	–	–	7.0	–	μV/°C
Input Offset Current T_A = T_{high} to T_{low}[1]	I_{IO}	– –	2.0 –	30 100	– –	5.0 –	30 75	– –	5.0 –	50 150	– –	5.0 –	50 200	– –	5.0 –	50 200	nA
Average Temperature Coefficient of Input Offset Current T_A = T_{high} to T_{low}[1]	$\Delta I_{IO}/\Delta T$	–	10	–	–	10	300	–	10	–	–	10	–	–	10	–	pA/°C
Input Bias Current T_A = T_{high} to T_{low}[1]	I_{IB}	– –	–45 –	–100 –300	– –	–45 –	–100 –200	– –	–45 –	–250 –500	– –	–45 –	–250 –500	– –	–45 –	–250 –500	nA
Input Common Mode Voltage Range[2] V_{CC} = 30 V (26 V for LM2902, V) V_{CC} = 30 V (26 V for LM2902, V), T_A = T_{high} to T_{low}	V_{ICR}	0 0	–	28.3 28	0 0	–	28.3 28	0 0	–	24.3 24	0 0	–	24.3 24	0 0	–	24.3 24	V
Differential Input Voltage Range	V_{IDR}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	V
Large Signal Open Loop Voltage Gain R_L = 2.0 kΩ, V_{CC} = 15 V, for Large V_O Swing, T_A = T_{high} to T_{low}[1]	A_{VOL}	50 25	100 –	–	25 15	100 –	–	25 15	100 –	–	25 15	100 –	–	25 15	100 –	–	V/mV
Channel Separation 10 kHz ≤ f ≤ 20 kHz, Input Referenced	CS	–	–120	–	–	–120	–	–	–120	–	–	–120	–	–	–120	–	dB
Common Mode Rejection, R_S ≤ 10 kΩ	CMR	70	85	–	65	70	–	65	70	–	50	70	–	50	70	–	dB
Power Supply Rejection	PSR	65	100	–	65	100	–	65	100	–	50	100	–	50	100	–	dB
Output Voltage–High Limit T_A = T_{high} to T_{low}[1] V_{CC} = 5.0 V, R_L = 2.0 kΩ, T_A = 25°C V_{CC} = 30 V (26 V for LM2902, V), R_L = 2.0 kΩ V_{CC} = 30 V (26 V for LM2902, V), R_L = 10 kΩ	V_{OH}	3.3 – 27	3.5 – 28	– – –	3.3 – 27	3.5 – 28	– – –	3.3 26 27	3.5 – 28	– – –	3.3 22 23	3.5 – 24	– – –	3.3 23 23	3.5 – 24	– – –	V

NOTES: 1. T_{low} = –25°C for LM224
= 0°C for LM324, A
= –40°C for LM2902
= –40°C for LM2902V
T_{high} = +85°C for LM224
= +70°C for LM324, A
= +105°C for LM2902
= +125°C for LM2902V

2. The input common mode voltage or either input signal voltage should not be allowed to go negative by more than 0.3 V. The upper end of the common mode voltage range is V_{CC} –1.7 V.

LM324, LM324A, LM224, LM2902, LM2902V

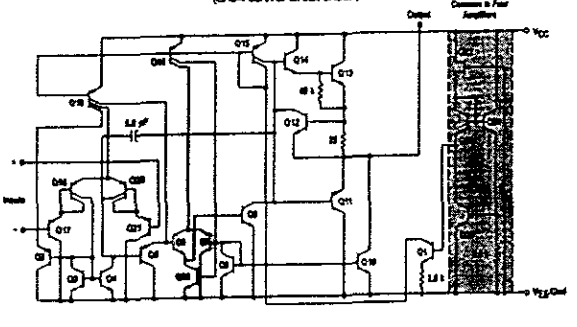
ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 V, V_{EE} = Gnd, T_A = 25°C, unless otherwise noted.)

Characteristic	Symbol	LM224			LM324A			LM324			LM2902			LM2902V			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Output Voltage – Low Limit, V_{CC} = 5.0 V, R_L = 10 kΩ, T_A = T_{high} to T_{low}[1]	V_{OL}	-	5.0	20	-	5.0	20	-	5.0	20	-	5.0	100	-	5.0	100	mV
Output Source Current (V_{in+} = 1.0 V, V_{in-} = 0 V, 10 V) T_A = 25°C; T_A = T_{high} to T_{low}[1]	I_O+	20 10	40 30	-	20 10	40 30	-	20 10	40 30	-	20 10	40 30	-	20 10	40 30	-	mA
Output Sink Current (V_{in-} = 1.0 V, V_{CC} = 10 V, T_A = 25°C; T_A = T_{high} to T_{low}[1] (V_{in+} = 1.0 V, V_O = 200 mV, T_A = 25°C)	I_O-	10 5.0 12	20 8.0 50	-	10 5.0 12	20 8.0 50	-	10 5.0 12	20 8.0 50	-	10 5.0 12	20 8.0 50	-	10 5.0 12	20 8.0 50	-	mA
Output Short Circuit to Ground[2]	I_{SC}	-	40	60	-	40	60	-	40	60	-	40	60	-	40	60	mA
Power Supply Current (T_A = T_{high} to T_{low}[1] V_{CC} = 30 V (24 V for LM2902, V); V_O = 0 V, R_L = ∞ V_{CC} = 5.0 V, V_O = 0 V, R_L = ∞	V_{CC}	-	-	3.0	-	1.4	3.0	-	-	3.0	-	-	3.0	-	-	3.0	mA
		-	-	1.2	-	0.7	1.2	-	-	1.2	-	-	1.2	-	-	1.2	

[1] T_{low} = −25°C for LM224 = 0°C for LM324, A = −40°C for LM2902 = −40°C for LM2902V
T_{high} = +85°C for LM224 = +70°C for LM324, A = +105°C for LM2902 = +105°C for LM2902V

2. The input common mode voltage or either input signal voltage should not be allowed to go negative by more than 0.3 V. The upper end of the common mode voltage range is V_{CC} − 1.5 V.

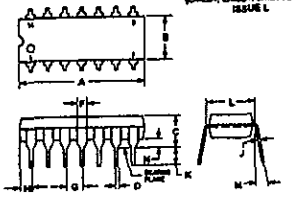
Representative Circuit Diagram
(One-Fourth of Circuit Shown)



LM324, LM324A, LM224, LM2902, LM2902V

OUTLINE DIMENSIONS

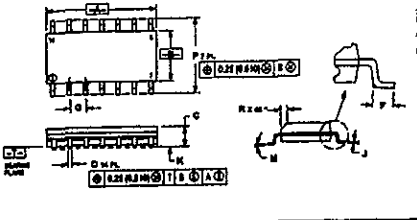
N SUFFIX
PLASTIC PACKAGE
CASE 646–06
(LM224, LM324, LM2902 Only)
ISSUE L



NOTES:
1. LEADS WITHIN 0.13 (0.005) RADIUS OF TRUE POSITION AT SEATING PLANE AT MAXIMUM MATERIAL CONDITION.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
4. ROUNDED CORNERS OPTIONAL.

DIM	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	0.715	0.770	18.16	19.56
B	0.240	0.260	6.10	6.60
C	0.145	0.185	3.69	4.69
D	0.015	0.021	0.38	0.53
F	0.040	0.070	1.02	1.78
G	0.100 BSC		2.54 BSC	
H	0.052	0.095	1.32	2.41
J	0.008	0.015	0.20	0.38
K	0.115	0.135	2.92	3.43
L	0.300 BSC		7.62 BSC	
M	0°	10°	0°	10°
N	0.015	0.035	0.39	0.89

D SUFFIX
PLASTIC PACKAGE
CASE 751A–03
(SO–14)
ISSUE F



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.
5. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.127 (0.005) TOTAL IN EXCESS OF D DIMENSION AT MAXIMUM MATERIAL CONDITION.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	8.55	8.75	0.337	0.344
B	3.80	4.00	0.150	0.157
C	1.35	1.75	0.054	0.068
D	0.35	0.49	0.014	0.019
F	0.40	1.25	0.016	0.049
G	1.27 BSC		0.050 BSC	
J	0.18	0.25	0.007	0.009
K	0.10	0.25	0.004	0.009
M	0°	7°	0°	7°
P	5.80	6.20	0.228	0.244
R	0.25	0.50	0.010	0.019

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution;
P.O. Box 20912; Phoenix, Arizona 85036. 1–800–441–2447 or 602–303–5454

MFAX: RMFAX0@email.sps.mot.com – TOUCHTONE 602–244–6609
INTERNET: http://Design-NET.com

JAPAN: Nippon Motorola Ltd.; Tatsumi-SPD-JDC, 6F Seibu-Butsuryu-Center, 3–14–2 Tatsumi Koto-Ku, Tokyo 136, Japan. 03–81–3521–8315

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd., 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

Bibliografía

Bibliografía.

- *User guide for the Magellan OEM GPS Module.*
  Magellan Sistems Corporation
  USA 1992

- *GPS satellite surveying.*
  Leick, Alfred.
  Ed. Willey.
  Nueva York 1995.

- *Manual técnico de giróscopo*
  URSS, Abril 1978

- *Manual de referencia MCU MC68HC11E1*
  México 1997

- *Microcontroller tecnology the 68HC11.*
  Spasov, Peter.
  Ed. Prentice-Hall.
  Nueva Yersey 1996.

- *Automatic Flight Control.*
  Pallett E.H.J.
  Ed. BPS Professional Books
  Gran Bretaña 1989

- *Los sistemas eléctricos en aviación.*
  Pallett E.H.J.
  Ed. Paraninfo.
  Madrid 1979.

- *Avionic systems.*
  Middlenton D. H.
  Ed. Longman Scientific & Technical.
  Gran Bretaña 1989.

- *El libro de las comunicaciones del PC.*
  José A. Carballar.
  Ed. Alfa-Omega
  México 1998.

- *C Manual de referencia.*
  Schildt, Herbert.
  Ed. Osborne McGraw Hill.
  Madrid 1990.

- *Progamming in c whit numerical methods for engineers.*
  Riojani K. B
  USA 1996

- http://www.mot-sps.com (Motorola Semiconductors).

- http://www.analogdevices.com

- http://www.national.com