ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE INGENIERÍA

DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE ADQUISICIÓN DE SEÑALES SÍSMICAS Y ALMACENAMIENTO DE DATOS DIGITALES, CON UTILIZACIÓN DE LA COMUNICACIÓN USB

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

ANDRÉS ALEJANDRO CADENA CASTILLO
JUAN CARLOS TAPIA MALDONADO

DIRECTOR: MSc. MARÍA SOLDEDAD JIMÉNEZ JIMÉNEZ

Quito, Diciembre 2006

DECLARACIÓN

Nosotros, Andrés Alejandro Cadena Castillo y Juan Carlos Tapia Maldonado, declaramos bajo juramento que el trabajo aquí escrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

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

Andrés Alejandro Cadena Castillo

Juan Carlos Tapia Maldonado

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Andrés Alejandro Cadena Castillo y Juan Carlos Tapia Maldonado, bajo mi supervisión.

MSc. María Soledad Jiménez Jiménez
DIRECTORA DE PROYECTO

AGRADECIMIENTO

Agradezco a mis padres por haberme dado la vida y por haberme inculcado valores tan importantes como la honestidad y la responsabilidad. Un agradecimiento especial a Danny López y Segundo Quizhpi por su colaboración en la elaboración de este proyecto; y a la Familia Tapia Maldonado, en especial a Adriana y Juan, por haberme acogido en su hogar en el desarrollo de este trabajo.

Andrés A. Cadena C.

AGRADECIMIENTO

Agradezco a todas las personas que han hecho posible la realización de este trabajo, a Dios sin el cual no es posible la consecución de nada y quien me ha dado la voluntad y constancia necesarias. De un modo muy especial mi gratitud a mis padres, a mi familia, maestros y mis amigos en particular a Danny, Ramiro Andrés y Segundo.

Juan Carlos Tapia M.

DEDICATORIA

Quiero dedicar este trabajo a mi familia: a mi hermana Gabriela, a mis tíos Dori y Raúl, a mis primos Saskia, Maggy y Raúl; y especialmente a mis padres Gladys y Gonzalo, quienes me han apoyado durante toda mi carrera estudiantil.

Andrés A. Cadena C.

DEDICATORIA

A mis padres, hermanos, tíos, abuelos en especial a mi papi Facundo, a mi sobrina Antonella y a Lili.

Juan Carlos Tapia M.

CONTENIDO

RESUMEN

El presente proyecto de titulación fue realizado para brindar una solución eficiente en el diseño y construcción de un sistema de adquisición y almacenamiento masivo de datos.

Como fuente de señal de entrada al equipo se tiene un sensor sísmico de tres componentes; estas señales, antes de ingresar a un conversor analógico digital, son acondicionadas con un circuito desplazador de voltaje y un circuito de ganancia variable. Una vez que se tienen los datos digitales se los almacena en una memoria USB, la cual es portátil, de bajo costo y de gran capacidad de almacenamiento.

Este trabajo se ha dividido en cuatro Capítulos, en el Capítulo 1 se presenta la teoría referente al manejo del Protocolo USB y de la Tabla de Asignación de Archivos, el funcionamiento del Sistema de Posicionamiento Global y de un Sistema Sísmico, e información referente a Microcontroladores AVR.

En el Capítulo 2 se describe los requisitos de diseño expuestos por el Instituto Geofísico y los dispositivos escogidos para satisfacerlos. También se detallan el diseño de Hardware y Software para cada bloque del equipo GEODAS_USB.

En el Capítulo 3 se detallan las pruebas realizadas al equipo GEODAS_USB, acompañadas de los resultados obtenidos y su respectivo análisis.

En el Capítulo 4 se presentan las Conclusiones y Recomendaciones de todo el Proyecto de Titulación.

Adicionalmente se incluyen cinco anexos, en el Anexo A se presentan los esquemas eléctricos del equipo GEODAS_USB, en ellos se reflejan todas las conexiones necesarias para el uso de cada componente.

En el Anexo B están las láminas PCB del Equipo final, las cuales son utilizadas para la fabricación del circuito impreso.

En el Anexo C se muestran fotos del equipo GEODAS_USB, por ejemplo la placa diseñada en vistas superior e inferior, también se muestran fotos de algunos componentes importantes como del controlador USB y del módulo GPS. Además una foto con todos los elementos necesarios para la instalación de una estación.

El Anexo D contiene el Manual de Usuario del Equipo GEODAS_USB, en donde se detalla el uso de las herramientas necesarias para la calibración del equipo y para el análisis de los datos. Además se muestran los componentes del sistema y cómo emplearlos correctamente.

Finalmente en el Anexo E se detalla el contenido del CD GEODAS_USB en el cual se incorporan los códigos fuentes de los programas cargados en los microcontroladores, además las herramientas para visualizarlos y modificarlos.

PRESENTACIÓN

El equipo GEODAS_USB es una herramienta portátil de fácil instalación, que será usada por el Instituto Geofísico para el monitoreo sísmico y volcánico en zonas de difícil acceso, donde establecer un enlace de radio para la transmisión de datos sería muy costoso o no factible.

Este equipo fue diseñado e implementado para adquirir datos de un sensor sísmico de tres componentes, sincronizarlos con la hora global, acondicionar las señales recibidas y grabarlas en un dispositivo de almacenamiento masivo USB.

Las características eléctricas del sismómetro empleado como fuente de información son: voltaje de salida pico - pico de ±2.5 V en el cual la respuesta del sensor es lineal, y sus frecuencias límites de respuesta son 1/20 Hz y 40 Hz.

Se implementa un filtro pasivo pasa bajos con una frecuencia de corte de 50 Hz para garantizar que los datos almacenados sean los generados por la fuente sísmica. Adicionalmente existe un circuito de ganancias analógicas variables el cual es controlado digitalmente. Los niveles de ganancias posibles van en un rango desde cero hasta quince en pasos de uno.

El equipo es capaz de sincronizarse con la hora global y además obtener su posición global por medio de un módulo receptor de GPS. Con esto se garantiza que los datos almacenados son coherentes en tiempo para todas las estaciones, permitiendo un análisis fiable de los datos.

Por último, por medio del controlador AT43USB380 el equipo funciona como *HOST* USB y es capaz de reconocer una gama de dispositivos conectados a su puerto, discriminando entre ellos las memorias USB (*Mass Storage Device*). Una vez establecida la comunicación con el dispositivo conectado, se implementa como capa superior el manejo de archivos utilizando FAT32, grabando así un archivo .GDU que es reconocido por cualquier sistema operativo.

CAPÍTULO 1 SUSTENTACIÓN TEÓRICA

1.1 PROTOCOLO USB [1],[2],[13]

1.1.1 INTRODUCCIÓN

En este proyecto se analizarán ciertos aspectos importantes del protocolo USB, enfocándose principalmente a los temas relacionados con los dispositivos de almacenamiento masivo que utilizan comunicación USB, por ejemplo tipo de transferencia de datos, protocolo de capa superior, y en el caso de los tipos de dispositivos clasificados por velocidad de transmisión, sólo se explicará el funcionamiento de *Low y Full speed*, debido a que no existen memorias USB que sean dispositivos *High speed*.

USB (*Universal Serial Bus*) es un protocolo *plug&play*[1] entre una computadora y sus periféricos, tales como un teclado, una cámara o un *mouse*. Fue creado en noviembre de 1994 con la participación de cuatro compañías *Campaq*, *Intel*, *Microsoft* y *NEC*; las cuales se unieron para brindar una solución de conexión entre una computadora y un teléfono, siendo de un fácil uso, bajo costo para altas velocidades de transmisión, y con una proyección a una expansión en sus puertos.

Reseña Histórica

USB 1.1

La primera versión generada de este protocolo, podía establecer una velocidad de transmisión de datos desde 1,5 Mbps hasta 12 Mbps. En donde se soportaba transmisiones de voz a tiempo real, audio, y video comprimido. Permitiendo que

[1] Plug&play: es una característica que se les otorga a los dispositivos que pueden conectarse a una computadora y empezar a trabajar inmediatamente sin necesidad de un reinicio del sistema operativo.

un mayor número de dispositivos puedan ser fácilmente conectados a la computadora, pudiendo llegar hasta 127 dispositivos añadidos a una sola computadora; conectados entre ellos en una topología de árbol con la ayuda de *HUBs* para aumentar la capacidad en los puertos.

Es un protocolo que aprovecha el ancho de banda disponible para aumentar la eficiencia en la transmisión, con sobrecarga baja en cada intercambio de datos. También permite una amplia gama de tamaños de paquetes, acoplándose así a los requerimientos de cada aplicación.

USB 2.0

Fue creado en octubre de 1999, como una evolución de la versión 1.1 y totalmente compatible con ésta. Permite mayores velocidades de transmisión, llegando hasta 480 Mbps; con lo cual se tiene un ancho de banda suficiente para transmitir video sin compresión.

Fue pensado principalmente para permitir la conexión de periféricos que necesitan mayor rendimiento como son lectores y grabadores de CD/DVD, discos duros de alta densidad, entre otros.

OTG (*On-the-Go*)

Es una adaptación a la versión USB 2.0, y fue creada en junio del 2002. Nació para aprovechar la aceptación de las versiones anteriores, debido al bajo costo de implementación, su alta velocidad, su alta difusión y la gran cantidad de periféricos que pueden ser encontrados en el mercado. Y cubrir una deficiencia encontrada en los sistemas desbalanceados, la cual no permite la conexión directa entre dos periféricos.

Esta nueva evolución del protocolo funciona con nuevos conectores mas pequeños, nuevos cables y adaptadores, limitando la capacidad de funciones en los equipos. Básicamente, se realiza un reconocimiento del tipo de conector que posee cada periférico; entonces se negocia cuál de los dos realizará la función de

Host para comandar el envío de datos, tomando en cuenta que los dos equipos poseen las capacidades duales.

1.1.2 INTERFAZ MECÁNICA

1.1.2.1 Cable

Para la versión 2.0, el cable utilizado es un *STP* (AWG 28) de 4 hilos, el cual posee dos hilos de datos (D- y D+) entrecruzados, VBus, GND, y la cubierta o *shield*.

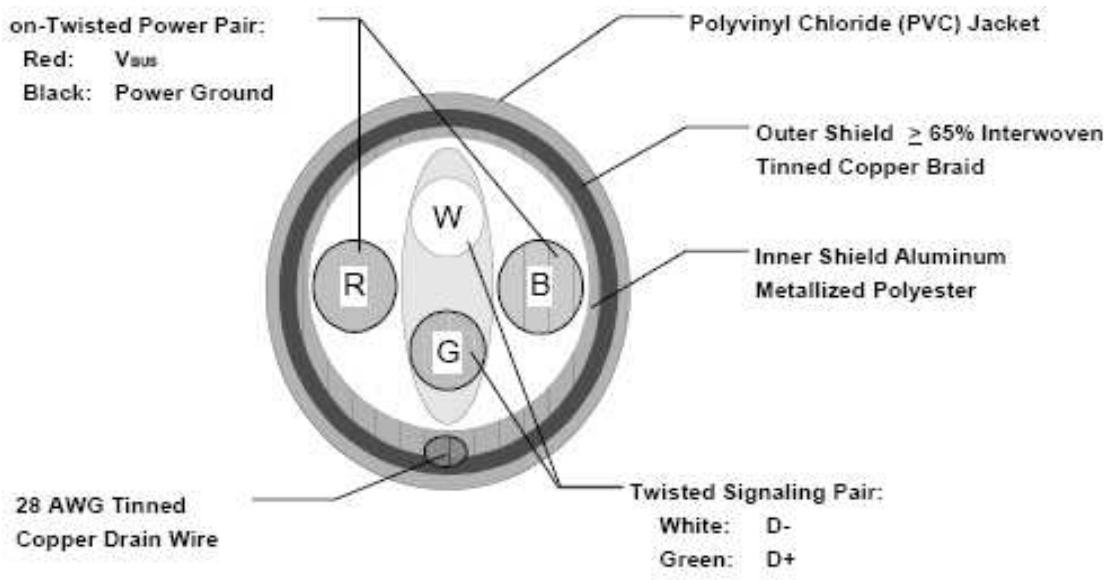


Gráfico 1-1. Corte de un cable tipo para la versión USB 2.0 [1]

Este tipo de cable es totalmente compatible con la versión 1.1, que al utilizar una velocidad de transmisor menor, no necesita que el par de datos sea trenzado ni acorazado, pudiendo utilizar simplemente un cable *UTP*.

1.1.2.2 Conectores

De acuerdo a la especificación dada para USB, existen dos tipos de conectores, Serie A y Serie B. Clasificados así dependiendo del tipo de dispositivo donde se encuentre el conector, de tal forma que en un dispositivo *Host* se tiene un Serie A, y en un *Slave* o *Device* se tiene un Serie B.

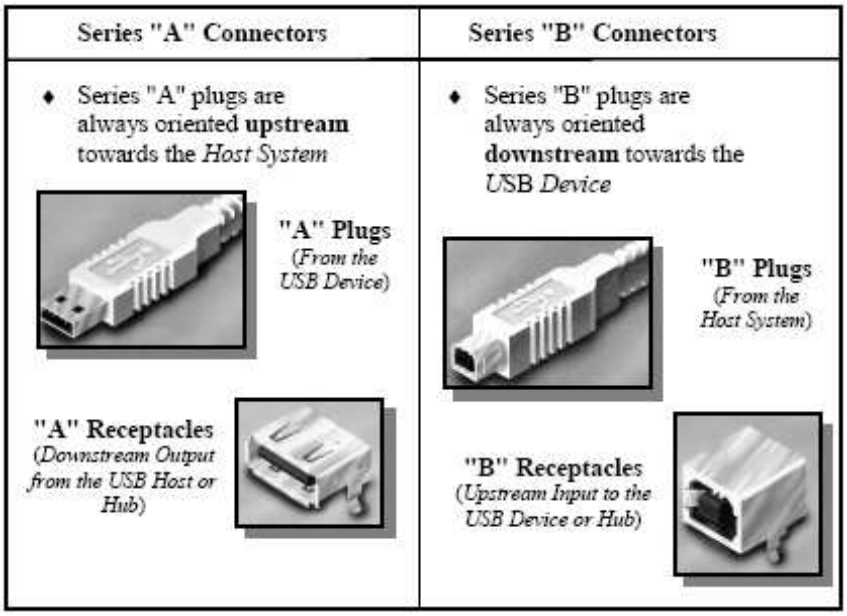
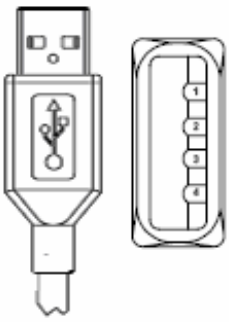


Gráfico 1-2. Tipo de conectores existentes según la normativa USB 2.0. [1]

Para cada uno de los tipos de conectores se especifica en la documentación el conector macho y el correspondiente conector hembra, así como también sus medidas y características físicas; los conectores fueron estandarizados de esta manera, para que su reemplazo en caso de fallo o pérdida sea sencillo y de fácil acceso.

A continuación se presenta un gráfico donde se especifica dentro del conector cómo están distribuidos los pines y las señales correspondientes a cada uno, acompañado del gráfico de un conector Serie A, para indicar su ubicación física.



Contact Number	Signal Name	Typical Wiring Assignment
1	VBUS	Red
2	D-	White
3	D+	Green
4	GND	Black
Shell	Shield	Drain Wire

Gráfico 1-3. Distribución de las señales en el conector Serie A. [1]

El elemento, ya sea *Host* o *Device*, tiene dimensiones estandarizadas para el conector y la huella a ser usada en el PCB. En el gráfico 1-4 se muestra un esquema acotado del conector, con la distribución de pines y además una huella modelo para ser colocada en la placa de un dispositivo *Host*.

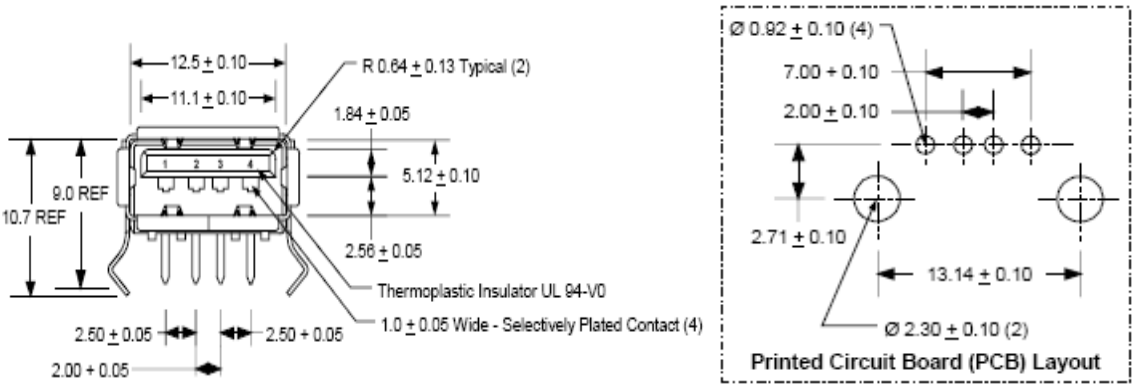


Gráfico 1-4. Un conector tipo A con su respectiva huella PCB, utilizado para un dispositivo Host. [1]

1.1.3 CAPA FÍSICA

En referencia a esta capa, se van a explicar los aspectos eléctricos, como son los niveles de voltaje en polarización y en señal, detección de la velocidad de un dispositivo conectado, codificación de línea y velocidades de transmisión.

1.1.3.1 Polarización y Consumo

En la siguiente tabla se pueden observar los valores mínimos y máximos para ciertos parámetros que están involucrados en la comunicación USB, como son voltaje de polarización y los valores lógicos aceptados en las líneas de datos D+ y D-. Además valores de corriente que puede consumir o suministrar de acuerdo a lo especificado por el estándar USB.

Parámetro	Símbolo	Min	Max	Unidades
Voltaje de Polarización:				
Puerto *High-power*	VBUS	4.75	5.25	V
Puerto *Low-power*	VBUS	4.75	5.25	V

Corriente de Polarización:				
Hub High-power (salida)	ICCPRT	500		mA
Hub Low-power (salida)	ICCUPT	100		mA
Dispositivo *High-power* (entrada)	ICCHPF		500	mA
Dispositivo *Low-power* (entrada)	ICCLPF		100	mA
Dispositivo/Hub no configurado (entrada)	ICCINIT		100	mA
Dispositivo *High-power* suspendido	ICCSH		2.5	mA
Dispositivo *Low-power* suspendido	ICCSL		500	µA
Niveles de entrada para *Low/Full-speed*:				
Alto	VIH	2.0		V
Bajo	VIL		0.8	V
Alta Impedancia	VIHZ	2.7	3.6	V
Sensibilidad entrada diferencial	VDI	0.2		V
Niveles de salida para *Low/Full-speed*:				
Alto	VOH	2.8	3.6	V
Bajo	VOL	0.0	0.3	V
SE1	VOSE1	0.8		V
Rango *Crossover*[2] de salida	VCRS	1.3	2.0	V

Tabla 1-1. Niveles de voltaje permitidos según la especificación USB 2.0 para dispositivos Low/Full-speed. [1]

Este tipo de comunicación desbalanceada obliga a tener dos puntos de vista para la validación de ciertas características eléctricas del protocolo, así un *Host Controller*[3] debe tener la capacidad de proporcionar un voltaje de polarización de 5V al dispositivo si éste lo necesita; de ser así, un Host genérico debería implementar un control de corriente, para que ésta no exceda los valores especificados en la tabla 1-1.

Al contrario, un *Device* puede ser de dos tipos: *Bus-powered*, los cuales obtienen su polarización desde el bus proporcionado por el *Host*, en éste el consumo de

[2] Voltaje *Crossover*: es un rango de voltaje donde se debe encontrar el cruce de las señales de datos D+ y D-.
[3] *Host Controller*: es el dispositivo ya sea un microcontrolador o un arreglo de *hardware*, que maneja el proceso USB. Este es el nodo principal en la distribución tipo estrella que tiene este protocolo.

corriente no debe exceder lo especificado en la tabla 1-1, así como sus señales D+ y D- deben estar dentro del rango de datos válidos.

Otros tipos de *Device* son los *Self-powered*, éstos son dispositivos en los que su consumo va a ser mayor al permitido por su *Host*, por lo que se usa una fuente de poder externa para alimentar a todo su circuito y la energía proporcionada por el *Host* no es utilizada; así mismo sus líneas de datos deben estar dentro del mismo rango de voltajes para que sean interpretadas debidamente.

Dentro de esta clasificación de dispositivos hay que tomar en cuenta que pueden existir *Device* tipo *HUB*, los cuales cumplen la función de *Host* para nuevos dispositivos conectados a ellos; sin exceder el límite de corriente permitido por el *Host* al cual se conectan.

1.1.3.2 Detección de Velocidad de un Dispositivo

Un *Host* reconoce qué tipo de *Device* ha sido conectado, identificando a qué velocidad éste será capaz de transmitir sus datos, todo esto es realizado en *hardware* permitiendo que el mismo *Host* pueda atender los requerimientos de varios dispositivos que manejen diferentes velocidades. Estas velocidades son estándar y han variado de acuerdo a la versión de la especificación, así un *Low-speed* tendrá una configuración física diferente que un dispositivo que puede trabajar a *Full-speed*.

La diferencia en *hardware* entre un dispositivo *Full-speed* con otro *Low-speed* se reduce a una resistencia de *pull-up*[4] de 1,5 kΩ conectada a 3.3 V desde uno de los pines de datos, por lo que si detecta que hay una resistencia conectada al D+ es un dispositivo *Full-speed*, caso contrario si se encuentra en el D- es un *Low-speed*.

[4] Resistencia de *pull-up*: es una resistencia conectada entre una entrada digital y el nivel de voltaje de polarización.

En el siguiente gráfico se explica la configuración de cada dispositivo, en donde el *Host* tiene dos resistencias de *pull-down*[5] de 15 kΩ cada una desde las líneas de datos a tierra.

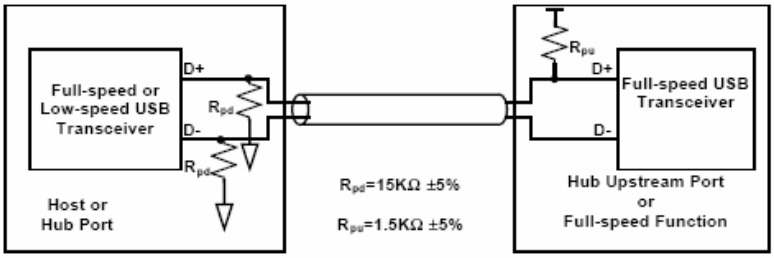


Gráfico 1-5. Diagrama de un *Host* conectado a un *Device Full-speed*. [1]

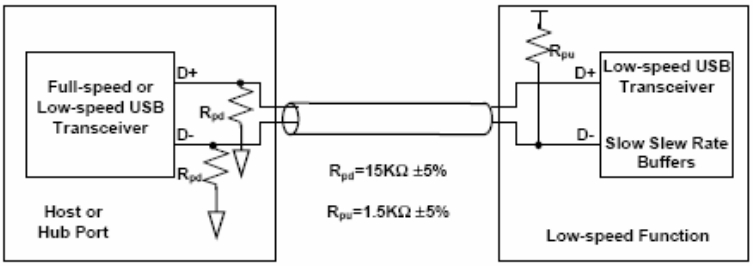


Gráfico 1-6. Diagrama de un Host conectado a un Device Low-speed. [1]

Estos valores de las resistencias fueron escogidos para que en conjunto satisfagan las condiciones eléctricas del protocolo. Para una detección de la velocidad el dispositivo es reseteado por el *Host*, el cual aterriza las dos señales de datos por un tiempo mínimo de 10ms, al salir de la condición de reset, la línea de datos que haya estado conectada con la resistencia de *pull-up* a 3.3 V va a tender a regresar a este valor por lo que se verá un cambio desde 0 V hasta VIH(min), con lo cual el *Host* detecta fácilmente qué línea de datos realizó ese cambio y determina a qué velocidad trabaja el dispositivo conectado.

1.1.3.3 Tipos de Señales

Dentro del protocolo para que éste cumpla con el *handshake*[6] necesario se han definido algunos tipos de señales, las cuales tienen una forma de expresarse

[5] Resistencia de *pull-down*: es una resistencia conectada entre una entrada digital y el nivel de voltaje de tierra eléctrica.
[6] *Handshake*: palabra utilizada para expresar un proceso de intercambio de datos o señales dentro de un protocolo.

mediante las líneas de datos D+ y D-, y los voltajes que éstas toman. En la siguiente tabla se indican algunas de ellas, y se utilizan los nombres en inglés para una mejor explicación del protocolo.

Estado del Bus	**Niveles de Voltaje**		
	Conector Origen[7]	Conector Final[8]	
		Requerido	Aceptable
Differential "1"	D+ > VOH (min) and D- < VOL (max)	(D+) - (D-) > 200 mV and D+ > VIH (min)	(D+) - (D-) > 200 mV
Differential "0"	D- > VOH (min) and D+ < VOL (max)	(D-) - (D+) > 200 mV and D- > VIH (min)	(D-) - (D+) > 200 mV
*Single-ended*0 (SE0)	D+ and D- < VOL (max)	D+ and D- < VIL (max)	D+ and D- < VIH (min)
*Single-ended*1 (SE1)	D+ and D- > VOSE1(min)	D+ and D- > VIL (max)	
Data J state: *Low-speed* *Full-speed*	 *Differential* "0" *Differential* "1"	 *Differential* "0" *Differential* "1"	
Data K state: *Low-speed* *Full-speed*	 *Differential* "1" *Differential* "0"	 *Differential* "1" *Differential* "0"	
Idle state (reposo): *Low-speed* *Full-speed*	NA	 D- > VIHZ (min) and D+ < VIL (max) D+ > VIHZ (min) and D- < VIL (max)	 D- > VIHZ (min) and D+ < VIH (min) D+ > VIHZ (min) and D- < VIH (min)
Resume state	*Data K state*	*Data K state*	
Start-of-Packet	D+ y D- cambian desde el estado de reposo (*Idle*) al K *state*.		

[7] Conector origen: hace referencia al conector en el *Host Controller*.
[8] Conector final: hace referencia al conector en el dispositivo USB, se diferencian a los conectores porque puede haber una extensión en medio de los dos.

(SOP)			
End-of-Packet (EOP)	SE0 por 2 tiempos de bit seguido por *J state* por un tiempo de bit.	SE0 ≥ 1 tiempo de bit seguido por J state por un tiempo de bit	SE0 ≥ 1 tiempo de bit seguido por un *J state*
Reset	D+ and D- < VOL (max) ≥10ms	D+ and D- < VIL (max) for ≥10 ms	D+ and D- < VIL (max) for ≥2.5 µs

Tabla 1-2. Tipos de señales dentro del protocolo USB, con sus respectivos valores límites.[1]

1.1.3.4 Codificación de Datos

El protocolo USB utiliza una codificación de datos *NRZI*, en este tipo de codificación un "1" se caracteriza por mantener el nivel de voltaje, y un "0" por un cambio en el nivel. En la figura siguiente se puede observar un ejemplo, donde existen varios "1" consecutivos y provocan periodos largos sin transiciones; en cambio que varios "0" seguidos provocan que el dato *NRZI* se invierta con cada bit.

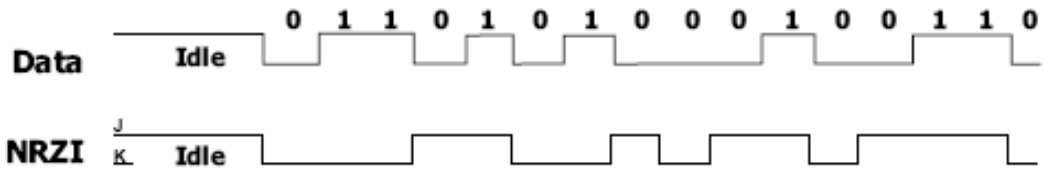


Gráfico 1-7. Ejemplo de codificación *NRZI*. [1]

1.1.3.5 Relleno de Bits

Dentro de esta codificación se utiliza un relleno de bits para asegurar que la comunicación no falle por falta de sincronismo entre el trasmisor y el receptor. Esta herramienta altera la secuencia de bits de datos cuando esto sea necesario.

Así se añade un "0" luego de seis "1s" consecutivos en la cadena de datos antes de la codificación con *NRZI*, para así forzar una transición en los datos, con lo que

se asegura que el receptor obtenga una transición al menos cada siete bits, para garantizar el sincronismo del reloj de recepción.

El receptor decodifica los datos *NRZI*, reconociendo los bits añadidos y retirándolos de los datos.

1.1.3.6 Velocidad de Transmisión

Según la velocidad de transmisión, a los dispositivos USB ya sean *Host* o *Device*, se los clasifica en tres tipos:

High-speed: estos dispositivos tienen una velocidad nominal 480 Mbps, y solo permiten una desviación de un ± 0.05%.

Full-speed: en este caso se tiene una velocidad nominal de 12 Mbps, con un tolerancia de ± 0.25%.

Low-speed: estos dispositivos son los más sencillos de construir debido a la velocidad relativamente baja de 1.5 Mbps con una tolerancia de ± 1.5%; esta desviación permite que se usen cristales de bajo costo para dispositivos de baja velocidad.

1.1.4 CAPA ENLACE

1.1.4.1 Campo de Sincronismo

Todo paquete comienza con un campo de sincronismo, éste es una secuencia de ocho bits que genera la mayor cantidad de transiciones; y es usado por el circuito de recepción para alinear los datos entrantes con el reloj local. Este campo no será incluido en los siguientes diagramas de paquetes, ya que solo es un mecanismo de sincronismo.

1.1.4.2 Campo de Identificador de Paquete PID

En un paquete USB, luego de un campo de sincronismo siempre se encuentra este identificador de cuatro bits donde se especifica el tipo de paquete, y cuatro bits para un chequeo de errores del campo anterior. Este segundo campo se genera con el complemento a 1 del primer campo, garantizando así una correcta decodificación del tipo de paquete a ser recibido.

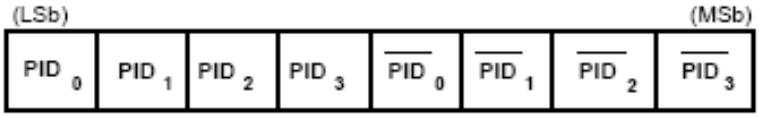


Gráfico 1-8. Formato del Identificador de Paquete. [1]

Los bits PID<1:0> entregan la clasificación dentro de tres grupos de codificación: *token*, *data* y *handshake*.

Tipo de PID	Nombre del PID	PID<3:0>
Token	*OUT*	0001b
	IN	1001b
	SOF	0101b
	SETUP	1101b
Data	*DATA0*	0011b
	DATA1	1011b
Handshake	*ACK*	0010b
	NAK	1010b
	STALL	1110b

Tabla 1-3. Tipo de paquetes de acuerdo al identificador recibido. [1]

1.1.4.3 Campo de Dirección

Con este campo se identifica al *Device*, del cual se puede recibir o al que se le puede enviar datos dependiendo del valor del *Token* en el campo PID.

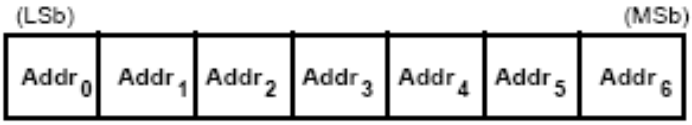


Gráfico 1-9. Campo de dirección. [1]

Como se observa en la figura, con siete bits se puede tener 128 direcciones, pero la dirección cero es reservada, y será la dirección por defecto que tendrá cada dispositivo hasta que el *Host*, en el proceso de enumeración, le asigne su dirección única y final con lo que solamente es posible conectar 127 dispositivos, lo cual cumple con la limitación dada por la especificación.

Este campo sólo es recibido cuando se tiene *Token IN*, *OUT* y *SETUP*.

1.1.4.4 Campo de Endpoint[9]

Este campo consiste de cuatro bits, lo cuales permiten una mayor flexibilidad en el direccionamiento cuando se necesita más de un *endpoint*. Especificando así a qué *endpoint* está direccionado el flujo de datos.

Todos los dispositivos deben tener un *endpoint* de control por defecto en el *endpoint* cero, los demás *endpoints* pueden ser configurados para una función específica.

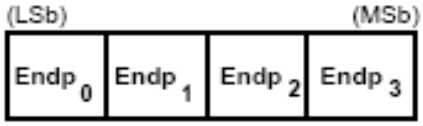


Gráfico 1-10. Campo de endpoint. [1]

En el gráfico 1-10 se puede ver que se tiene cuatro bits para este campo, lo cual limita la cantidad de *endpoints* que un dispositivo puede manejar, pero también existe una limitación por el tipo de dispositivo. Para un *Device Low-speed* se tienen un número máximo de tres *endpoints*: uno de control en el *endpoint* cero y

[9] *Endpoint*: es un elemento del *Device* USB con dirección única, que será una fuente o destino de información en la comunicación entre el *Host* y el *Device*.

dos que pueden ser de control o de interrupción. Un *Device Full-speed* es capaz de manejar los 16 *endpoints* los cuales pueden ser de entrada o salida.

Este campo sólo es recibido cuando se tiene *Token IN*, *OUT* y *SETUP*.

1.1.4.5 Campo de Datos

Este campo puede variar desde cero hasta 1024 bytes, esta longitud de datos varia de acuerdo al tipo de transferencia que se tiene. Siendo USB un protocolo con transmisión serial, primero se envía el bit menos significativo y todos los bytes son consecutivos (sin sobrecarga entre ellos), como se muestra en el siguiente gráfico:

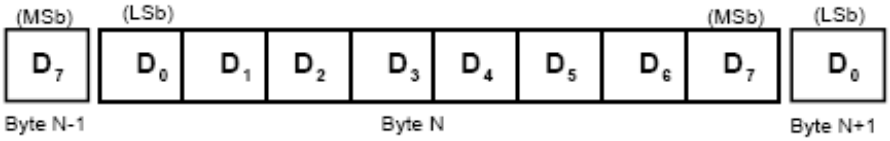


Gráfico 1-11. Muestra la distribución de los bits en la transmisión de datos, en donde se envía primero el menos significativo. [1]

1.1.4.6 Campo de CRC[10]

Este campo es generado para asegurar una transmisión de datos exitosa, teniendo así unos polinomios generadores capaces de detectar errores de uno o dos bits. Este chequeo se lo hace a todos los campos después del PID, y si se detecta algún error el receptor puede eliminar un campo o todo el paquete.

Existen dos polinomios generadores, uno usado para los *Token IN*, *SETUP* y *OUT*, en donde se chequea los campos de Dirección y *Endpoint*, y sólo ocupa cinco bits, como se indica a continuación:

$$G(X) = X^5 + X^2 + 1$$

[10] CRC: *Cyclic Redundancy Check*: es una forma de examinar los datos para observar si existió algún error en la transmisión, lectura o escritura de datos.

El otro polinomio generador es de 16 bits, y es sólo aplicado al campo de datos:

$$G(X) = X^{16} + X^{15} + X^{2} + 1$$

1.1.4.7 Formato de los Paquetes

1.1.4.7.1 Paquete tipo Token

Este tipo de paquete consiste de cuatro campos, el campo *PID* especifica el tipo de *Token*: *IN*, *OUT* o *SETUP*. A continuación con los campos Dirección y *Endpoint* se indica la dirección específica del *endpoint* que está involucrado en la transacción de datos. Al final se tiene un campo CRC asociado a este tipo de paquetes.

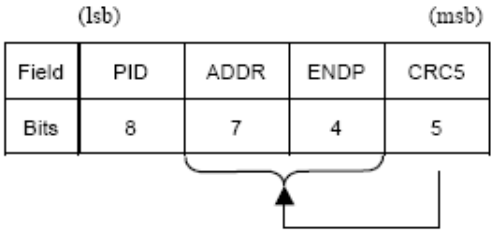


Gráfico 1-12. Formato del Paquete tipo Token. [1]

1.1.4.7.2 Paquete tipo Inicio de Trama

Este tipo de paquete es enviado por el *Host* a una velocidad nominal (12 Mbps) cada 1 ms para dispositivos *Full-speed*. Éste consiste en un campo de *PID* donde se indica el tipo de paquete; y a continuación un campo de once bits con el número de la trama.

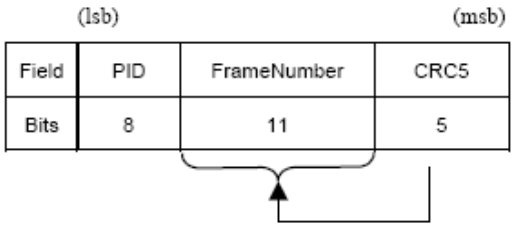


Gráfico 1-13. Formato del Paquete tipo Inicio de Trama. [1]

Al final se tiene un campo *CRC*, de cinco bits, que utiliza el mismo polinomio generador de un paquete tipo *Token*.

1.1.4.7.3 Paquete tipo Datos

Este tipo de paquetes consiste de: su respectivo *PID*, el cual indica si es *DATA0* o *DATA1*, lo cual ayuda en el sincronismo de los datos en esta capa; luego se tiene el campo de datos, el cual tiene una longitud variable, que puede ir desde cero hasta el máximo permitido por el tipo de dispositivo.

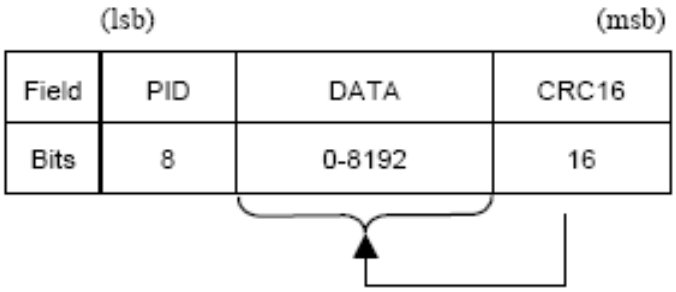


Gráfico 1-14. Formato del Paquete tipo Datos. [1]

De esta manera un dispositivo *Low-speed* sólo puede tener una cantidad de datos de ocho bytes, y uno que sea *Full-speed* puede llegar hasta 1024 bytes, dependiendo del tipo de transferencia utilizado.

Cabe aclarar que los datos tienen que ser una cantidad entera de bytes, y el campo de *CRC* que sigue sólo involucra el chequeo de los datos, y no del *PID*, ya que éste tiene su propio campo para chequeo de errores.

1.1.4.7.4 Paquete tipo Handshake

Estos paquetes sólo consisten del campo de PID, con el cual se discriminan los tres tipos diferentes de paquetes que se pueden tener, los mismos que son utilizados para reportar un estatus dentro de una transacción de datos.

(lsb) (msb)

Field	PID
Bits	8

Gráfico 1-15. Formato del Paquete tipo Handshake. [1]

ACK: Indica que el paquete fue recibido sin ningún error detectado por el relleno de bits o por el campo *CRC*, y además que el campo *PID* fue recibido correctamente. Este paquete es transmitido por el *Host* en una transacción *IN*, y por el *Device* en las transacciones *OUT* y *SETUP*.

NAK: Es un paquete que sólo puede ser enviado por el *Device*, e indica que éste no fue capaz de aceptar datos desde el *Host* (*OUT*), o que no tiene datos para enviar al *Host* (*IN*). Es utilizado para un control de flujo en donde se indica que el *Device* es incapaz de transmitir o recibir datos, pero que es capaz de regresar a su funcionamiento normal sin la intervención del *Host*.

STALL: Sólo un *Device* puede devolver este tipo de paquetes. Esta posibilidad puede ocurrir cuando el *Host* trata de enviar algún comando por el *endpoint* de control, osea cuando éste envía un paquete de *SETUP*, y el *Device* no soporta el comando recibido.

1.1.4.8 Transacciones tipo Bulk

Este tipo de transacciones se caracterizan porque brindan una transferencia de datos libre de errores entre el *Host* y el *Device*, lo cual significa que reconoce si existió un error y realiza una retransmisión de los datos. Como se muestra en el gráfico 1-16, una transacción *Bulk* consiste de tres fases: *Token*, *Data*, y *Handshake*, las cuales son utilizadas de acuerdo a lo que suceda dentro del protocolo.

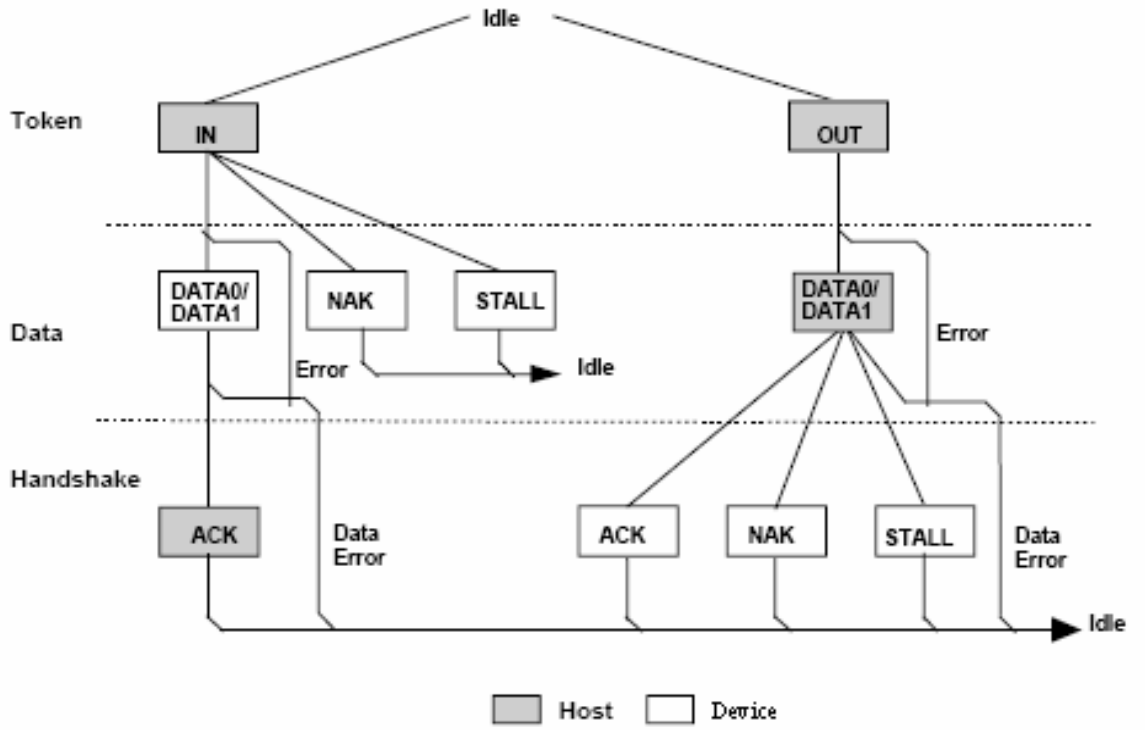


Gráfico 1-16. Formato de una Transacción Bulk. [1]

Para una transferencia *IN*, el *Host* inicia la comunicación con un *Token IN*, el *Device* tiene tres opciones: enviar los datos, responder que no tiene datos, o rechazar el comando recibido. Una vez que se han recibido los datos, el *Host* verifica si existió algún error, de no existir ninguno, éste responde con un ACK, comunicándole al *Device* que la transacción fue exitosa.

Para una transacción *OUT*, el *Host* le comunica al *Device* que le va a enviar datos con un *Token OUT*, a continuación envía los datos. El *Device* comprueba que no existan errores en la comunicación, y responde con un *ACK* si la recepción de datos fue exitosa, con un *NAK* si no pudo recibir los datos, o con un *STALL* si el comando recibido no es válido.

En cualquiera de los casos si se detecta un error en la recepción de los datos, no se envía ningún paquete de *handshake*, lo que indicará al dispositivo que envió los datos que éstos no fueron recibidos.

1.1.4.9 Transacciones tipo Control

En este tipo de transferencia de datos, existen por lo menos dos etapas, una donde se envía un *Token SETUP*, y una donde el *Host* recibe el estatus del comando enviado. Es posible una etapa intermedia donde el *Host* envíe o reciba datos, dependiendo del comando enviado en el *Token*.

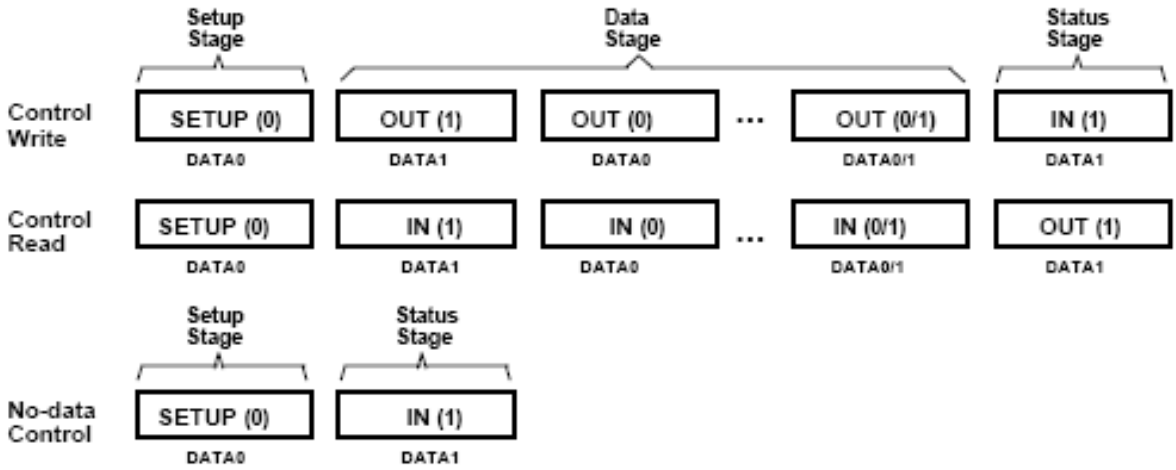


Gráfico 1-17. Secuencias de Escritura y Lectura en una Transacción tipo Control. [1]

Este tipo de transferencia de datos sigue el mismo procedimiento que el tipo *Bulk*, sólo que no se envía un *Token IN* o un *Token OUT*, sino que dependiendo del comando dentro del *Token SETUP*, el *Device* reconoce cómo debe proceder. De igual manera se tiene las señales de *Handshake* y la verificación de errores para el control de flujo en la transferencia de datos.

1.1.4.10 Transacciones tipo Interrupción

Este tipo de transferencia también es confiable, es decir, utiliza las señales de *handshake* para asegurar una transferencia de datos. Lo único que la diferencia de una transacción tipo *Bulk*, es que el *Host* determina un ancho de banda fijo para un cierto dispositivo, ya sea para una transferencia con un *Token IN* o un *Token OUT*; es decir cada cierto tiempo fijo dependiendo del tipo de *Device*, el *Host* inicia una transferencia de datos.

1.1.4.11 Transacciones tipo Isócronas

Este tipo de transacciones tienen las fases de envío del *Token* y de Datos, pero no la fase de *handshake*. Con lo cual se evita el control de errores, teniendo una velocidad de envío de datos mayor pero no confiable.

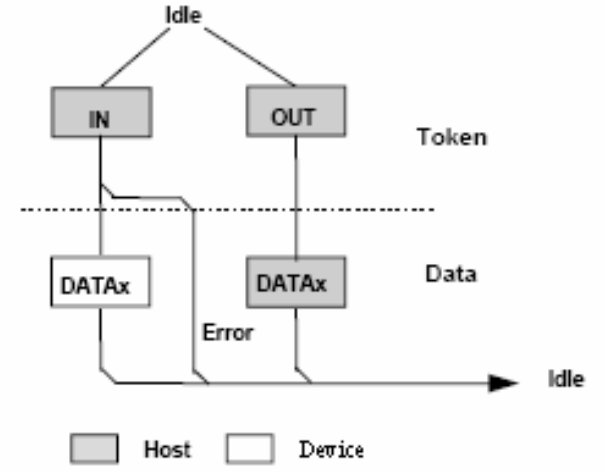


Gráfico 1-18. Formato de una transacción tipo Isocrónica. [1]

Como se muestra en la figura 1-18 dependiendo del tipo de *Token* enviado por el *Host* se tiene la dirección de flujo de los datos. La única comprobación de errores la realiza el dispositivo con los *Token IN*, y de existir alguno se descartan los datos recibidos en la siguiente etapa.

1.1.4.12 Sincronización de Datos

El protocolo USB tiene un mecanismo para garantizar una secuencia correcta en el envío de datos entre el *Host* y el *Device*. Esto se consigue con los paquetes identificados como *DATA0* y *DATA1*. Ambos *Host* y *Device*, comienzan un proceso de transferencia de datos con el *DATA0*, el que actúe de receptor de datos, únicamente cambiará la secuencia a *DATA1* cuando reciba los datos libres de errores, y el transmisor hará lo mismo sólo si recibe un *ACK* de los datos enviados. De no recibir ninguna respuesta, el dispositivo transmisor asumirá que los datos no han sido recibidos e intentará una nueva transmisión con el mismo valor ya sea *DATA0/1*. Un ejemplo similar se muestra en la siguiente figura, donde

los datos recibidos no fueron aceptados por el Device y éste responde con un *NAK*, y el transmisor intenta enviar los mismos datos hasta recibir un *ACK*.

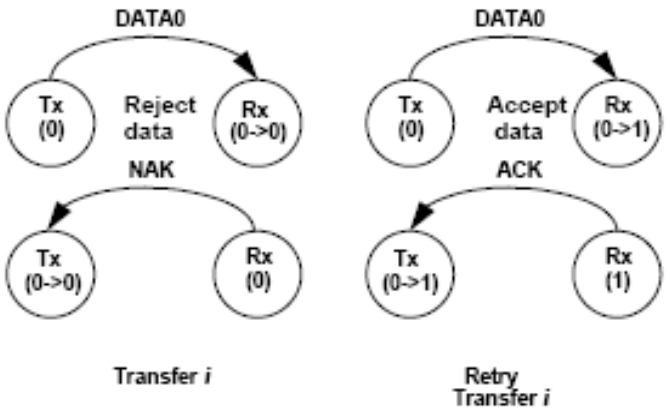


Gráfico 1-19. Ejemplo de retransmisión de datos. [1]

En el caso de una transacción Isócrona no existe esta sincronización por el intercambio entre el *DATA0* y *DATA1*, debido a que no se tiene la fase de *handshake*.

Ejemplo de una Transferencia de Datos *Bulk*

Un tipo de transacción *Bulk* es de los más completos dentro del protocolo USB, ya que realiza todos los procesos para así garantizar un arribo correcto de los datos. En la figura 1-20 se muestra el ejemplo de una transferencia *IN*.

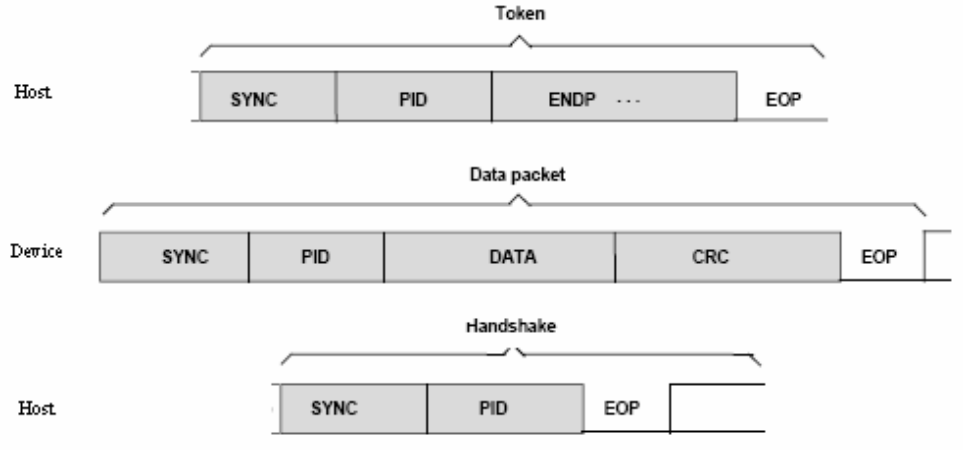


Gráfico 1-20. Ejemplo de un flujo de datos para una transferencia Bulk – IN. [1]

Primero el *Host* envía el *Token IN*, con su campo de sincronismo, su respectivo *PID* y la dirección del dispositivo y *endpoint* al que va dirigido el comando.

El *Device* verifica que el *PID* no contenga errores, de ser así continua a la etapa de enviar los datos solicitados, con todos los campos necesarios, donde ya se incluye el de *CRC* que sólo protege el campo de Datos.

Al final el *Host* verifica que no existió ningún error tanto en el campo de *PID*, como en el de Datos usando los campos de *CRC*; de ser exitosa esta verificación, el *Host* envía la señal de *Handshake ACK*, culminando con la transferencia de datos.

1.1.5 ESTADOS DE UN DISPOSITIVO

Un dispositivo tiene que pasar por ciertas etapas cuando es conectado a un puerto USB, de esta manera el *Host* puede conocer el tipo de *Device* y su modo de operación (entendiendo el tipo de transacción que utilizará, si es auto polarizado, y qué tipo de función ofrece).

En la siguiente tabla se detallan los posibles estados que puede tomar un dispositivo:

Conectado	***Powered***[11]	***Default***[11]	***Address***[11]	***Configured***[11]	**Estatus**
No	-	-	-	-	El dispositivo no está conectado al puerto USB.
Si	Si	No	-	-	El dispositivo ha sido conectado y polarizado, pero no ha sido reseteado.
Si	Si	Si	No	-	El dispositivo ya ha sido reseteado, pero no ha sido asignada

[11] Powered, Default, Address, Configured: Palabras tomadas de la especificación del USB, no han sido traducidas para una correcta explicación del protocolo.

					una dirección única. El dispositivo responde con la dirección por defecto.
Si	Si	Si	Si	No	El dispositivo ya recibe su dirección única, pero no ha sido configurado.
Si	Si	Si	Si	Si	El dispositivo luego de ser configurado está listo para que el *Host* haga uso de la función del *Device*.

Tabla 1-4. Estados de un Dispositivo. [1]

Estado Conectado

USB es un protocolo *plug&play*, por lo que se toma en cuenta este estado, el proceso que realiza el *Host* para conocer al *Device* sólo se da cuando el dispositivo está conectado al puerto USB.

Estado *Powered*

Un dispositivo USB puede alimentarse directamente del puerto USB y/o de una fuente externa, por lo que el dispositivo ya puede estar polarizado antes de estar conectado al bus, puesto que el dispositivo llega a este estado una vez que las líneas de Datos ya tienen voltaje.

Estado *Default*

Después que un dispositivo ha sido polarizado, éste no debe hacer caso a ninguna señal que pueda cruzar por el bus si es que no ha sido reseteado. Una vez pasado el proceso de reseteado el *Host* reconoce a qué velocidad funciona el *Device*, el cual luego de la señal de reset tiene la dirección por defecto.

Estado *Address*

Una vez reseteado el dispositivo responde a la dirección por defecto, hasta que el *Host* le asigne una dirección única, que será la que utilice desde ese momento para cualquier transacción a realizarse.

Estado *Configured*

Antes que la función proporcionada por el *Device* pueda ser utilizada, el dispositivo tiene que ser configurado, esto significa que todos los valores asociados con los *endpoints* a ser utilizados deben ser puestos a su valor por defecto, por ejemplo configurar que el primer dato a ser recibido sea el *DATA0*.

1.1.6 PROCESO DE ENUMERACIÓN

Este proceso es realizado por el *Host* USB cada vez que un dispositivo es conectado a su puerto, mediante un conjunto de transacciones éste puede conocer de qué tipo es el *Device* y cómo debe manejarlo. Este proceso consta de algunos pasos:

1. Una vez que el *Host* reconoce que un dispositivo ha sido conectado en el puerto, éste espera 100 ms para permitir una buena conexión física, y realiza un *Reset* al *Device*.
2. Una vez terminada la señal de *Reset*, el *Device* se encuentra en el estado *Default* y responde a la dirección por defecto.
3. Empleando esta dirección y a través del endpoint de Control el *Host* lee el descriptor del dispositivo, en el cual encontrará información como el tamaño máximo de bytes de datos en el *endpoint* por defecto.
4. El *Host* también lee la información de cada configuración disponible para el *Device*, este proceso puede durar algunos milisegundos hasta obtener todos los descriptores, entre ellos los de cada interfaz y los de cada *endpoint*.

5. Ahora el *Host* con toda la información necesaria, asigna una dirección única al *Device*, entonces éste pasa al estado *Address*.
6. El *Host* basado en toda la información recibida, selecciona una configuración para el *Device* y éste pasa al estado *Configured*. En este punto el *Device* configura sus *endpoints* con la descripción entregada en los diferentes descriptores, y el dispositivo ahora se encuentra listo para ser utilizado.

Una vez desconectado el dispositivo del puerto, toda esa información recibida es descartada y la dirección única asignada estará libre para ser usada por otro dispositivo.

1.1.7 DESCRIPTORES

Un dispositivo USB utiliza los descriptores para informar sus atributos. Un descriptor es una estructura definida de datos, en donde el primer campo es un byte donde se indica su longitud, seguido de un byte donde se indica el tipo de descriptor.

Si el *Host* dentro del proceso de enumeración recibe un valor menor al establecido para un tipo de descriptor, la información recibida con el descriptor es desechada; en cambio si recibe un número mayor al definido, los bytes sobrantes son ignorados.

Hay un tipo especial de descriptor que es el *String*, en el cual se entrega una información adicional como un texto asociado a cada descriptor, éste es opcional por lo que existe un campo en cada descriptor donde se indica si existe el *String* que acompaña a esa información.

1.1.7.1 Descriptor tipo Device

Éste contiene una información general sobre el *Device*. Lo recibido en este descriptor se tomará como global para todas las configuraciones que el dispositivo tenga, por lo tanto solo existirá un descriptor tipo *Device*.

En la siguiente tabla se muestra cada campo de este tipo de descriptor.

Campo	Tamaño	Valor[12]	Descripción
bLength	1	12h	Tamaño del descriptor en bytes
bDescriptorType	1	01h	Tipo de descriptor *DEVICE*
bcdUSB	2	0200	Versión del protocolo USB utilizado
bDeviceClass	1	-	Código de Clase, asignado por el *USB-IF*[13]
bDeviceSubClass	1	-	Código de Sub Clase, asignado por el *USB-IF*
bDeviceProtocol	1	-	Código de Protocolo, asignado por el *USB-IF*
bMaxPacketSize0	1	8,16,32 o 64	Tamaño máximo para los paquetes enviados por el Endpoint cero.
idVendor	2	-	ID del Vendedor, asignado por el *USB-IF*
idProduct	2	-	ID del Producto, asignado por el fabricante del *Device*.
bcdDevice	2	-	Versión del *Device*, expresado con un número BCD
iManufacturer	1	-	Índice del descriptor tipo *String*, describiendo al fabricante
iProduct	1	-	Índice del descriptor tipo *String*, describiendo al producto
iSerialNumber	1	-	Índice del descriptor tipo *String*, describiendo el número de serie del

[12] Valor: en este campo se pondrán valores que son comunes o pueden ser tomados como ejemplo.
[13] *USB-IF*: *USB Implementers Forum*: es una corporación no lucrativa que fue formada para facilitar el desarrollo de productos que utilicen el Protocolo USB.

			producto
bNumConfigurations	1	-	Es el número de posibles configuraciones que tiene el dispositivo

Tabla 1-5. Tabla de los campos de un Descriptor tipo Device.[1]

1.1.7.2 Descriptor tipo Configuración

Este descriptor entrega información específica relacionada a una de las posibles configuraciones en un dispositivo. Una misma configuración puede manejar varias interfaces, y este valor también se especifica dentro de este descriptor.

En la siguiente tabla se muestra una explicación de cada campo que contiene este tipo de descriptor.

Campo	Tamaño	Valor	Descripción
bLength	1	09h	Tamaño del descriptor en bytes
bDescriptorType	1	02h	Tipo de descriptor CONFIGURATION
wTotalLength	2	-	Longitud total para esta configuración, incluyendo todos los descriptores de interfaz y *endpoints*
bNumInterfaces	1	-	Número de interfaces soportadas por esta configuración
bConfigurationValue	1	-	Valor que es usado como argumento para seleccionar esta configuración
iConfiguration	1	-	Índice del descriptor tipo *String*, describiendo a esta configuración
bmAttributes	1	1xx00000b	Características de la configuración: D7: 1L D6: Auto-polarizado D5: Despertar remoto D4…0: 0L
bMaxPower	1	-	Máximo consumo de corriente que el dispositivo necesitará del puerto USB, expresado en unidades de 2 mA

Tabla 1-6. Tabla de los campos de un Descriptor tipo Configuración. [1]

1.1.7.3 Descriptor tipo Interfaz

Para un tipo de configuración, este descriptor dará información de los tipos de interfaces con las que se puede trabajar; dentro de estas puede haber o no *endpoints*. En el número de *endpoints* con los que la interfaz puede trabajar no se toma en cuenta el número cero porque es el *endpoint* de control por defecto.

En la siguiente tabla se muestra una explicación a cada campo que contiene este tipo de descriptor.

Campo	Tamaño	Valor	Descripción
bLength	1	09h	Tamaño del descriptor en bytes
bDescriptorType	1	04h	Tipo de descriptor INTERFACE
bInterfaceNumber	1	-	Es el número identificador de esta interfaz, dentro del grupo que puede soportar la configuración
bAlternateSetting	1	-	Valor para seleccionar una opción adicional para la interfaz arriba escogida
bNumEndpoints	1	-	Número de *endpoints* usados por esta interfaz, excluyendo el *endpoint* de control
bInterfaceClass	1	-	Código de Clase, asignado por el *USB-IF*
bInterfaceSubClass	1	-	Código de Sub Clase, asignado por el *USB-IF*
bInterfaceProtocol	1	-	Código de Protocolo, asignado por el *USB-IF*
	1	-	Índice del descriptor tipo *String*, describiendo a esta interfaz

Tabla 1-7. Tabla de los campos de un Descriptor tipo Interfaz.[1]

1.1.7.4 Descriptor tipo Endpoint

Para cada endpoint a ser usado dentro de la transferencia de datos existe un descriptor, en el cual se explica el ancho de banda que solicita y el tipo de transferencia de datos que soporta, explicando también si es *IN*, *OUT* o de

CONTROL, dependiendo del sentido de transferencia que tendrán los datos. No existe un descriptor únicamente para el *endpoint* de control por defecto.

En la siguiente tabla se muestra una explicación de cada campo que contiene este tipo de descriptor.

Campo	**Tamaño**	**Valor**	**Descripción**
bLength	1	07h	Tamaño del descriptor en bytes
bDescriptorType	1	05h	Tipo de descriptor *ENDPOINT*
bEndpointAddress	1	x000xxxxb	Es la dirección del *endpoint* definida por una configuración, y tiene la siguiente forma: Bit 3…0: Número del *endpoint* Bit 6…4: 0L Bit 7: Dirección 0 = endpoint *OUT* 1 = endpoint *IN*
bmAttributes	1	-	Este campo describe los atributos del *endpoint*: Bits 1…0: Tipo de Transferencia 00 = Control 01 = Isocrónica 10 = *Bulk* 11 = Interrupción Bits 5…2: Definidos para *endpoints* Isócronos Bits 7…6: 0L
wMaxPacketSize	2	-	Indica el máximo tamaño de paquetes en bytes que el endpoint es capaz de enviar o recibir.
bInterval	1	-	Este campo es usado de acuerdo al tipo de transferencia seleccionada. Para un *endpoint* de Interrupción, puede tener valores desde 1 a 255, indicando el tiempo en el que el *Host* debe acceder con

			información a este *endpoint*, ya sea de lectura o escritura.

Tabla 1-8. Tabla de los campos de un Descriptor tipo Endpoint.[1]

1.1.7.5 Descriptor tipo String

Como fue mencionado antes, estos descriptores son opcionales, y lo único que proporcionan es información adicional en forma de texto, como una explicación de la funcionalidad de cada descriptor.

Es evidente que al ser un texto, un dispositivo puede soportar varios tipos de lenguaje, por lo que se tiene un descriptor tipo *String* que es especial y para obtenerlo se utilizará el Índice con valor cero; en la siguiente tabla se explica cada campo de este tipo de descriptor.

Campo	Tamaño	Valor	Descripción
bLength	1	-	Tamaño del descriptor en bytes
bDescriptorType	1	03h	Tipo de descriptor *STRING*
wLANGID[0]	2	-	Código cero del ID del lenguaje
…	…	…	…
wLANGID[x]	2	-	Código x del ID del lenguaje

Tabla 1-9. Tabla de los campos de un Descriptor tipo String Cero. [1]

Luego de haber descrito el lenguaje en que serán expresados los datos dentro de los demás descriptores tipo *String*, cabe mencionar a los descriptores *UNICODE*[14], que son *String* también, pero éstos ya poseen el texto explicativo a una función específica, como se muestra en la siguiente tabla.

Campo	Tamaño	Valor	Descripción
bLength	1	-	Tamaño del descriptor en bytes
bDescriptorType	1	03h	Tipo de descriptor *STRING*

[14] *UNICODE*: Es un estándar mundial para una codificación de caracteres. (URL: http://www.unicode.com).

bString	N	-	Cadena de datos codificada en el formato *UNICODE*

Tabla 1-10. Tabla de los campos de un Descriptor tipo String UNICODE. [1]

1.1.8 PROTOCOLO COMANDO/DATOS

Para el proceso de lectura y escritura de datos en los dispositivos USB de almacenamiento masivo se utiliza un protocolo confiable que consta de tres fases.

En primera instancia el *Host* envía los comandos *CBW*[15], los cuales están formados de 31 bytes distribuidos de la siguiente forma:

bit / Byte	7	6	5	4	3	2	1	0
0-3	*dCBWSignature*							
4-7	*dCBWTag*							
8-11 (08h-0Bh)	*dCBWDataTransferLength*							
12 (0Ch)	*bmCBWFlags*							
13 (0Dh)	Reserved (0)				*bCBWLUN*			
14 (0Eh)	Reserved (0)			*bCBWCBLength*				
15-30 (0Fh-1Eh)	*CBWCB*							

Gráfico 1-21. Campos del comando *CBW*. [2]

dCBWSignature: este campo contiene el código 43425355h (USBC) el cual ayuda a identificar este tipo de comandos.

dCBWTag: este campo es un identificador de trama; el *Host* coloca un número, el cual tiene que ser devuelto por el *Device* en un campo similar en el comando de respuesta.

[15] *CBW*: *Command Block Wrapper*: es un paquete que contiene un comando de bloque y su información asociada.

dCBWDataTransferLength: en este campo se especifica el tamaño de datos a ser transferidos en la siguiente fase del protocolo. Si este campo es cero, indica que no existe la segunda fase y el *Host* sólo espera el comando de respuesta.

bmCBWFlags: los bits de este campo están definidos de acuerdo a la siguiente tabla.

Bit	Función
7	Dirección del flujo de datos 0 = *Data-Out* 1 = *Data-In*
6	Obsoleto. 0L
5…0	Reservados. 0L

Tabla 1-11. Funciones de los bits del campo bmCBWFlags. [2]

bCBWLUN: este campo especifica el valor del número de la unidad lógica[16] que debe recibir el comando. De no manejar múltiples unidades lógicas el *Host* debe poner este campo a cero.

bCBWCBLength: este campo indica el número de bytes válidos en el campo *CBWCB*. Los únicos valores permitidos son desde el 01h hasta el 10h.

CBWCB: este campo es el conjunto de bytes que forman el comando a ser ejecutado por el dispositivo.

La segunda fase de este protocolo, es la transferencia de datos, existe sólo si el campo *dCBWDataTransferLength* es diferente de cero. La dirección que tendrán los datos depende del bit siete del campo *bmCBWFlags*.

[16] Numero de Unidad Lógica: es la dirección para una unidad de disco, este término es utilizado en el protocolo *SCSI* (*Small Computer System Interface*) para diferenciar varias unidades dentro de un mismo bus.

La tercera fase es realizada únicamente por el *Device*, el cual envía un comando de estatus llamado *CSW*[17] de 13 bytes que están organizados de la siguiente forma:

bit / Byte	7	6	5	4	3	2	1	0
0-3	*dCSWSignature*							
4-7	*dCSWTag*							
8-11 (8-Bh)	*dCSWDataResidue*							
12 (Ch)	*bCSWStatus*							

Gráfico 1-22. Campos del comando CSW. [2]

dCSWSignature: este campo contiene el código 53425355h (USBS) el cual ayuda a identificar este tipo de comandos.

dCSWTag: en este campo el *Device* copia el valor recibido en el campo *dCBWTag* del comando *CBW* respectivo.

dCSWDataResidue: este campo es utilizado por el *Device* para informar al *Host* que cantidad de datos faltan por procesar del número especificado por el campo *dCBWDataTransferLength*.

bCSWStatus: en este campo se indica el éxito o el fallo del comando recibido. El *Device* envía el valor de cero si todo el proceso fue exitoso, o el valor 01h si existió algún error en la ejecución del comando.

[17] *CSW*: *Command Status Wrapper*: es un paquete que contiene el estatus de la ejecución de un comando de bloque.

1.2 TABLA DE ASIGNACIÓN DE ARCHIVOS: FAT [3]

1.2.1 INTRODUCCIÓN

FAT (*File Allocation Table*) este sistema de archivos fue creado por Microsoft para acoplarse de mejor manera a las nuevas capacidades de discos duros, ya que los anteriores sistemas de archivos FAT 12 y 16 generan un cluster[18] demasiado grande.

FAT32 permite una distribución y organización de la memoria para poder acceder a los archivos, directorios y subdirectorios de una manera simple. Para poder acceder a la memoria lo primero que se realiza es la lectura del sector 0 conocido como *Master Boot Record* (*MBR*) como se observa en la figura 1-23.

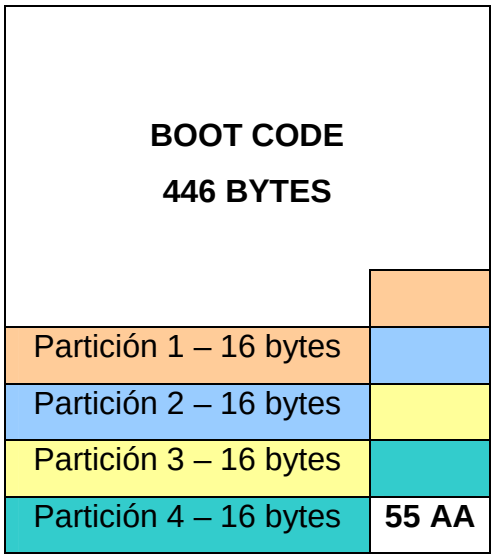


Gráfico 1-23. Campos principales del MBR. [3]

El campo *Boot Code* está formado por 446 bytes que contienen el código de máquina a ser ejecutado por el sistema operativo en el caso de que la unidad de memoria sea empleada para el arranque del sistema. A continuación de este campo se encuentran cuatro campos de 16 bytes cada uno confinado a una partición. Finalmente se debe encontrar el código 55AA que indica que el sector 0 contiene información válida.

[18] Cluster.- Unidad básica de memoria para disco duro

Como se muestra en a figura 1-23, la unidad de memoria puede ser dividida en secciones (conocidas como particiones), en la mayoría de los casos se utiliza una única partición y su información se encuentra en el primer campo de 16 bytes confinados a la partición 1, mientras que los otros tres se encuentran llenos de ceros.

El campo de una partición se encuentra dividido de la siguiente manera:

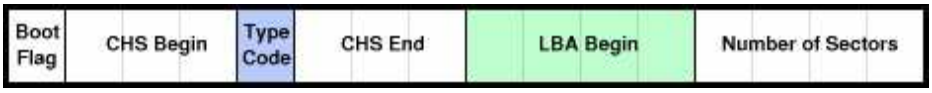


Gráfico 1-24. Campo de partición. [3]

De los campos mostrados en la figura anterior, sólo los que se encuentran resaltados con color tienen importancia para una aplicación básica del FAT32. Para verificar que el sistema de archivos empleado es FAT32 se nenecita chequear que el campo *Type Code* de un byte tenga el código 0Bh o 0Ch.

El campo *Number of Sectors* indica el número de sectores que tiene la memoria; y el campo *LBA Begin* indica el sector donde comienza la partición en el disco, y donde se encuentra el volumen ID; éste sector contiene información sobre la partición, los campos que se necesitan para acceder al mismo, se detallan a continuación:

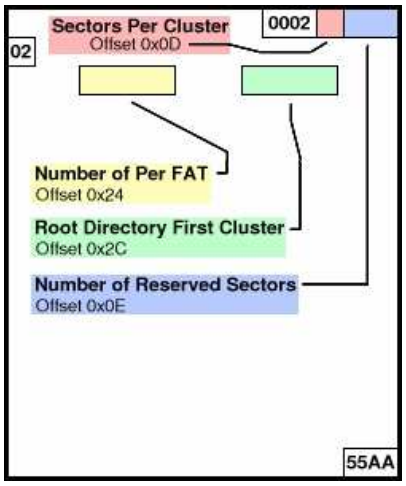


Gráfico 1-25. Campos críticos del Volumen ID. [3]

Campos	Desplazamiento	Tamaño	Valor[19]
Bytes por Sector	0x0B	16 Bits	Siempre 512 Bytes
Sectores por Cluster	0x0D	8 Bits	1,2,4,8,16,32,64,128
Números de Sectores Reservados	0x0E	16 Bits	Usualmente 20h
Número de FATs	0x10	8 Bits	Siempre 2
Sectores por FAT	0x24	32 Bits	Depende del tamaño del disco, valor calculado al ser formateada la unidad
Cluster Inicial del Directorio Raíz	0x2C	32 Bits	Usualmente 0x00000002
Fin del Sector	0x1FE	16 Bits	Siempre 0x55AA

Tabla 1-12. Valores por defecto de los campos críticos del Volumen ID. [3]

En base a estos campos se calcula lo siguiente:

- Dirección del Primer Cluster = Dirección de la Partición + Número de Sectores Reservados + [(Número de FATs)*(Número de sectores por FAT)]

- Dirección del FAT = Dirección de la Partición + Número de Sectores

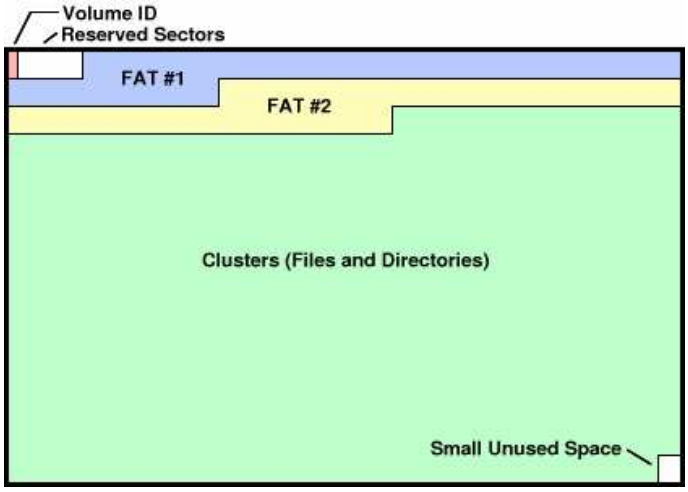


Gráfico 1-26. Mapeo interno de la memoria. [3]

[19] Valor: en esta columna se encontraran algunos valores por defecto o típicos para cada campo.

Al acceder al cluster inicial del directorio raíz se encuentran las entradas de archivos, los cuales están formados por grupos de 32 bytes detallados a continuación:

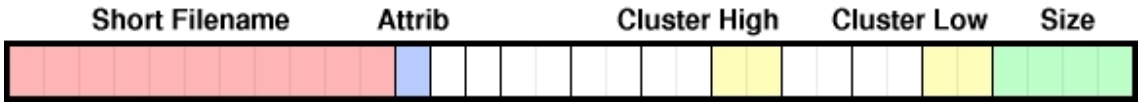


Gráfico 1-27. Campo de entrada de archivo. [3]

Short Filename.- Este campo está formado por 11 caracteres que deben ser letras mayúsculas, los ocho primeros corresponden al nombre y los tres restantes a la extensión. Si el nombre del archivo es menor a ocho caracteres los restantes deben ser llenados con espacios (20h). Nótese que el nombre del archivo no contiene punto.

Attrib.- Este campo está formado por un byte en el cual cada uno de los ocho bits que lo forman indica un atributo particular de archivo como se detalla en la siguiente tabla.

Bits de Atributo	Función	Estado	Comentario
0 (LSB)	Solo lectura	1	No permite la escritura
1	Oculto	1	No se debe mostrar en la inscripción de directorio
2	Sistema	1	El archivo es del sistema operativo
3	Volumen ID	1	Nombre del Volumen
4	Directorio	X	Subdirectorio
5	Archivo	X	Archivo
6	No usado	0	Debería ser cero
7 (MSB)	No usado	0	Debería ser cero

Tabla 1-13. Campo de atributo de archivo. [3]

Como se puede observar la diferencia entre un archivo, un volumen o un directorio es su atributo.

Low Cluster.- Este campo indica el cluster inicial del archivo, conociendo este número se puede ir a la tabla del FAT para recuperar la información que éste contiene.

Size.- Este campo indica el tamaño en bytes que ocupa el archivo.

1.2.2 MANEJO DEL FAT

La tabla del FAT32 está dividida en paquetes de 32 bits (si fuera FAT16 serian paquetes de 16 bits), cada uno de ellos está asociado a un cluster de la memoria. Se tiene que buscar el paquete de 32 bits correspondiente al cluster inicial del archivo, este paquete contiene la dirección del siguiente paquete, y este último contiene la dirección del siguiente y así sucesivamente hasta encontrar un paquete que contiene el código FFFFFFFFh que indica fin de archivo.

Hay que tener presente que los cluster 0 y 1 no son utilizados, por lo tanto los paquetes asociados a ellos en la tabla del FAT tampoco se utilizan.

Si un cluster de la memoria está utilizado, su correspondiente paquete en la tabla del FAT debe contener un número mayor o igual a dos y menor a 0FFFFFFFh o puede tener el código de fin de archivo. Y si un cluster no está utilizado, el código en su paquete correspondiente tiene que ser cero.

En el caso de un directorio, el *low cluster* contiene el inicio de la tabla del nuevo directorio raíz; si éste llegara a ocupar mas de un cluster el principio de búsqueda es igual al de buscar un archivo.

En el siguiente ejemplo se puede entender de mejor manera cómo se encuentra distribuido un archivo o un directorio en la memoria. El directorio empieza en el cluster 00000002h por lo tanto se debe ir al paquete 00000002h de la tabla del

FAT, el cual apunta al 00000009h y éste al 0000000Ah el cual apunta al 0000000Bh y éste al 00000011h. Finalmente en el paquete 00000011h se encuentra el fin de directorio por lo tanto el directorio está formado por el cluster 2h, 9h, Ah, Bh, 11h consecutivamente.

De igual forma se ve que el archivo #1 se encuentra formado por los clusters 3h, 4h, 5h, 7h, 8h.

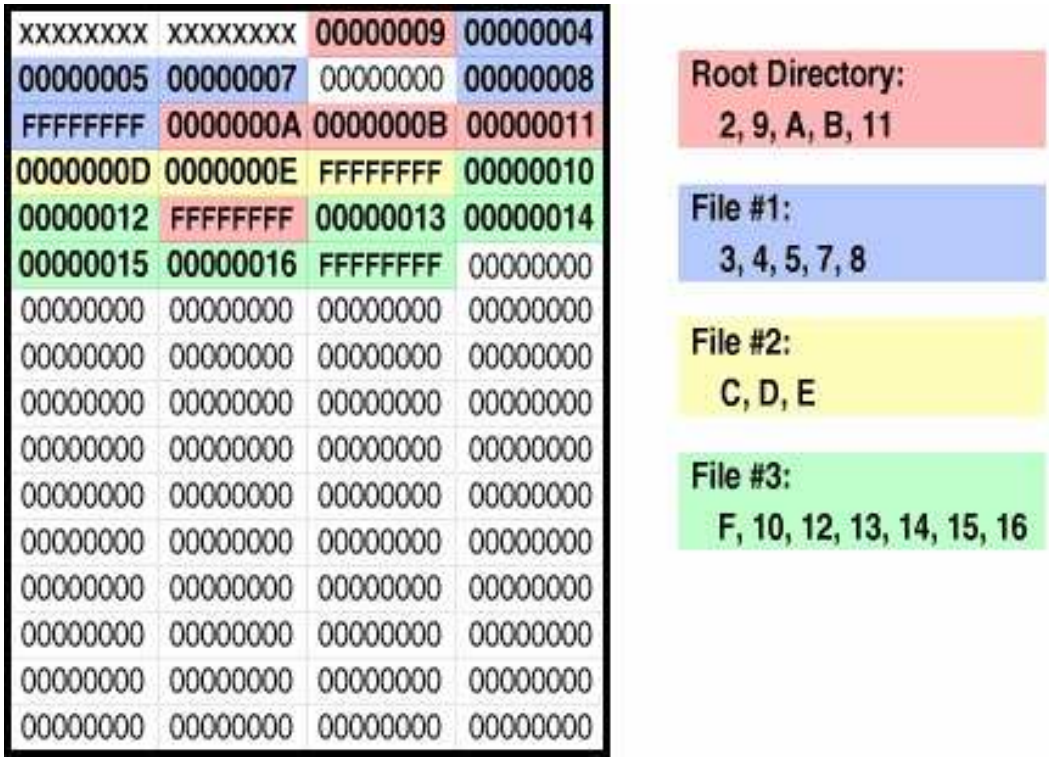


Gráfico 1-28. Ejemplo de uso de la tabla del FAT. [3]

1.3 SISTEMA DE POSICIONAMIENTO GLOBAL [4]

1.3.1 INTRODUCCIÓN

El sistema de posicionamiento global mediante satélites (*GPS*: *Global Positioning System*) supone uno de los más importantes avances tecnológicos de las últimas décadas. Es un sistema de localización, que fue diseñado e implementado desde el final de los 50’s hasta 1995 por el Departamento de Defensa de los Estados

Unidos con fines militares para proporcionar estimaciones precisas de posición, velocidad y tiempo. Por razones de seguridad, las señales GPS generadas para uso civil se someten a una degradación deliberada, al tiempo que su emisión se restringe a una determinada frecuencia.

La precisión intrínseca del receptor de GPS depende del número de satélites visibles en un momento y posición determinados. Con ocho satélites a la vista se tiene un estimado en la posición de seis a quince metros, sin aplicar ningún tipo de corrección a la señal recibida.

1.3.2 ARQUITECTURA DEL SISTEMA GPS

El sistema se descompone en tres segmentos básicos, los dos primeros de responsabilidad militar:

A) Segmento espacio: formado por 24 satélites GPS (21 unidades operativas y tres de repuesto) con una órbita de 20000 Km. de radio y un periodo de doce horas, con trayectorias sincronizadas para cubrir toda la superficie del globo y cuyo abastecimiento se lo realiza mediante energía solar.

B) Segmento control: que consta de cinco estaciones monitoras encargadas de mantener en órbita los satélites y supervisar su correcto funcionamiento, tres antenas terrestres que envían a los satélites las señales que deben transmitir y una estación experta de supervisión de todas las operaciones.

C) Segmento usuario: formado por las antenas y los receptores situados en tierra. Los receptores, a partir de los mensajes que provienen de cada satélite visible, calculan distancias y proporcionan una estimación de posición y tiempo.

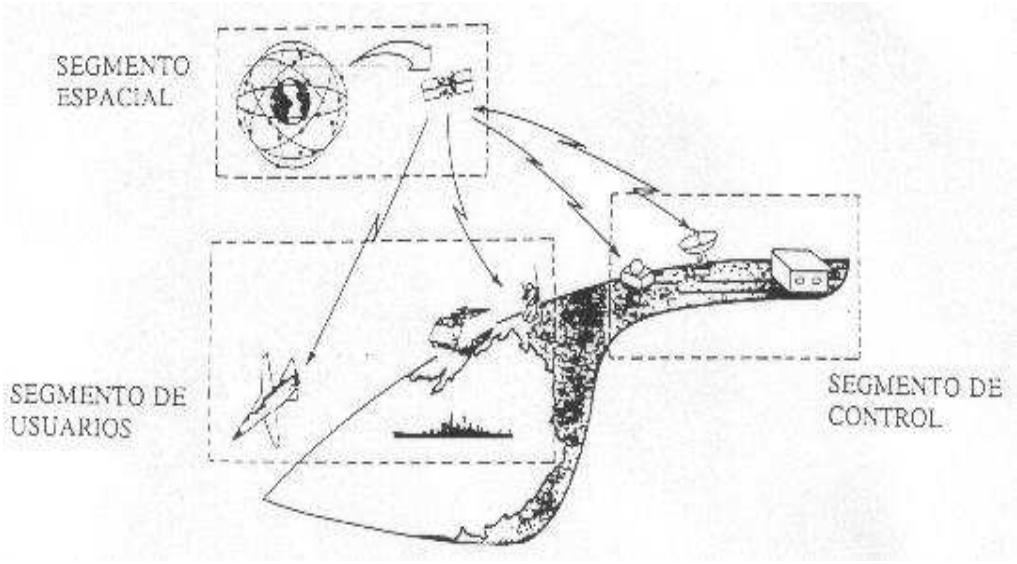


Gráfico 1-29. Gráfico de la composición del sistema de GPS. [4]

1.3.3 PRINCIPIOS DE FUNCIONAMIENTO DEL SISTEMA GPS

El sistema GPS tiene por objetivo calcular la posición de un punto cualquiera en un espacio de coordenadas (x,y,z) partiendo del cálculo de las distancias del punto a un mínimo de tres satélites cuya localización es conocida, utilizando el método llamado Triangulación.

La triangulación consiste en averiguar el ángulo de cada una de las tres señales respecto al punto de medición. Teniendo los tres ángulos se determina fácilmente la propia posición relativa respecto a los tres satélites. Conociendo además las coordenadas o posición de cada uno de ellos por la señal que emiten, se obtiene las posiciones absolutas o coordenadas reales del punto de medición. También se consigue una exactitud extrema en el reloj del GPS, similar a la de los relojes atómicos que desde tierra sincronizan a los satélites.

La distancia entre el usuario (receptor GPS) y un satélite se mide multiplicando el tiempo de vuelo de la señal emitida desde el satélite por su velocidad de propagación. Para medir el tiempo de vuelo de la señal de radio es necesario que los relojes de los satélites y de los receptores estén sincronizados, pues deben generar simultáneamente el mismo código.

El tiempo de retardo necesario para sincronizar ambas señales es igual al tiempo de viaje de la señal proveniente del satélite. Conociendo este tiempo (supóngase 0.06 segundos), se multiplica por la velocidad de la luz (a la que se supone viaja la onda de radio proveniente de los satélites) y se obtiene la distancia hasta el satélite.

Tiempo de retardo (0.06 seg) x Vel. de la luz (300.000 km/seg) = Dist. (18.000 km)

Ahora bien, mientras los relojes de los satélites son muy precisos los de los receptores son osciladores de cuarzo de bajo costo y por tanto imprecisos. Las distancias con errores debidos al sincronismo se denominan pseudodistancias. La desviación en los relojes de los receptores añade una incógnita más que hace necesario un mínimo de cuatro satélites para estimar correctamente las posiciones

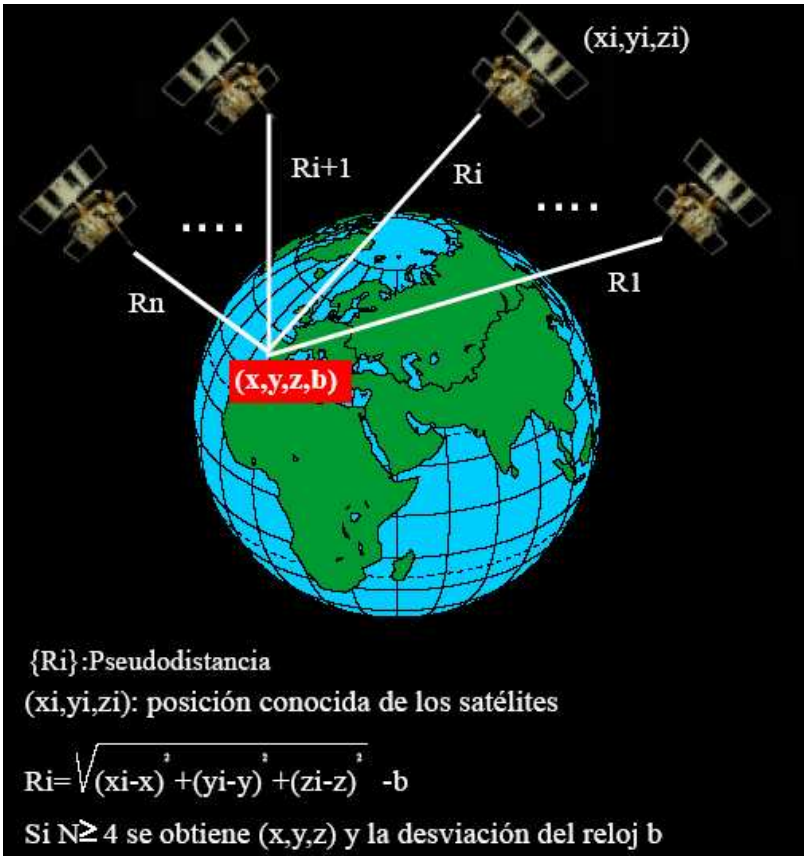


Gráfico 1-30. Modelo de cómo un receptor GPS se conecta a los satélites en orbita. [4]

En el cálculo de las pseudodistancias hay que tener en cuenta que las señales GPS son muy débiles y se hallan inmersas en el ruido de fondo inherente al planeta en la banda de radio. Este ruido natural está formado por una serie de pulsos aleatorios, lo que motiva la generación de un código pseudo-aleatorio artificial por los receptores GPS como patrón de fluctuaciones. En cada instante un satélite transmite una señal con el mismo patrón que la serie pseudo-aleatoria generada por el receptor. En base a esta sincronización, el receptor calcula la distancia realizando un desplazamiento temporal de su código pseudo-aleatorio hasta lograr la coincidencia con el código recibido, este desplazamiento corresponde al tiempo de vuelo de la señal. Este proceso se realiza de forma automática, continua e instantánea en cada receptor.

El código pseudo-aleatorio utilizado tiene un nivel alto de implementación, existen varias y muy buenas razones para esta complejidad. La complejidad del código ayuda a asegurarse que el receptor GPS no sintonice accidentalmente alguna otra señal. Siendo el modelo tan complejo, es altamente improbable que una señal cualquiera pueda tener exactamente la misma secuencia.

Dado que cada uno de los satélites tiene su propio y único Código Pseudo Aleatorio, esta complejidad también garantiza que el receptor no se confunda accidentalmente de satélite. De esa manera, también es posible que todos los satélites trasmitan en la misma frecuencia sin interferirse mutuamente. Esto complica a cualquiera que intente interferir el sistema desde el exterior al mismo.

1.3.4 SINCRONISMO CON HORA GLOBAL

Un sistema receptor GPS no sólo entrega la posición global, también entrega una hora global, la misma que contiene información del año, mes, día, hora, minuto, y segundo, tomando como referencia al meridiano de Greenwich.

Debido a que los relojes implementados dentro de los satélites son increíblemente precisos es posible tomarlos como sistema de referencia para sincronizar los relojes locales de los sistemas que empleen un receptor GPS.

La hora de cada uno de los relojes internos de los satélites, está monitoreada por estaciones en tierra, en caso de que llegase a desigualarse el reloj de un satélite éste es igualado inmediatamente desde la estación de control tomando como referencia a un reloj atómico.

La ventaja de emplear un sistema GPS como reloj, es la de poder tener un gran número de estaciones completamente sincronizadas sin importar su ubicación geográfica al momento de su instalación.

La hora que entrega un receptor GPS está desfasada de la hora global apenas con el tiempo de propagación de la señal enviada por satélite al receptor, y el tiempo que demora el receptor en ponerla a disposición del usuario, este tiempo está en el orden de las decenas de milisegundos.

1.3.5 APLICACIONES DE LOS GPS

Son múltiples los campos de aplicación de los sistemas de posicionamiento tanto como sistemas de ayuda a la navegación, como en modelización del espacio atmosférico y terrestre o aplicaciones con requerimientos de alta precisión en la medida del tiempo. A continuación se detallan algunos de los campos civiles donde se utilizan en la actualidad sistemas GPS:

- Sincronización de Sistemas Electrónicos: Cuando se tienen varias estaciones remotas, se necesita tener un mismo reloj de sincronismo y a su vez cada estación debe conocer su posición real. Todo esto en busca de un correcto análisis de los datos obtenidos por dichas estaciones, dependiendo de la aplicación establecida para cada equipo, por ejemplo el análisis de señales sísmicas, en donde cada estación debe presentar su ubicación y poseer un reloj sincronizado para que sus datos sean entendibles y confiables.

- Sincronización de señales**:** La industria eléctrica utiliza el GPS para sincronizar los relojes de sus estaciones monitoras a fin de localizar posibles fallos en el

servicio eléctrico. La localización del origen del fallo se realiza por triangulación, conociendo el tiempo de ocurrencia desde tres estaciones con relojes sincronizados.

- Estudio de fenómenos atmosféricos: Cuando la señal GPS atraviesa la troposfera, el vapor de agua, principal causante de los distintos fenómenos meteorológicos, modifica su velocidad de propagación. El posterior análisis de la señal GPS es de gran utilidad en la elaboración de modelos de predicción meteorológica.

- Localización y navegación: El sistema GPS se utiliza como ayuda en expediciones de investigación en regiones de difícil acceso y en escenarios caracterizados por la ausencia de marcas u obstáculos. Un ejemplo son los sistemas guiados por GPS para profundizar en el conocimiento de las regiones polares o desérticas.

- Modelos geológicos y topográficos: Los geólogos comenzaron a aplicar el sistema GPS en los 80’s para estudiar el movimiento lento y constante de las placas tectónicas, para la predicción de terremotos en regiones geológicamente activas. En topografía, el sistema GPS constituye una herramienta básica y fundamental para realizar el levantamiento de terrenos y los inventarios forestales y agrarios.

- Ingeniería civil: En este campo se utiliza la alta precisión del sistema GPS para monitorizar en tiempo real las deformaciones de grandes estructuras metálicas o de cemento sometidas a cargas.

- Sistemas de alarma automática**:** Existen sistemas de alarma conectados a sensores dotados de un receptor GPS para supervisión del transporte de mercancías tanto contaminantes de alto riesgo como perecederas (productos alimentarios frescos y congelados). En este caso la generación de una alarma permite una rápida asistencia al vehículo.

- Guiado de disminuidos físicos: Se están desarrollando sistemas GPS para ayuda en la navegación de invidentes por la ciudad. En esta misma línea, la industria turística estudia la incorporación del sistema de localización en guiado de visitas turísticas a fin de optimizar los recorridos entre los distintos lugares de una ruta.

- Navegación y control de flotas de vehículos: El sistema GPS se emplea en planificación de trayectorias y control de flotas de vehículos. La policía, los servicios de socorro (bomberos, ambulancias), las centrales de taxis, los servicios de mensajería, empresas de reparto, etc. organizan sus tareas optimizando los recorridos de las flotas desde una estación central. Algunas compañías ferroviarias ya utilizan el sistema GPS para localizar sus trenes, máquinas locomotoras o vagones, supervisando el cumplimiento de las señalizaciones.

- Sistemas de aviación civil: En 1983 la caída del vuelo 007 de la compañía aérea coreana al invadir cielo soviético, por problemas de navegación, acentúo la necesidad de contar con la ayuda de un sistema preciso de localización en la navegación aérea. Hoy en día el sistema GPS se emplea en la aviación civil tanto en vuelos domésticos, transoceánicos, como en la operación de aterrizaje.

- Navegación desasistida de vehículos: Se están incorporando sistemas GPS como ayuda en barcos para maniobrar de forma precisa en zonas de intenso tráfico, en vehículos autónomos terrestres que realizan su actividad en entornos abiertos en tareas repetitivas, de vigilancia en medios hostiles (fuego, granadas, contaminación de cualquier tipo) y en todos aquellos móviles que realizan transporte de carga, tanto en agricultura como en minería o construcción. La alta precisión de las medidas ha permitido importantes avances en el espacio en órbitas bajas y así tareas de alto riesgo de inspección, mantenimiento y ensamblaje de satélites artificiales pueden ahora realizarse mediante robots autónomos.

1.4 SISTEMA SÍSMICO [5]

1.4.1 ONDAS SÍSMICAS

Al romper un objeto (supóngase un esfero de plástico) se produce un chasquido u ondas sonoras que se desplazan por el aire. De igual forma cuando se arroja una piedra a un estanque también se producen unas ondas (en este caso pequeñas olas) que se propagan desde donde cayó la piedra hacia las orillas del estanque. Algo similar ocurre con los terremotos: al romperse la roca se generan ondas que se propagan a través de la Tierra, tanto en su interior como por su superficie.

En un sólido pueden transmitirse dos tipos de ondas. El primer tipo es conocido como onda de compresión, porque consiste en la transmisión de compresiones (P) y rarefacciones como en el caso de la transmisión del sonido, en este caso las partículas del medio se mueven en el mismo sentido en que se propaga la onda. El segundo tipo es conocido como ondas transversales (S) o de cizalla; las partículas se mueven ahora en dirección perpendicular a la dirección de propagación de la onda. La figura muestra esquemáticamente la propagación de estas ondas en un bloque sólido.

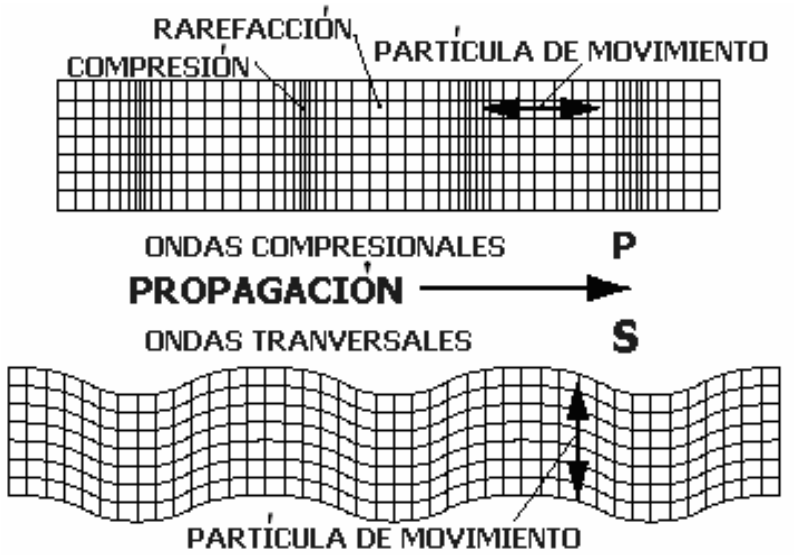


Gráfico 1-31. Ejemplo de la propagación de ondas en un elemento sólido.[5]

Dentro de este grupo de ondas, las P son las que se propagan con mayor velocidad (de ahí su nombre, primarias), presentando además la característica de poder propagarse por cualquier tipo de material, sea sólido o líquido. Las ondas S viajan a una velocidad algo menor (secundarias) y no se propagan por masas líquidas. Las velocidades de las diferentes ondas dependen de las características del medio; por ejemplo, en rocas ígneas la velocidad de las ondas P es del orden de 6 Km/seg, mientras que en rocas poco consolidadas es de aproximadamente 2 Km/seg ó menor [4].

A pesar de que la velocidad en las ondas varía en un factor de diez o más en la Tierra, la relación entre las velocidades promedio de las ondas P y S es constante. Este hecho posibilita a los sismólogos a simplemente tomar el tiempo de diferencia en el arribo de las ondas P y el de las ondas S para tener una rápida y razonable estimación de la distancia desde donde se provocó el fenómeno y la estación de observación. Solo se tiene que multiplicar el tiempo S-P en segundos, por el factor de 8 Km/s para la distancia en kilómetros.

1.4.2 SENSORES SÍSMICOS

Un sismómetro o sismógrafo es un instrumento muy sensible, que es esencial para estudiar los temblores o los terremotos para la sismología. Éste es un aparato que registra el movimiento del suelo causado por el paso de una onda sísmica. Los sismógrafos fueron ideados a fines del siglo pasado y perfeccionados a principios del presente. En la actualidad, estos instrumentos han alcanzado un alto grado de desarrollo electrónico, pero el principio básico empleado no ha cambiado como se verá a continuación.

Básicamente existen dos tipos de sensores sísmicos: Sismómetros Inertes, los cuales miden el movimiento de la tierra en forma relativa a un cuerpo inerte (una masa suspendida); y Medidores de Tensión o Extensómetros, los cuales miden el movimiento de un punto de la tierra en relación a otro. Como en la mayoría de los casos el movimiento de la tierra con respecto a un cuerpo inerte es de mayor amplitud que el tomado de un equipo diferencial, los Sismómetros Inertes son

generalmente más sensibles a señales de temblor de la tierra. Sin embargo, a frecuencias muy bajas es muy difícil el mantener la referencia inerte, por lo que para observación de oscilaciones libres de bajo orden de la tierra, maremotos y deformaciones cuasi – estáticas, los Extensómetros son conceptualmente más simples que los Inertes, aunque su construcción implique un proceso más elaborado.

Un sismómetro inerte convierte el movimiento de la tierra en una señal eléctrica, pero sus propiedades no pueden ser descritas por un factor dentro de una escala, como por ejemplo voltios por milímetro de movimiento de la tierra. La respuesta de este sismómetro a los movimientos producidos por la tierra depende de varios factores, por ejemplo de la amplitud del movimiento y también de la escala de tiempo o qué tan frecuente es su movimiento; esto se debe a que la masa inerte tiene que ser conservada en su sitio por fuerzas mecánicas o electromagnéticas. Cuando el movimiento sea lento, la masa se moverá con el resto del instrumento, y la señal de salida será más pequeña.

El modelo físico más simple para la parte mecánica de un sismómetro inerte es el sistema formado por una masa (M) y un resorte (S) con un amortiguamiento viscoso (R), como se muestra en la figura 1-32:

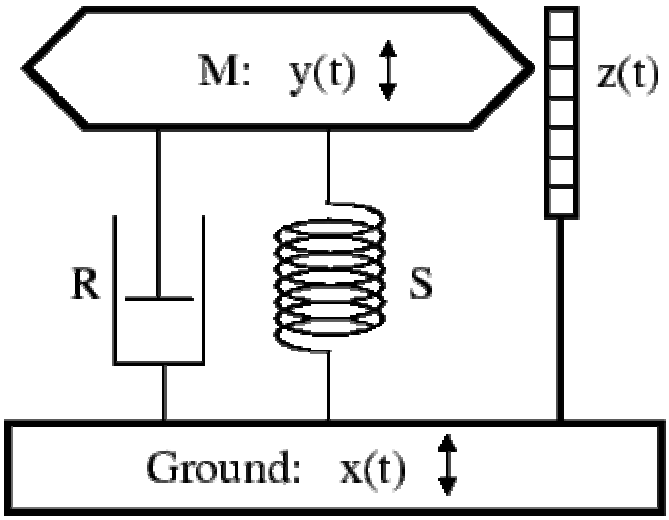
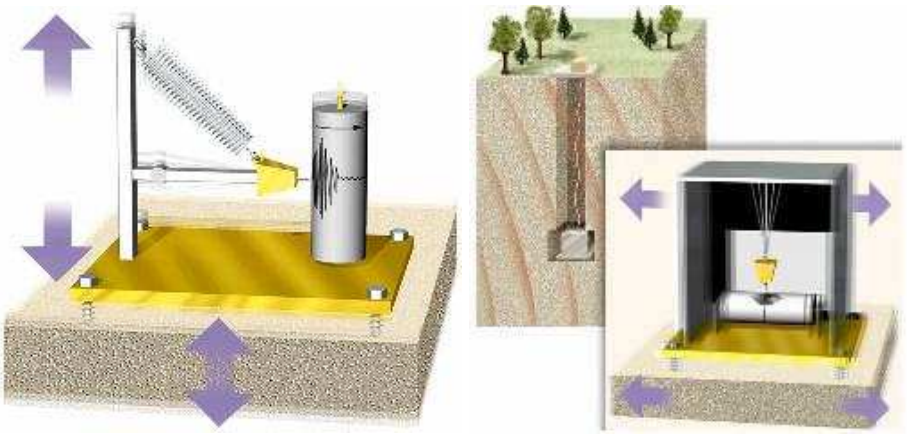


Gráfico 1-32. Modelo físico de un sismómetro. [5]

Otro modelo muy básico para obtener las señales que se propagan por las rocas luego de un movimiento telúrico es el de un péndulo o masa suspendida de un bastidor, el cual se desplaza en forma relativa al suelo, en virtud de su inercia. Este principio determina que todos los cuerpos tienen una resistencia al movimiento o a variar su velocidad. Así, el movimiento del suelo puede ser medido con respecto a la posición de una masa suspendida por un elemento que le permita permanecer en reposo por algunos instantes con respecto al suelo. Estos movimientos, detectados mediante transductores y correctamente amplificados, se registran en función del tiempo. Esto se lo puede realizar sobre un papel ahumado adosado a un tambor que gira a velocidad constante.



Péndulo Vertical *Péndulo Horizontal*

Gráfico 1-33. Ejemplo de un sismómetro de péndulo. [5]

Actualmente todo sismógrafo utilizado es electromagnético, la única variación de este sismógrafo en relación al mecánico, es que el desplazamiento de la masa genera electricidad al mover una bobina dentro del campo magnético de un imán. Al producirse el movimiento del suelo se genera corriente en la bobina proporcional a la velocidad de movimiento del suelo. Este sistema electromagnético reemplaza al sistema amortiguador de los sistemas antiguos, lo cual ayuda a que la oscilación de la masa sea realmente proporcional al movimiento del suelo. Adicionalmente se tiene un galvanómetro el cual sirve de amplificador del movimiento, para facilitar la visibilidad de los resultados si son impresos sobre un papel.

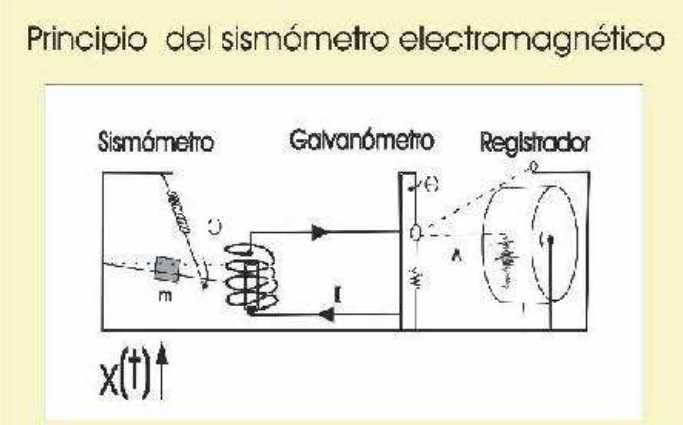


Gráfico 1-34. Principio del sismómetro electromagnético. [5]

Continuamente se ha estudiado la forma de mejorar los elementos sísmicos, hoy en día los sensores sísmicos, llamados de banda ancha, utilizan el sistema de fuerza de retroalimentación o sistemas de fuerza balanceada. Este sistema consiste de un circuito de retroalimentación negativo, el cual ejerce una fuerza proporcional al desplazamiento de la masa inercial para cancelar el movimiento relativo. Un transductor eléctrico convierte el movimiento de la masa en una señal eléctrica, la cual genera una fuerza de retroalimentación que debe ejercerse para anular el movimiento del péndulo corresponde a la aceleración del suelo. Este sistema permite extender el ancho de banda y la linealidad de los sismómetros, porque no permite grandes movimientos de la masa que deformen los resortes o los niveles. La señal de salida de estos sistemas posee un gran rango dinámico debido a que los transductores electromagnéticos también lo poseen.

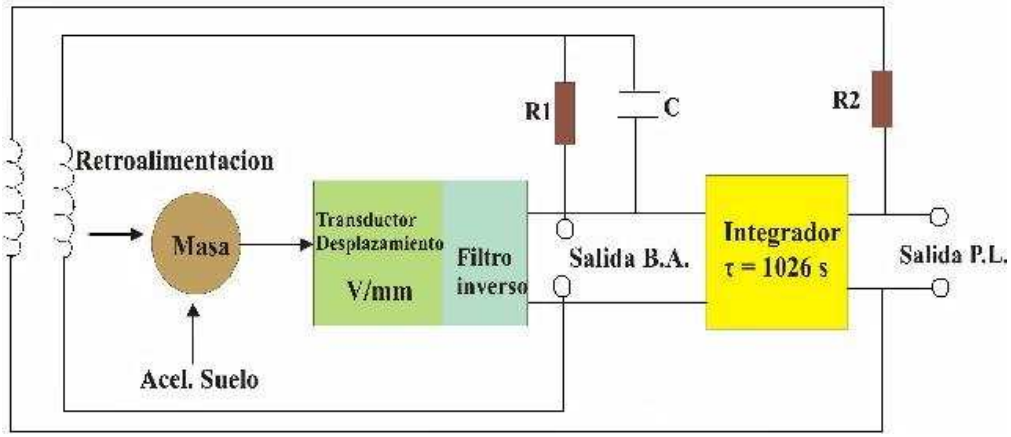


Gráfico 1-35. Diagrama de un sismómetro de banda ancha. [5]

1.5 MICROCONTROLADORES AVR

Estos microcontroladores son desarrollados por ATMEL, el objetivo que se planteo esta empresa fue el de obtener microcontroladores más veloces, de bajo consumo de potencia y económicos.

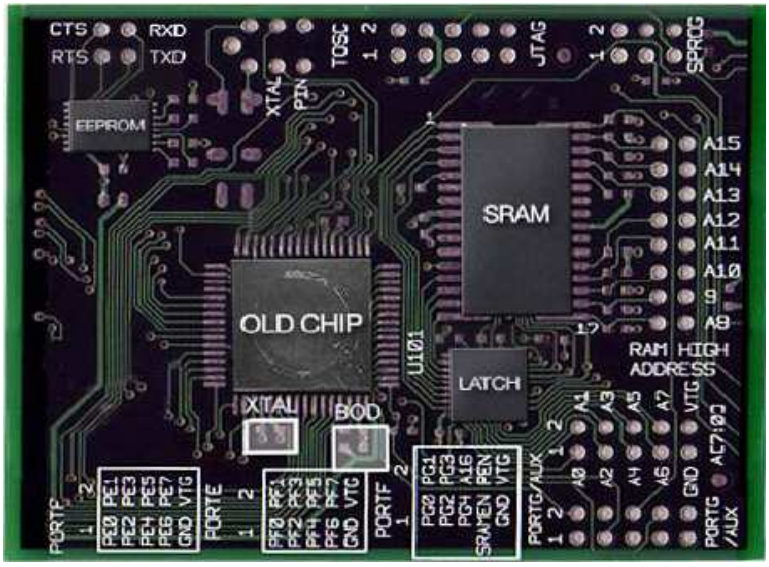


Gráfico 1-36. Vista interna de un microcontrolador AVR

Gráfico 1-37. Vista externa de un microcontrolador AVR

1.5.1. CARACTERÍSTICAS DE LA TECNOLOGÍA AVR

A continuación se expone las características más relevantes que tienen estos microcontroladores.

Esta tecnología fue creada en base a un microprocesador RISC, Computadora con Conjunto de Instrucciones Reducido (*Reduced Instruction Set Computer*),

esto no significa que el microprocesador tenga menos instrucciones sino que éstas se ejecutan con mayor velocidad debido a que el microprocesador tiene más registros de propósito general, lo cual provoca que el tiempo de ejecución de cada instrucción sea menor.

Para poder ejecutar una instrucción el microprocesador tiene que leerla, decodificarla, cargarla en memoria, ejecutarla y dar la respuesta. Este tipo de microprocesador puede comenzar a leer la siguiente instrucción mientras la anterior está siendo decodificada, esto implica que se está trabajando en dos instrucciones.

Todo el set de instrucciones fue desarrollado pensando en emplear compiladores de lenguajes de alto nivel como por ejemplo el lenguaje C.

Los microprocesadores AVR realizan una operación aritmética en un ciclo de reloj excepto la multiplicación que toma dos, cualquier acceso a la RAM demora dos ciclos de reloj, los movimientos entre registros de entrada y/o salida demora un ciclo de reloj, mover 8 o 16 bits entre registros demora un ciclo de reloj y leer la memoria de programa demora 3 ciclos de reloj.

En los microprocesadores AVR las instrucciones están organizadas en grupos de 16 o 32 bits, teniendo un bus interno de 8 bits.

Los microcontroladores AVR pueden trabajar con frecuencias de reloj desde 0 a 16 MHz, algunos hasta 20 MHz, y pueden llegar a voltajes de polarización de 1.8 V mínimo.

Debido a que los microcontroladores AVR emplean un núcleo RISC su consumo de potencia se reduce en comparación con tecnologías antiguas.

1.5.2. FAMILIA DE PRODUCTOS

En el siguiente gráfico se muestra la evolución de los productos que emplean la tecnología AVR desde su núcleo hasta la familia de los Mega.

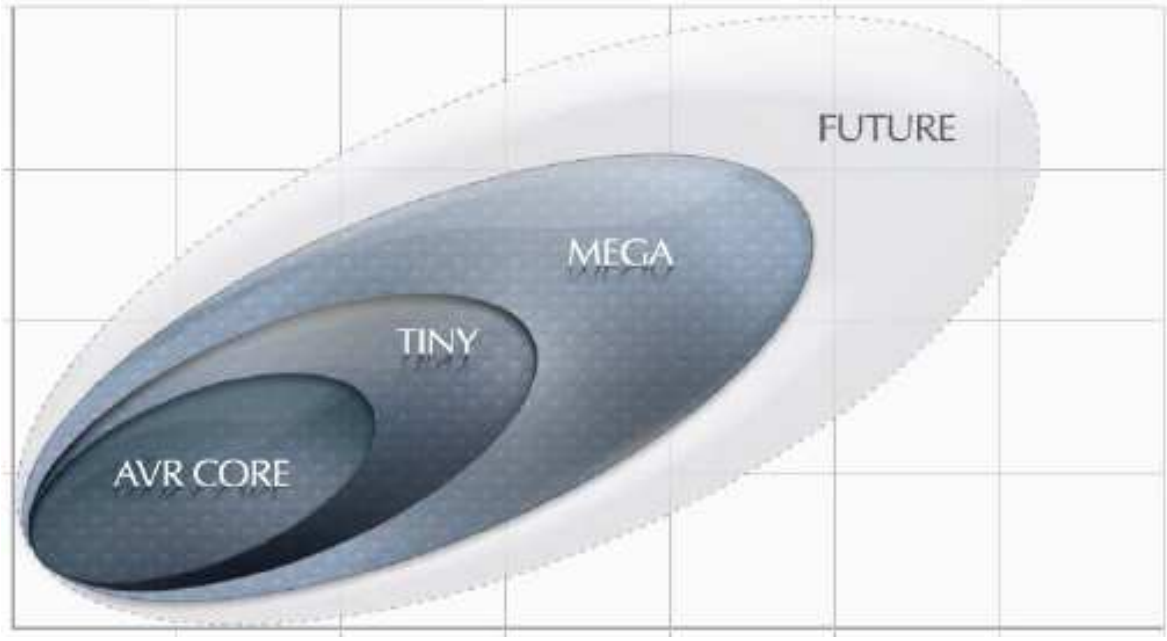


Gráfico 1-38. Evolución de la Familia de productos con tecnología AVR

TinyAVR

Gráfico 1-39. Figura de un microprocesador Tiny

Estos microcontroladores son de propósito general tienen hasta 4 K bytes de memoria flash para programa, 128 bytes de memoria SRAM, conversores ADC de 10 bits, 128 bytes en memoria EEPROM y son los más económicos.

Esta familia fue diseñada para aplicaciones específicas y que demandan muy pocos recursos.

MegaAVR

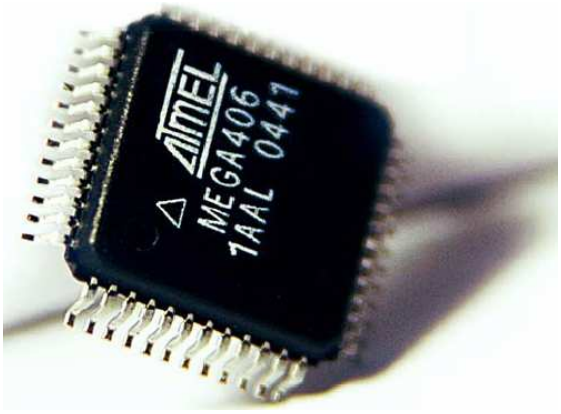


Gráfico 1-40. Figura de un microcontrolador Mega

Estos microcontroladores tienen la característica de auto programación lo cual permite una reprogramación remota sin circuito adicional para la grabación de la memoria de programa, tienen hasta 256 K bytes de memoria flash para programa, 4 K bytes de memoria SRAM y 4 K bytes de memoria EEPROM.

Pueden tener hasta 8 canales multiplexados con un ADC de 10 bits.

Estos microcontroladores fueron diseñados para aplicaciones que requieren gran cantidad de código de programa.

CAPÍTULO 2 DISEÑO Y CONSTRUCCIÓN DEL EQUIPO

El equipo solicitado por el Instituto Geofísico es una solución alternativa en su continua labor de estudiar los fenómenos sísmicos y volcánicos. La idea nace para satisfacer la necesidad de monitorear la actividad sísmica en lugares de difícil acceso y en los cuales un enlace por radio sería muy costoso o no factible.

Actualmente todos los equipos de monitoreo instalados utilizan un enlace de radio a tiempo real, por el cual se envían los datos registrados por los sensores sísmicos. Normalmente son usados equipos analógicos, los mismos que modulan la señal entregada por el sensor para que pueda ser transmitida al Instituto Geofísico, en donde se recupera la señal sísmica y se la grafica en tambores giratorios, los cuales tienen una capacidad de almacenamiento máxima de un día por cada lámina, las mismas que son almacenadas para llevar un historial del monitoreo.

Debido a que se debe prescindir del enlace de radio que permite vincular una estación con el Instituto Geofísico, el equipo debe ser capaz de: almacenar una gran cantidad de información en una memoria portátil, sincronizar los datos con la hora global, e implementar un control de voltaje en la fuente de alimentación.

En este capítulo se definen los requerimientos de diseño definidos por el Instituto Geofísico y el sistema diseñado que permita satisfacerlos.

2.1 REQUISITOS DE DISEÑO

El equipo debe ser capaz de:

- Admitir tres señales analógicas bipolares provenientes de un sensor sísmico (GEODEVICE FBS-3B) en un rango de ±2.5V máximo, con una frecuencia máxima de respuesta de 40Hz.

- Amplificar con ganancias variables por igual a las tres señales analógicas de entrada.

- Mantener un sincronismo de los datos adquiridos con la hora global, para que en el análisis de los datos se pueda correlacionar la información de varias estaciones.

- Registrar su posición global para identificar la fuente de las señales obtenidas.

- Almacenar los datos adquiridos en un dispositivo de almacenamiento masivo.

- Implementar una comunicación hacia un computador para una visualización de las señales adquiridas, con el objetivo de facilitar una calibración del equipo.

- Controlar el consumo de energía en caso de detectar que el nivel de voltaje en las baterías es menor a un nivel crítico, y reestablecer el sistema cuando se alcance un nivel óptimo de voltaje.

Además el equipo debe ser portátil, de fácil instalación y manejo sencillo para su aplicación en el campo.

2.2 JUSTIFICACIÓN DE LOS DISPOSITIVOS EMPLEADOS

Para poder adquirir tres señales analógicas bipolares de baja frecuencia se puede utilizar un conversor tipo Sigma–Delta por su precisión en la cuantificación de los datos a baja frecuencia, en un rango mínimo de 0 a 50Hz. Este conversor debe manejar por lo menos tres canales de entrada multiplexados con un rango dinámico de 5V cada uno; además permitir insertar una etapa de amplificación

previa a la conversión de analógico a digital. Por lo que el dispositivo elegido para esta función es el AD7738.

Debido a que el conversor escogido no puede trabajar con voltajes bipolares, se acondicionan las señales del sensor sumándoles un voltaje DC de 2.5V a cada componente. Obligando a que la etapa de amplificación variable también mantenga la misma referencia desplazada.

El sistema debe incluir un dispositivo GPS para la sincronización del equipo con la hora global y para determinar su posición en el planeta. Se escogió el Motorola M12+ debido a su tamaño y bajo costo comercial. Para suplir una falencia encontrada en un pulso por segundo generado por este GPS, se utiliza el RTC[20] DS1307 el cual se sincroniza con la hora del M12+ llevando una cuenta paralela del tiempo y generando una señal cuadrada confiable a 1Hz.

El dispositivo de almacenamiento masivo escogido fue una memoria flash con acceso USB, debido a su creciente popularidad, su alta velocidad de acceso, su portabilidad, su característica plug&play, y la alta capacidad de almacenamiento de datos.

Para el control de la memoria se necesita un dispositivo que implemente las funcionalidades de Host dentro del protocolo USB, debido a que la memoria tiene una naturaleza de Device. Para esto se eligió el controlador AT43USB380.

El microcontrolador principal elegido es el ATmega128L. El cual se encarga de establecer una comunicación con el controlador USB, el módulo GPS y el RTC antes mencionados, y una comunicación con el computador mediante el pórtico serial.

Debido a que la grabación en la memoria flash conlleva un tiempo máximo de 400ms por cada 600 bytes grabados, la adquisición de datos no puede ser

[20] RTC: *Real Time Clock*: Reloj a tiempo real. Es un circuito integrado diseñado para llevar la cuenta en segundos, minutos, horas, días, meses y años.

manejada por el ATmega128L, por lo que para cumplir con la función de recolección de datos, realizada a 100Hz, se utiliza el microcontrolador ATmega16L, el cual está conectado como un periférico mas del microcontrolador principal.

Tomando como referencia la aplicación se designó un nombre para el sistema completo: GEODAS_USB (GEO = Tierra, DAS = Data Acquisition System, USB = Protocolo en el Dispositivo de Almacenamiento Masivo).

2.3 DESCRIPCIÓN DE LOS DISPOSITIVOS EMPLEADOS

A continuación se detallan las características relevantes de los dispositivos empleados en la aplicación.

2.3.1 AT43USB380

Este controlador trabaja a un voltaje nominal de 3.3 V con un cristal de 6 MHz. Puede ser configurado en modo Host USB 2.0 sólo para manejar dispositivos Full/Low speed. Internamente contiene dos microcontroladores; uno se encarga de establecer el protocolo USB realizando tareas como: el proceso de enumeración, el manejo de cada tipo de transferencia (Bulk, Interrupción, Control e Isócrona), la detección de errores y la retransmisión de datos para endpoints no isócronos. El segundo microcontrolador está encargado de la comunicación entre el AT43USB380 y el microcontrolador que lo maneja; utilizando para la transacción de datos un bus variable entre 8/16/32 bits.

El AT43USB380 tiene una memoria RAM interna para datos tipo FIFO[21] de 1K para Transmisión y 1K para Recepción, a la cual se accede enviando/leyendo datos hacia/desde una dirección especial del mapa de registros.

[21] FIFO: Fist In – First Out: Primero en entrar, primero en salir.

Hay que tomar muy en cuenta que los microcontroladores internos tienen como memoria de programa una RAM, la cual debe ser grabada con los datos de sus respectivo Firmware[22]. Este proceso se lo hace cada vez que se inicializa el controlador también utilizando la dirección de la memoria FIFO.

A continuación se puede ver la distribución de pines del AT43USB380.

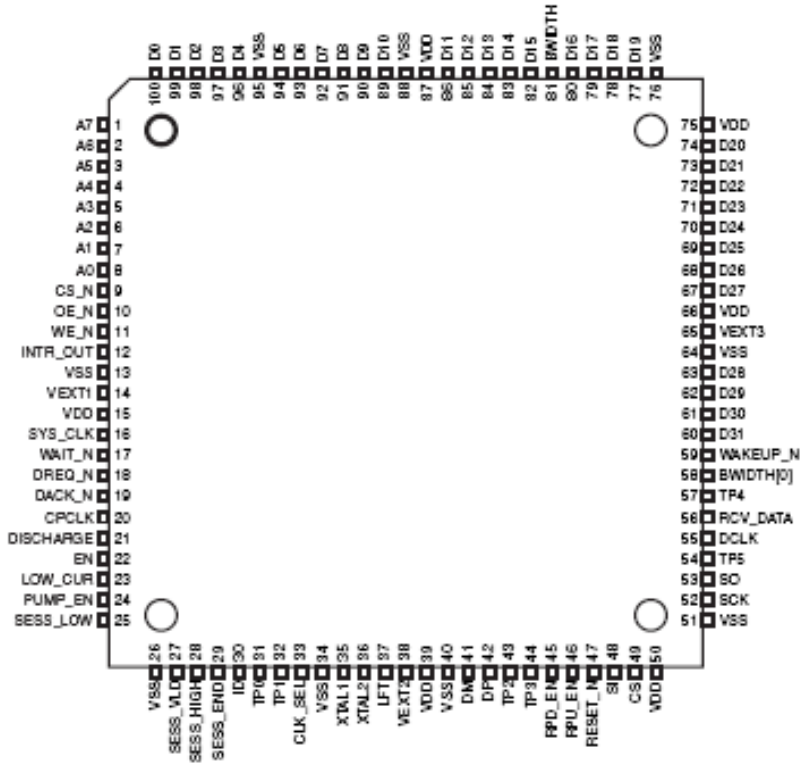


Gráfico 2-1. Distribución de pines del AT43USB380. [6]

2.3.1 ATMEGA128L

Este microcontrolador puede trabajar en el rango de voltaje desde 2.7 hasta 5.5 V y con un cristal de hasta 8 MHz. Tiene un encapsulado de 64 pines de los cuales 53 son programables como I/O (entrada/salida). Posee ocho interrupciones

[22] Firmware: es un programa utilizado para una función específica, el cual funciona como intermediario entre comandos externos a un dispositivo y el hardware que lo ejecuta.

externas, dos puertos USART[23] configurables, una memoria de programa de 128 Kbytes y una memoria RAM interna de 4Kbytes.

El programa es descargado al ATmega128L mediante una interfaz SPI utilizando la característica conocida como *In-System Programming*, la cual permite modificar el programa cuando el microcontrolador ya se encuentra soldado en placa.

A continuación se observa la distribución de pines para el ATmega128L.

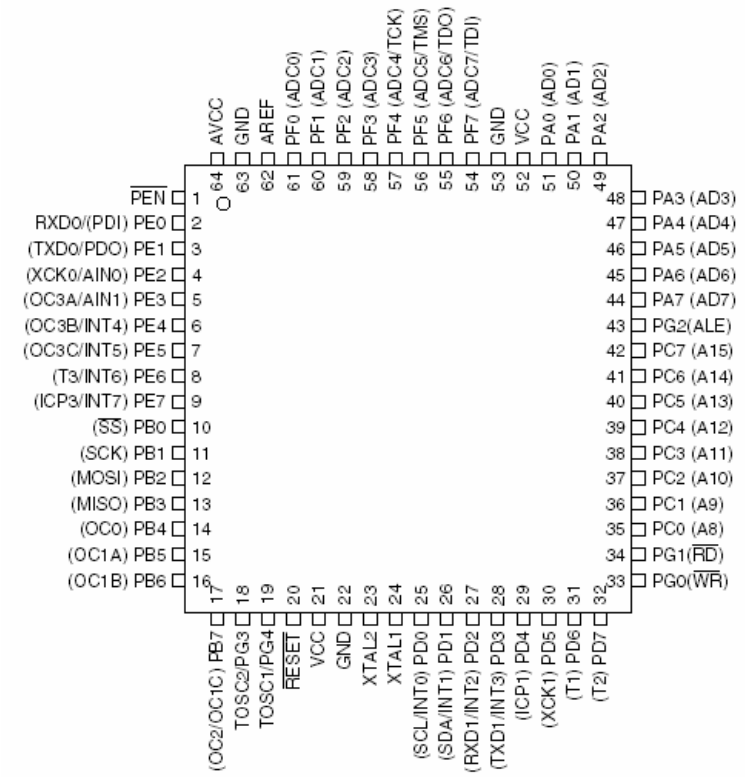


Gráfico 2-2. Distribución de pines del ATmega128L. [7]

2.3.2 ATMEGA16L

Este microcontrolador tiene características similares a las descritas para el ATmega128L, pero con algunos recursos más limitados, como son: la memoria de programa es de 16 Kbytes, la memoria RAM interna es de 1 Kbytes, solo tres

[23] USART: Universal Synchronic Asynchrony Reception Transmition: Puerto universal para transmisión/recepción sincrónica y asincrónica.

interrupciones externas, un puerto USART y solo 32 líneas programables como I/O de sus 44 pines.

A continuación se observa la distribución de pines para el ATmega16L.

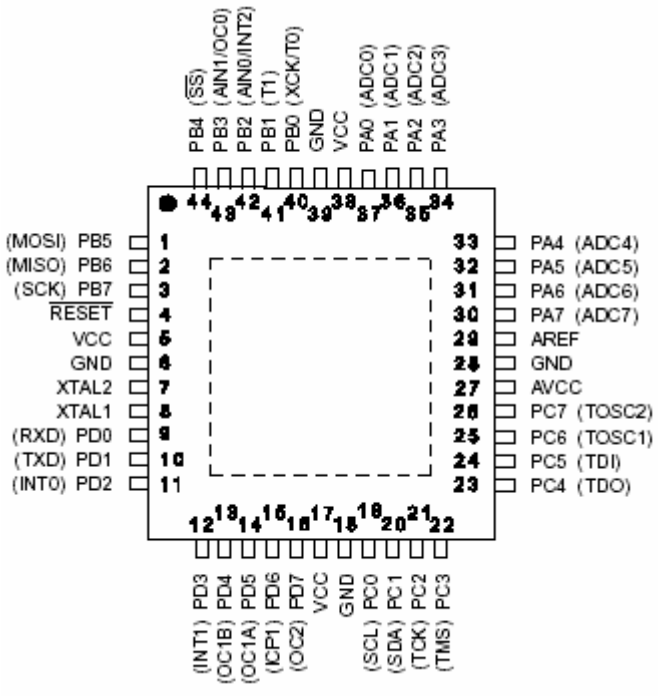


Gráfico 2-3. Distribución de pines del ATmega16L. [8]

2.3.3 MOTOROLA M12+

Este dispositivo GPS trabaja en un rango de voltaje que va de 2.85 a 3.15 V. Una de sus principales características es que puede manejar 12 canales en paralelo para la recepción de señales desde los satélites, con lo cual brinda una precisión de máximo 25 metros en un posicionamiento en tres dimensiones.

Tiene un puerto de comunicación serial a 3V TTL, por el cual se envían comandos de configuración y lectura de datos.

A continuación se presenta el diagrama de bloques interno del Motorola M12+.

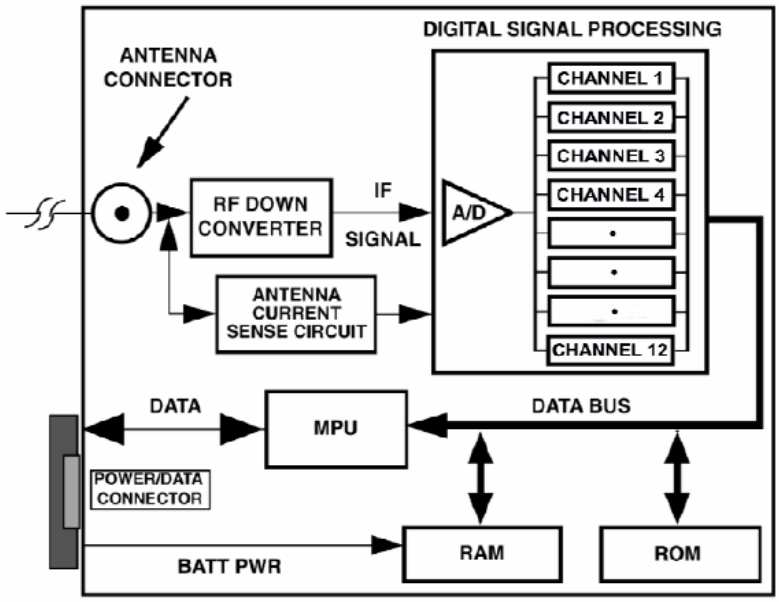


Gráfico 2-4. Diagrama de bloques interno del Motorola M12+.

El M12+ puede recibir señales desde 12 satélites simultáneamente, desde la antena se recibe una señal en la banda de frecuencia L1 (1575.42 MHz) de la cual se extrae la señal de datos en una frecuencia más baja, la cual es insertada en un conversor A/D de alta velocidad para luego ser mandada en paralelo a los 12 canales para la respectiva detección de las señales, una correlación de código y un seguimiento de la portadora.

Internamente el M12+ tiene un microprocesador, el cual se encarga de decodificar los datos enviados por los satélites para el cálculo de la posición y tiempo. También se encarga de implementar un protocolo binario serial con una velocidad de transmisión de 9600 bps con un bit de inicio, ocho bits de datos y un bit de parada.

La antena que utiliza tiene una impedancia de entrada de 50 Ω y está diseñada para el trabajo en la frecuencia L1 (1575.42 MHz, +/- 1.02 MHz) con un voltaje de polarización de 3 V. La ganancia tomando en cuenta la atenuación sufrida en el cable es de 24dB.

Para la comunicación con una aplicación externa, el M12+ tiene un conector de diez pines distribuidos de la siguiente forma:

Pin #	Signal Name	Description
1	TxD1	Transmit Data (3V logic)
2	RxD1	Receive Commands (3V logic)
3	+3V PWR	Regulated 3Vdc Input
4	1PPS	1 pulse-per-second output
5	Ground	Signal and Power common
6	Battery	Optional External Backup
7	Reserved	Not currently used
8	RTCM In	RTCM correction input
9	Antenna Bias	3V-5V antenna bias input
10	Reserved	Not currently used

Gráfico 2-5. Distribución de pines del conector de datos del Motorola M12+. [9]

El M12+ utiliza el protocolo binario de Motorola el cual se organiza de la siguiente manera:

Primero tiene dos caracteres de inicio de trama: 40h, 40h (@,@). Luego tiene un identificador de comando que ocupa dos bytes. A continuación está el campo de datos que puede tener desde uno hasta 150 bytes dependiendo del comando ejecutado. Inmediatamente luego del campo de datos se tiene un byte como campo de verificación de la integridad de los datos, el cual consiste en una XOR lógica entre cada byte después del identificador de inicio de trama. El mensaje es finalizado por dos bytes 0Dh y 0Ah (CR y LF respectivamente).

Para que un mensaje sea considerado válido se observa que luego de recibido el inicio de trama (@@) la longitud del mensaje esté de acuerdo al identificador de comando, que el campo de CheckSum sea correcto y que el mensaje termine en los bytes CR y LF. En caso de no recibir algún campo correcto el mensaje es descartado.

El comando SET TO DEFAULTS COMMAND es un reset que se le da al M12+ para poner todos sus parámetros a los valores por defecto, borrando así toda

información sobre posición, tiempo y fecha. Este comando está compuesto de la siguiente forma: @@CfC<CR><LF>.

Existe otro comando que permite leer la información completa desde el M12+, éste es el ASCII Position Message. La solicitud de datos se la realiza al enviar este comando: @@EqmC<CR><LF>, donde la m representa cada cuantos segundos el M12+ enviará la respuesta; si m vale 00h solo lo hará una vez.

El comando de respuesta enviado por el M12+ tiene 96 bytes con los siguientes campos:
@@Eq,mm,dd,yy,hh,mm,ss,dd,mm.mmmm,n,ddd,mm.mmmm,w,shhhh.h,sss.s,h,m,t,dd.d,nn,rrrr,aa,CC <CR><LF>; donde,

Fecha		
	mm=Mes	1..12
	dd=Dia	1..31
	yy=Año	98..18 (full date = 1998..2018)
Tiempo UTC		
	hh=horas	0..23
	mm=minutos	00..59
	ss=segundos	00..59
Latitud		
	dd=grados	000..180
	mm.mmmm=minutos	00..59.9999
	n=dirección	N = Norte, S = Sur
Longitud		
	ddd=grados	000..180
	mm.mmmm=minutos	00..59.9999
	w=dirección	W = Oeste, E = Este
Altura		
	s=signo	+ or -
	hhhh.h=altura en metros	-1000.0..18,000.0
Velocidad		

	sss.s=velocidad en nudos	000.0..999.9
	hhh.h=grados	000.0..359.9
Estatus Receptor		
	m=modo de corrección	0 = autónomo
		1 = diferencial
	t=tipo de corrección	0=no corrección
		1=corrección 2D
		2=corrección 3d
		3=Modo de propagación
	dd.d=pérdida de precisión	00.0…99.9
	nn=número de satélites en uso	00..37
	rrrr= ID de estación	0000..1023
	aa=tiempo del dato diferencial	00..90
	CCC=CheckSum	000 .. 255

2.3.4 DS1307

Este integrado fue diseñado para contabilizar el segundo, el minuto, la hora, el día, el mes y el año; y es capaz de contabilizar incluso años bisiestos. Utiliza un cristal externo de 32768 Hz, e internamente se divide la frecuencia hasta alcanzar una frecuencia de 1 Hz.

Para poder igualar este reloj se accede a la memoria RAM del RTC por medio de una interfaz I2C[24]. La RAM está mapeada de la siguiente manera:

ADDRESS	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	FUNCTION	RANGE
00H	CH	10 Seconds			Seconds				Seconds	00–59
01H	0	10 Minutes			Minutes				Minutes	00–59
02H	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM
		24	PM/AM							00–23
03H	0	0	0	0	0	DAY			Day	01–07
04H	0	0	10 Date		Date				Date	01–31
05H	0	0	0	10 Month	Month				Month	01–12
06H	10 Year				Year				Year	00–99
07H	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08H-3FH									RAM 56 x 8	00H–FFH

Gráfico 2-6. Mapa de la memoria RAM del DS1307. [10]

[24] I2C: es un bus de comunicaciones serie. Su principal característica es que sólo utiliza dos líneas para la comunicación. Una línea de datos y otra para la señal de reloj que es generada por el maestro.

El bit CH (*Clock Halt*) de la dirección 00h, controla el accionar del oscilador local, con 0L lo activa. Cada dirección contiene datos referentes a cada campo en la fecha u hora, los cuales están representados en un formato BCD[25].

Los bits RS1 y RS0 seleccionan la frecuencia a la cual oscila el pin SQW (*Square Wave*), el cual es habilitado por el bit SQWE (*Square Wave Enable*). En la siguiente tabla se observa las frecuencias permitidas de oscilación.

RS1	RS0	SQUARE-WAVE OUTPUT FREQUENCY
0	0	1Hz
0	1	4.096kHz
1	0	8.192kHz
1	1	32.768kHz

Tabla 2-1. Tabla de frecuencias de oscilación en el pin SQW. [10]

En el gráfico siguiente se observan la distribución de pines del DS1307.

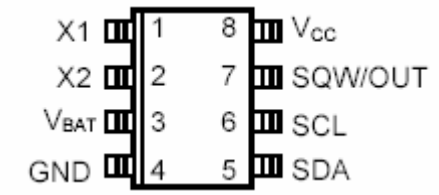


Gráfico 2-7. Distribución de pines del DS1307. [10]

2.3.5 AD7738

Este integrado es un conversor Análogo–Digital del tipo Sigma–Delta de 16 o 24 bits configurables. Trabaja con dos rangos de voltajes, uno para su sección digital que va de 2.7 a 3.6 V; y, para la sección analógica el rango es de 4.75 a 5.25 V.

A continuación se puede ver el diagrama con los bloques internos del AD7738.

[25] BCD: Binary-Coded Decimal: es un sistema numérico usado para codificar números enteros positivos, en donde cada cifra representa un digito decimal (0,1,2…..9).

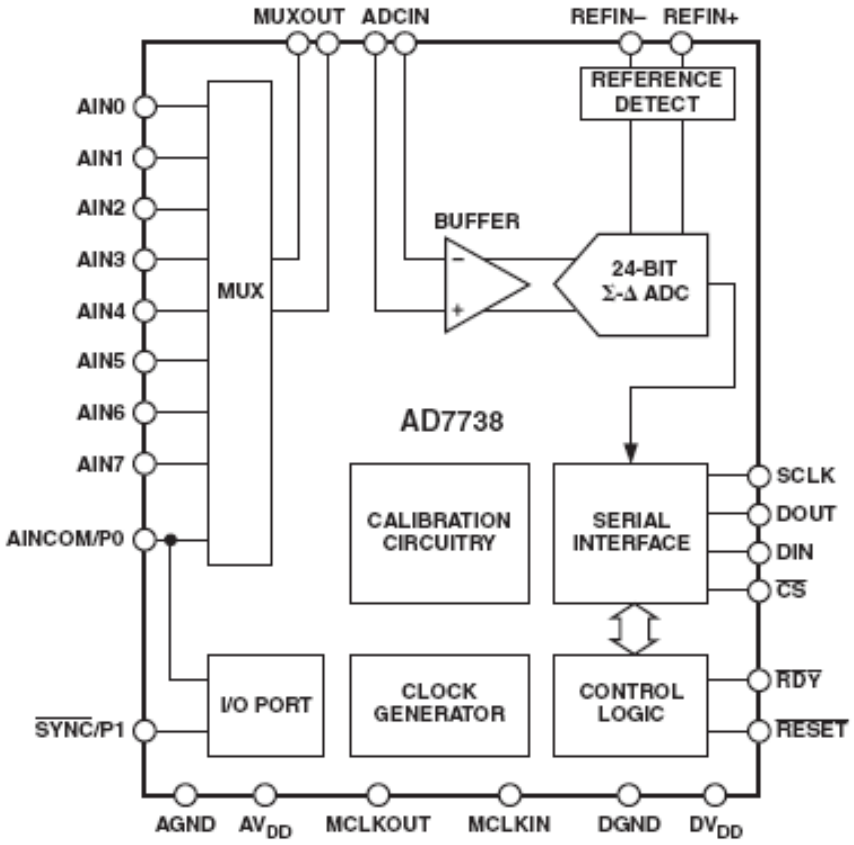


Gráfico 2-8. Diagrama de bloques interno del AD7738. [11]

Este conversor tiene ocho entradas, con una sola referencia negativa, conectadas a un multiplexor de alta velocidad. La salida de este multiplexor es accesible externamente por medio de dos pines (MUXOUT), lo cual permite acondicionar todas las señales multiplexadas, éstas son ingresadas de nuevo al conversor por medio de dos pines (ADCIN).

El AD7738 tiene una interfaz de comunicación SPI (*Serial Peripheral Interface*), por medio de la cual se accede a los registros de configuración y de datos, los cuales son independientes para cada canal. Dentro de los registros de configuración se puede modificar los rangos de voltaje (+625 mV, +1.25 V, +2.5 V, ±625 mV, ±1.25 V, ±2.5 V), cantidad de bits para cada conversión (16, 24 bits), la velocidad de conversión (tiempo de conversión) y el tipo de conversión (continua o simple).

Con una conversión continua, el AD7738 envía a 0L el pin RDY (*Ready*) indicando que los datos, de todos los canales habilitados, están actualizados. Al contrario en una conversión simple, el AD7738 sólo actualiza los datos del canal solicitado con el mismo pin RDY.

La siguiente expresión muestra cómo calcular el tiempo de conversión de un canal en el AD7738, donde se involucra la frecuencia del cristal usado y el valor del registro de configuración (FW).

$$Tiempo\ de\ Conversión\ (\mu s) = \left(FW\ x\ 64 + 206\right) / \ Frecuencia\ cristal\ (MHz) \quad \textbf{Ec. 2-1}$$

A continuación se presenta la distribución de pines del AD7738.

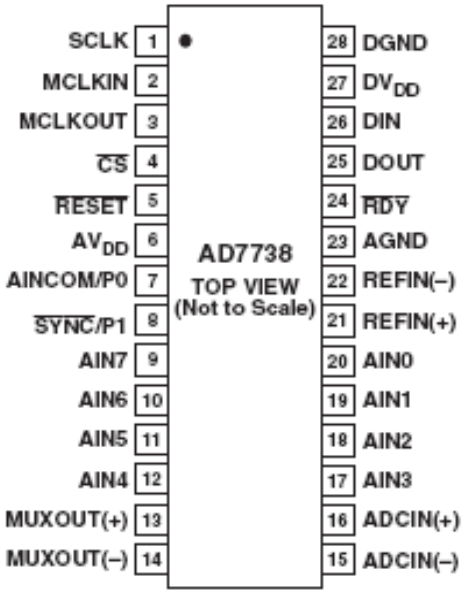


Gráfico 2-9. Distribución de pines del AD7738. [11]

2.4 DISEÑO DE HARDWARE

A continuación se muestra un diagrama de bloques del sistema GEODAS_USB, donde se especifica las configuraciones relevantes en cada dispositivo y el tipo de interfaz de comunicación entre cada uno de los dispositivos.

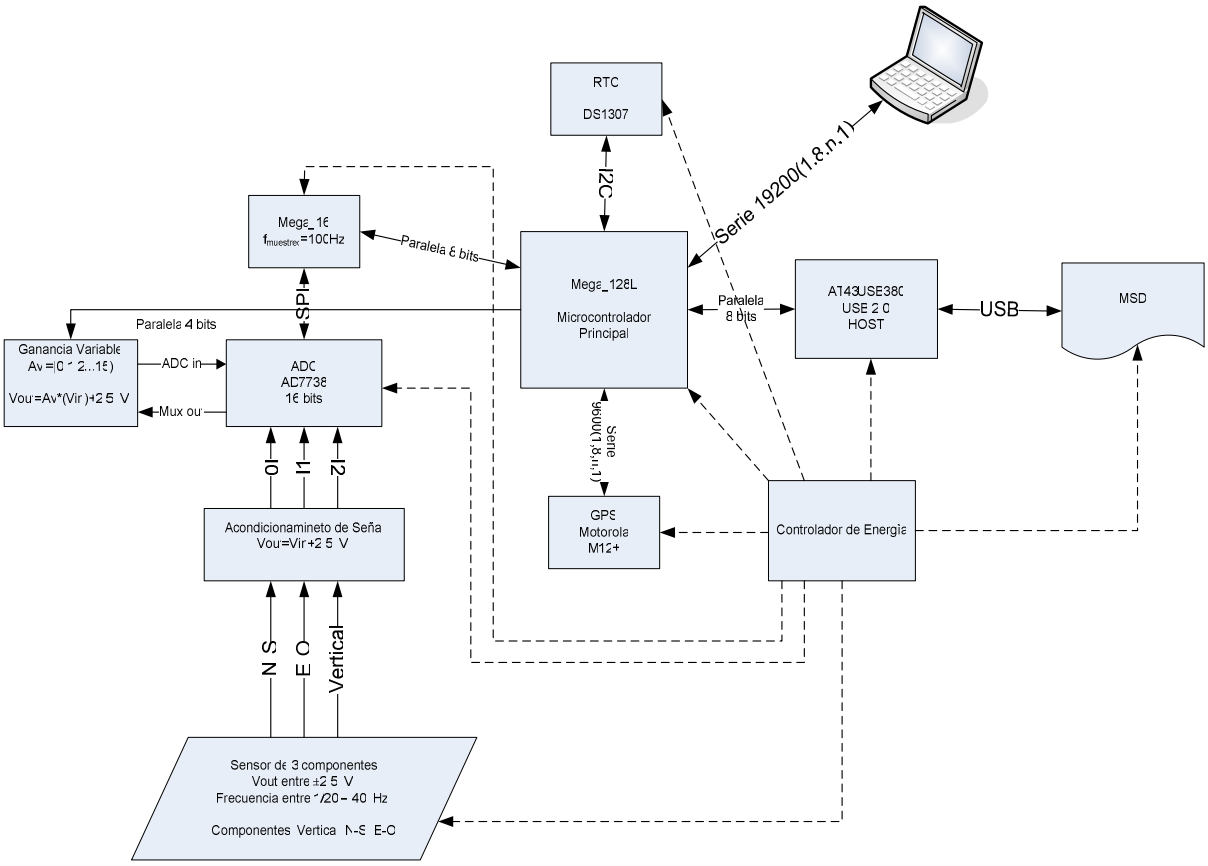


Gráfico 2-10. Diagrama de bloques del sistema GEODAS_USB.

El bloque Sensor de Tres Componentes representa al sismómetro empleado como la fuente de la información sísmica. Sus características eléctricas de salida para cada componente son: un voltaje de salida pico - pico de ±2.5 V en el cual la respuesta del sensor es lineal, sus frecuencias límites de respuesta son 1/20 Hz y 40 Hz.

Como se puede apreciar en el diagrama, el bloque Sensor de Tres Componentes genera las señales de entrada al bloque Acondicionamiento de Señal, el cual tiene la función de desplazar a las señales de entrada desde un nivel de 0V hasta 2.5V. Para esto se utiliza un desplazador de voltaje por cada componente configurado en base a un amplificador operacional de la siguiente manera:

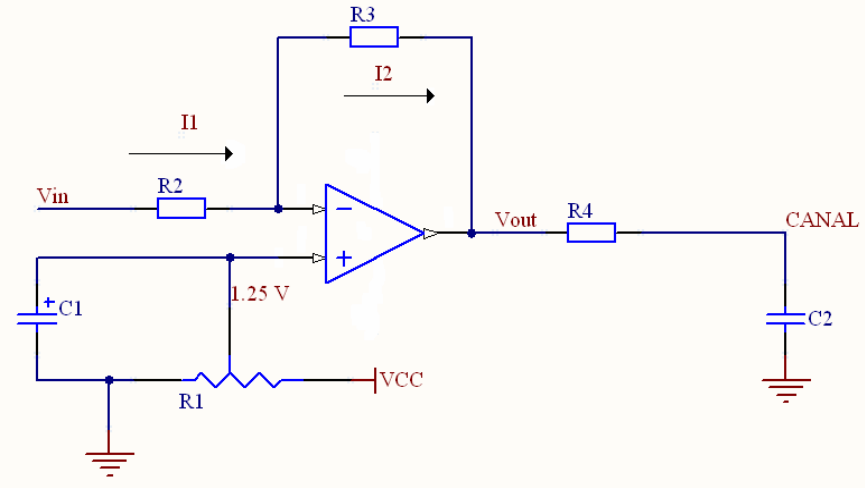


Gráfico 2-11. Circuito desplazador de voltaje.

Donde,

$$Vdiferencial \approx 0V \qquad \Rightarrow I_1 = \frac{V_{in} - 1.25V}{R_2} \qquad \Lambda \qquad I_2 = -\frac{V_{out} - 1.25V}{R_3}$$

$$I_1 = I_2 \qquad \Rightarrow \frac{V_{in} - 1.25V}{R_2} = -\frac{V_{out} - 1.25V}{R_3}$$

$$\text{Si } R_2 = R_3 \qquad \underline{\Rightarrow V_{out} = -V_{in} + 2.5V}$$

A la salida del operacional se tiene un filtro pasivo pasa bajos con una frecuencia de corte de 50Hz debido a que la frecuencia de muestreo para cada componente es de 100Hz.

Donde,

$$f_C = \frac{1}{2\pi R_4 C_2} = 50Hz$$

$$\text{Si } \underline{C_2 = 2.2\mu F} \qquad \Rightarrow R_4 = \frac{1}{2\pi f_C C_2} = 1446.86\Omega$$

$$\underline{\Rightarrow R_4 = 1.1K\Omega + 330\Omega} \qquad \underline{\therefore fc = 50.59\,Hz}$$

Como salidas del bloque Acondicionamiento de Señal se tienen los canales 0,1 y 2, los cuales ingresan al bloque ADC. Internamente el conversor AD7738 tiene un

multiplexor analógico de ocho canales (sólo se usan tres canales). Las señales multiplexadas son accesibles por medio de los pines MUXOUT en donde se incorpora el bloque Ganancia Variable.

El bloque de ganancias tiene una configuración eléctrica que permite amplificar únicamente la componente alterna de la señal de entrada (Vin + 2.5V) sin distorsionarla. Se utiliza un amplificador de AC de ganancia variable en base a un amplificador operacional configurado de la siguiente manera:

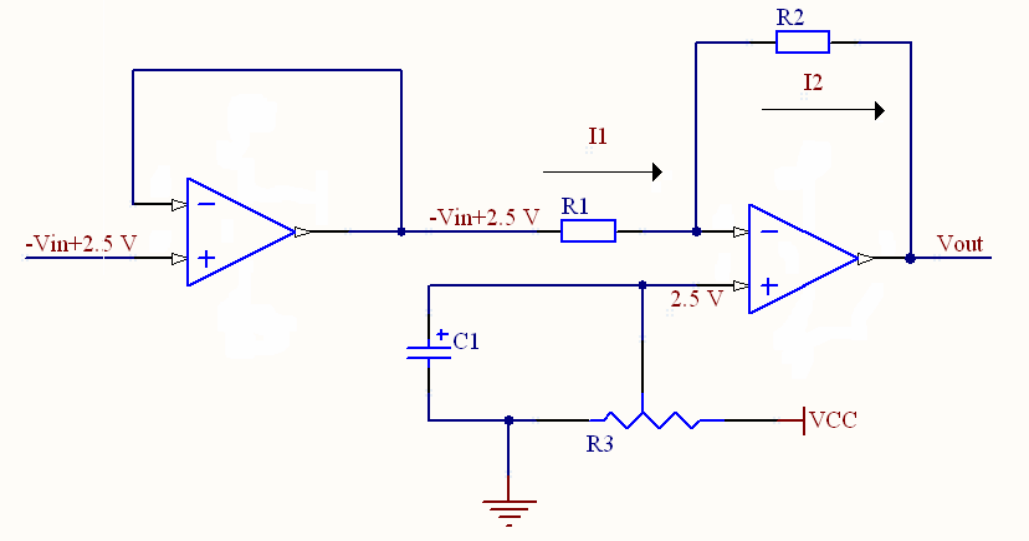


Gráfico 2-12. Circuito amplificador AC de ganancia variable.

Donde,

$$Vdiferencial \approx 0V \quad \Rightarrow I_1 = \frac{-V_{in} + 2.5V - 2.5V}{R_1} \quad \Lambda \quad I_2 = -\frac{V_{out} - 2.5V}{R_2}$$

$$I_1 = I_2 \quad \Rightarrow \frac{-V_{in} + 2.5V - 2.5V}{R_1} = -\frac{V_{out} - 2.5V}{R_2}$$

$$\Rightarrow \frac{-V_{in}}{R_1} = -\frac{V_{out}}{R_2} + \frac{2.5V}{R_2}$$

$$\Rightarrow V_{out} = V_{in}\frac{R_2}{R_1} + 2.5V$$

La resistencia R2 es un potenciómetro digital de 16 pasos, el mismo que está configurado de la siguiente manera:

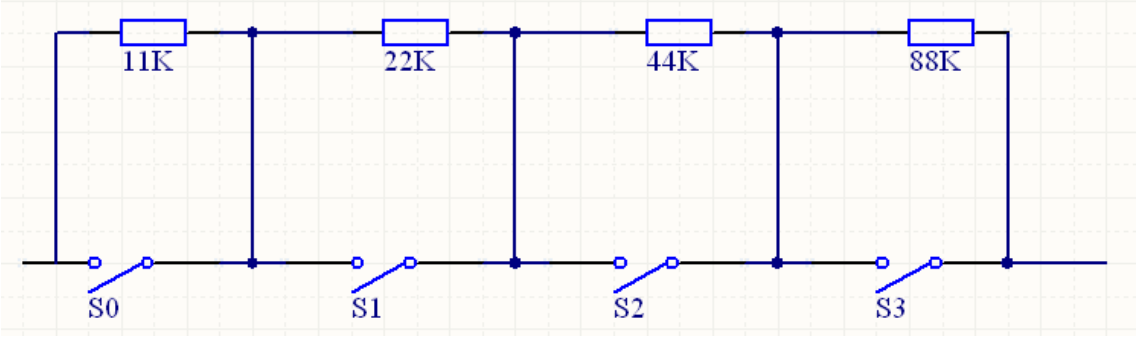


Gráfico 2-13. Potenciómetro digital de 16 pasos.

Para poder controlar la ganancia desde el microcontrolador principal (ATmega128L) se emplea el integrado 4066, el cual contiene cuatro interruptores analógicos internamente.

Este bloque de Ganancia Variable envía la señal unipolar amplificada a los pines ADCIN, que son la entrada de conversor configurado con una ventada dinámica de ±2.5V (0-5V) y con una resolución de 16 bits.

El bloque ADC se comunica con el bloque Mega_16 mediante una interfaz SPI, donde el microcontrolador ATmega16 administra la comunicación. Las funciones del bloque Mega_16 son las de resetear y configurar al AD7738, además se encarga de pedir datos de los tres canales cada 10ms almacenándolos en su memoria RAM para luego enviarlos al microcontrolador ATmega128L cuando éste lo solicite, utilizando una comunicación paralela de ocho bits.

El bloque Mega_128 contiene al microcontrolador principal (ATmega128L) el cual administra todo los dispositivos del sistema con los que se encuentra vinculado directamente, como son: el GPS Motorola M12+, el RTC DS1307, al controlador AT43USB380, y los bloques de Ganancia Variable y Mega_16.

Además el ATmega128L es capaz de comunicarse con un puerto serial de un computador, el cual selecciona las configuraciones abiertas al usuario como es el

nivel de ganancia variable y si los datos adquiridos son enviados por el puerto serial o grabados en la memoria MSD[26] por medio del controlador AT43USB380.

El bloque Controlador de Energía monitorea el voltaje de la fuente principal y se encarga de la conexión y desconexión de todos los bloques vinculados al mismo. El circuito que permite implementar esta función se detalla a continuación:

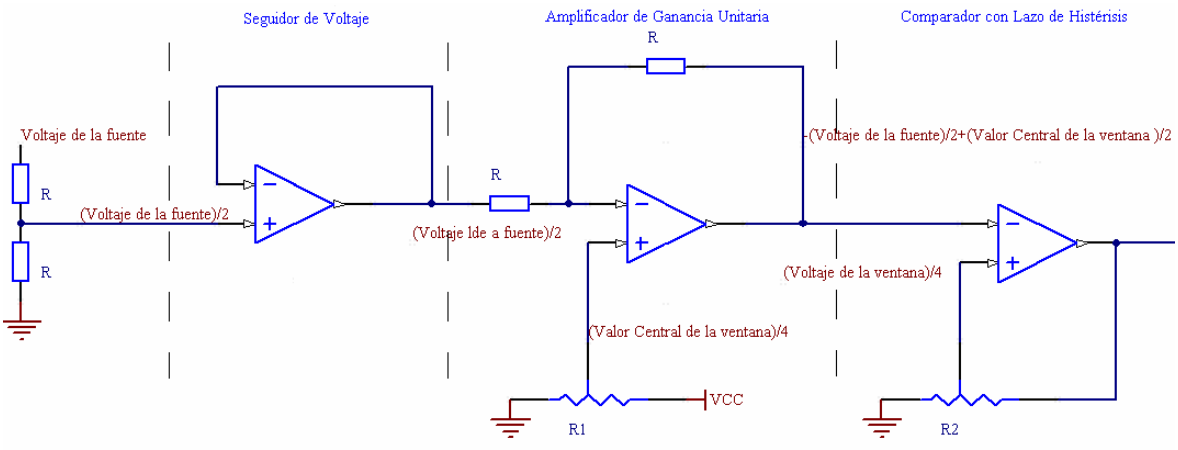


Gráfico 2-14. Circuito controlador de energía.

Se toma una muestra del voltaje de la fuente por medio de un divisor de tensión (Voltaje de la fuente/2), y para aislar las impedancias con la etapa siguiente se coloca un seguidor de voltaje.

A continuación con un amplificador de ganancia unitaria y un potenciómetro R1 calibrado a (Valor Central de la ventana)/4 se obtiene un valor diferencial entre el Voltaje entregado por la fuente y el voltaje central de la ventana deseada (-(Voltaje de la fuente)/2 + (Voltaje Central de la Ventana)/2).

Este voltaje diferencial entra a un comparador con lazo de histéresis, en donde el potenciómetro R2 regula el tamaño de la ventana. Esta etapa entrega valores lógicos al microcontrolador ATmega128L, el cual al reconocer una descarga del equipo (-Vcc=0L) se prepara para la desconexión.

[26] MSD: Mass Storage Device: es un medio de almacenamiento de gran capacidad, que es más barato que un disco duro o cualquier unidad de almacenamiento primario.

La misma señal para la conexión o desconexión llega a un relé pasando por una etapa de retardo, calculado a unos cinco segundos, lo cual permite que no exista actividad en el dispositivo USB, y se pueda desconectar a todo el equipo sin problema.

A continuación se tienen el circuito de la etapa de retardo de la señal, el cual consiste en una carga y descarga de un capacitor (malla RC) como una entrada a un comparador, entregando una señal para saturar o abrir un transistor el cual controla al relé.

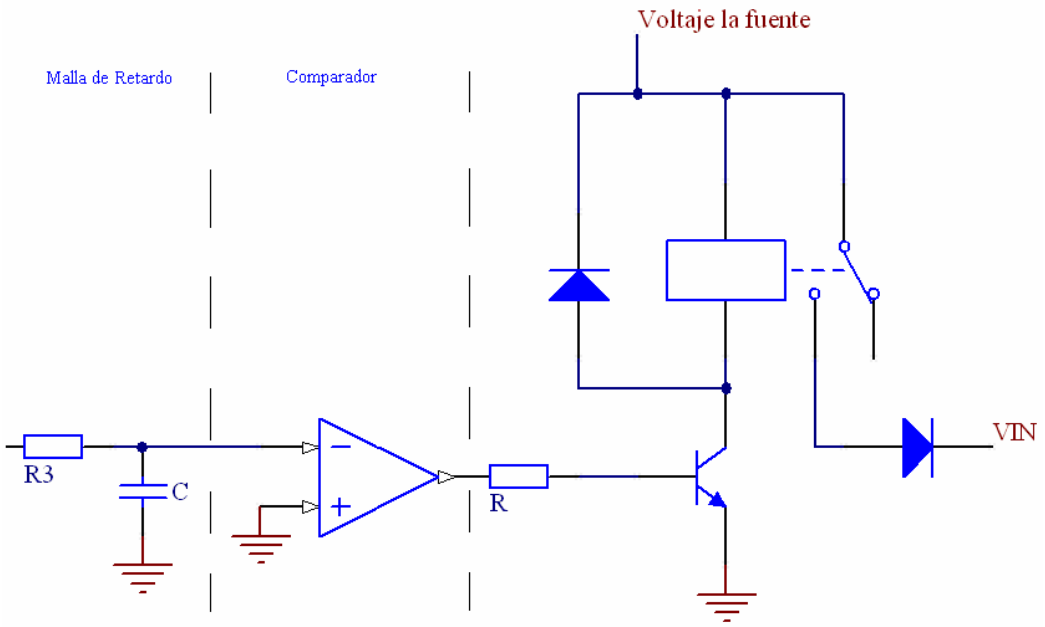


Gráfico 2-15. Circuito controlador de relé.

Hay que aclarar que este sistema no desconecta la fuente de los operacionales, permitiendo mantener el control sobre el voltaje de la fuente, para reiniciar el equipo cuando el voltaje sea el adecuado.

2.4.1 ESQUEMÁTICOS

Todos los esquemáticos fueron diseñados en Protel DXP 2004, y clasificados en varias hojas según los elementos principales involucrados en el equipo.

En el Anexo A se encuentran los siguientes esquemáticos correspondientes al equipo GEODAS_USB:

- ATmega128L.SchDoc.- Microcontrolador ATmega128L, conector grabador, RTC DS1307, conector GPS M12+, leds indicadores.
- ATmega16L.SchDoc.- Microcontrolador ATmega16L, conector grabador.
- AT43USB380.SchDoc.- Controlador USB AT43USB380, conector USB tipo A.
- ADC7738. SchDoc.- Conversor AD7738.
- Ganancia.SchDoc.- Filtros pasivos, circuito de ganancia variable.
- Fuentes.SchDoc.- Reguladores de voltaje de 8 V, 3.3 V, 3 V, 5 V y -8 V; circuito controlador de voltaje.

2.4.2 PCB

El circuito final o PCB fue diseñado en Protel DXP 2004 a partir de los esquemáticos del Anexo A.

En el Anexo B se puede ver las siguientes láminas del PCB:

- Capa superior.
- Capa inferior.
- Solo texto capa superior.
- Solo texto capa inferior.
- PCB grabador.
- PCB conector sensor.

2.4.3 PROTOTIPO

Como Anexo C se adjuntan fotos del equipo GEODAS_USB.

- Foto superior de la tarjeta.

- Foto inferior de la tarjeta.
- Equipo GEODAS_USB.

2.5 DISEÑO DE SOFTWARE

Haciendo referencia al Gráfico 2-10, el software desarrollado permite, por medio de librerías, generar las aplicaciones implementadas en cada uno de los dos microcontroladores escogidos (ATmega128L y ATmega16L).

En esta parte primero se detallan las librerías y sus métodos, para después explicar los programas principales en cada microcontrolador.

2.5.1 DESCRIPCIÓN DE LAS LIBRERÍAS

En este punto se detallan las funcionalidades de cada librería empleadas en el proyecto, mostrando un árbol detallando los métodos de acceso a cada una; y, el flujo de ciertos procesos que se desarrollan dentro de cada librería.

2.5.1.1 USB_MSD.h [1],[12]

Aquí se explica cómo se utilizan algunos métodos declarados en una librería pre-compilada creada por Atmel para el uso del controlador AT43USB380. Esta librería fue editada en GCC[27] y pre-compilada por el WinAVR[28], entregando como resultado una librería .a (USBP_StandAlone_lib.a).

A continuación se muestra un gráfico donde se puede observar un árbol de métodos, creados para vincular la librería .a con la aplicación.

[27] GCC: GNU C Compiler: son un conjunto de compiladores que funcionan en sistemas operativos de código abierto que trabajan en un lenguaje C.
[28] WinAVR: es una herramienta de código abierto para desarrollo utilizando los microcontroladores de la familia Atmel AVR. Este programa incluye un compilador GCC.

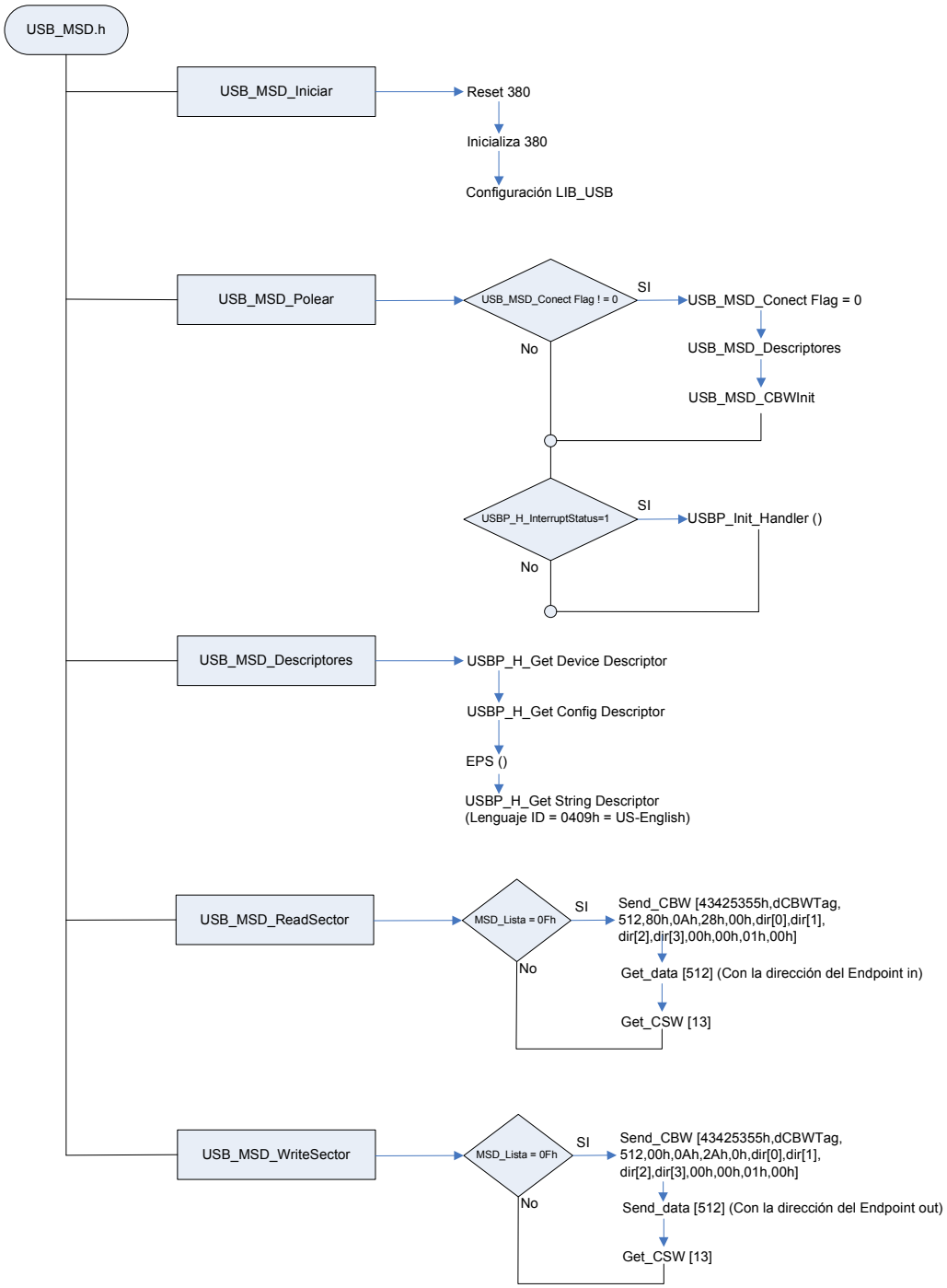


Gráfico 2-16. Árbol de métodos de la librería USB_MSD.h.

USB_MSD_Iniciar:

Esta función primero genera un pulso de Reset al AT43USB380 por medio de un pin (USB_MSD_Rst) controlado por el ATmega128L. A continuación se procede a la descarga de los firmwares para *Host*, por medio del FIFO con un bus de ocho bits.

El AT43USB380 responde con un flanco positivo en el pin INT0 del ATmega128L, indicando que el proceso culminó, el microcontrolador entonces debe verificar que el estatus de la descarga sea exitoso, de ser así continua con el programa.

Debido a que la librería USBP_StandAlone_lib.a es el único vínculo que existe entre la aplicación y el controlador USB, se deben tomar ciertas consideraciones especiales para su correcto uso, sobre todo cómo atender los eventos de interrupción generados por el AT43USB380 (INT0). En cada proceso de interrupción se invoca a la función USBPlib_Int_Handler(), la cual genera un respaldo de cada evento en el controlador USB y provoca que el estatus de la función USBPH_InterruptStatus() cambie y se llame al método USBP_Int_Handler(). Este método discrimina, de acuerdo al evento solicitado, qué método propietario debe ejecutarse.

Previamente se vinculan los métodos propietarios con ciertos eventos de interrupción, como por ejemplo Conexión o Desconexión de un dispositivo. Además la librería .a permite hacer un filtro a los tipos de dispositivos que serán reconocidos al ser conectados al controlador USB. En este filtro se consideran la Clase_USB (MSD), la SubClase_USB y el Protocolo_USB (Bulk).

USB_MSD_Polear:

Éste es un método especial que debe ser invocado continuamente en el programa principal. Dentro de este proceso se verifica si el estatus de las interrupciones ha cambiado para atender cada evento solicitado por el AT43USB380.

Dentro de este método también se monitorea si un dispositivo ha sido conectado, para invocar los métodos USB_MSD_Descriptorores y USB_MSD_CBWInit; dejando al dispositivo MSD listo para ser usado.

USB_MSD_Descriptores:

Por medio de este método, se configuran las características del dispositivo conectado. Primero se solicitan los descriptores tipo Device y Config en donde se obtiene la información de cuántos endpoints maneja el dispositivo, cuál de ellos es Endpoint IN y cuál es Endpoint OUT.

Luego se pide el descriptor tipo String con el ID de idioma 0409h (US-English). Como una información adicional del dispositivo conectado.

USB_MSD_ReadSector:

Este método permite acceder a un bloque de memoria del dispositivo MSD USB por medio del siguiente proceso. Primero se envía un comando CBW con los siguientes argumentos: [43425355h, dCBWTag, 512, 80h, 0Ah, CBWCB].

El campo CBWCB contiene el comando de lectura: [28h, 00h, dir[0], dir[1], dir[2], dir[3], 00h, #sectores, 00h], donde el primer byte identifica el tipo de comando. Los bytes identificados con la palabra dir[x] contienen la dirección del sector inicial para la lectura. Al final se tiene el campo #sectores que especifica la cantidad de sectores a ser leídos desde la dirección inicial. Los bytes segundo, séptimo y último son reservados y su valor es 00h.

A continuación viene la etapa de datos, en donde el Device envía tantos bytes como fueron solicitados. Y luego envía el comando CSW completando el proceso.

USB_MSD_WriteSector:

Este método permite grabar un bloque de memoria del dispositivo MSD USB mediante el siguiente proceso. El Host envía el comando CBW con los siguientes argumentos: [43425355h, dCBWTag, 512, 00h, 0Ah, CBWCB].

El campo CBWCB contiene el comando de escritura de datos el cual tiene como datos: [2Ah, 0h, dir[0], dir[1], dir[2], dir[3], 00h, #sectores, 00h]. El primer byte es el identificador del comando, de igual manera los bytes con el identificador dir[x]

contienen la dirección del sector inicial. Con el campo #sectores se indica cuántos sectores a partir del inicial serán grabados. El resto de bytes son reservados y su valor es 00h.

Luego el Host envía la cantidad de datos indicados y éste espera la respuesta del Device por medio del comando CSW.

2.5.1.2 FAT32.h [3]

Esta librería permite implementar el sistema de archivos FAT32 con dominio de nombres cortos, con operaciones orientadas al sector y no al cluster para permitir que esta librería pueda correr en el Atmega128L (4Kbytes en RAM).

A continuación se muestra un árbol de métodos incluidos en esta librería.

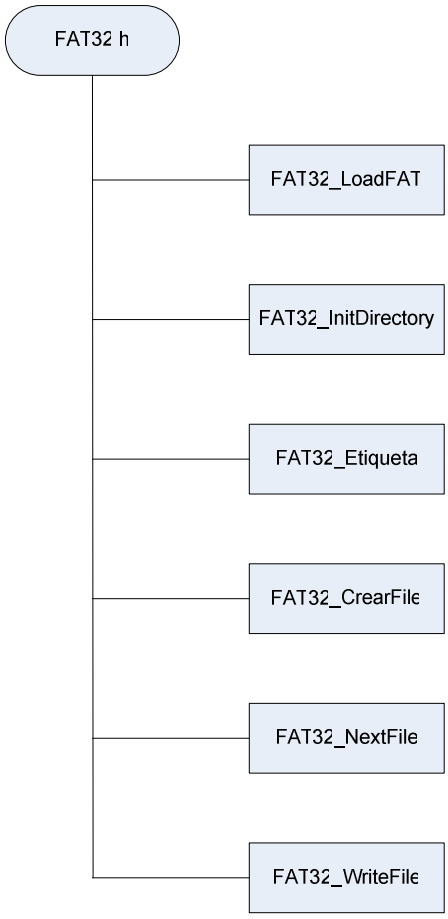


Gráfico 2-17. Árbol de métodos de la librería FAT32.h.

FAT32_LoadFAT:

Este método lee el sector cero para buscar el paquete asociado a la partición uno, donde se verifica que el campo Type Code sea 0Bh ó 0Ch que indican que el sistema de archivos es FAT32.

A continuación lee el Volumen ID de la partición uno, y se recuperan los Campos Críticos descritos en la Tabla1-12.

FAT32_InitDirectory:

Este método devuelve el primer archivo dentro del directorio raíz.

FAT32_NextFile:

Este método devuelve el siguiente archivo dentro del directorio raíz.

FAT32_Etiqueta:

Este método busca dentro de todas las entradas de archivo en el directorio raíz si existe una entrada con un atributo de volumen. Si la encuentra, la reemplaza con un texto que es un argumento de la función. De no ser así, busca una entrada de archivo vacía (00h) o libre (E5h), la modifica y crea la etiqueta de volumen.

FAT32_CrearFile:

Este método busca una entrada de archivo vacía (00h) o libre (E5h) en el directorio raíz, la modifica y crea el nuevo archivo, con una longitud inicial de cero.

FAT32_WriteFile:

Este método abre el último sector del último cluster del archivo y lo modifica con un incremento de máximo 512 bytes.

2.5.1.3 GPS.h [9]

Esta librería establece el protocolo necesario para la comunicación de la aplicación con el dispositivo GPS Motorola M12+. A continuación se observa el árbol de métodos de esta librería.

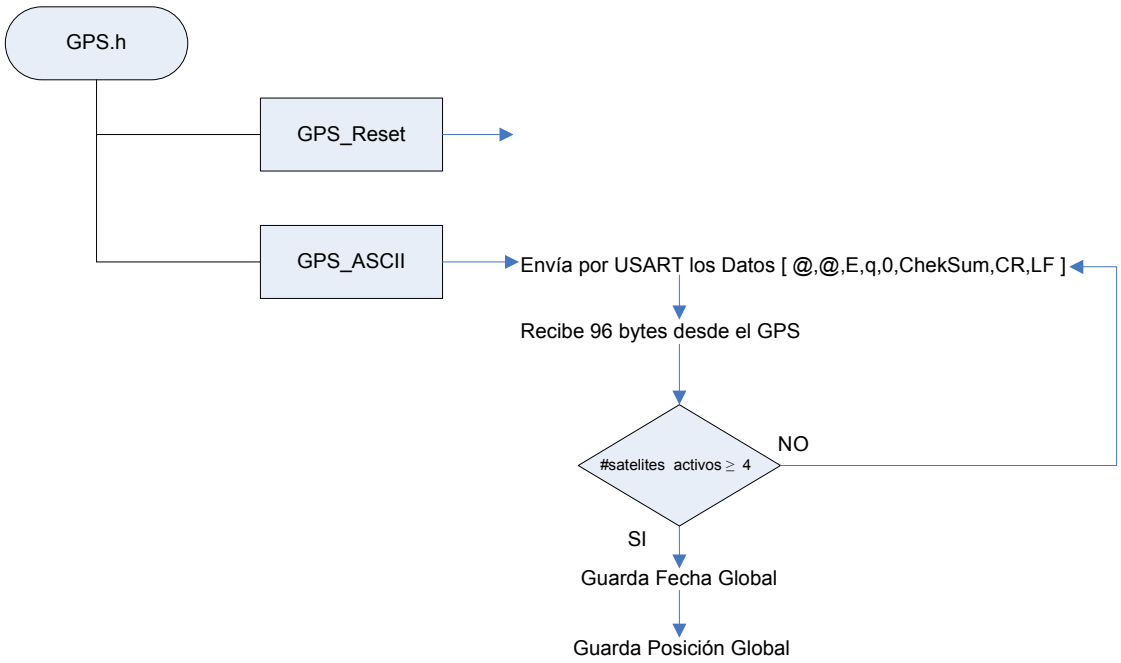


Gráfico 2-18. Árbol de métodos de la librería GPS.h.

GPS_Reset:

Este método envía el comando de Reset al GPS, con lo cual se borra toda información que podía haber estado almacenada con una sincronización anterior.

GPS_ASCII:

En este método se piden datos al GPS de posición, tiempo y número de satélites activos. Este proceso se realiza de una manera continua hasta lograr una sincronización del GPS con al menos cuatro satélites.

Una vez conseguida esta sincronización, se almacena la información de la fecha global y de la posición global.

2.5.1.4 RTC.h [10]

Esta librería permite igualar, configurar y leer el RTC DS1307 utilizando un protocolo I2C. En el siguiente gráfico se tiene el árbol de métodos incorporados.

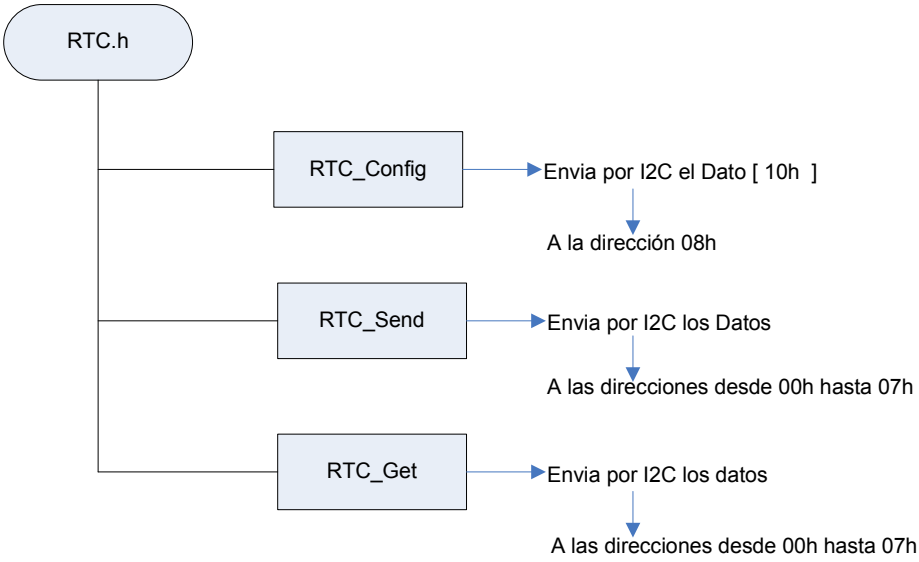


Gráfico 2-19. Árbol de métodos de la librería RTC.h.

RTC_Config:

Este método envía el comando 10h a la dirección 08h. Configurando el RTC con oscilación externa a 1HZ.

RTC_Send:

Este método envía los datos de fecha completa (año, mes, día del mes, día de la semana, hora, minuto, segundo) a la direcciones desde la 07h hasta la 00h respectivamente en el RTC.

RTC_Get:

Este método lee los datos de fecha completa que se encuentran en las direcciones desde la 00h hasta la 07h.

2.5.1.5 ADC.h [11]

Esta librería permite trabajar con el ADC 7738, al cual se le puede resetear, configurar y pedir datos de cualquier canal. El acceso a este conversor es a través de un protocolo SPI (*Serial Peripheral Interface*). A continuación se muestran los métodos de la librería.

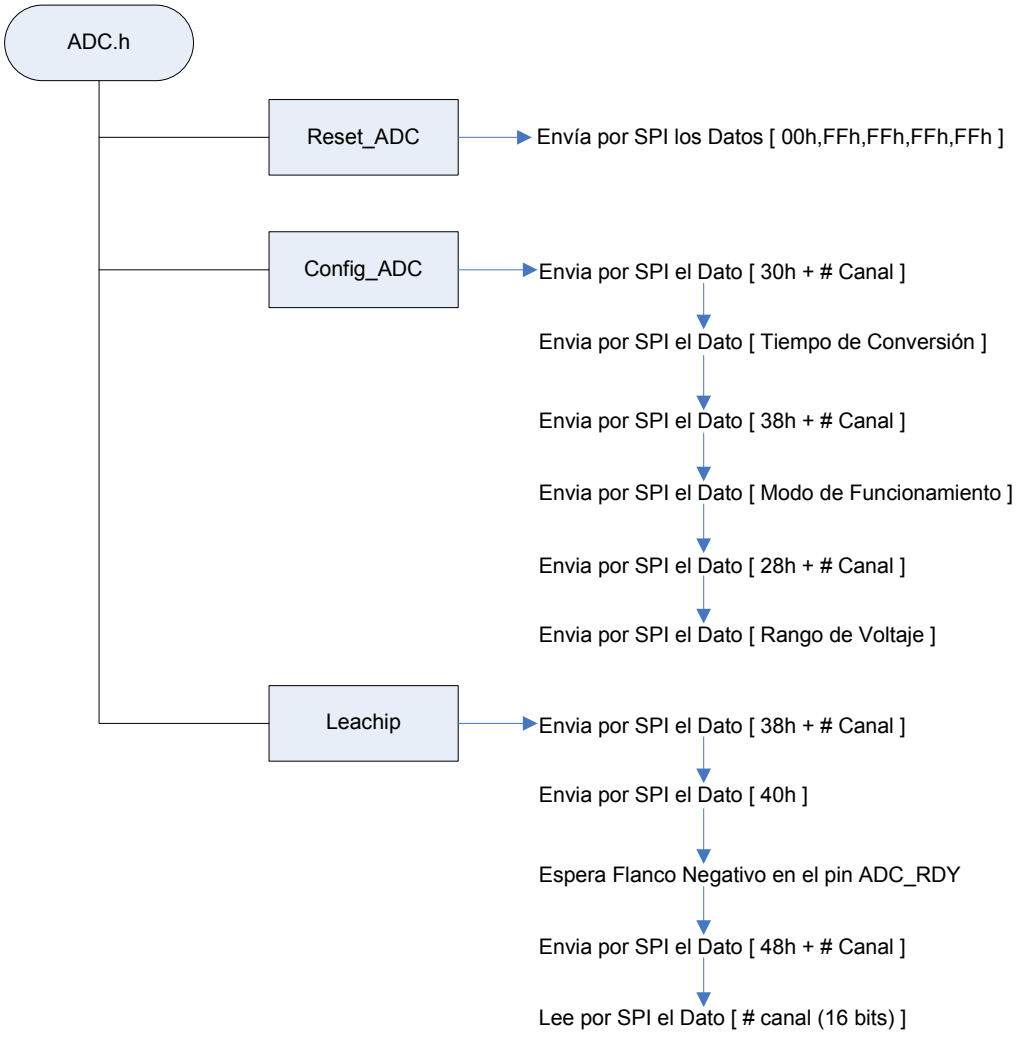


Gráfico 2-20. Árbol de métodos de la librería ADC.h.

Reset_ADC:

Este método envía la cadena de caracteres 00h, FFh, FFh, FFh, FFh por el puerto SPI para resetear al conversor.

Config_ADC:

Este método permite modificar los valores de configuración para cada canal. Estableciendo características de tiempo de conversión, modos de funcionamiento (conversión continua o conversión simple y número de bits por conversión 16 o 24) y el rango de voltaje de la ventana dinámica de trabajo.

Lea_Canal:

Este método activa la conversión simple de un canal enviando por el puerto SPI los comandos: 38h + #canal; 40h. A continuación espera que la señal ADC_RDY indique el final de la conversión. Se pide los datos actualizados del canal por medio del comando: 48h + #canal. A continuación el conversor envía los 16/24 bits de datos.

2.5.1.6 UART.h

Esta librería permite establecer una comunicación serie asincrónica entre el microcontrolador y una computadora, con un bit de inicio, ocho bits de datos, sin bit de paridad y un bit de parada; con velocidades de transmisión de hasta 19200 bps.

Todas las funciones de transmisión y recepción están implementadas en programa, con lo que se puede realizar una codificación de línea invertida (TTL).

En el siguiente gráfico se puede ver una lista de los eventos para esta librería.

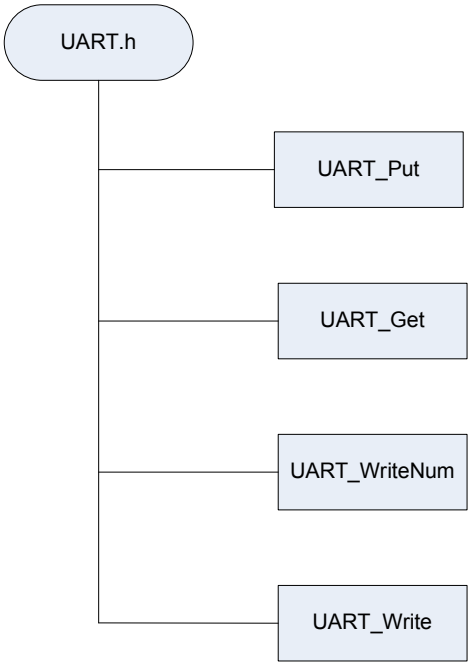


Gráfico 2-21. Árbol de métodos de la librería UART.h.

UART_Put:

Este método envía su argumento directamente por el pórtico serial.

UART_Get:

Este método devuelve un caracter recibido por el pórtico serial.

UART_WriteNum:

Este método utiliza el UART_Put para enviar dos caracteres correspondientes al código ASCII de su argumento.

UART_Write:

Este método envía por el pórtico serial una cadena de caracteres especificada en su argumento.

2.5.1.7 USART.h

Esta librería hace uso de los recursos en hardware del microcontrolador para establecer una comunicación serie asincrónica. A continuación se observa un árbol de métodos incluidos en la librería.

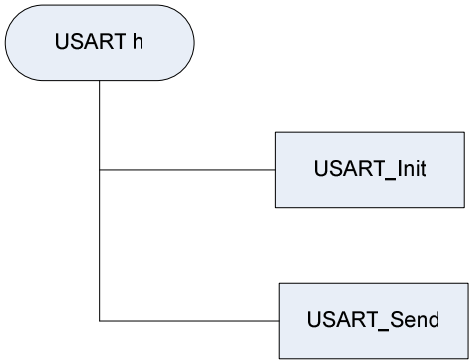


Gráfico 2-22. Árbol de métodos de la librería USART.h.

USART_Init:

Este método permite configurar el puerto serial del microcontrolador por medio de los registros asociados a éste. Se utiliza una transmisión con un bit de inicio, ocho bits de datos, sin bit de paridad y un bit de parada. La velocidad de transmisión será introducida se acuerdo al argumento del método.

USART_Send:

Este método envía un caracter por el puerto serial en hardware del microcontrolador.

2.5.1.8 SPI.h

Esta librería implementa en código una comunicación sincrónica desbalanceada SPI, solo con la función de maestro. Permite modificar la velocidad en la señal de reloj hasta 100 KHz., así como los pines a usar para el protocolo.

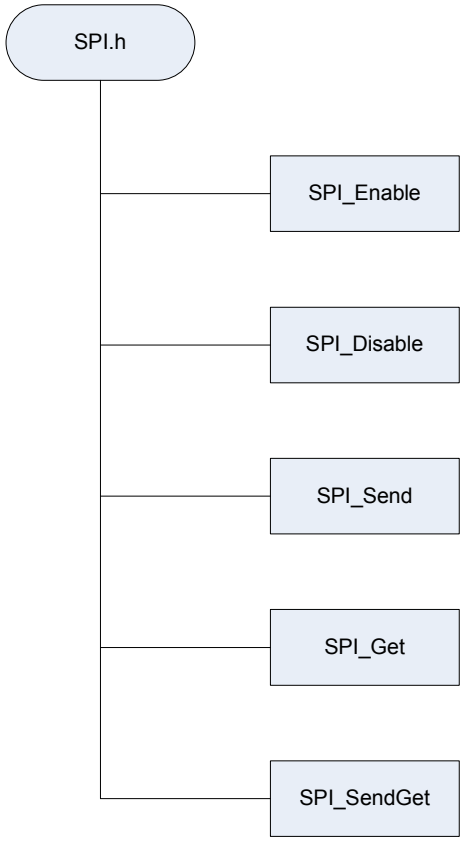


Gráfico 2-23. Árbol de métodos de la librería SPI.h.

SPI_Enable:

Este método habilita el puerto SPI, activando el pin de selección del elemento esclavo.

SPI_Disable:

Este método deshabilita el puerto SPI, poniendo el valor de reposo en el pin de selección del elemento esclavo.

SPI_Send:

Este método envía su argumento por el puerto SPI.

SPI_Get:

Este método devuelve un caracter leído en el puerto SPI.

SPI_SendGet:

Este método permite enviar (DO) y recibir (DI) un caracter al mismo tiempo por el puerto SPI.

2.5.2 PROGRAMAS DE LOS MICROCONTROLADORES

2.5.2.1 Programa ATmega128L

Este punto explica como está estructurado el programa principal que ejecuta el ATmega128L. A continuación se muestra un gráfico del diagrama de bloques de macros y declaraciones.

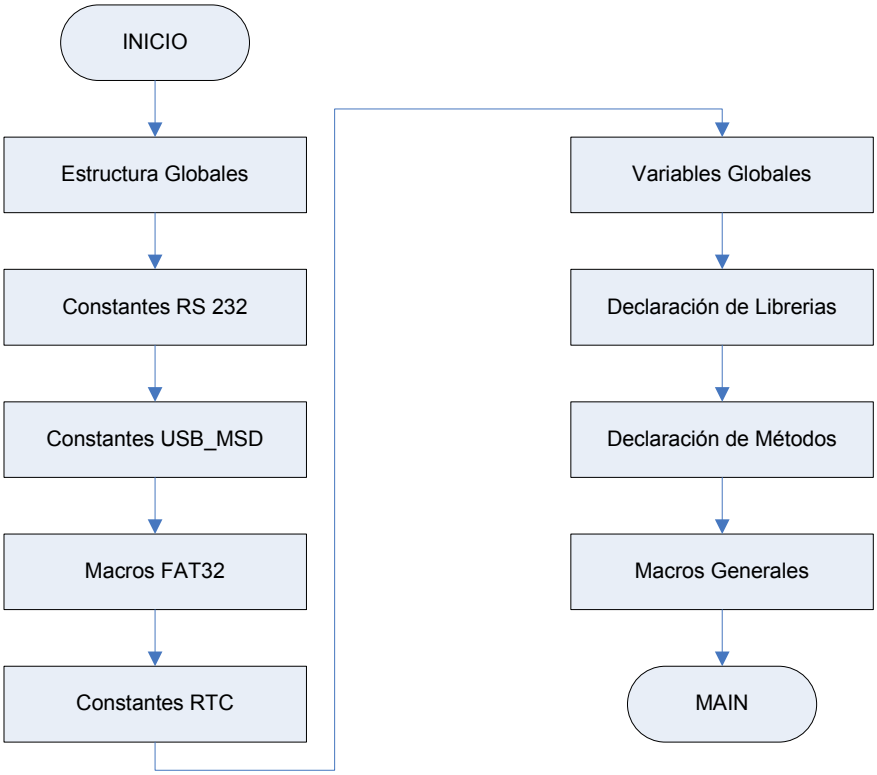


Gráfico 2-24. Diagrama de bloques de macros y declaraciones en el ATmega128L

Al inicio se encuentran las declaraciones de:

- Estructuras globales empleadas en el programa principal.
- Las constantes empleadas en la comunicación serial entre el microcontrolador ATmega128L y el computador, como son: velocidad de transmisión (19200 bps) y los pines del microcontrolador empleados para la transmisión y recepción de datos. Configurando así un puerto virtual con un bit de inicio, ocho bits de datos, un bit de parada y sin bit de paridad. Además se utiliza una lógica invertida en los niveles de voltaje en la comunicación serial para prescindir del integrado MAX232.
- Se define la velocidad de transmisión serial entre el microcontrolador y el modulo GPS M12+ a 9600 bps, con un bit de inicio, ocho bits de datos, un bit de parada y sin bit de paridad. Esta comunicación es implementada utilizado los recursos en hardware que posee el ATmega128L.
- Los pines empleados por el ATmega128L para comunicarse con el AT43USB380. Utilizando un bus de ocho bits para datos, y un bus de 8 bits para direccionamiento y cinco líneas para control de flujo.
- Las macros de lectura y escritura empleadas por la librería *FAT32* vinculadas con los métodos de correspondientes de la librería USB_MSD. Enlazando así dos librerías de funcionamiento independiente.
- Los pines empleados por el ATmega128L para comunicarse con el *RTC* DS1307. El cual trabaja empleando una capa física I2C.
- Variables globales.
- Declaración de Métodos.
- Macros generales empleadas en el programa principal.

A continuación se describe el programa principal en el ATmega128L con su respectivo diagrama de bloques.

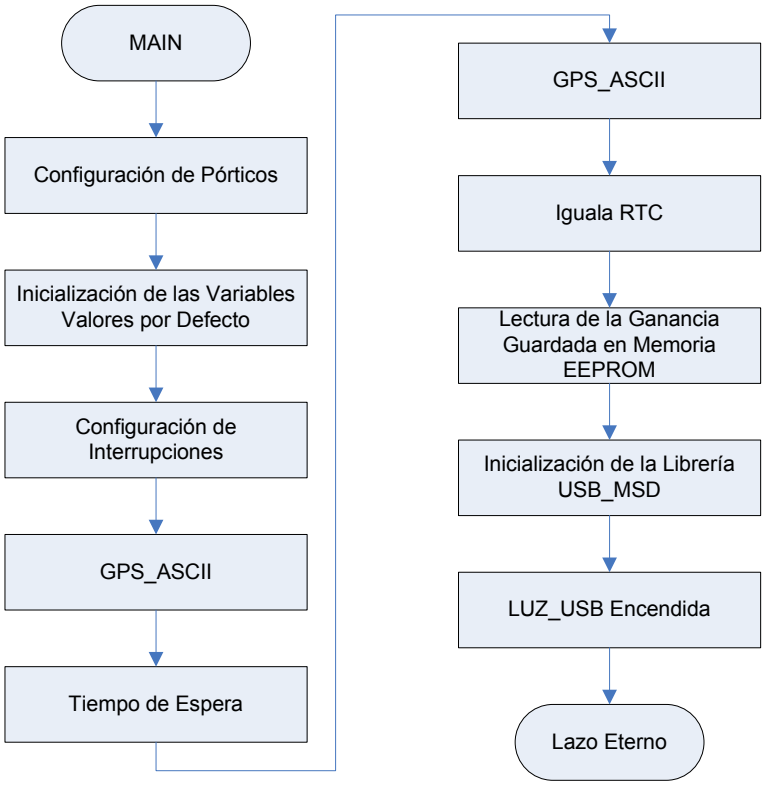


Gráfico 2-25. Diagrama de flujo del programa principal en el ATmega128L.

Al iniciar el programa principal se configuran los pórticos como entradas o salidas de acuerdo a las declaraciones, se asignan valores por defecto a las variables empleadas en el programa principal y se configuran las interrupciones. Luego de esto el programa pide datos al módulo GPS M12+ y verifica que el número de satélites encontrados por el receptor GPS sea al menos cuatro, si esta condición se cumple el programa lee los datos de latitud, longitud y altura.

A continuación el programa espera un tiempo de aproximadamente un minuto y toma el valor de la hora global del GPS M+12 con la que iguala al *RTC* DS1307, después de esto el RTC genera una señal cuadrada a una frecuencia de 1 Hz, la cual es empleada por el ATmega128L para sincronismo.

Al tener un sincronismo garantizado, se procede a la inicialización de la librería USB_MSD, con lo que se activa el controlador AT43USB380 como *HOST USB* y ahora el quipo es capaz de reconocer cualquier dispositivo USB conectado a su

puerto, discriminando para un establecimiento de una comunicación en capa superior, solo las memorias USB (MSD).

A continuación se muestra un diagrama de flujo del lazo eterno.

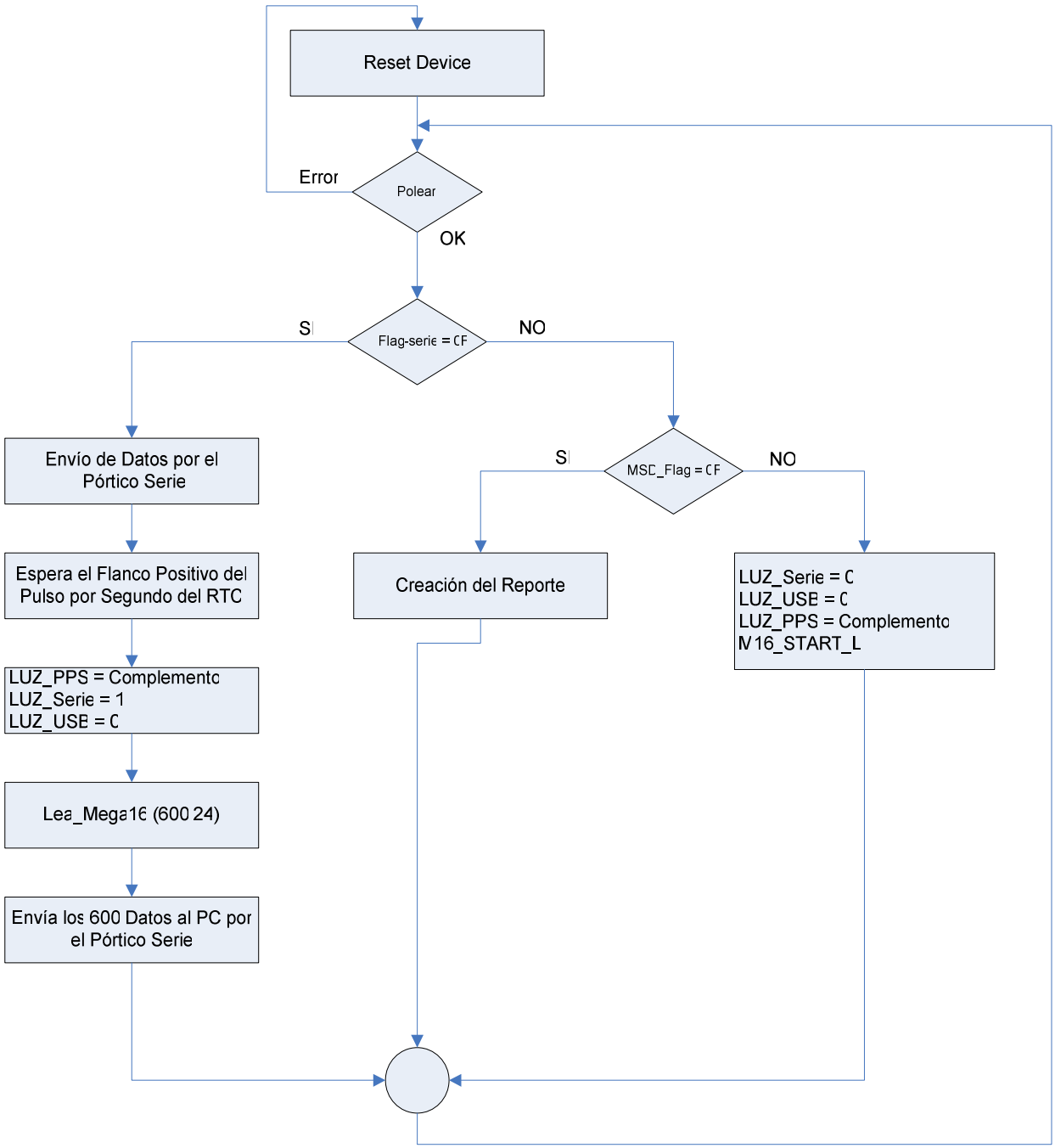


Gráfico 2-26. Diagrama de flujo del lazo eterno en el ATmega128L.

Al culminar con los anteriores procedimientos el programa entra a un lazo eterno, dentro del cual se puede direccionar el flujo del programa con ayuda de banderas las cuales son alteradas de acuerdo al dato ingresado cuando se genera una interrupción por recepción debido a una transmisión serial provocada por el computador.

El computador puede enviar los siguientes comandos:

- *Comando "T":* el microcontrolador envía como respuesta la "L" indicando que el equipo esta conectado al puerto serial.
- *Comando "I":* el microcontrolador responde con una trama de datos la cual contiene la latitud, longitud, altura y un identificador de la estación.
- *Comando "S":* el programa polea los datos del ATmega16L y envía bloques de 600 bytes de datos cada segundo hacia el computador.
- *Comando "M":* el programa polea los datos del ATmega16L y los almacena en la memoria como un archivo binario de extensión GDU.
- *Comando "P":* el programa deja de polear al ATmega16L y no envía datos ni a la memoria ni al puerto serial.
- *Comando "Ganancia":* para este comando se envían datos entre 30h para ganancia de 0 hasta 3Fh para ganancia de 15, estas ganancias se almacenan en la memoria eeprom del microprocesador.

Los paquetes de 600 bytes contienen 100 muestras cuantificadas a 16 bits correspondientes a los tres canales, organizadas en grupos de seis bytes, dos por cada canal.

A continuación se muestra un diagrama de flujo de la creación del reporte

.

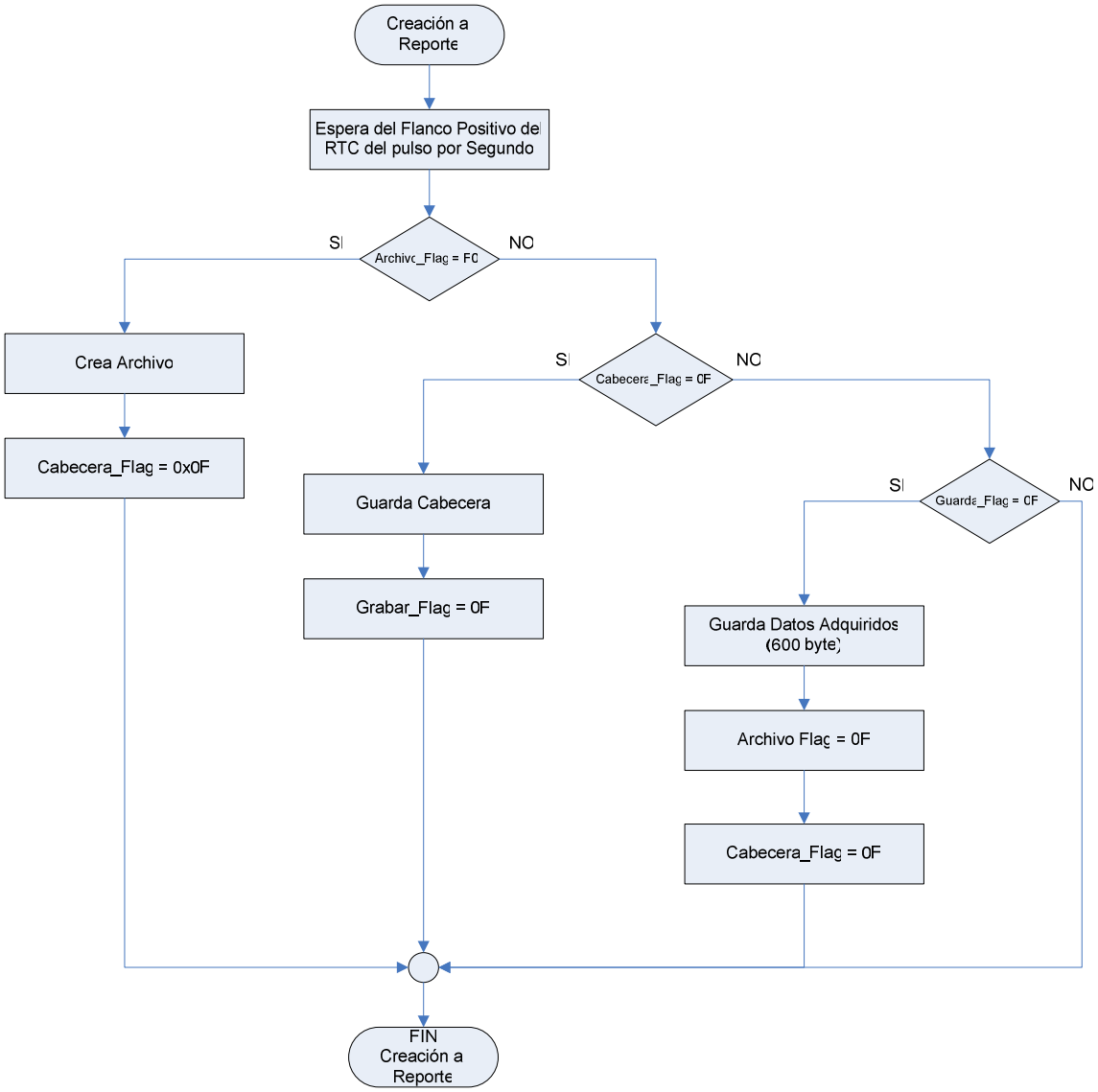


Gráfico 2-27. Diagrama de flujo de la creación del reporte.

Cuando se encuentra activado el envío de datos a la memoria USB, el programa busca si existe un archivo creado con el nombre de “GEODAS.GDU” en el caso de no encontrarlo crea un archivo en blanco con el mismo nombre y luego agrega una cabecera la cual se encuentra estructurada de la siguiente forma:

GEODAS_USB V1.0

Posicion
Latitud: 00°08.8949'S

Longitud: 078°27.7640'W
Altura: 02934.5

Fecha: 2006/10/09
Hora: 17:31:59

ID_Estacion: 1

AUTORES:
Andres Cadena
Juan Carlos Tapia

Datos:xxyyzzxxyyzzxxyyzz….xxyyzz

Esta cabecera es agregada al archivo cada vez que se genera una discontinuidad en los datos adquiridos, esto puede suceder por tres motivos: que el computador suspenda la adquisición, que el usuario retire la memoria o que el voltaje en la fuente principal sea menor al fijado en el sistema de control de voltaje.

Debido a que el envío de datos a la memoria USB es continuo, el sistema tiene implementado un pulsante el cual suspende el envío de datos a la memoria garantizando que no exista actividad en la misma cuando el usuario la remueva, con esto se protege la integridad de los datos almacenados.

Se necesita adquirir datos del conversor análogo-digital AD7738 con una frecuencia de 100 muestras por segundo lo cual genera un tiempo entre muestras de apenas 10ms, por este motivo se diseñó un sistema que adquiera los datos durante un segundo los cuales son descargados en paquetes de 24 bytes por cada slot de tiempo libre entre muestra y muestra; como son 24 bytes los descargados desde el ATmega16L al ATmega128L la descarga de los 600 bytes requiere de 25 slots de tiempo libre con lo que se gana 750 ms para almacenar paquetes de 600 bytes en la memoria o para ser transmitirlos al computador.

Para aclarar cómo se genera este proceso de adquisición, se presenta un diagrama de tiempos en el cual se indica el tiempo que demora el ATmega16L en recoger los datos de los tres canales del ADC, el tiempo que demora en transmitir los datos hacia el ATmega128L, el tiempo que demora trasmitir los datos hacia el computador el microprocesador ATmega128L y el tiempo que demora en guardar los datos en la memoria USB

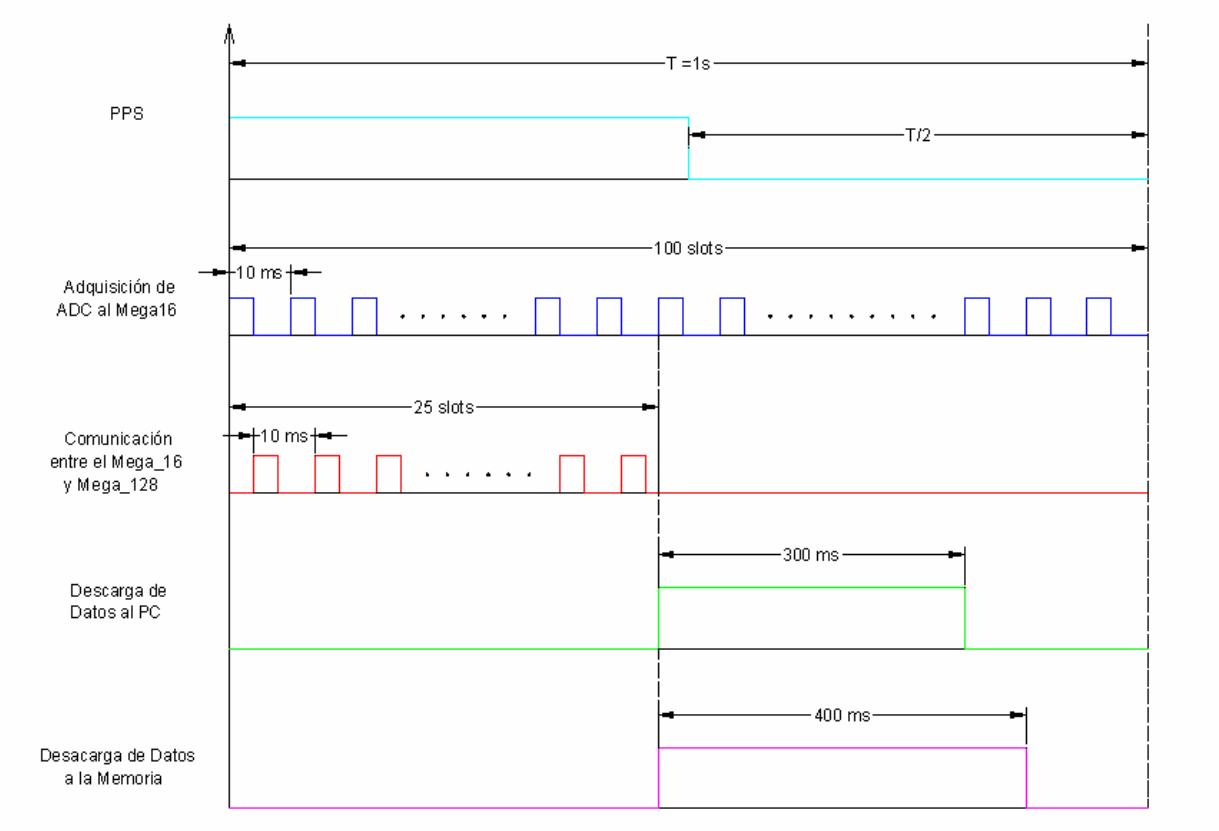


Gráfico 2-28. Diagrama de flujo de la creación del reporte.

2.5.2.2 Programa ATmega16L

En este punto se detalla en bloques como está estructurado el programa en el microprocesador ATmega16L. En la primera parte constan las declaraciones de macros y variables globales, esta porción de código sólo se ejecuta una vez al resetear el microcontrolador. En cada bloque de macros se definen los pines del microcontrolador a ser usados en cada una de las diferentes comunicaciones.

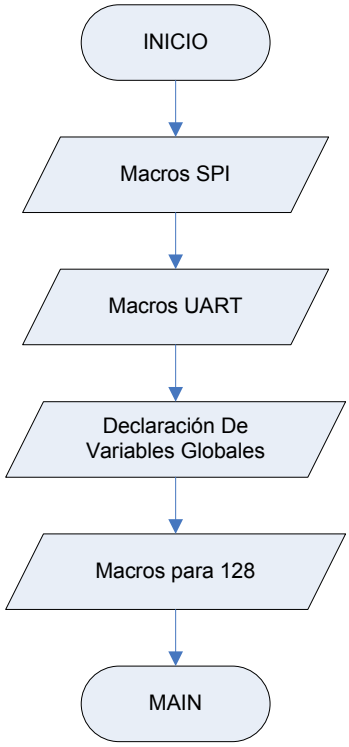


Gráfico 2-29. Diagrama de flujo de macros en el ATmega16L.

A continuación se presenta el programa principal; el cual comienza con una configuración de los puertos del microprocesador definiendo si son utilizados como entradas o salidas. Luego se inicializa el conversor AD7738 con un reset generado por medio de un comando para después configurar cada canal con una ventana dinámica de ±2.5 V, un tiempo de conversión de 36.3 µs y un modo de conversión simple. Para la inicialización del *Timer1* se cargan los valores necesarios para que éste genere una interrupción cada 10 ms. En la última parte de la configuración están las interrupciones, las cuales son habilitadas para su comunicación con el ATmega128L y para el muestreo de las señales por medio del conversor.

A partir de este momento el ATmega16L se encierra en un lazo infinito, en el cual sólo valida un flanco positivo en la señal CKL_128 para copiar un byte desde RAM hacia el pórtico de salida de datos.

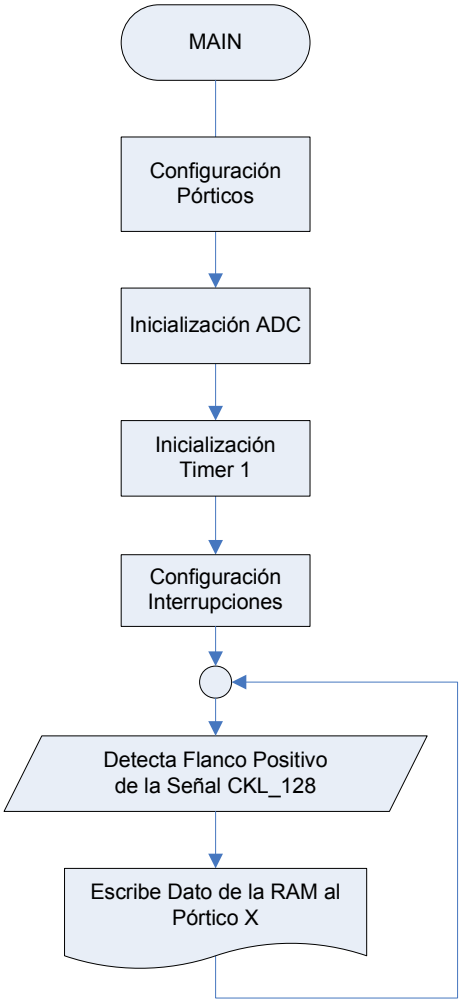


Gráfico 2-30. Diagrama de flujo del programa principal en el ATmega16L.

La interrupción externa es configurada por flanco positivo para un inicio del *Timer1*, como se observa en el siguiente diagrama.

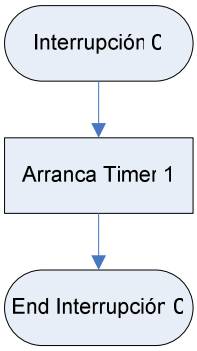


Gráfico 2-31. Diagrama de flujo de la interrupción externa en el ATmega16L.

Por último se incluye el bloque de código que es ejecutado en la interrupción del Timer1 a 100 Hz. Aquí primero se carga el valor en el *Timer1* para que siga contando, y luego se leen desde el conversor los datos de cada canal, los cuales son almacenados en RAM.

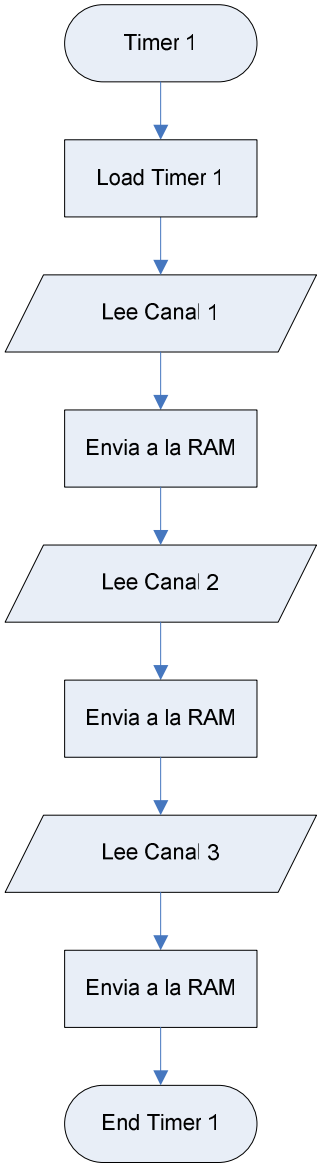


Gráfico 2-32. Diagrama de flujo de la interrupción del Timer1 en el ATmega16L.

2.6 ANÁLISIS DE COSTOS

En este punto se listan todos los materiales con los cuales fue construido el equipo GEODAS_USB con su respectiva cantidad, descripción y precio.

Índice	Cantidad	Elemento	Descripción	Precio unitario	Precio
1	20	Capacitor	10 uF	$0,17	$3,40
2	1	Capacitor	3.3 nF	$0,21	$0,21
3	1	Capacitor	33 nF	$0,03	$0,03
4	3	Capacitor	470 pF	$0,03	$0,10
5	5	Capacitor	2.2 uF	$0,09	$0,46
6	8	Capacitor	33 pF	$0,03	$0,26
7	17	Resistencia	330 ohm	$0,07	$1,24
8	18	Resistencia	1.1 Kohm	$0,07	$1,31
9	36	Resistencia	11 Kohm	$0,07	$2,63
10	2	Resistencia	15 Kohm	$0,07	$0,15
11	1	Resistencia	680 Kohm	$0,10	$0,10
12	6	Potenciómetro	10 Kohm	$1,24	$7,44
13	4	2N3904	Transistor NPN	$0,20	$0,80
14	3	Led	Rojo	$0,20	$0,59
15	1	Led	Verde	$0,23	$0,23
16	1	Led	Anaranjado	$0,18	$0,18
17	3	1N4007	Diodo	$0,10	$0,30
18	3	Receptáculo	Mini 5x2	$3,82	$11,46
19	1	Bornera	2x1	$0,30	$0,30
20	1	Header	3x1	$0,40	$0,40
21	1	Header	5x2	$0,78	$0,78
22	1	Relee	12 VDC	$1,00	$1,00
23	1	Switch	Doble	$0,40	$0,40
24	1	Pulsante	Simple	$0,20	$0,20
25	1	ICL7662	Convertidor DC -DC	$1,65	$1,65
26	3	LM324	Amplificador 4 Compuertas	$0,85	$2,55
27	1	AD7738	ADC Sigma-Delta 16-24 bits	$15,06	$15,06
28	1	AT43USB380	Controlador USB	$10,05	$10,05
29	1	LM317	Regulador de voltaje variable	$1,00	$1,00
30	1	MC33269	Regulador de voltaje 3.3 V	$1,30	$1,30
31	1	LM7805	Regulador de voltaje 5 V	$1,00	$1,00
32	1	LM7808	Regulador de voltaje 8 V	$1,00	$1,00
33	1	TLC4066	Switch analógico	$0,55	$0,55
34	1	ATMEGA16L	Microcontrolador AVR	$10,05	$10,05
35	1	ATMEGA128L	Microcontrolador AVR	$18,00	$18,00
36	1	DS1307	RTC	$3,29	$3,29
37	1	Conector	USB tipo A female	$0,80	$0,80
38	4	Cristal	6 Mhz	$1,00	$4,00
39	1	Cristal	32.768 KHz	$1,00	$1,00
40	1	Caja	Pelican 1060	$20,00	$20,00
41	1	Conector	Sensor Sìsmico (19 pines)	$5,00	$5,00
42	1	Conector	3 pines	$1,50	$1,50
43	1	Conector	DB9 macho	$0,80	$0,80
44	1	M12+	Módulo GPS	$150,00	$150,00
45	1	cable	4 hilos	$0,65	$0,65
46	2	Conector	DB9 hembra	$0,80	$1,60
					$284,81
47	1	PCB	17x8 cm^2	$80,00	$80,00
48	1	Soldado		$80,00	$80,00
49	1	Ensamblaje		$20,00	$20,00
					$464,81
				IVA 12%	$55,78
				Precio Final:	$520,59

Tabla 2-2.Tabla de costos de materiales y manufactura.

A partir de los costos finales se puede apreciar que el equipo es económico en comparación con un equipo americano de funciones similares (*RefTeK* modelo: *130 Broadband Seismic Recorder*[29]), cuyo costo está alrededor de $10000.

Vale recalcar que todo el equipo, excepto el módulo GPS, fue construido en el país y los materiales de montaje superficial fueron adquiridos por importación directa.

[29] Costo proporcionado por la empresa REFRACTION TECHNOLOGY, INC. Cotización # 10013.

CAPÍTULO 3 PRUEBAS Y RESULTADOS

En este capitulo se muestran los resultados obtenidos y su análisis en base a las mediciones realizadas en el equipo GEODAS_USB.

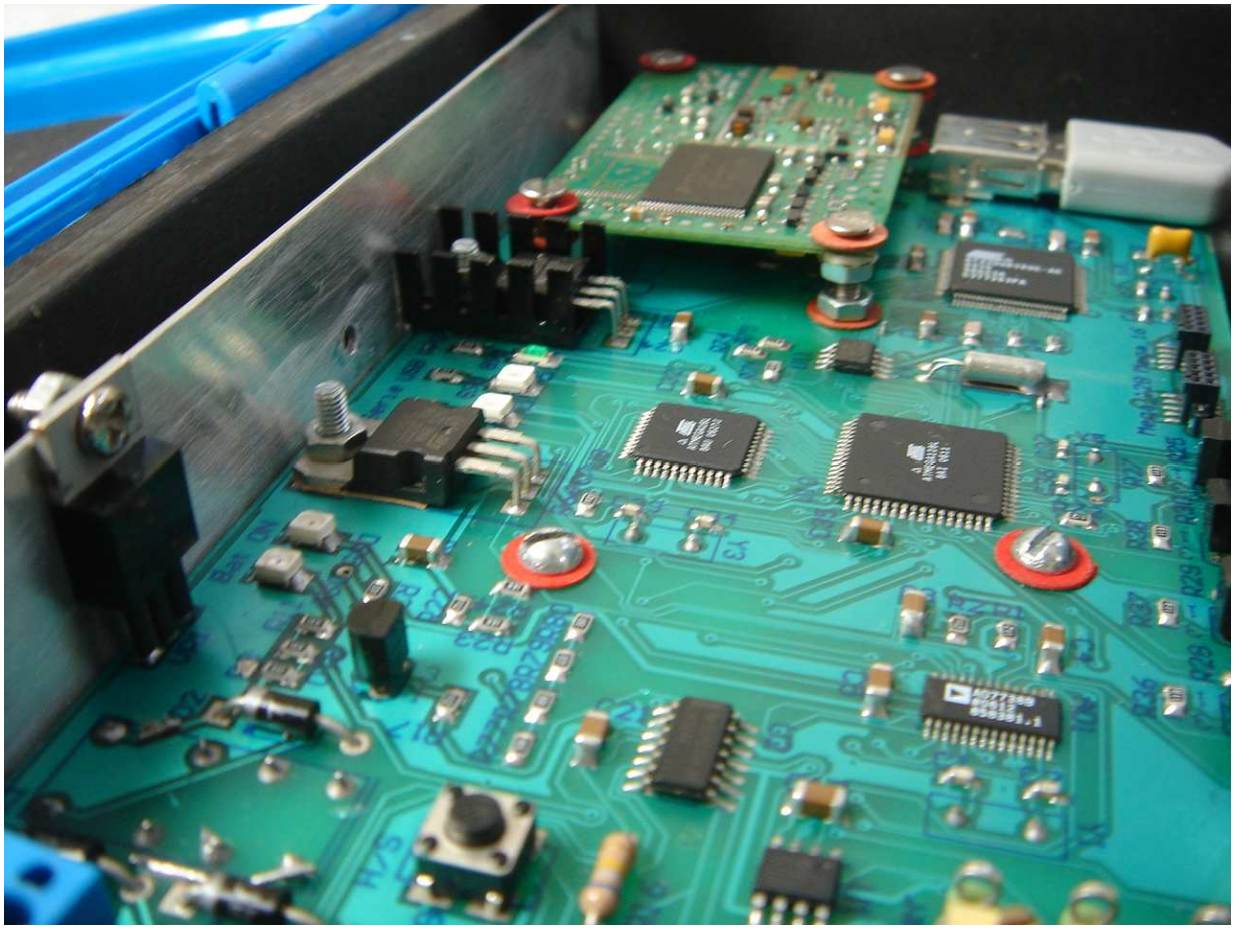


Gráfico 3-1. Foto interna del equipo GEODAS_USB.

Fueron efectuadas las siguientes pruebas: Voltajes de Alimentación, Control de Voltaje, Consumo de Corriente, Respuesta Analógica, Dispositivos USB y Sensor Real. Este tipo de pruebas reflejarán el comportamiento eléctrico y analógico de los circuitos diseñados y de los elementos utilizados.

A continuación se especifica el procedimiento realizado en cada prueba y los resultados obtenidos.

3.1. VOLTAJES DE ALIMENTACIÓN

Con un voltímetro se tomaron los valores de entrada y salida de cada uno de los reguladores de voltaje empleados como fuentes de alimentación.

A continuación se muestran los valores teóricos y medidos para cada fuente de voltaje:

Fuente [Vdc] (valor teórico)	**Vin[30] [Vdc]**	**Vout [Vdc] (valor medido)**
8 V	12	7.8
-8 V	7.8	-7.4
5 V	7.8	5
3.3 V	7.8	3.2
3 V	7.8	3

Tabla 3-1. Valores medidos en cada regulador de voltaje.

Análisis de resultados: los voltajes de salida obtenidos son aceptables como respuesta de los reguladores comerciales utilizados. Estos valores están dentro de los rangos de voltaje aceptados como polarización para todos los dispositivos que emplean estas fuentes.

El mayor desvío de voltaje se observa en la fuente negativa, esto se debe a que la corriente máxima que puede manejar este integrado (20 mA) es muy reducida en comparación con la que pueden manejar los otros reguladores (1000 mA y 800 mA).

3.2. CONTROL DE VOLTAJE

Basados en el circuito del Gráfico 3-2 y su explicación teórica, se busca un valor central de la ventana de 12 V y una ventana de 1 V (Vdesconexión = 11.5 V,

[30] La fuente de alimentación principal que se utiliza es una batería de 12 V dc, lo cual garantiza que la señal de entrada no tiene rizo.

Vconexión = 12.5 V). Por lo cual los voltajes medidos en los puntos 1 y 2 deberían ser 3 V y 0.25 V respectivamente.

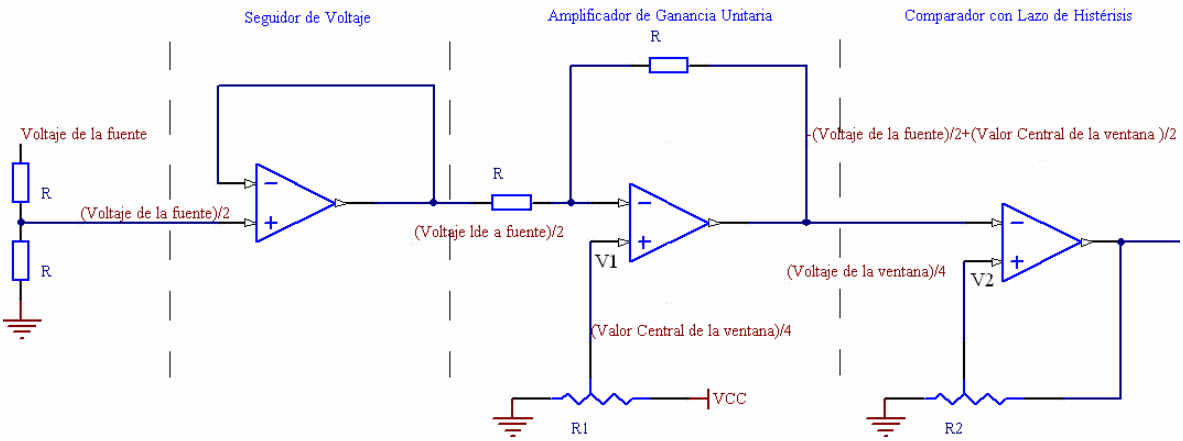


Gráfico 3-2. Circuito controlador de voltaje.

Los resultados obtenidos son:

V1 [V]	V2 [V]	V desconexión [V]	V conexión [V]	Eje [V]	Ventana [V]
3	0.25	11.6	13	12.3	1.4

Tabla 3-2. Respuesta a los valores teóricos en el circuito controlador de voltaje.

Al no obtenerse los resultados deseados en los voltajes de conexión y desconexión, se procede a calibrar los potenciómetros R1 y R2 con los siguientes valores:

V1 [V]	V2 [V]	V desconexión [V]	V conexión [V]	Eje [V]	Ventana [V]
2.93	0.208	11.5	12.5	12	1

Tabla 3-3. Respuesta a los valores calibrados en el circuito controlador de voltaje.

Análisis de resultados: Como se puede apreciar para obtener los valores de conexión y desconexión deseados se tuvo que hacer un pequeño ajuste en los voltajes V1 y V2; esto se debe a que en el análisis teórico se considera un voltaje diferencial de 0 V, pero en la realidad este voltaje es pequeño pero no es igual a cero.

3.3. CONSUMO DE CORRIENTE

Con un amperímetro conectado al borne positivo de la batería se mide la corriente para diferentes escenarios de funcionamiento del equipo GEODAS_USB.

Escenario	Corriente [mA]
Sin periféricos	296
Con sensor sísmico	330
Con dispositivo USB	365
Con sensor sísmico y dispositivo USB	396

Tabla 3-4. Valores de corriente para los diferentes escenarios de prueba.

Leyendo la información de los manuales de una tarjeta entrenadora comercializada por Atmel (AT43DK380), se observa que usan el regulador de voltaje LT1083, el cual es una fuente voltaje regulable y con una corriente máxima de 7.5 A.

La utilidad del entrenador es similar al equipo GEODAS_USB en el manejo del integrado AT43USB380, pero no utiliza un microcontrolador de ocho bits para realizar este proceso, en su lugar utiliza un microprocesador de 32 bits. [16]

Análisis de resultados: Como se puede apreciar la corriente de consumo es la esperada al trabajar con dispositivos que procesan a alta velocidad. Además es reducida en comparación a una tarjeta entrenadora la cual podría ser empleada para realizar la misma aplicación del equipo GEODAS_USB.

3.4. RESPUESTA ANALÓGICA

Para evaluar la respuesta del circuito analógico (ganancia variable y respuesta de frecuencia), se emplea un generador de señales, un osciloscopio digital y el programa GEODAS_GRAPHICS y se realizan las siguientes pruebas.

3.4.1. GANANCIAS

Se toma como señal de entrada una sinusoide a 1 Hz de 100 mV de amplitud, esta señal se envía por igual a los tres canales disponibles. En las siguientes gráficas se muestra la respuesta del circuito a cuatro valores de ganancia.

Señal de entrada:

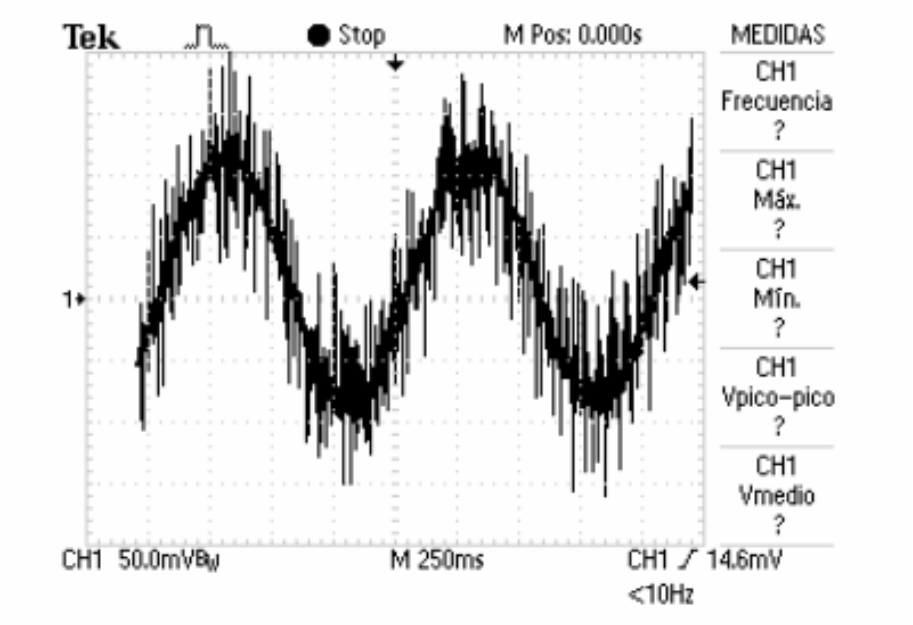


Gráfico 3-3. Señal de entrada: sinusoide, 1 Hz, 100 mV.

Ganancia = 1 :

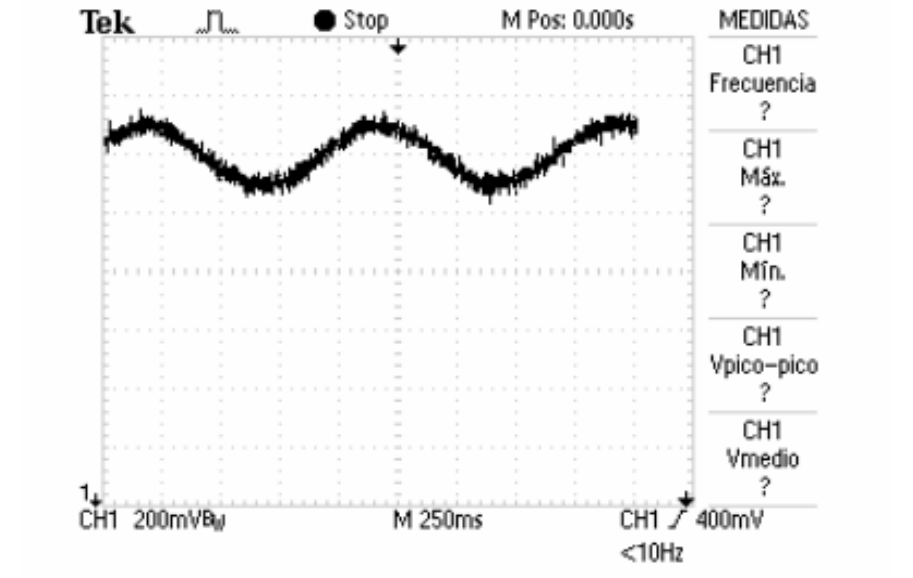


Gráfico 3-4. Señal de salida, osciloscopio digital.

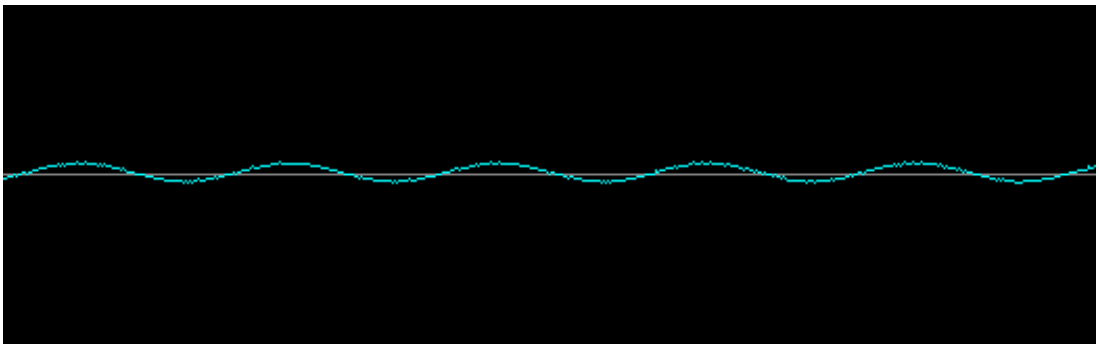


Gráfico 3-5. Señal de salida, programa GEODAS_GRAPHICS.

Ganancia = 5 :

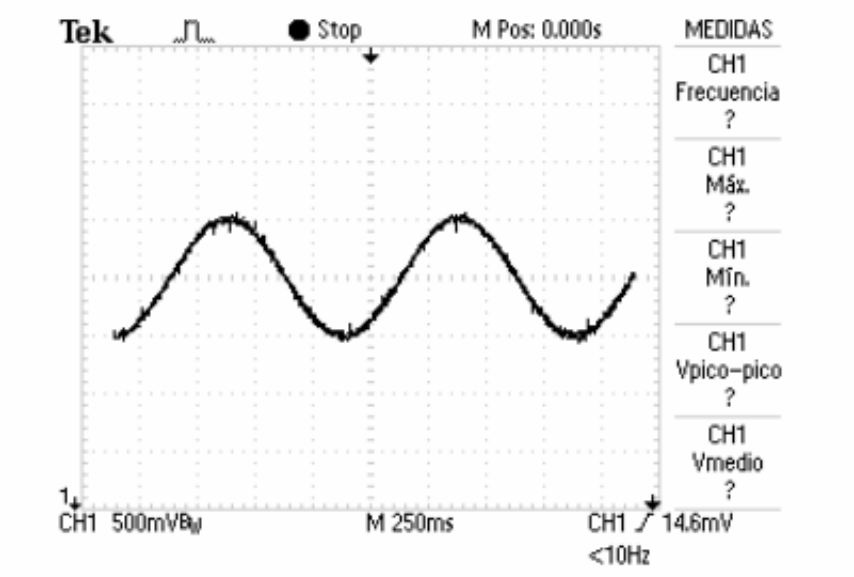


Gráfico 3-6. Señal de salida, osciloscopio digital.

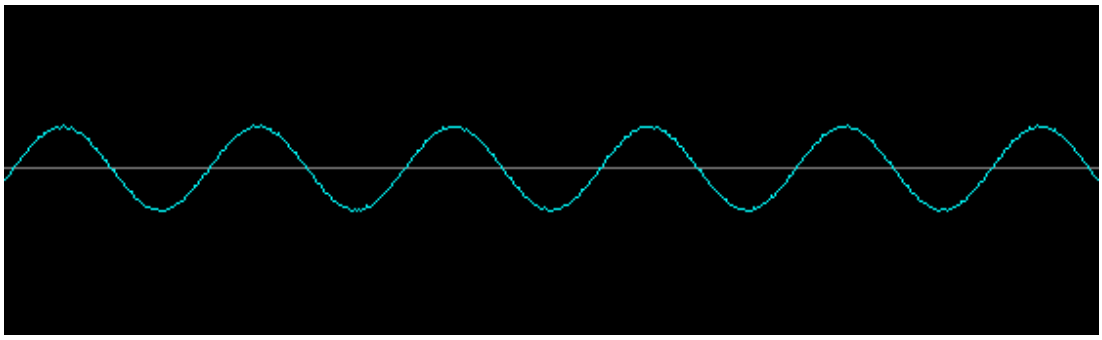


Gráfico 3-7. Señal de salida, programa GEODAS_GRAPHICS.

Ganancia = 10 :

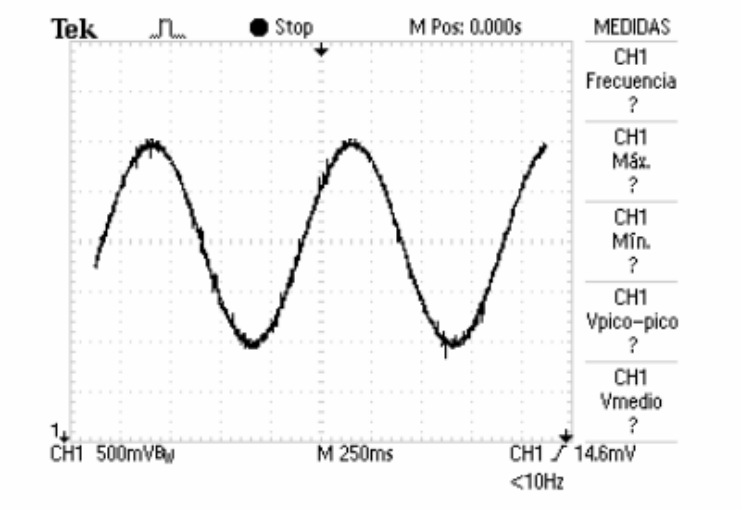


Gráfico 3-8. Señal de salida, osciloscopio digital.

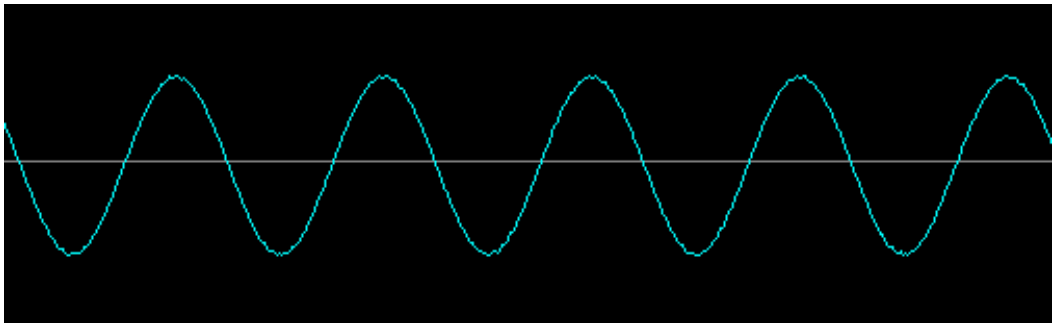


Gráfico 3-9. Señal de salida, programa GEODAS_GRAPHICS.

Ganancia = 15 :

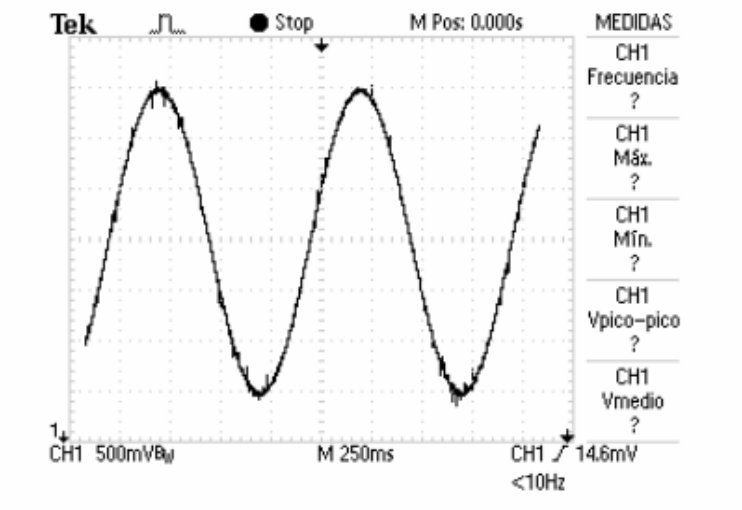


Gráfico 3-10. Señal de salida, osciloscopio digital.

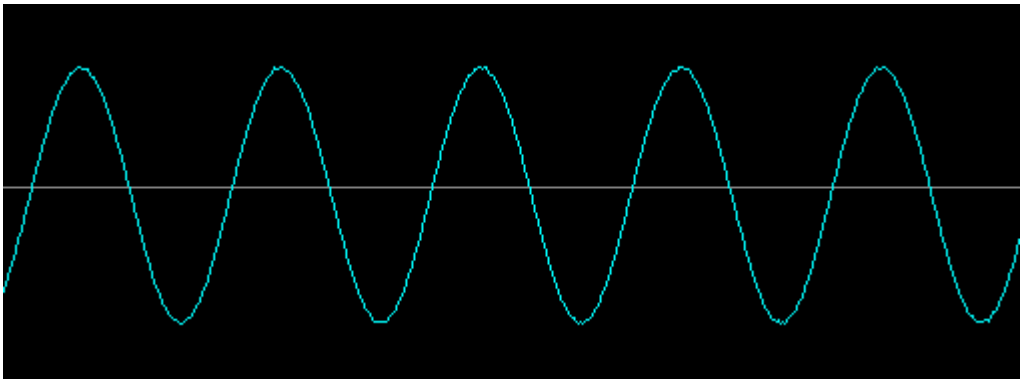


Gráfico 3-11. Señal de salida, programa GEODAS_GRAPHICS.

A continuación se muestra una tabla con los valores medidos con un voltímetro para una señal de entrada de 200 mV dc, con todas las ganancias disponibles.

Ganancia teórica	Vin [mV]	Vout [mV]	Ganancia real	|% error|
1	200	203	1.015	1.5
2	200	403	2.015	0.75
3	200	599	2.995	0,167
4	200	798	3.99	0.25
5	200	994	4.97	0.6
6	200	1193	5.965	0.58
7	200	1389	6.945	0.786
8	200	1590	7.95	0.625
9	200	1787	8.935	0.722
10	200	1984	9.92	0.8
11	200	2182	10.91	0.818
12	200	2382	11.91	0.75
13	200	2579	12.895	0.808
14	200	2776	13.88	0.857
15	200	2974	14.87	0.867

Tabla 3-5. Tabla de ganancias.

Análisis de resultados: El error mostrado en la tabla 3-5 es el resultado de dos variables no consideradas en el cálculo teórico. La primera se debe a que los

switches analógicos (4066) se construyen con dispositivos semiconductores, esto provoca que presenten una impedancia en el orden de ohmios cuando se encuentran cerrados, influyendo en el equivalente resistivo del arreglo mostrado en el gráfico 2-13. La segunda variable es la tolerancia en el valor nominal de las resistencias empleadas en el arreglo (1%).

Para minimizar la influencia de la impedancia presentada por los switches el arreglo fue elaborado con resistencias de un valor alto (11 KΩ) en relación a la impedancia de los switches.

3.4.2. RESPUESTA DE FRECUENCIA

Para evitar que se cuantifiquen señales de frecuencia mayor a 50 Hz se implementó un filtro pasivo RC pasa bajos en cada uno de los canales de entrada.

Para poder evaluar la respuesta de frecuencia del filtro se ingresa una señal sinusoidal de amplitud fija (2 V) y frecuencia variable obteniéndose los siguientes resultados.

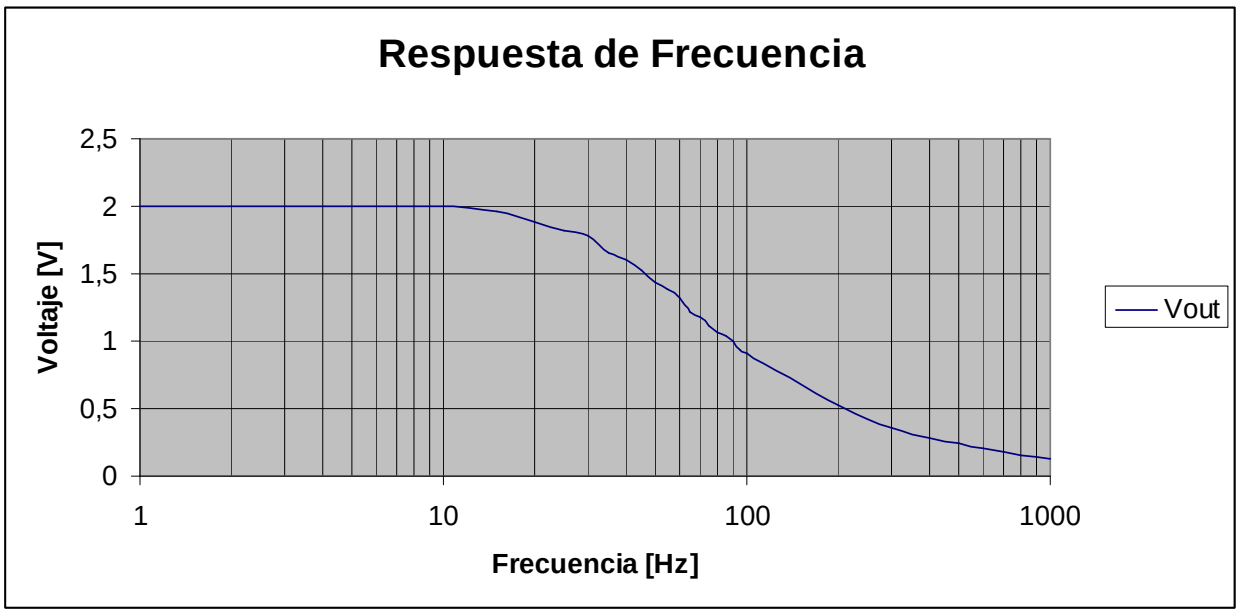


Gráfico 3-12. Respuesta de frecuencia del filtro RC.

Análisis de resultados: La respuesta obtenida en frecuencia tiene una caída muy lenta (14,96 dB/dec) debido a que se empleó un filtro pasivo. Se observa que las señales arriba de la frecuencia de corte calculada (50.59 Hz) se atenúan progresivamente hasta el 1 KHz, y a partir de esta frecuencia se puede considerar que las señales están atenuadas totalmente.

Para esta prueba se ingresaron señales de 0 a 1 KHz solamente para evaluar la respuesta del filtro; sin embargo en la aplicación final del equipo GEODAS_USB la fuente de información (sensor sísmico) entrega señales en un rango de frecuencia limitado (1/20 Hz a 40 Hz).

3.5. DISPOSITIVOS USB

A continuación se presentan todos los tipos de memorias USB con las cuales el equipo GEODAS_USB fue probado.

Marca de memoria USB	Densidad	Resultado
Kingston	1 Gbytes	Exitoso
Memorex	512 Mbytes	Exitoso
Dane-elec	256 Mbytes	Exitoso
Kozumi	128 Mbytes	Exitoso
Sony	128 Mbytes	Exitoso

Tabla 3-6. Tipos de memorias probadas con el equipo GEODAS_USB.

Gráfico 3-13. Foto de las memorias probadas con el equipo GEODAS_USB.

Análisis de resultados: Como se puede apreciar el equipo GEODAS_USB al implementar la comunicación USB como *Host* es capaz de manejar cualquier tipo de memorias USB que trabaje con un modo de transferencia *Bulk*, independientemente del fabricante o de la densidad de memoria.

Ahora se muestra una tabla con el tiempo máximo en que una memoria USB, conectada al equipo GEODAS_USB, se llena de información (se supone que la memoria fue conectada vacía y en adquisición continua).

$$t_{duración} = \frac{1}{f_{muestreo}} x \frac{1}{\#bytes\ x\ muestra} x densidad\ de\ memoria$$

$$t_{duración} = \frac{1}{100\ muestras/segundo} x \frac{1}{6\ bytes/muestra} x \frac{1\ dia}{86400\ segundos} x densidad\ de\ memoria$$

Densidad	Duración [días]
512 Mbytes	≈ 10
1 Gbytes	≈ 20
2 Gbytes	≈ 41
4 Gbytes	≈ 82

Tabla 3-7. Tiempo máximo de datos almacenados por el equipo GEODAS_USB.

Análisis de resultados: el tiempo de duración resultante del cálculo realizado es aproximado debido a que la densidad de la memoria no es exacta, esto se puede comprobar al ver las propiedades de una memoria USB conectada a un computador.

3.6. SENSOR REAL

En esta prueba se busca comprobar que el equipo GEODAS_USB cumple con la tarea para la que fue diseñado.

Se realiza paso a paso los procedimientos de instalación, desde nivelar el sensor sísmico, conectar periféricos, instalación de la antena del módulo GPS, calibrar cada canal analógico y el almacenamiento de datos en una memoria USB.

Luego de eso se ilustrará el proceso para reemplazar una memoria con datos por una memoria en blanco, acompañado de una visualización de los datos almacenados.

3.6.1. NIVELAR SENSOR SÍSMICO

El sensor sísmico utilizado (Geodevice FBS-3B) tiene en su parte exterior un nivelador, y en su parte inferior tres patas ajustables en altura.

Se busca una superficie plana horizontal donde se asentará al sensor, y variando la altura de las patas se busca que la burbuja indicadora se encuentre centrada como se muestra en la figura:

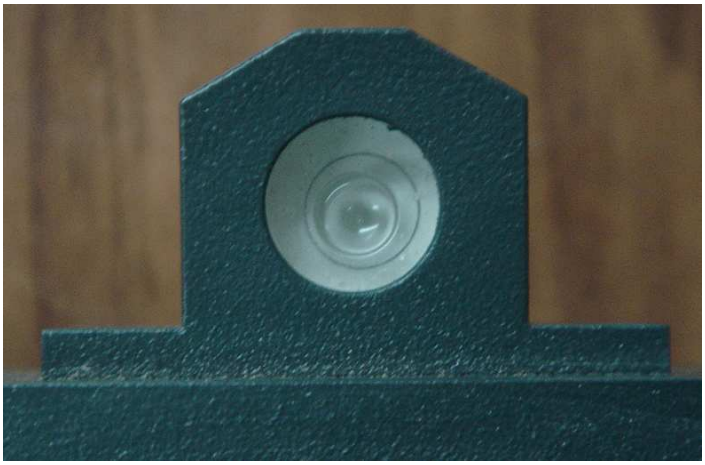


Gráfico 3-14. Foto del nivelador del sensor sísmico.

3.6.2. CONECTAR PERIFÉRICOS

El equipo GEODAS_USB tiene tres conectores, uno para alimentación (12 V dc), otro conector compatible con el sensor sísmico (Geodevice FBS-3B) y un tercero para el puerto de comunicación serial (DB9).

Terminado este procedimiento el equipo debería verse de la siguiente manera:

Gráfico 3-15. Foto de los conectores acoplados.

3.6.3. INSTALACIÓN DE LA ANTENA DEL GPS

Se debe ubicar la antena de tal manera que el logotipo de la empresa fabricante (Motorola) quede apuntando hacia arriba, y además garantizar que no exista ningún obstáculo que interfiera con la línea de vista hacia los satélites.

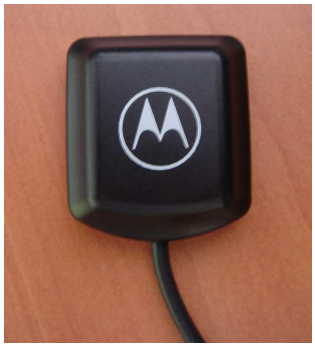


Gráfico 3-16. Foto de la antena del módulo GPS (M12+).

3.6.4. CALIBRACIÓN DE LOS CANALES ANALÓGICOS

Se energiza la tarjeta con el switch de alimentación general, se comprueba que los dos leds rojos de polarización estén encendidos, y se espera a que el led rojo que indica modalidad Adquisición se encienda.

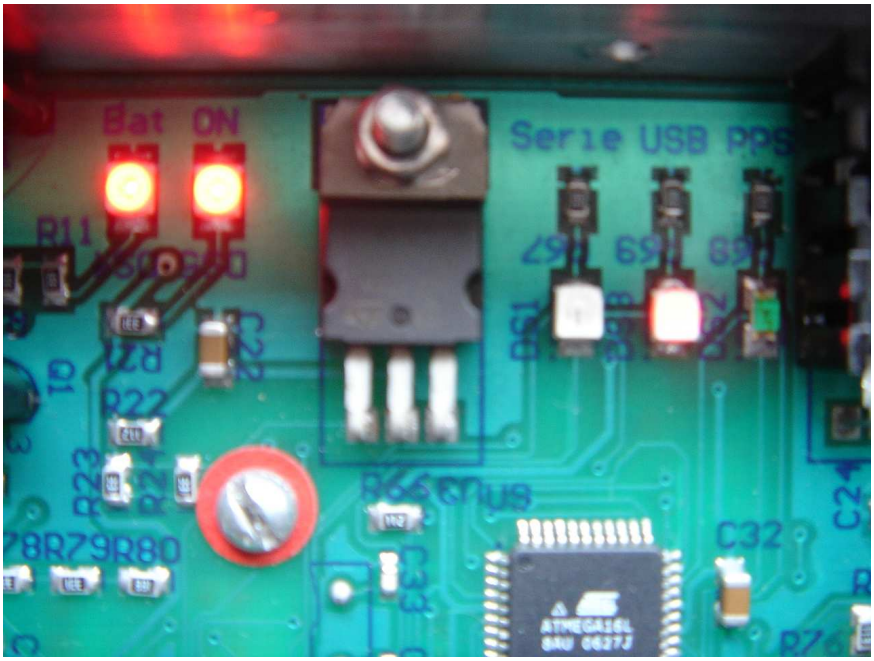


Gráfico 3-17. Foto leds indicadores.

Se corre la aplicación GEODAS GRAPHICS. Se da un clic en la etiqueta conectar y se espera que se enlace con el equipo GEODAS_USB, entonces aparecerá en la pantalla la información correspondiente a la estación (Latitud, Longitud, Altura, ID de estación).

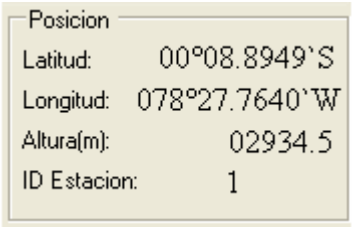


Gráfico 3-18. Información de la estación.

Continuando con el proceso se elije la opción “Calibración”, se observará en el equipo que el led naranja (Serie) está encendido y el led verde parpadeando cada segundo. En el computador se observará en tiempo real la información de cada canal. Normalmente la primera vez las señales estarán descalibradas como se muestra en la siguiente figura:

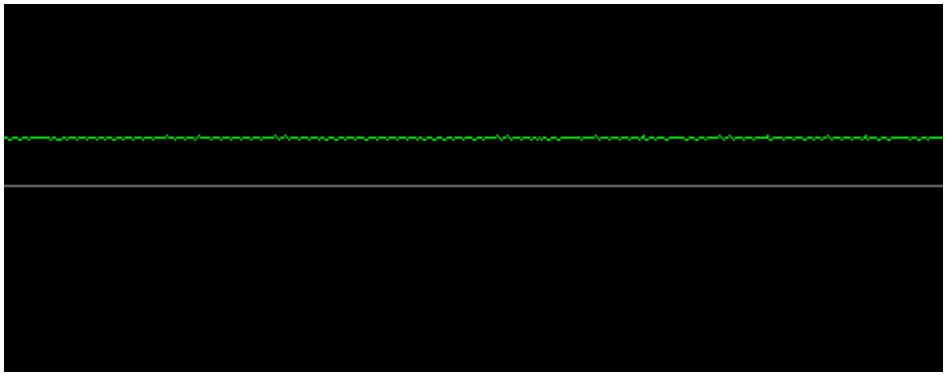
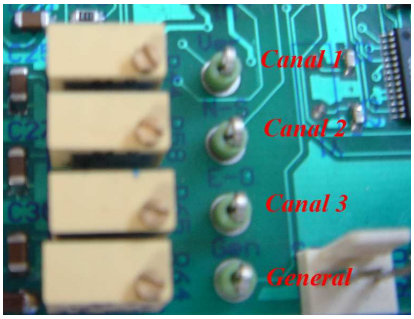


Gráfico 3-19. Canal descalibrado.

Para proceder a una correcta calibración se debe escoger la amplificación máxima que permite el equipo: “Ganancia = 15”. Y con ayuda de los potenciómetros de calibración del circuito analógico se deben mover las señales hasta llegar al nivel de referencia.



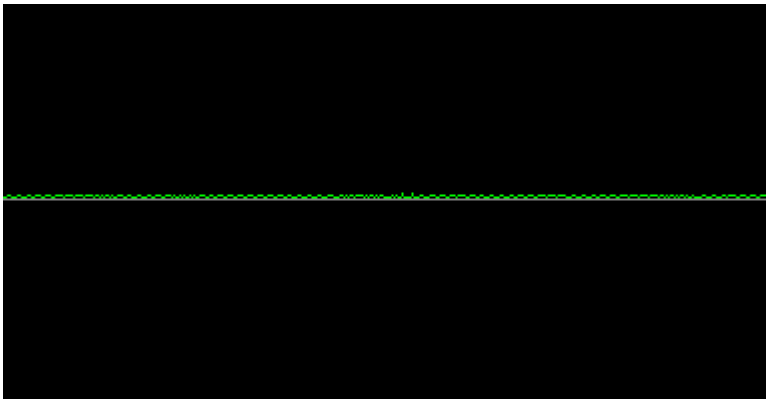


Gráfico 3-20. Potenciómetros de calibración y un canal calibrado.

A continuación se deben provocar perturbaciones en el sensor sísmico para que la selección de la ganancia sea adecuada, de acuerdo al criterio del sismólogo a cargo de la instalación del equipo.

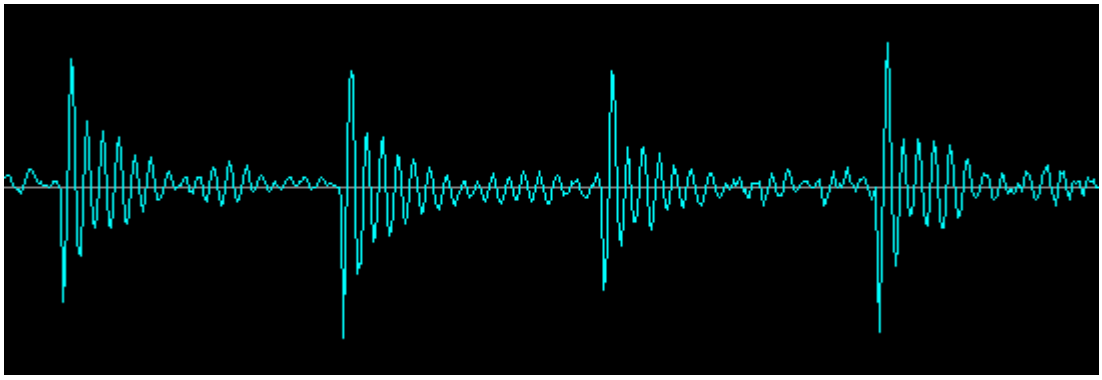


Gráfico 3-21. Señal sísmica con Ganancia = 5.

3.6.5. ALMACENAMIENTO DE DATOS

Se selecciona la opción “Adquisición” con lo cual el equipo GEODAS_USB está listo para almacenar los datos de los canales, previamente calibrados, en un dispositivo USB.

Una vez que se conecte la memoria USB al conector tipo A del equipo GEODAS_USB, se podrá ver la actividad en la memoria por medio del led indicador ubicado dentro de la misma.

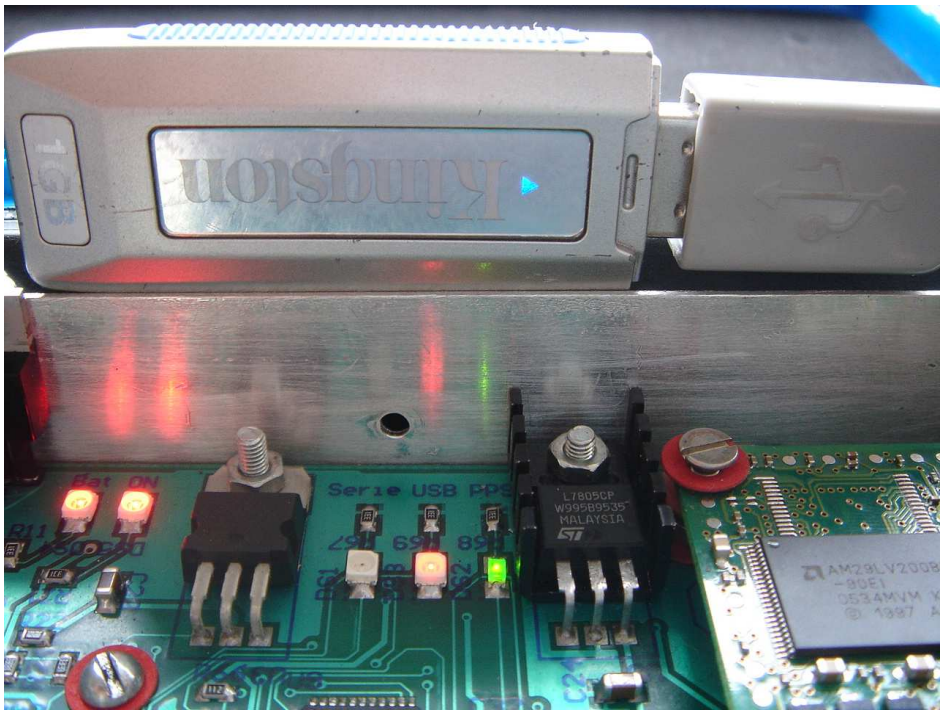


Gráfico 3-22. Foto con dispositivo USB conectado al equipo GEODAS_USB.

3.6.6. REEMPLAZO DEL DISPOSITIVO USB

Cuando se quiera visualizar los datos almacenados en el dispositivo USB, se tiene que retirar el hardware con seguridad manteniendo presionado el pulsante H/S hasta que la actividad en el led indicador de la memoria sea nula.

Luego se la pude retirar del conector tipo A, y se conecta una nueva memoria en blanco para que se sigan almacenando los datos (no es necesario reiniciar el equipo).

3.6.7. VISUALIZACIÓN DE DATOS

Para poder visualizar los datos almacenados por el equipo se debe ejecutar el programa USB, y abrir el archivo GEODAS.GDU.

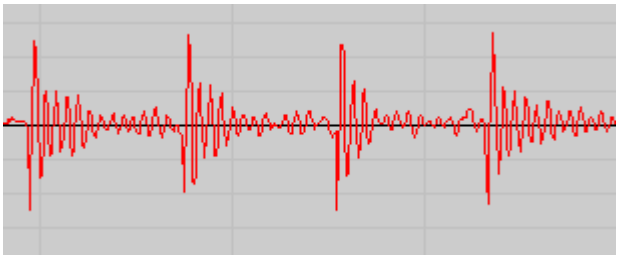


Gráfico 3-23. Gráfico de un archivo GEODAS.GDU.

Análisis de resultados: Si el proceso de instalación fue realizado correctamente (nivel del sensor sísmico, instalación de antena del módulo GPS, calibración de cada canal), el equipo GEODAS_USB cumple con todos los requerimientos planteados al inicio de su diseño.

Esta prueba fue expuesta a miembros del Instituto Geofísico de la Escuela Politécnica Nacional, los mismos que aprobaron el funcionamiento del equipo, basados en su experiencia en manejo de equipos sísmicos.

CAPÍTULO 4 CONCLUSIONES Y RECOMENDACIONES

4.1.CONCLUSIONES

El equipo GEODAS_USB maneja un dispositivo comercial (memoria USB) para el almacenamiento masivo de datos obtenidos como resultado de la cuantificación de las señales provenientes de un sensor sísmico de tres componentes.

El sistema de adquisición de datos del equipo GEODAS_USB, valiéndose de un circuito amplificador de ganancia variable, es capaz de registrar señales en el orden de milivoltios.

En la construcción del equipo GEODAS_USB se emplearon elementos eficientes y de bajo costo, además la alta calidad alcanzada en el circuito impreso permite minimizar el impacto del ruido eléctrico.

El resultado del eficiente trabajo de diseño y construcción hace que el equipo GEODAS_USB sea portátil y de muy sencilla instalación y manejo, facilitando el trabajo del personal del Instituto Geofísico en su continua labor de mitigar el impacto de los desastres naturales en nuestro país.

Para el manejo de la comunicación USB implementada en el equipo GEODAS_USB, se utilizó el controlador AT43USB380 (lanzado al mercado en febrero del 2005), el cual brinda una solución práctica y económica en el manejo de la comunicación USB como *HOST*.

El consumo de potencia del equipo GEODAS_USB es bajo en comparación con el entrenador comercial AT43DK380 con el cual se podría desarrollar una aplicación similar. La diferencia principal radica en los microcontroladores utilizados en cada tarjeta.

Para minimizar la sobrecarga en la información almacenada en la memoria USB se incorpora una cabecera, ésta contiene el tiempo donde fue tomada la primera muestra de datos y la información de la estación. A partir de esta cabecera los datos son almacenados en forma consecutiva en un formato binario. Para la interpretación de los datos se tiene en cuenta que existen tres canales de 16 bits cada uno y que la frecuencia de muestreo es de 100 Hz.

Una de las ventajas de los dispositivos que tienen incorporado el protocolo USB es su característica *plug&play,* por lo tanto, no se requiere que el sistema que actúa como *Host* sea reiniciado cada vez que se conecte un dispositivo USB, sino que lo reconoce y se configura para establecer la comunicación con dicho dispositivo.

Debido a que el protocolo USB ha tenido una muy buena aceptación en el mercado, existe una gran variedad de periféricos para diferentes aplicaciones con bajo costo.

Uno de los dispositivos más difundidos son las memorias flash USB (*memory stick*), debido a su bajo costo, reducido tamaño y gran capacidad de almacenamiento, actualmente llegan hasta 4 Gbytes de datos.

La velocidad de transmisión en el protocolo USB varía de acuerdo a la versión de la especificación, la versión 1.1 maneja velocidades de 1.5Mbps y 12Mbps, la versión 2.0 maneja velocidades de 1.5Mbps, 12Mbps y 480 Mbps.

El consumo de corriente en un sistema digital es proporcional a la velocidad de procesamiento del mismo.

En capa física el protocolo USB es un protocolo balanceado, utiliza dos líneas de datos (D+ y D-), además el receptor decodifica un dato diferencial lo cual ayuda a eliminar el ruido que se pueda introducir en el cable.

En protocolo USB emplea una codificación de línea *NRZI* con relleno de bits para evitar cadenas largas de unos, provocando mayores transiciones en la señal codificada, mejorando así el sincronismo.

El protocolo USB en capa enlace es desbalanceado, definiendo un sistema maestro (*HOST*) y un sistema esclavo (*DEVICE*), por este motivo el canal de transmisión es administrado por el *HOST*.

Para la detección de errores el protocolo USB emplea un campo *CRC*, si se detecta un error empleando un modo de transmisión confiable se pide retransmisión.

El protocolo USB puede implementar cuatro modos de transmisión de los cuales tres son confiables (*Bulk*, Interrupción y Control) y uno no confiable (Isócrono).

Cuando un dispositivo USB es conectado al bus pasa por los siguientes estados: polarizado, reseteado, direccionado y configurado.

Para que el *HOST* USB pueda determinar cómo debe administrar la comunicación con un cierto dispositivo conectado a su puerto, pide los descriptores mediante el *endpoint* de control, en ellos se indica información del dispositivo como por ejemplo: modo de transmisión, corriente máxima de consumo, período de poleo, número de *endpoints* empleados, etc.

Para poder crear un archivo en la memoria que pueda ser interpretado por el sistema operativo de un computador, se requiere de un sistema de manejo de archivos conocido como *FAT* (Tabla de asignación de archivos).

Existen varias versiones de *FAT* como son *FAT12*, *FAT16* y *FAT32,* la diferencia más significativa entre ellos es la cantidad de bytes por cluster que manejan; para volúmenes de alta densidad se emplea *FAT32,* el cual genera el menor número de bytes por cluster.

Un sistema GPS está formado por tres partes: segmento espacio (los satélites), segmento control (estaciones encargadas del correcto funcionamiento de los satélites) y segmento usuario (los receptores en tierra).

Para triangular una señal espacial se requiere de al menos tres puntos conocidos; pero si se toma en cuenta una variable adicional, como es el tiempo de propagación de la señal en distancias grandes, se requiere un mínimo de cuatro puntos.

Un sensor sísmico genera señales eléctricas, cuando una bobina suspendida oscila dentro de un campo magnético fijo, las señales inducidas son proporcionales al movimiento de la bobina.

Para poder obtener un sensor sísmico de banda ancha se realiza una retroalimentación negativa de posición de la bobina suspendida, con lo que el sistema sísmico puede detectar frecuencias muy bajas, menores a 1 Hz.

El controlador AT43USB380 internamente tiene dos microcontroladores, uno implementa las capas física y enlace del protocolo USB, y el otro gestiona la interfaz de comunicación con un microcontrolador externo.

Para que el AT43USB380 trabaje, se necesita descargar un programa por cada microcontrolador interno, este proceso se realiza cada vez que es inicializado el controlador.

El AT43USB380 tiene un bus de ocho bits para dirección y un bus variable para datos (8,16 o 32 bits).

Los microcontroladores AVR utilizados presentan grandes ventajas como son: gran capacidad de memoria de programa, pueden ser grabados en la placa, tienen memoria eeprom interna; existen compiladores en lenguaje de alto nivel para programarlos, son veloces (1 ciclo de máquina = 1 ciclo de reloj) y trabajan en un rango de voltaje de polarización de 2.7 a 5 V.

Existen módulos GPS económicos y con una interfaz de fácil manejo, lo cual permite acceder al servicio de datos GPS (posición y hora global) e incorporarlos a cualquier sistema.

En el mercado existes chips que son relojes a tiempo real, por ejemplo el DS1307. Su función es la de llevar la cuenta de la fecha y hora, inclusive registrando años bisiestos. Además generan una señal cuadrada a un 1Hz utilizada para sincronismo.

Un conversor *Sigma-Delta* tiene un mejor rechazo al ruido de entrada para la cuantificación, pero es lento comparado con conversores veloces, como uno de rampa. Esto limita a este tipo de conversor a trabajar en un rango de frecuencias bajas.

El integrado AD7738 solo maneja señales unipolares en diferentes rangos de voltaje (+625 mV, +1.25 V, +2.5 V, ±625 mV, ±1.25 V, ±2.5 V), configurables por medio de una interfaz SPI, la misma que es usada para la lectura de datos de cada canal.

El conversor AD7738 puede multiplexar hasta 8 canales, y permite agregar una etapa de acondicionamiento externo para la señal multiplexada.

Con ayuda de un amplificador operacional se puede desplazar cualquier señal a un voltaje DC requerido.

Con un arreglo de resistencias en ponderación binaria y *switches* analógicos (4066), se puede obtener un potenciómetro digital de avance lineal, que al formar parte de un amplificador provoca una variación de la ganancia, proporcional al valor del potenciómetro.

El error obtenido al realizar las pruebas del circuito de ganancias variables se debe a que el switch analógico utilizado (4066) no tiene contactos ideales y cada

uno presenta una impedancia pequeña, la cual sumada al arreglo de resistencias, varía un poco la respuesta del circuito.

Los circuitos diseñados con amplificadores operacionales tienen un excelente rechazo al ruido. Así la señal de entrada puede ser ruidosa, pero al momento de pasar por un seguidor de tensión o por un circuito desplazador de voltaje la señal de salida estará ya filtrada.

Al analizar las señales de baja frecuencia y de amplitud reducida (ej. 100 mV) con un osciloscopio digital se tienen problemas de visualización y no se puede apreciar claramente la repuesta de un circuito analógico a ese tipo de señales.

Para alargar el tiempo de vida útil de una batería no se la debe descargar por completo, así el equipo debe controlar que si el voltaje decae demasiado se debe reducir el consumo de energía a lo mínimo para permitir que ésta se pueda recargar por medio de un panel solar.

4.2.RECOMENDACIONES

Debido a la tendencia en electrónica a emplear sistemas de montaje superficial, los diseños de nuevos equipos deben ser diseñados usando elementos superficiales (*SMD*). Además por su reducido tamaño se pueden alcanzar equipos mucho más portátiles.

Para facilitar el montaje del equipo en el campo, se debe diseñar una aplicación para un dispositivo portátil (*handheld*) que permita la calibración en tiempo real de cada canal del sistema, utilizando el puerto USB ya implementado en el equipo.

Para darle mas versatilidad al equipo se recomienda que se implemente un sistema de envió de datos por radio frecuencia, utilizando un nuevo dispositivo que utilice el puerto USB ya existente en el equipo.

Se recomienda que se incorpore dentro del pensum académico un estudio más a fondo del protocolo USB, el cual por su versatilidad está reemplazando a interfaces como serie y paralelo.

Se recomienda que se incentiven más proyectos prácticos como temas de titulación, porque su desarrollo involucra un mayor dominio del tema seleccionado.

REFERENCAS BIBLIOGRÁFICAS

[1] Universal Serial Bus Revision 2.0 specification
(http://www.usb.org/developers/docs/)

[2] Mass Storage Bulk Only 1.0
(http://www.usb.org/developers/devclass_docs#approved)

[3] Understanding FAT32 Filesystems
(http://www.pjrc.com/tech/8051/ide/fat32.html)

[4] GPS Global Position System
(http://www.geocities.com/puertoweb1/gps.html)

[5] Sismógrafos mecánicos y electromagnéticos
(http://www.ssn.unam.mx/SSN/instrumentacion1.html)

[6] AT43USB380 Datasheet complete
(http://www.atmel.com/products/usb/forms/splash.asp#380)

[7] ATmega128L
(http://www.atmel.com/dyn/products/product_card.asp?family_id=607&family_name=AVR+8%2DBit+RISC+&part_id=2018)

[8] ATmega16L
(http://www.atmel.com/dyn/products/product_card.asp?family_id=607&family_name=AVR+8%2DBit+RISC+&part_id=2010)

[9] M12+ Oncore Users Guide
(http://www.synergy-gps.com/content/view/35/60/)

[10] DS1307 Datasheet
(http://www.maxim-ic.com/quick_view2.cfm/qv_pk/2688)

[11] AD7738 Datasheet
(http://www.analog.com/en/prod/0,2877,AD7738,00.html)

[12] AT43USB380 Integration Guide
(http://www.atmel.com/products/usb/forms/splash.asp#380)

[13] AXELSON, Jan, USB Complete: Everything You Need to Develop Custom USB Peripherals, Segunda Edición.

[14] AT43USB370/380 USB Precessor Library Software Development Guide for Host Mode
(http://www.atmel.com/products/usb/forms/splash.asp#380)

[15] HORENSTEIN, Mark, Circuitos y Dispositivos Microelectrónicas, Pretince-Hall Hispanoamericana, Segunda Edición, Mexico, 1997.

[16] LT1083-Fixed – 7.5 A Low Drop Positive Fixed Regulator
(http://www.linear.com/pc/productDetail.do?navId=H0,C1,C1003,C1040,C1055,P1278)

[17] JAMSA, Kris, Aprenda c++ paso a paso, Editorial Alfaomega, Segunda Edición, México, 1997.

[18] AVR Product Line Introduction
(http://www.atmel.com/products/avr/overview.asp)

[19] HYDE, John, USB Design by Example, Segunda Edición, 2004.

Anexos

ANEXO A

Esquemáticos del Equipo GEODAS_USB

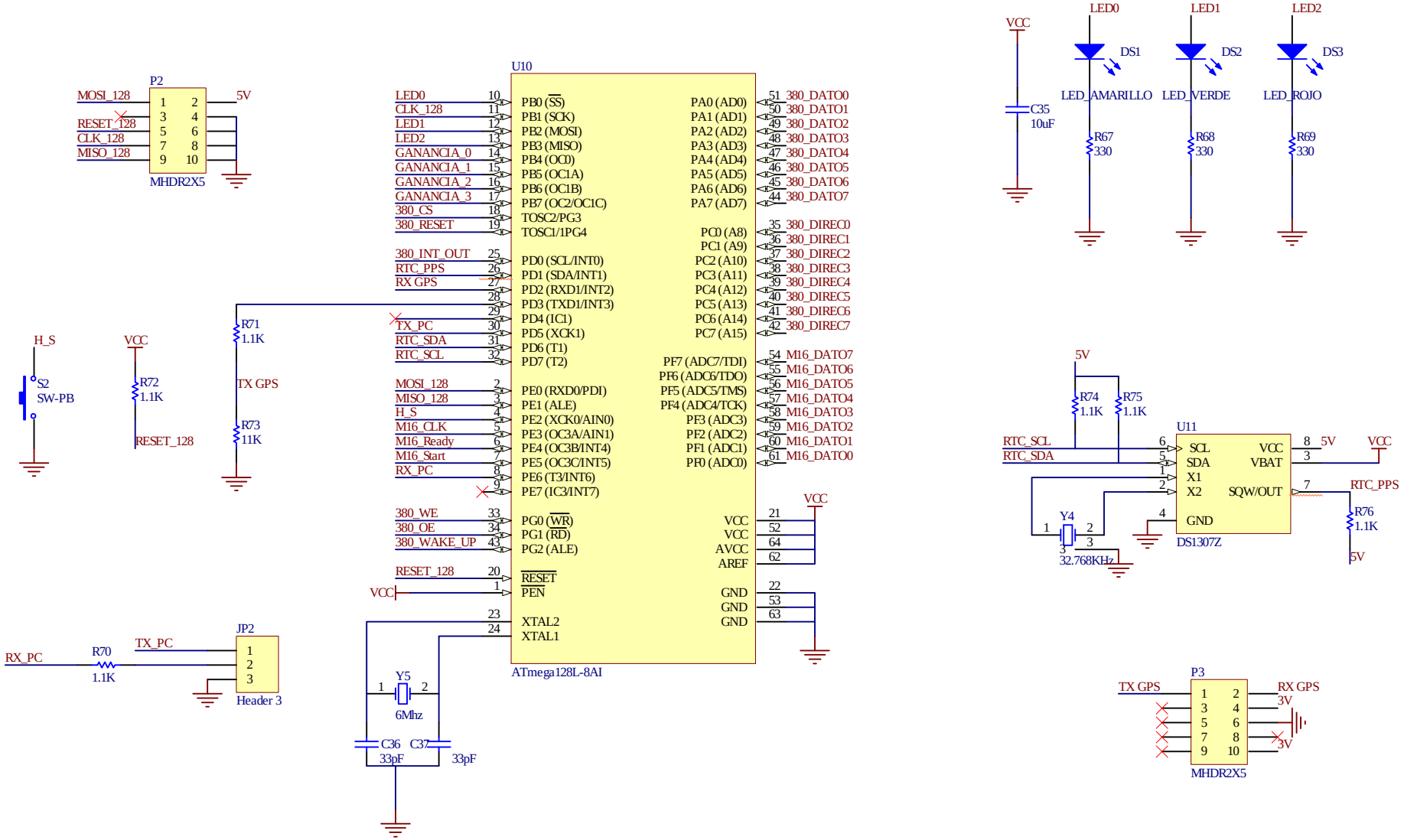


Gráfico A-1. ATmega128L.SchDoc

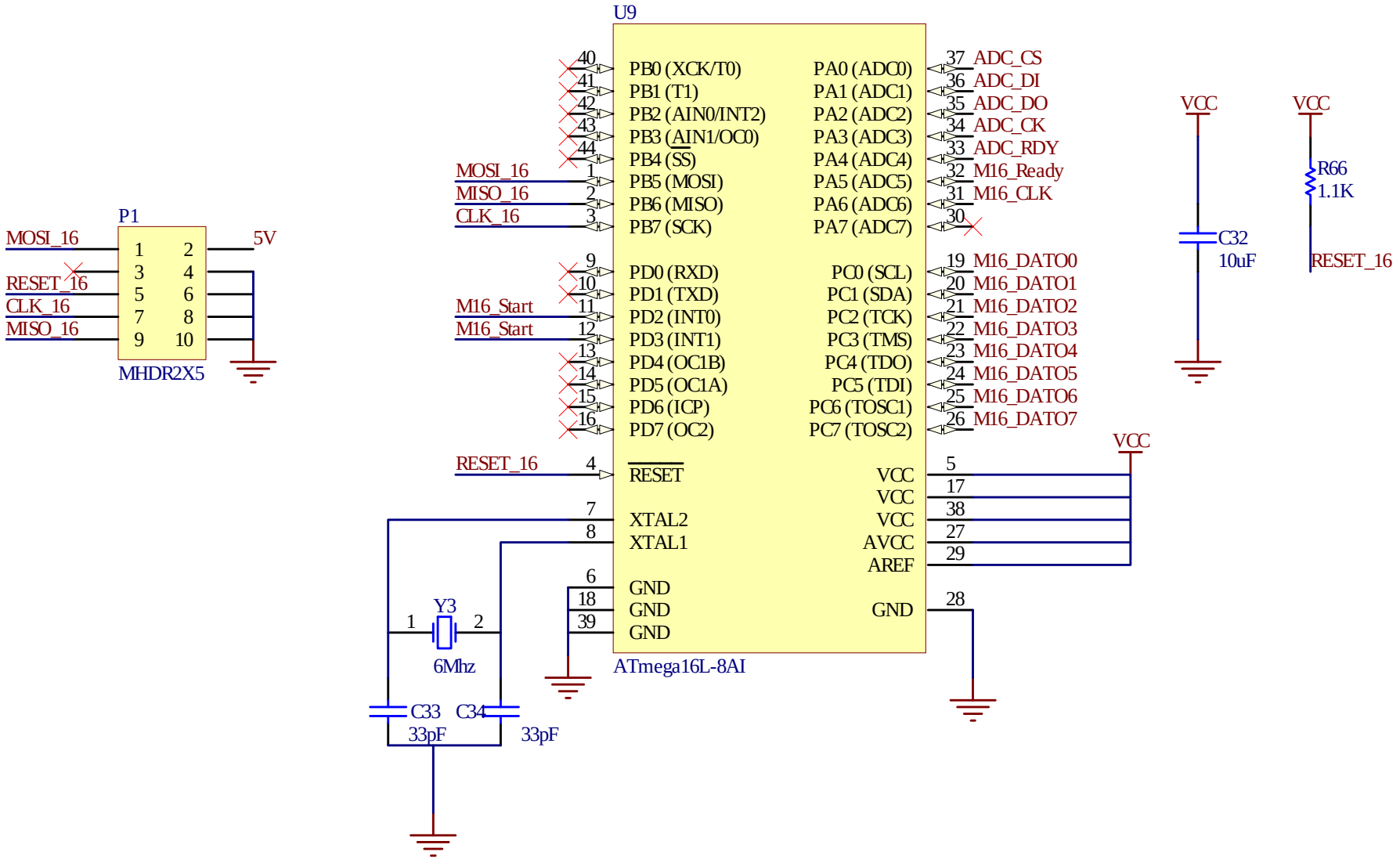


Gráfico A-2. ATmega16L

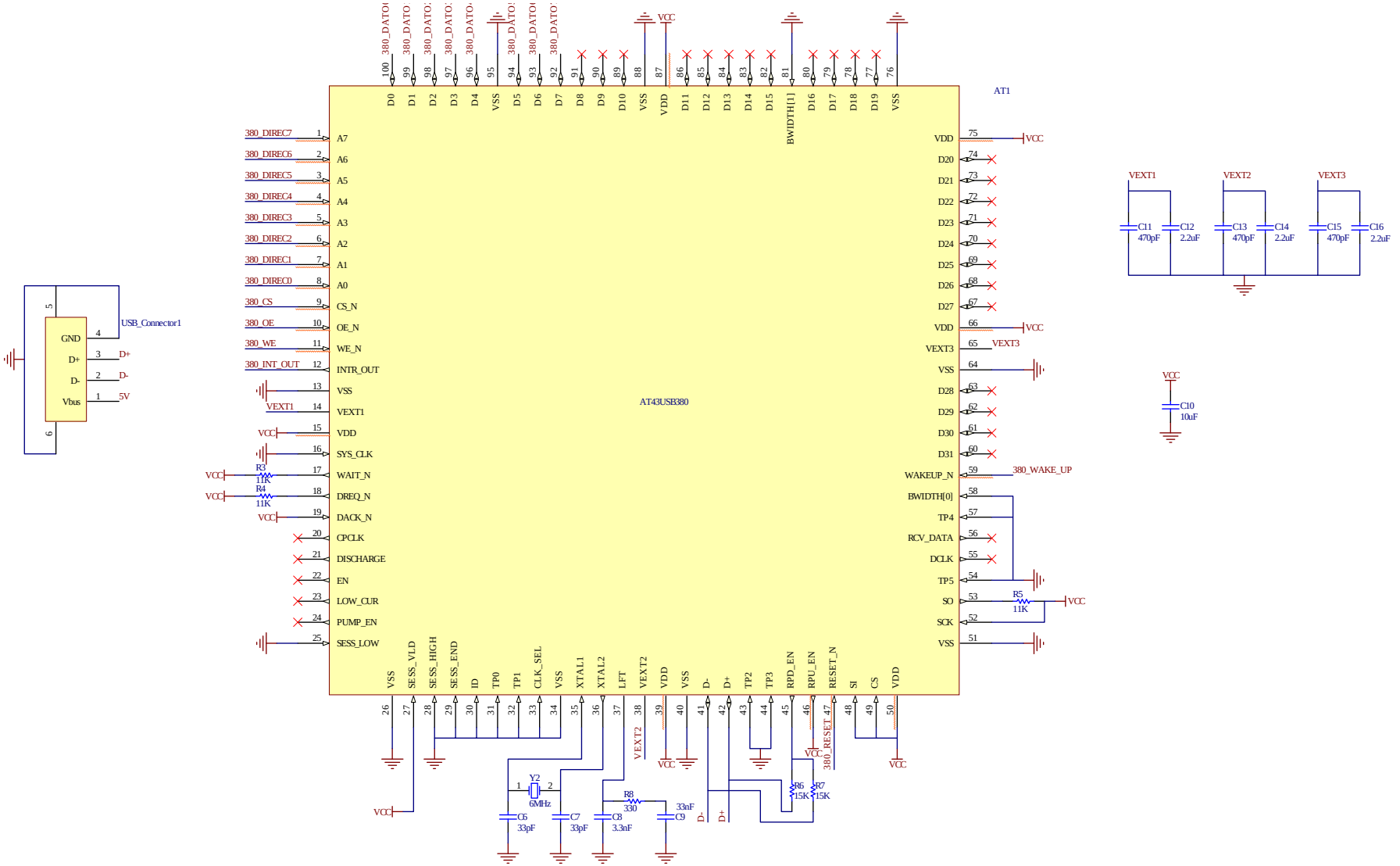


Gráfico A-3. AT43USB380.SchDoc

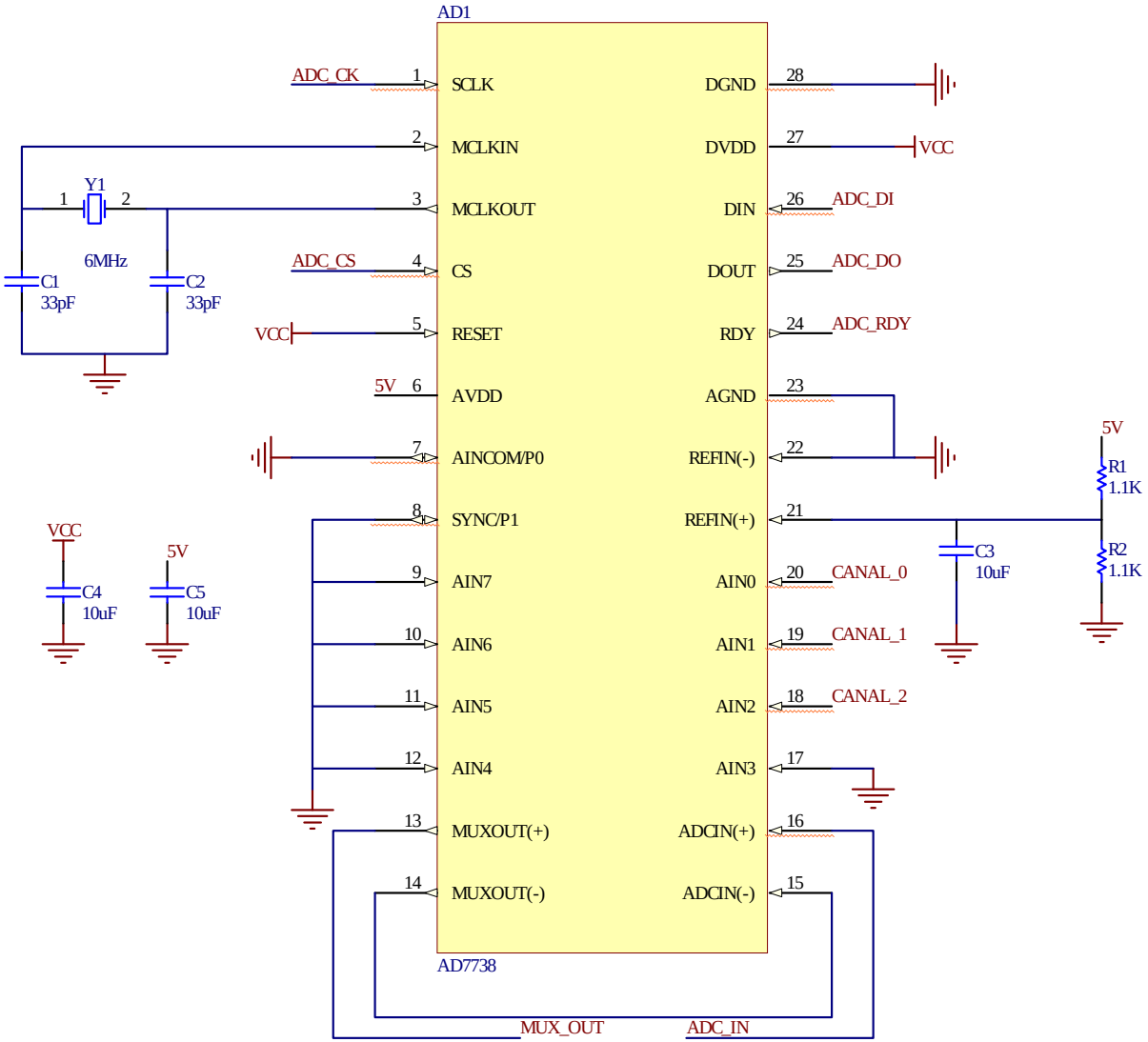


Gráfico A-4. ADC7738.SchDoc

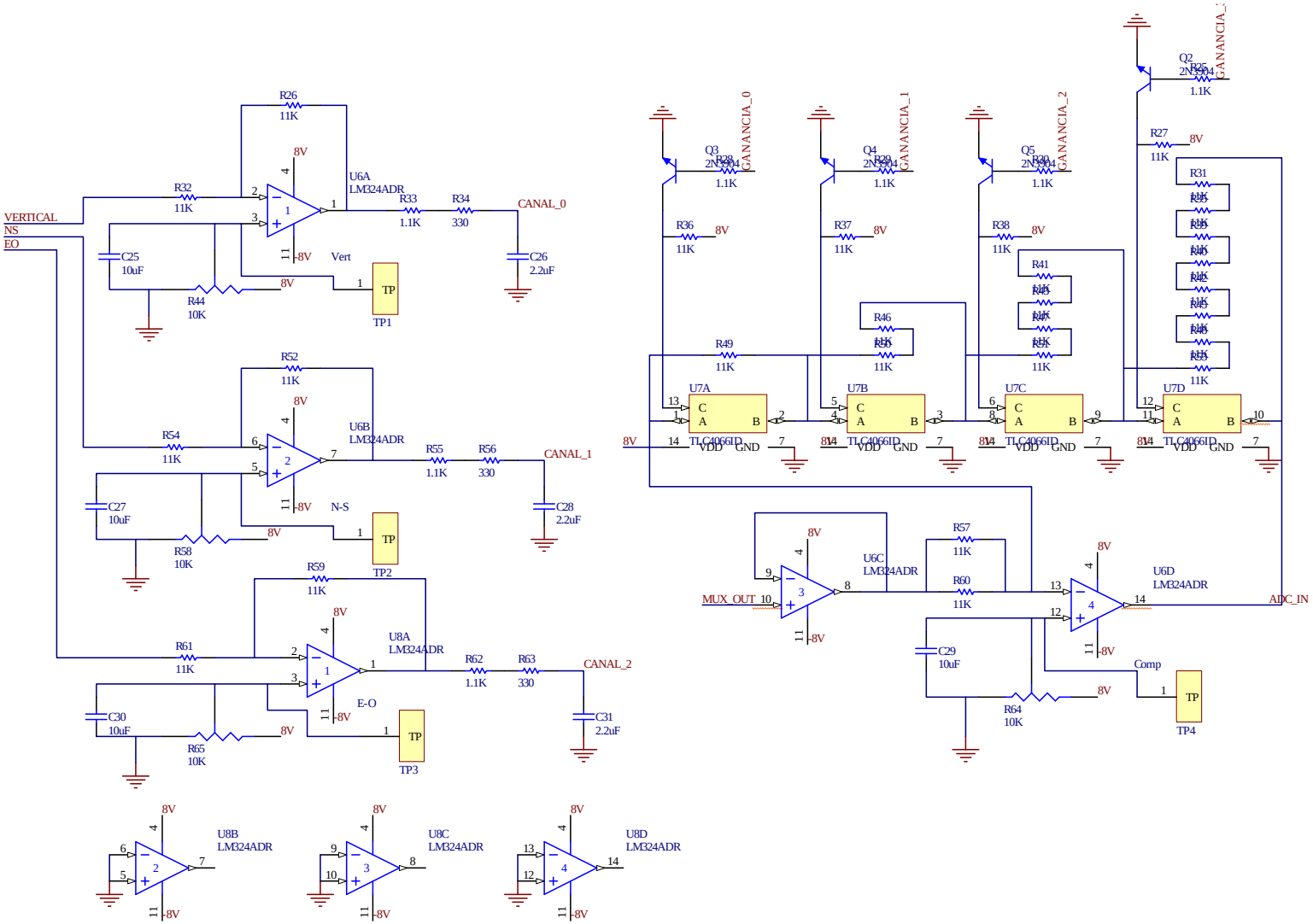


Gráfico A-5. Ganancia.SchDoc

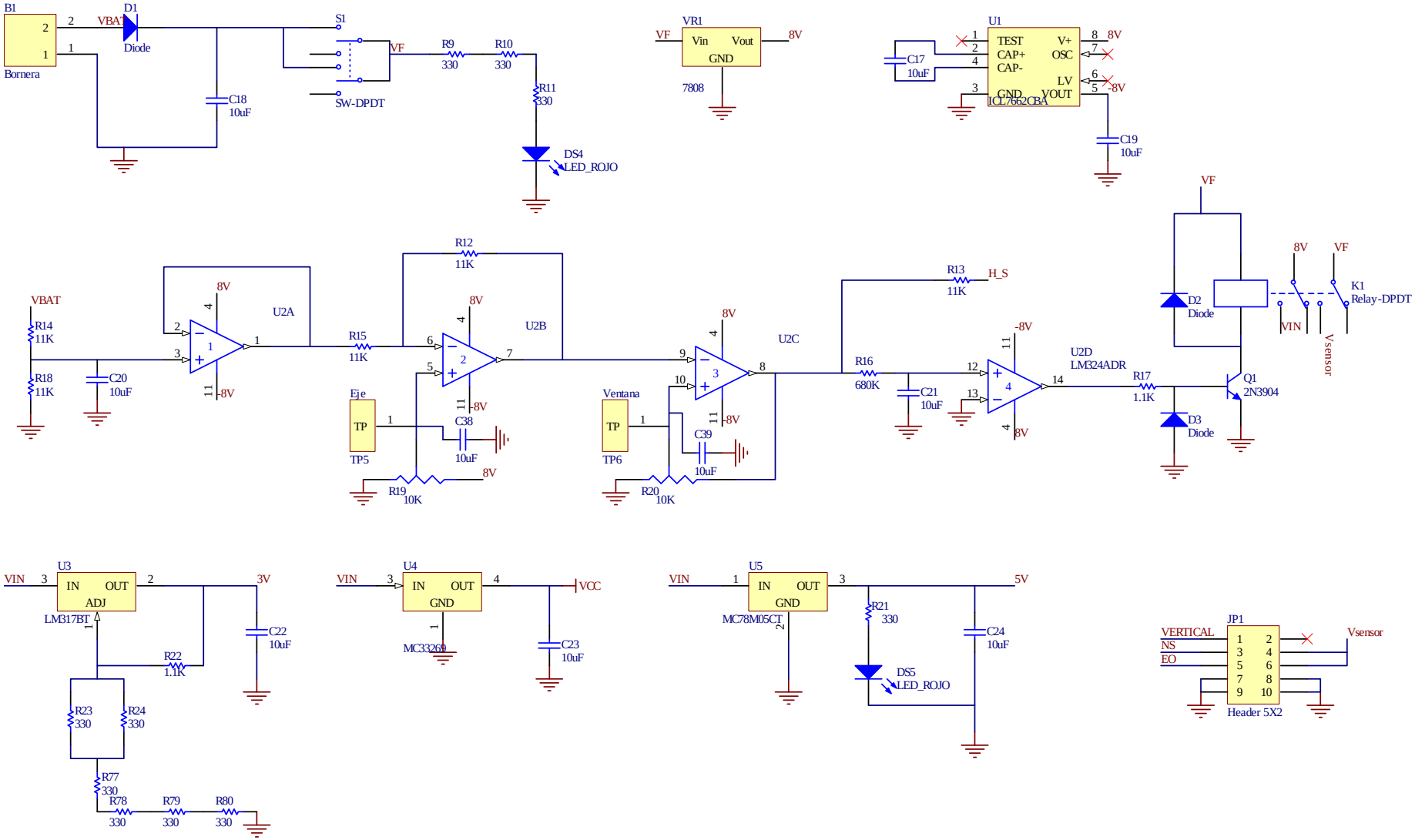


Gráfico A-6. Fuentes.SchDoc

ANEXO B

Láminas PCB del Equipo GEODAS_USB

Gráfico B-1. Capa Superior

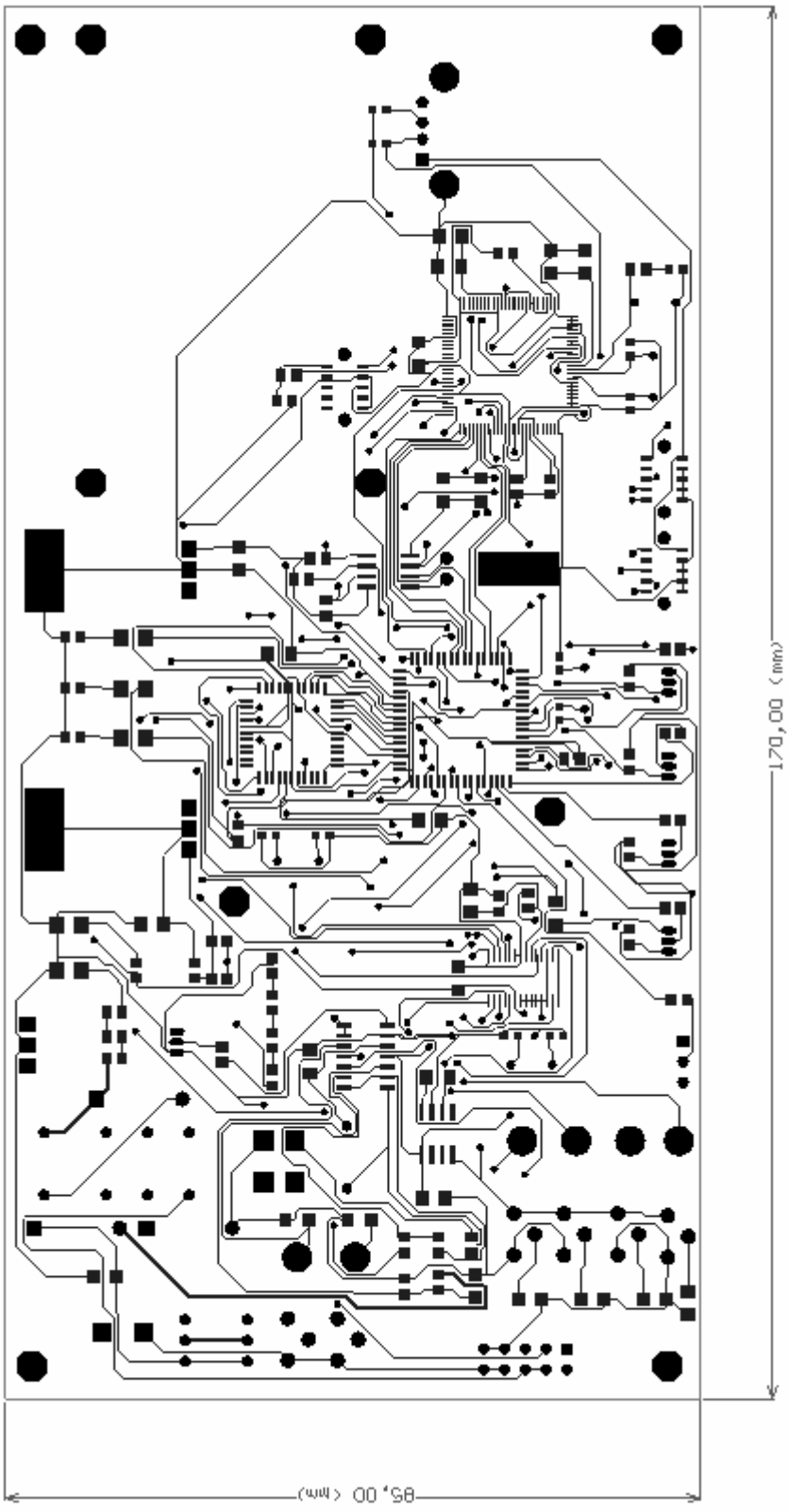


Gráfico B-2. Capa Inferior

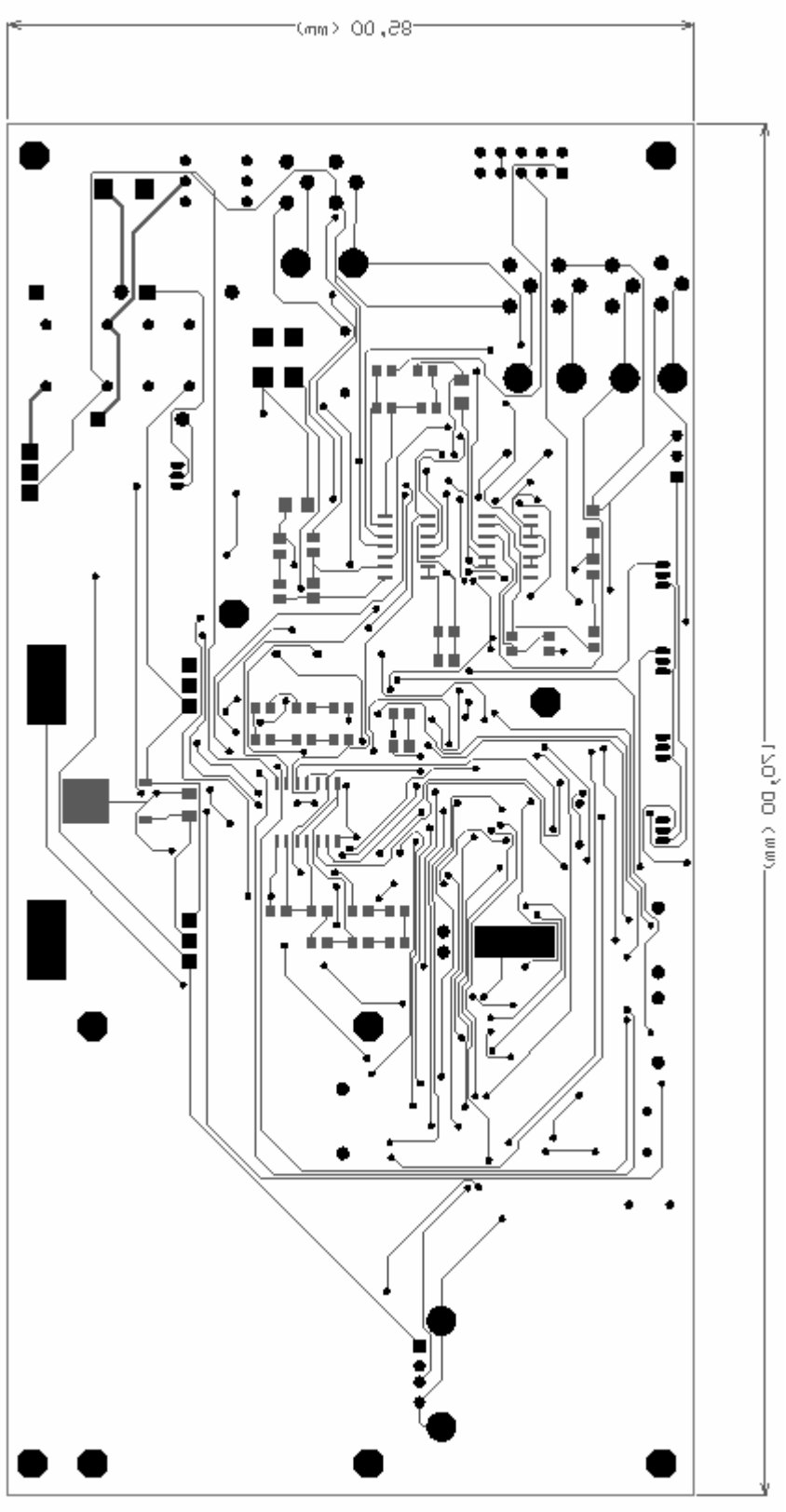


Gráfico B-3. Texto Capa Superior

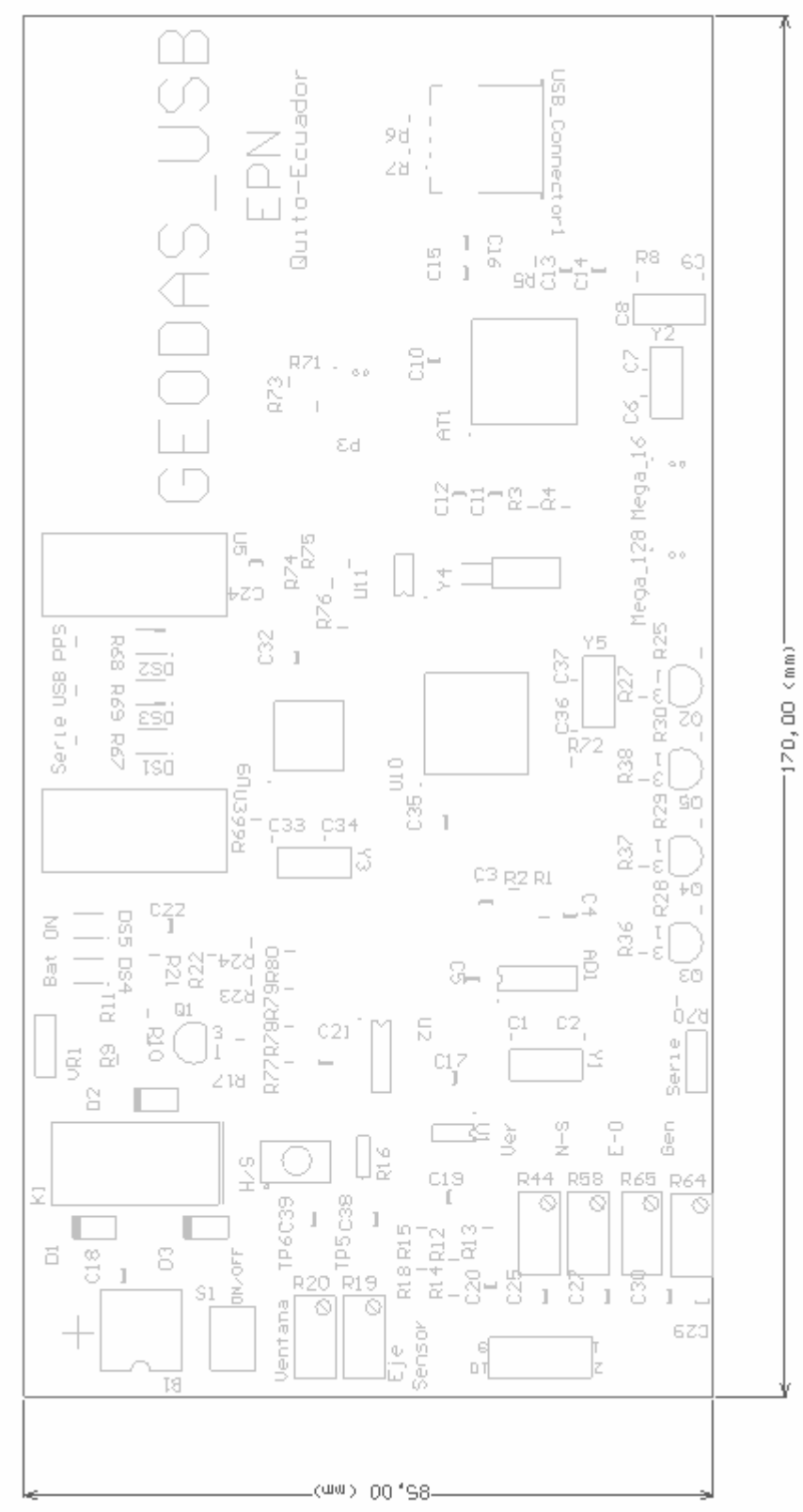


Gráfico B-4. Texto Capa Inferior

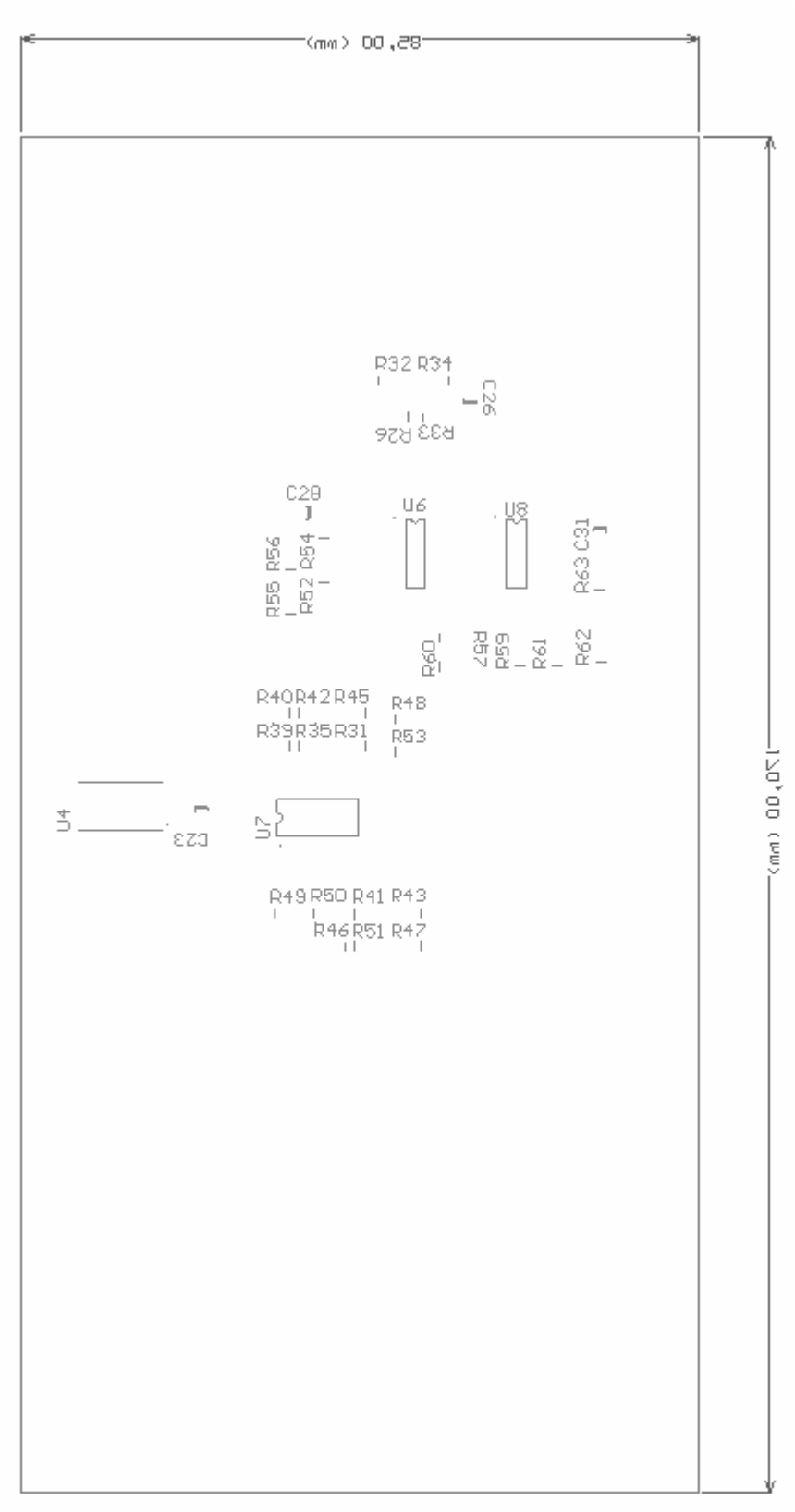


Gráfico B-5. PCB Grabador

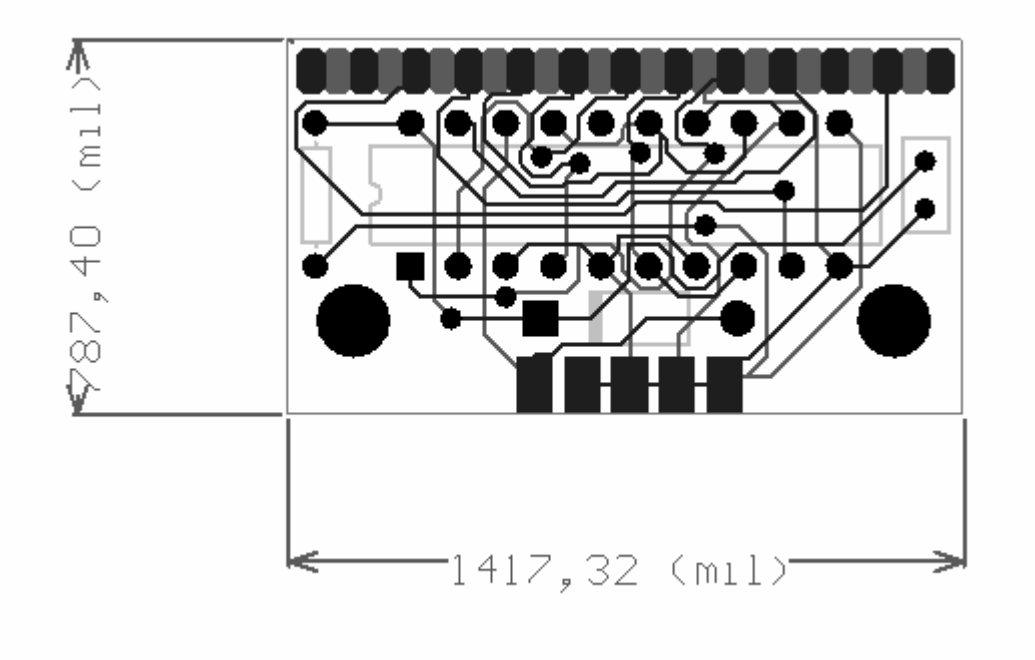
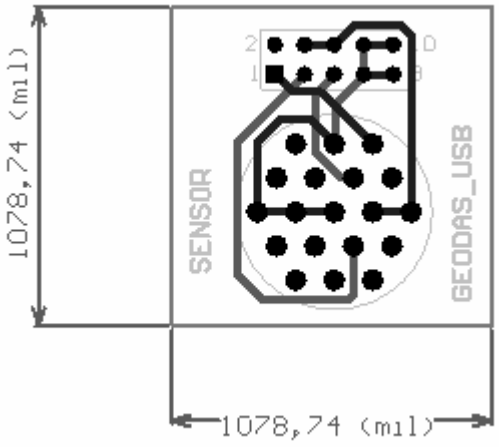


Gráfico B-6. PCB Conector Sensor



ANEXO C

Fotos del Equipo GEODAS_USB

Gráfico C-1. Vista Superior Equipo GEODAS_USB

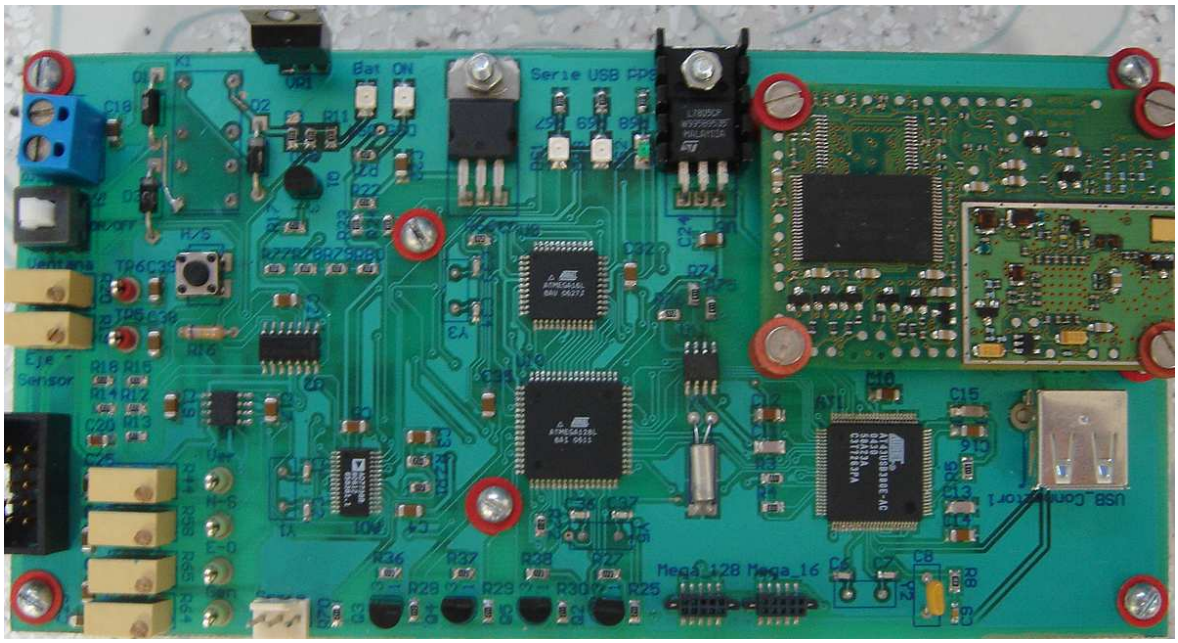


Gráfico C-2. Vista Inferior Equipo GEODAS_USB

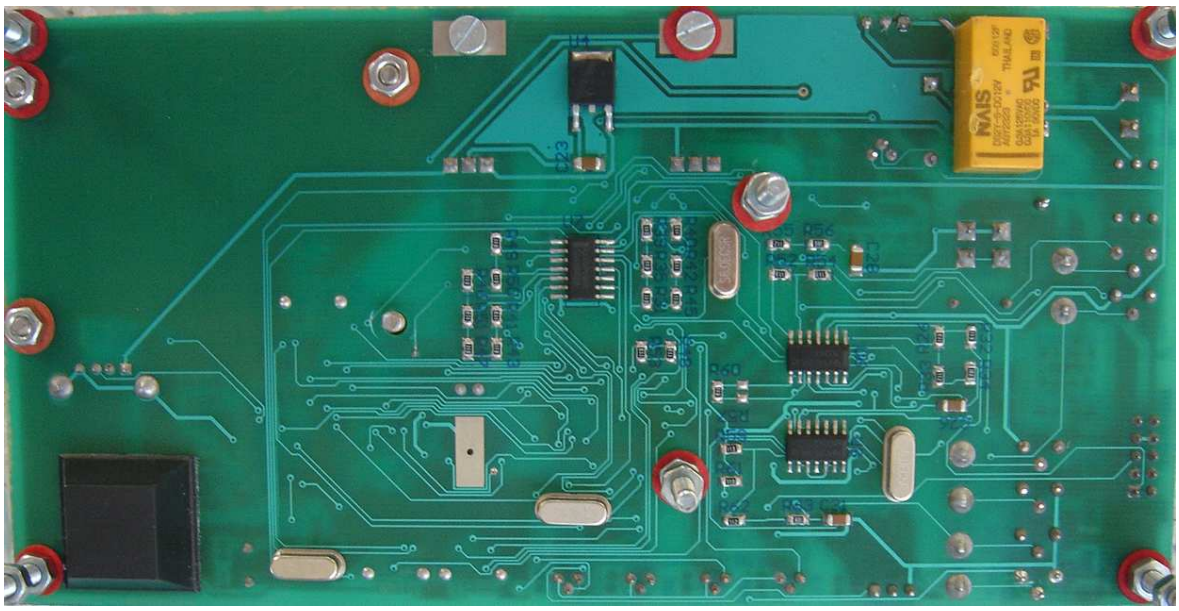


Gráfico C-3. Vista Módulo GPS M12+

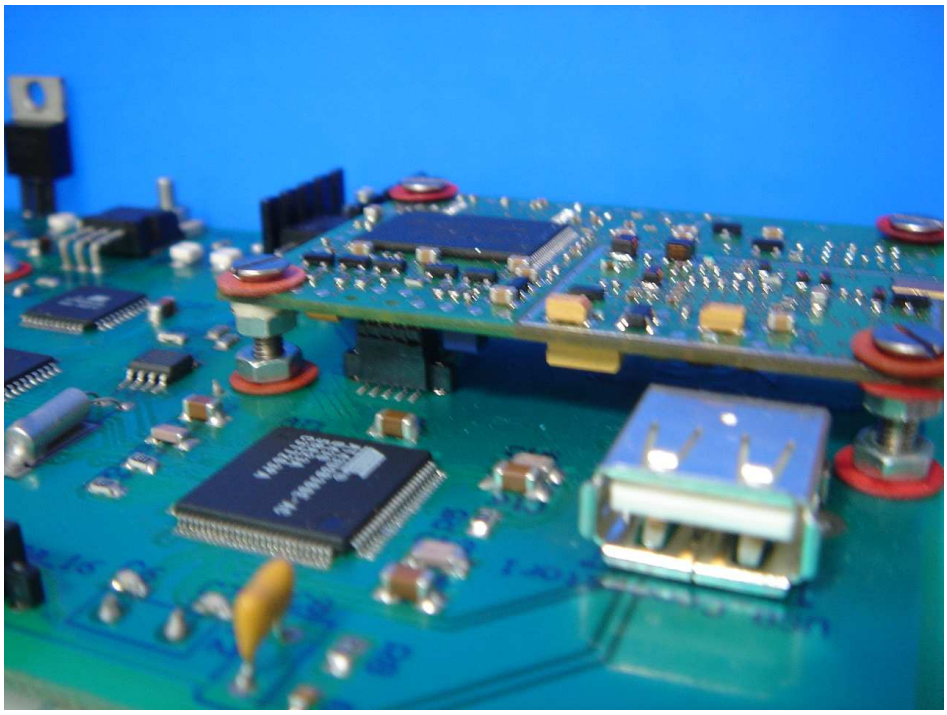


Gráfico C-4. Circuito de Ganancia Variable

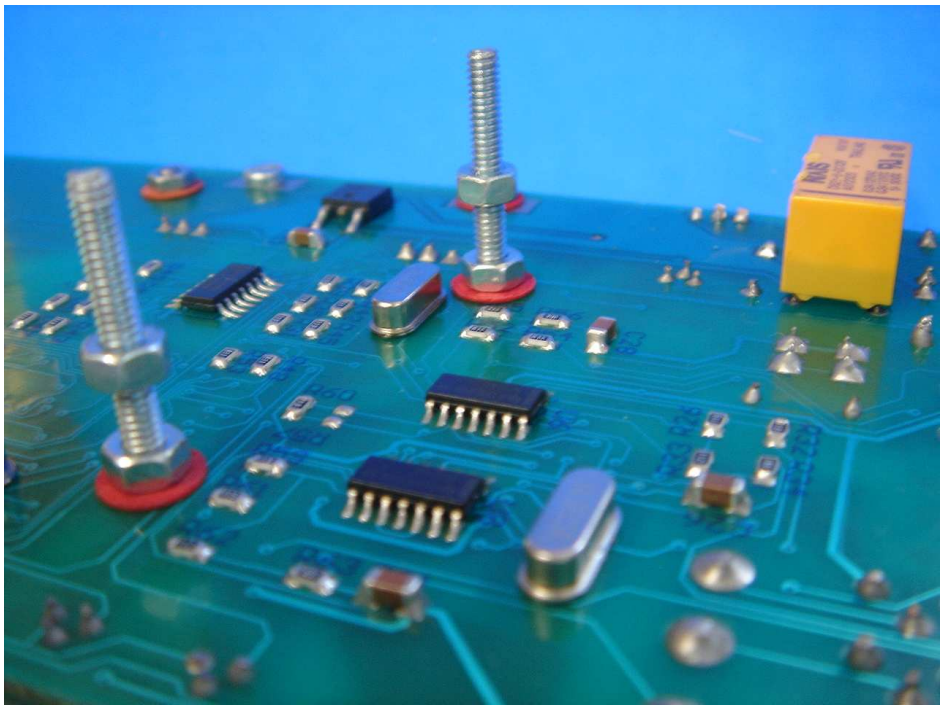


Gráfico C-5. Controlador USB

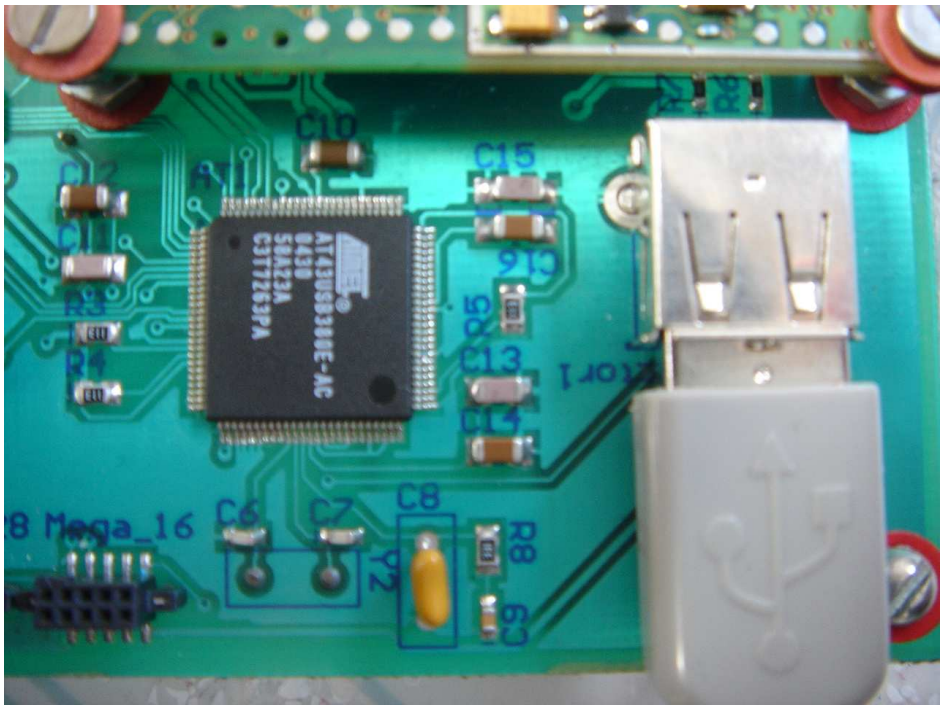


Gráfico C-6. Equipo GEODAS_USB

ANEXO D

Manual de Usuario del Equipo GEODAS_USB

MANUAL DE USUARIO

GEODAS_USB V 1.0

ÍNDICE

MANUAL DE USUARIO

GEODAS_USB v 1.0

1. INTRODUCCIÓN

El equipo GEODAS_USB es una herramienta para el monitoreo sísmico y volcánico, el cual recoge datos provenientes de un sensor sísmico banda ancha de tres componentes, los sincroniza con la hora global por medio de un módulo GPS y los almacena en un dispositivo USB.

En este manual se indicará el procedimiento correcto de calibración, condiciones de uso y análisis de los datos. Así también se explicará el uso de los programas utilitarios GEODAS GRAPHICS y USB.

2. PROGRAMAS UTILITARIOS

2.1. GEODAS GRAPHICS

Este programa fue desarrollado en la plataforma de Visual .NET 2005 y para su funcionamiento sólo se necesita tener instalado el controlador: Microsoft .NET Framework 2.0. El cual puede ser descargado gratuitamente desde el Internet.

El programa GEODAS GRAPHICS funciona como un osciloscopio digital para las señales aceptadas por el equipo GEODAS_USB. Al ejecutarlo se observa la siguiente ventana:

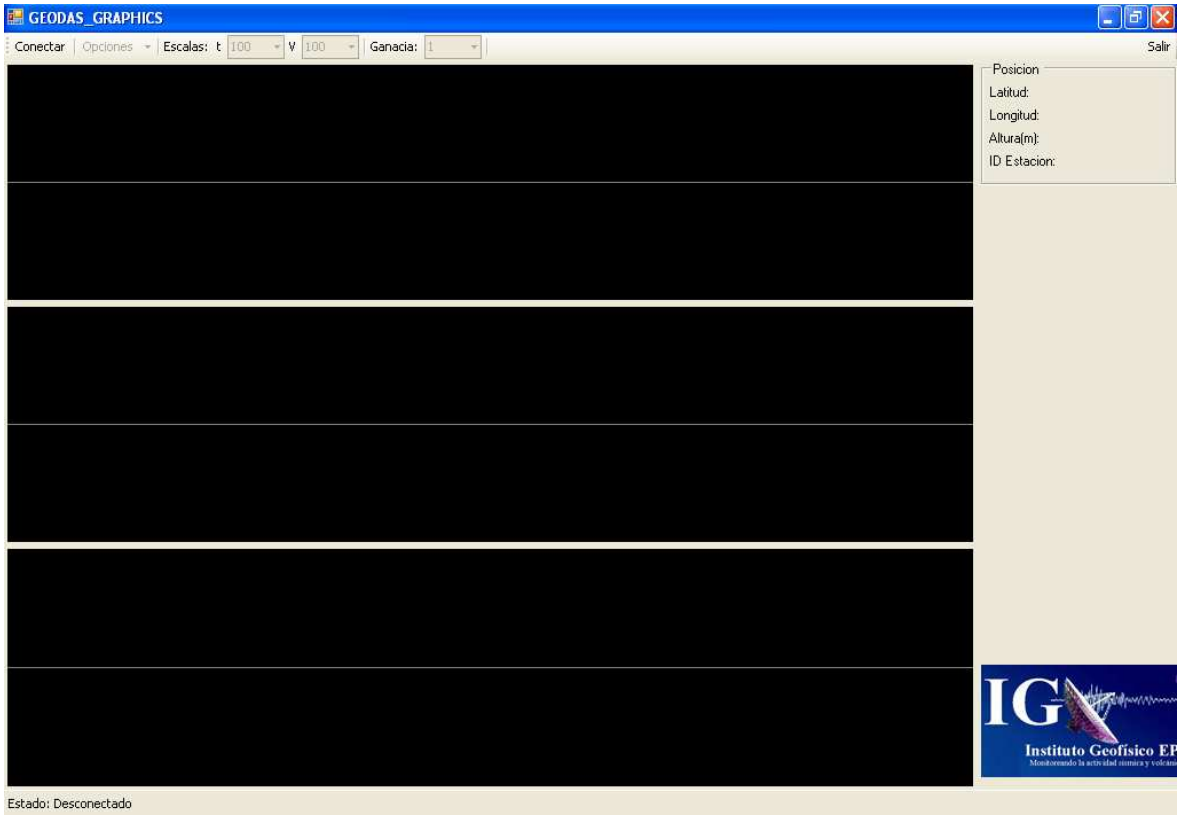


Gráfico D-24. Interfaz gráfica del programa GEODAS GRAPHICS

Teniendo encendido el equipo GEODAS_USB se debe dar un clic sobre la palabra “Conectar”, el programa automáticamente encuentra en qué puerto está conectado el equipo y muestra a la derecha de la pantalla la información del equipo: Latitud, Longitud, Altura y el ID de estación. Además la leyenda del botón accionado cambia a “Desconectar”.

De no estar conectado el equipo al puerto, o de encontrarse apagado, en la parte inferior izquierda se muestra el mensaje “Estado: Equipo no encontrado”, de haberlo encontrado se observará el mensaje “Estado: Conectado Com x”.

Una vez enlazados el programa y el equipo se habilita la palabra “Opciones”, donde se despliega un menú: “Calibración”, “Adquisición” e “Inactividad”.

Calibración: Al dar un clic sobre esta opción, el equipo GEODAS_USB envía los datos adquiridos por el puerto serial, mostrándose en pantalla las trazas de cada componente. Solo deja habilitada la opción “Inactividad”.

Se habilitan las ventanas de Escalas en tiempo y voltaje, y la de Ganancia.

Al modificar la escala de tiempo se puede observar las señales adquiridas ampliadas o reducidas en tiempo. La escala de voltaje permite hacer una ampliación o reducción en software de los datos recibidos del equipo y que serán mostrados en la pantalla.

La opción de Ganancias, envía un comando al equipo con la ganancia analógica deseada, este cambio se notará en la pantalla al ser graficados los datos enviados por el equipo GEODAS_USB.

Adquisición: Al escoger esta opción el equipo GEODAS_USB hablita su opción de guardar los datos adquiridos el una memoria USB. Este proceso de grabación comienza siempre y cuando una memoria esté conectada al puerto del equipo. Sólo deja habilitada la opción “Inactividad”.

Inactividad: Esta opción detiene los otros procesos, ya sea de Calibración o Adquisición. Habilita las opciones “Calibración” y “Adquisición”.

Una vez calibrado el equipo se puede cerrar el programa dando un clic en la opción “Salir” ubicada en la parte superior derecha.

El programa por defecto, al cerrarse envía un comando al equipo GEODAS_USB para que se active la opción de Adquisición de datos (grabar datos en el dispositivo USB), si se pasó por alto activar esa opción en el equipo quede activada su función primordial.

2.2. USB

Este programa fue desarrollado en JAVA y para su funcionamiento sólo necesita tener instalado el controlador: Java 2 SDK, SE v1.4.2_09. El cual puede ser descargado gratuitamente desde el Internet.

Este programa fue creado para el análisis de los datos generados y almacenados por el equipo GEODAS_USB en las memorias USB. Al ejecutar el programa USB se mostrará la siguiente pantalla:

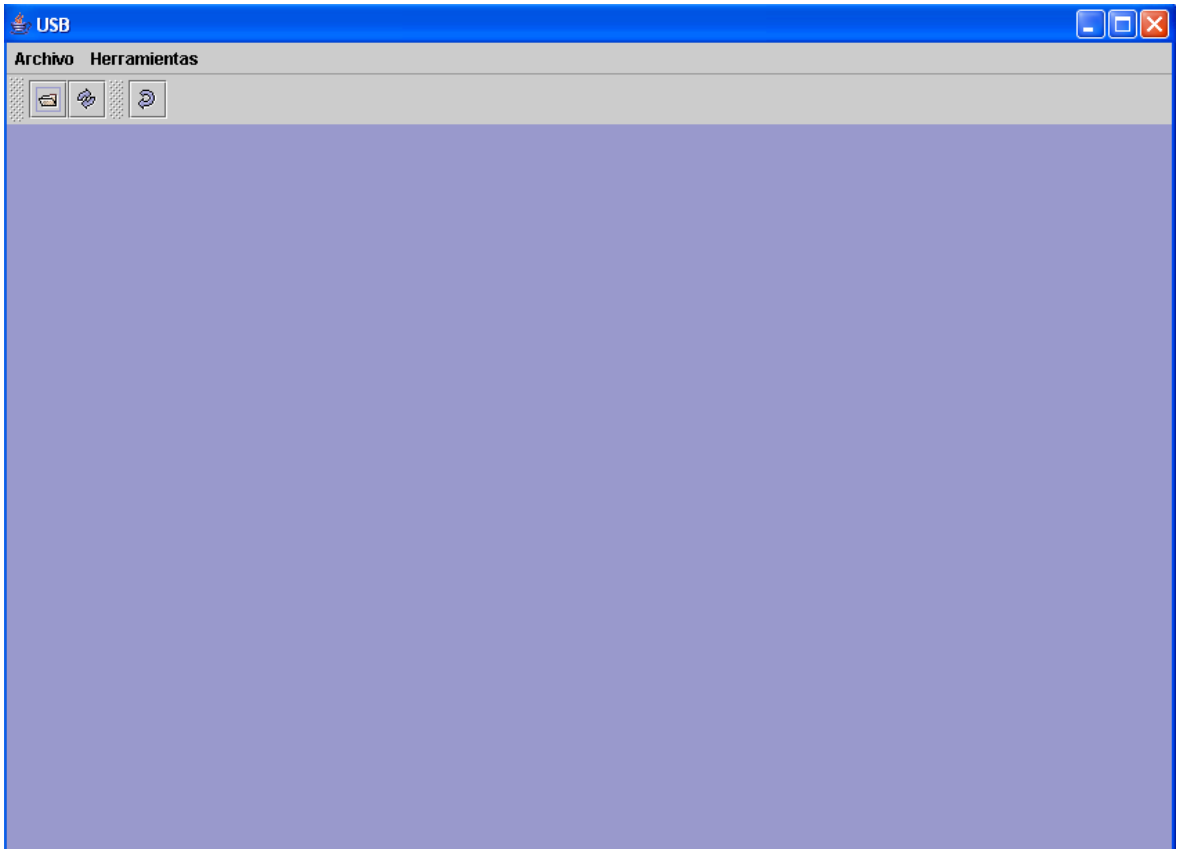


Gráfico D-25. Interfaz gráfica programa USB.

En el menú “Archivo” se encuentra la opción “Abrir” la cual despliega la siguiente ventana:

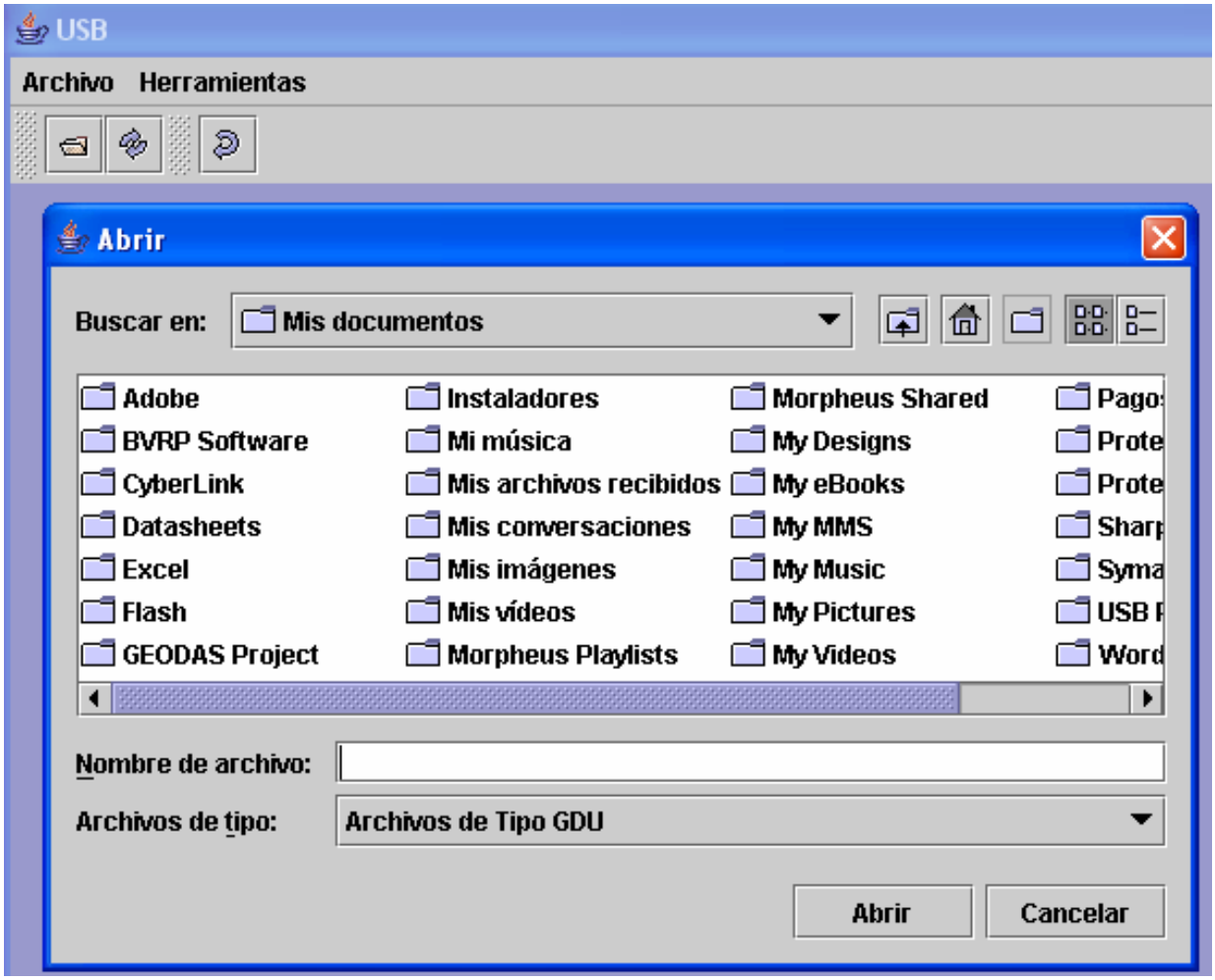


Gráfico D-26. Caja de dialogo para abrir los archivos Tipo GDU.

A continuación sólo se debe seleccionar al archivo ".GDU" que se desee abrir para su análisis.

En el menú "Herramientas" se tiene la opción "Generar SAC", con la cual se recupera sólo el campo de datos del archivo GDU escogido y con esa información genera tres archivos nuevos, los cuales están en un formato estándar y compatible con programas mas avanzados de análisis de datos sísmicos.

3. EQUIPO GEODAS_USB

El equipo GEODAS_USB consta de las siguientes partes:

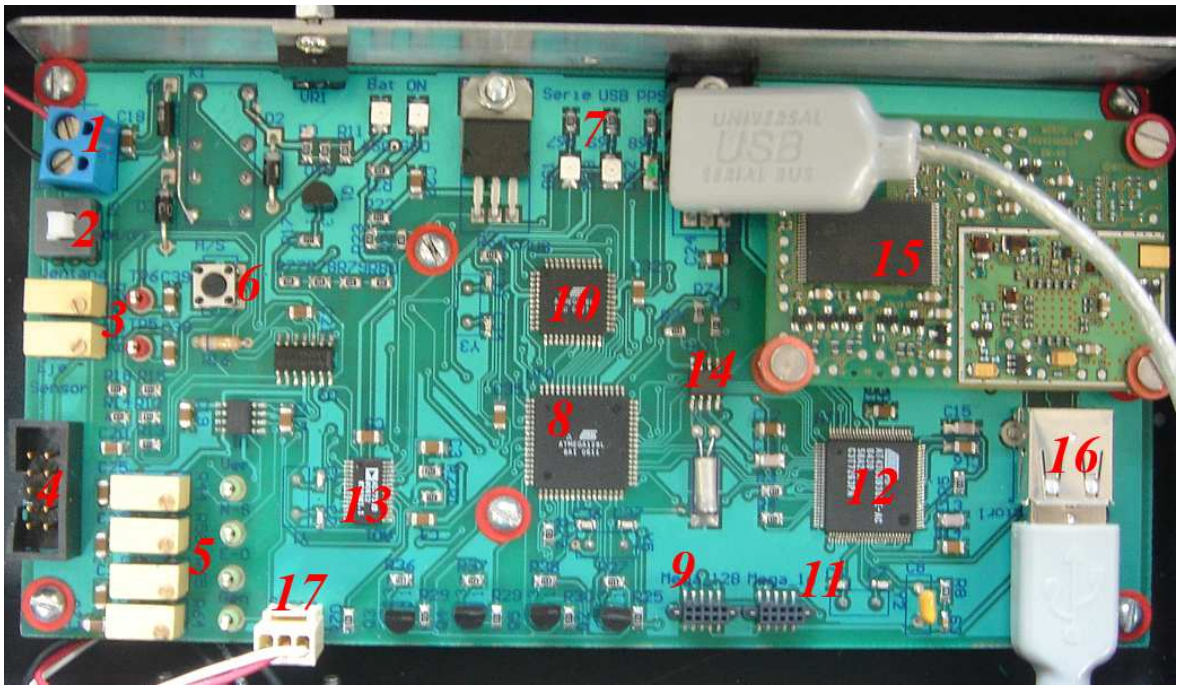


Gráfico D-27. Foto de la tarjeta principal GEODAS_USB.

1. Bornera de alimentación
2. Switch de encendido
3. Calibración circuito controlador de voltaje
4. Correa de señales de entrada
5. Calibración circuito analógico
6. Pulsante Hardware con seguridad
7. Leds indicadores
8. Microcontrolador ATmega128L
9. Receptáculo grabador Microcontrolador ATmega128L
10. Microcontrolador ATmega16L
11. Receptáculo grabador Microcontrolador ATmega16L
12. Controlador AT43USB380
13. Conversor sigma delta AD7738
14. RTC DS1307

15. Módulo GPS
16. Conector USB tipo A
17. Puerto de Comunicación Serial

3.1. CALIBRACIÓN

3.1.1. CIRCUITO CONTROLADOR DE VOLTAJE

Este circuito tiene la intención de controlar que el voltaje de la batería de alimentación no baje de un cierto nivel para que no se degrade, y permite que el equipo se conecte de nuevo cuando la batería recupera su carga por medio de un panel solar.

Para esto se tiene dos niveles de voltaje que permiten controlar los valores de un eje y una ventana, con los cuales se regula los niveles de voltaje mínimos para una desconexión y conexión del sistema.

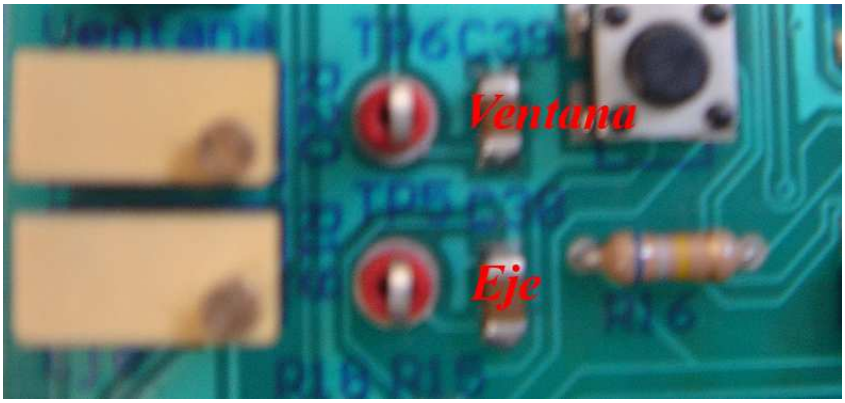


Gráfico D-28. Potenciómetros calibradores del circuito controlador de voltaje.

En la figura anterior se puede ver dos potenciómetros que regulan los voltajes en las puntas de prueba adyacentes a ellos, los valores medidos con un voltímetro deben ser:

V1 [V]	V2 [V]	V desconexión [V]	V conexión [V]	Eje [V]	Ventana [V]
2.93	0.208	11.5	12.5	12	1

Respetando las siguientes ecuaciones:

$$V_1 = \frac{V_{EJE}}{4{,}09} \qquad \text{y} \qquad V_1 = \frac{V_{VENTANA}}{4{,}8}$$

3.1.2. CIRCUITO ANALÓGICO

El equipo GEODAS_USB tiene un circuito de ganancias analógicas las cuales son controladas digitalmente, estas ganancias son modificadas utilizando el programa GEODAS GRAPHICS. El cual debe ser ejecutado y conectado al equipo seleccionando la opción de “Calibración”.

Para una correcta calibración de la respuesta del circuito analógico se debe escoger la ganancia mas alta posible (Ganancia=15). Y debe estar conectada la fuente de las señales a ser analizadas, ya que cada fuente tiene una respuesta diferente.

Se tienen cuatro potenciómetros para la calibración analógica como se muestran en el siguiente gráfico:

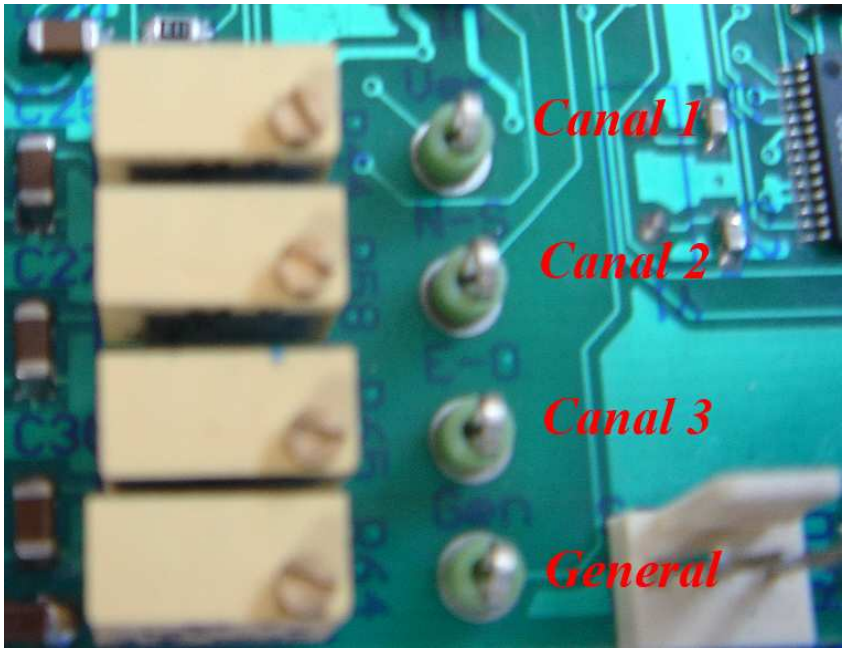


Gráfico D-29. Potenciómetros para calibración del circuito análogo.

Los tres primeros son para calibrar la componente continua de entrada de cada canal, y el cuarto potenciómetro es para la componente continua del amplificador de ganancia variable. Los valores medidos en las puntas de pruebas adyacentes a cada uno deben estar cerca de 1,25 V para cada componente y 2,5 V para el cuarto potenciómetro.

A continuación se debe poner una fuente nula (0 V), o garantizar que el sensor no esté detectando actividad sísmica. Entonces se procede a calibrar cada canal moviendo su respectivo potenciómetro para que la respuesta de cada uno esté en el eje “Cero” como se observa en la siguiente figura:

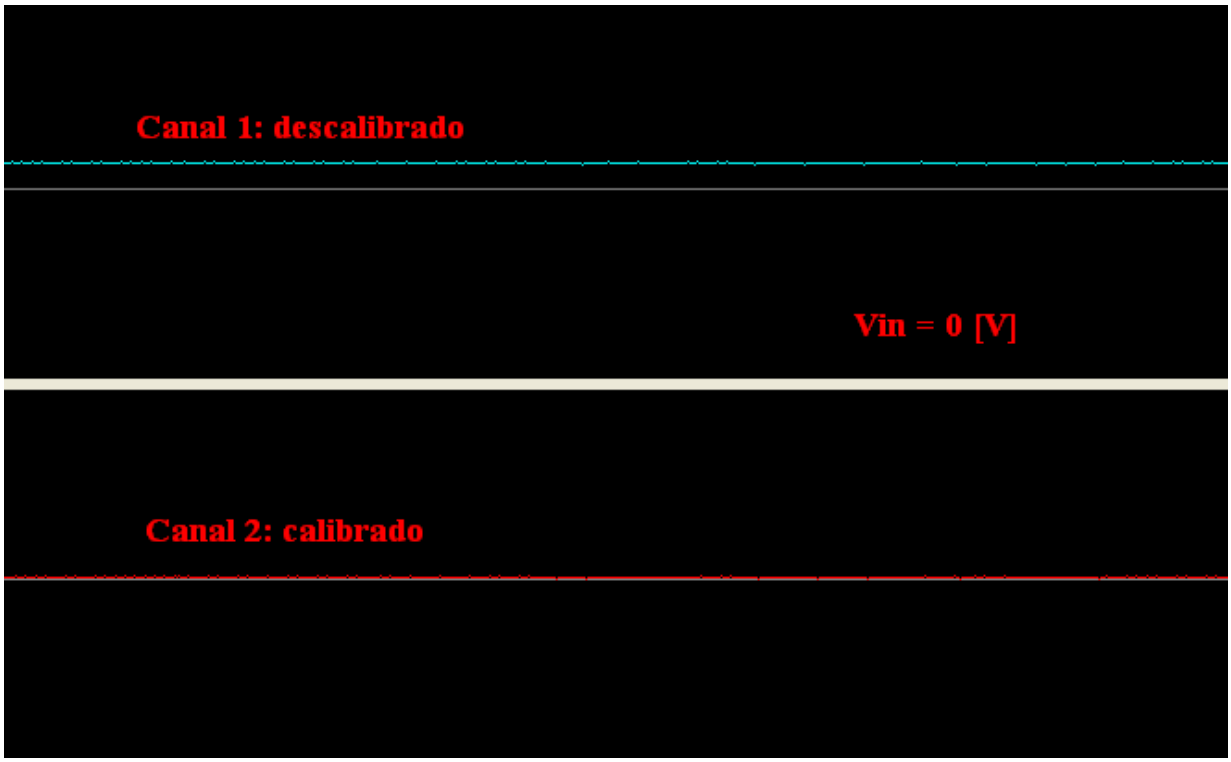


Gráfico D-30. Gráfico visualizado en el programa GEODAS GRAPHICS con un canal calibrado y uno descalibrado.

Una vez culminada la calibración de cada canal se debe proceder a escoger la ganancia adecuada para la respuesta del sensor utilizado.

Por último se debe activar la opción “Adquisición” para grabar los datos en el dispositivo USB.

3.2. MANEJO DEL EQUIPO

Una vez realizadas las calibraciones pertinentes al equipo GEODAS_USB, su uso es muy sencillo y práctico.

Para encender o apagar el equipo de debe accionar el switch que se muestra en la figura:

Gráfico D-31. Foto del switch de alimentación principal.

Existen dos leds rojos como indicadores de voltaje: un para la batería de alimentación y otro para alimentación a los microcontroladores. Y tres leds adicionales que indicaran el tipo de proceso en el que se encuentra el equipo: Serie, USB y PPS.

Cuando se conecta una memoria USB para almacenar los datos adquiridos por el equipo se puede ver el led USB encendido y el led PPS intercambiando entre encendido y apagado cada segundo.

Una vez que se requiera retirar el dispositivo USB del puerto del equipo, hay que tener precaución y mantener presionado el pulsante “Hardware con Seguridad” mostrado en la siguiente figura, hasta que la actividad en el led indicador de la memoria USB se haya detenido.

Gráfico D-32. Foto del pulsante H/S.

Si se desea volver a grabar más datos, sólo se debe volver a conectar la memoria en el puerto del equipo y automáticamente éste la reconocerá y continuará con el proceso de adquisición y almacenamiento.

ANEXO E

Contenido del CD GEODAS_USB

Contenido del CD GEODAS_USB[31]

- Programa: AVR - IDE GCC v 2.0
- Programa: GEODAS GRAPHICS
- Programa: USB
- Condigo fuente: Geodas_USB
- Condigo fuente: Geodas_M16
- Librerías: GEODAS_USB_LIB
- Instalador: WinAVR
- Instalador: Microsoft .NET Framework 2.0
- Instalador: Java 2 SDK, SE v1.4.2_09
- README.txt

[31] El CD GEODAS_USB estará adjunto al texto del Proyecto de Titulación.