Diseño e Implementación de un Sistema para la Adquisición Automatizada y Procesamiento de Datos para Prácticas de Laboratorio

Edwin Fernando Giraldo Aristizábal
Código: 200110162010
Teléfono: 2619500 extensión 9291
Celular: 3147233797
edgiral@eafit.edu.co

Daniel José Ramírez Valencia
Código: 200327500010
Teléfono: 2640640
Celular: 3006177075
dramir14@eafit.edu.co

Asesor
Álvaro Andrés Velásquez Torres
Físico Ph.D.
Teléfono: 2619500 extensión 9550

UNIVERSIDAD EAFIT
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE INFORMÁTICA Y SISTEMAS
MEDELLÍN
2009

Diseño e Implementación de un Sistema para la Adquisición Automatizada y Procesamiento de Datos para Prácticas de Laboratorio

EDWIN FERNANDO GIRALDO ARISTIZÁBAL
DANIEL JOSÉ RAMÍREZ VALENCIA

Proyecto de Grado para optar por el título de Ingeniero de Sistemas

Asesor
Álvaro Andrés Velásquez Torres
Físico Ph.D.
Profesor de Tiempo Completo, Departamento de Ciencias Básicas

UNIVERSIDAD EAFIT
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE INFORMÁTICA Y SISTEMAS
MEDELLÍN
2009

Firma del presidente del jurado

Firma del Jurado

Firma del Jurado

Medellín 22 de mayo de 2009

DEDICADA A...

Mis padres que me han visto crecer como persona y como profesional, a mi hermano que me ha apoyado desde siempre, a los profesores que me han formado durante la carrera, a los compañeros que me han aguantado hasta el final.

Daniel

Mi hija Isabela, por cederme tiempo del que debía dedicarle a ella para mi estudio, a mi esposa, a mis padres y hermanos por su constante ayuda y apoyo incondicional a lo largo de mi estudio.

Edwin

AGRADECIMIENTOS

Los autores expresan sus agradecimientos a:

Álvaro Andrés Velásquez Torres por su apoyo y recomendaciones, por su interés y por su deseo de hacer de este proyecto algo mejor.

Hugo Alberto Murillo Hoyos por su gran ayuda, su colaboración, paciencia y conocimiento. Su trabajo y orientación fue vital para la realización del proyecto.

Vanessa Rodríguez Lora por su ayuda no solo durante la tesis sino durante toda la carrera, además por la ayuda incondicional en busca siempre de que este proyecto fuera mejor.

Daniel Giraldo por su ayuda incondicional en el diseño de la carcasa y sus recomendaciones al respecto.

A los profesores y estudiantes de la EIA que nos ayudaron con la mejora de la comunicación USB.

Al personal del laboratorio de Física que nos ayudo en la elaboración del prototipo.

CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

GLOSARIO

CIRCUITO: conjunto de elementos que consumen o generan corriente eléctrica unidos por conductores por los que circula dicha corriente.

CONTROL: regulación, manual o automática, sobre un sistema. Tablero o panel donde se encuentran los mandos. Dispositivo que regula a distancia el funcionamiento de un aparato, mecanismo o sistema.

DIP: encapsulado en el cual el circuito se encuentra bajo una superficie que lo recubre.

DOO: Diseño Orientado a Objetos. Es un método de diseño que abarca el proceso de descomposición orientado a objetos y una notación para representar los modelos lógico y físico tal como los modelos estáticos y dinámicos del sistema bajo diseño.

DRIVER: Controlador. Es un programa que interactúa con un dispositivo de hardware en particular. Contiene todos los datos necesarios del dispositivo para que el resto de programas y el sistema operativo sepan como han de utilizarlo.

EDA: (Electronic Design Automation). Diseño electrónico automatizado. Se dice de todas aquellas herramientas de software que tienen como finalidad el diseño de circuitos.

ENCODER: sensor electro-opto-mecánico que unido a un eje proporciona información de la posición angular. Su fin es actuar como un dispositivo de realimentación en sistemas de control integrado.

EOP: End of Packet. Señal que indica que se ha llegado al final de un paquete de transmisión.

HANDSHAKE: confirmación. Comunicación básica entre dispositivos electrónicos para asegurar la transmisión de datos. Normalmente se necesitan dos líneas de control, adicionales a las de datos

HOST: anfitrión, puede ser el PC o un dispositivo

MICROCONTROLADOR: computadora completa situada en un único chip, que contiene todos los elementos del microprocesador básico además de otras funciones especializadas.

NRZI: Non Return to Zero Invert. Código de no retorno a cero que invierte los ceros, introduciendo capacidad de sincronización en ausencia de transmisión.

PID/VID: Product ID/ Vendor ID. En USB son unos los descriptores que se emplean para el reconocimiento del dispositivo.

PROTOCOLO: conjunto de reglas predeterminadas que hacen posible el intercambio coordinado de mensajes entre usuarios, procesos, máquinas, esto incluye mecanismos de control de las relaciones entre las entidades comunicantes, la localización de los recursos y el flujo ordenado de la comunicación

SISTEMA EMBEBIDO: sistemas que están diseñados alrededor de un microcontrolador que integra en un chip memoria de programa, memoria de datos y varias funciones periféricas.

USB: Universal Serial Bus. Interfaz de hardware que permite conectar periféricos de baja velocidad, como el teclado, el ratón, la impresora o cámaras digitales, a una computadora.

CPU: Acrónimo inglés de "Central Processing Unit", en castellano, "Unidad Central de Proceso". Aquella unidad incluida íntegramente en el microprocesador (solo en PC's) de un computador que controla el resto de unidades. Formada por la unidad aritmético-lógica, la unidad de control y de pequeños registros principalmente. El control lo lleva a cabo mediante la interpretación y la ejecución de instrucciones, sincronizadas por un reloj.

Ensamblador: lenguaje de programación de medio nivel, el cual es traducible directamente a lenguaje máquina. Está constituido por un juego de instrucciones característico de cada máquina con distinta arquitectura.

Hardware: conjunto de dispositivos físicos, metálicos y de materiales plásticos que forman el computador u otro dispositivo conectado al mismo.

Software: son los programas, incluyendo procedimientos, utilidades, sistemas operativos, programas de aplicación y paquetes informáticos, implementados para un sistema informático.

Tiempo real: un sistema de este tipo es aquel que necesita de tiempos de respuesta muy cortos, incluso del orden de microsegundos, en el caso de procesos críticos.

INTRODUCCIÓN

En los últimos años se ha visto la importancia a nivel institucional que tienen los laboratorios de física para los procesos de enseñanza y aprendizaje de los estudiantes, ya que estos dependen, en gran medida, del papel que juegue el laboratorio como instrumento de articulación entre la teoría y la práctica.

Por ello el objetivo central del laboratorio es ser complemento de los conocimientos adquiridos en clase teóricas, en donde la manipulación de instrumentos de medida ayuden a clarificar conceptos y a experimentar de forma práctica los diferentes fenómenos.

Sin embargo, para llevar a cabo este objetivo central deben darse las condiciones adecuadas y sobre todo, una nueva actitud que permita ubicar al laboratorio como un escenario primordial para la enseñanza.

Esta nueva actitud implica, también, la búsqueda de nuevos métodos de trabajo que permitan cumplir fomentar en el alumno la creación de nuevos hábitos de trabajo, estudio, organización y responsabilidad.

Por todo lo anterior se hace necesario el desarrollo de aplicaciones que involucren problemas de medición, adquisición y procesamiento de datos, de manera que se disminuya el error por toma de datos, además del tiempo que toma una nueva adquisición por causa de algún error de lectura, procurando que el esfuerzo de los estudiantes se concentre en el análisis de los datos, en la formulación de hipótesis sobre los experimentos, en los márgenes de error de las medidas y las relaciones de las diferentes variables.

Este documento presenta el desarrollo de los módulos de hardware y software para un sistema dedicado a la adquisición automatizada de datos en los laboratorios de física básica de la Universidad EAFIT. El desempeño del sistema es probado por medio de dos prácticas de laboratorio, la primera de ellas corresponde a la medida del momento de inercia de un disco rígido y la segunda corresponde a la medida de la constante de tiempo de un circuito RC.

DEFINICIÓN DEL PROBLEMA

El desarrollo de aplicaciones que involucren problemas de medición, manejo de datos y su procesamiento, es una de las tareas más comunes en las que se encuentran interviniendo los sistemas de cómputo actualmente. El presente trabajo de grado tiene como objetivo brindar un ambiente asistido por computador a una serie de prácticas de laboratorio de los cursos de Física I y II de la Universidad EAFIT (Códigos CB0237 y CB0236), de manera que se habilite el manejo sistemático de la información, y la toma de datos de manera automatizada, con el fin de facilitar el desarrollo de las prácticas de laboratorio, por medio de la creación de tablas y gráficos que permitan que el estudiante pueda hacer mayor énfasis en el análisis de los resultados, tener un menor margen de error en las medidas y disminuir el error producido en el proceso manual de toma de datos.

OBJETIVOS

Objetivo General

Diseñar e implementar los sistemas de hardware y software que permitan la adquisición de datos en los laboratorios de física básica, generando tablas y gráficos para las variables de estudio en las prácticas.

Objetivos Específicos

- Identificar los elementos necesarios para desarrollar las prácticas.
- Identificar los tipos de salida y los protocolos utilizados por los diferentes sensores utilizados en las prácticas.
- Modelar e implementar el hardware de interfaz de captura de datos analógicos y digitales desde los sensores y con transmisión serial por protocolo USB a un PC para su posterior procesamiento.
- Adaptar las conexiones de los sensores existentes en el laboratorio de física, y que sean necesarios para el desarrollo de las prácticas, para que sean compatibles con el hardware de interfaz de captura.
- Modelar e implementar el software necesario para el funcionamiento del hardware de interfaz físico de adquisición de datos.

- Implementar una práctica general, asistida por computador, de manera que el usuario pueda hacer uso del hardware para la captura de datos y del software para la tabulación y monitoreo de los datos.

- Implementar una práctica de “Dinámica rotacional” asistida por computador de manera que se alcancen los siguientes objetivos (Especificados en la guía de laboratorio):
  - Medir experimentalmente el momento de inercia (inercia de rotación) de un cuerpo rígido.
  - Identificar en un sistema de varios cuerpos la relación entre variables angulares y variables lineales.
  - Experimentar cómo la energía potencial gravitacional puede transformarse en energía cinética translacional y rotacional.

- Implementar una práctica de “Condensadores” asistida por computador de manera que se alcancen los siguientes objetivos (Especificados en la guía de laboratorio):
  - Analizar el proceso de carga y descarga de un condensador.

    - Por medio de la medición del voltaje en los terminales de un condensador, obtener su energía almacenada.

- Construir la documentación necesaria para que el sistema pueda ser modificado y reproducido por el personal encargado del laboratorio de física.

IMPORTANCIA

La adquisición de datos y su procesamiento no son actividades centrales en las prácticas de laboratorio, sin embargo, en muchas ocasiones, es necesario emplear mucho tiempo para obtenerlos y aplicar algunos procesos necesarios para transformarlos en información útil para entender los fenómenos que se recrean en las prácticas.

Dado que el entendimiento de los fenómenos y la verificación de los modelos físicos y matemáticos juegan un papel central en la práctica de los laboratorios de física, se hace necesario desarrollar herramientas que faciliten la adquisición de datos y su manipulación.

El desarrollo del sistema, objeto de este proyecto de grado, permitirá que los estudiantes dediquen la mayor parte del tiempo de las prácticas al análisis y comprensión de los resultados de las medidas, al cálculo de las magnitudes físicas de interés, al análisis dimensional de las variables y a la formulación de hipótesis para explicar las discrepancias entre los resultados experimentales y las predicciones del modelo teórico. Los aspectos mencionados anteriormente incentivan el aprendizaje significativo de los conceptos de la física.

1 MARCO TEÓRICO

Para el desarrollo de este proyecto es necesario conocer los conceptos más relevantes que involucran un proyecto como el Diseño e Implementación de un Sistema para la Adquisición Automatizada y Procesamiento de Datos para Prácticas de Laboratorio, como se ampliará a continuación.

1.1 SENSORES

Los términos sensor y transductor se suelen aceptar como sinónimo, aunque si se tuviera que hacer una distinción, el término sensor es quizás más amplio, incluyendo una parte sensible o captador propiamente dicho y algún tipo de circuito de acondicionamiento de la señal.

1.1.1 CLASIFICACIÓN DE LOS SENSORES

ANALÓGICOS:

Aquellos que dan como salida un valor de tensión o corriente variable en forma continua dentro del campo de medida. Es frecuente en este tipo de sensores la presencia de una etapa de salida para suministrar señales normalizadas de 0 – 10V o 4-20mA.

DIGITALES:

Son aquellos que dan como salida una señal codificada en forma de pulsos o en forma de palabra digital codificada en hexadecimal, binario BCD u otro sistema cualquiera.

1.1.2 CARACTERÍSTICAS GENERALES DE LOS SENSORES

CAMPO DE MEDIDA:

Es el rango de valores de la magnitud de entrada comprendida entre el máximo y el mínimo detectables por el sensor con una tolerancia de error aceptable.

RESOLUCIÓN:

Resolución es el mínimo incremento en la variable de entrada que produce un incremento medible en la variable de salida.

PRECISIÓN:

La precisión hace referencia al grado de concordancia entre los resultados obtenidos al efectuar una medida en las mismas condiciones, el valor real de la variable no juega un papel decisivo en la precisión.

REPETIBILIDAD:

Característica que indica la máxima desviación entre valores de salida obtenidos al medir varias veces en un mismo valor de entrada con el mismo sensor y en idénticas condiciones ambientales.

LINEALIDAD:

Se dice que un sensor es lineal, si existe una constante de proporcionalidad única que relaciona los incrementos de señal de salida con los correspondientes incrementos de la señal de entrada, en todo el campo de salida.

SENSIBILIDAD:

Característica que indica la mayor o menor variación de la salida por unidad de la magnitud de entrada.

1.1.3 TIPOS DE SENSORES

TRANSDUCTORES DE POSICIÓN:

Los transductores de posición permiten medir la distancia de un objeto respecto a un punto o eje de referencia o simplemente detectar la presencia de un objeto a una cierta distancia.

TRANSDUCTORES DE PROXIMIDAD:

Los detectores de proximidad pueden estar basados en distintas propiedades físicas, siendo los más frecuentes los siguientes:

- Detectores inductivos.
- Detectores capacitivos
- Detectores ópticos
- Detectores ultrasónicos

Por lo general, se trata de sensores con respuesta digital.

TIPOS DE SENSORES SEGÚN LA SALIDA

DETECTORES TODO O NADA DE AC:

Se trata de detectores cuya salida es un interruptor estático de CA basándose en tiristores o triacs, por lo general no pueden utilizarse más que para CA ya que para CC una vez cebados no se desactivan.

DETECTORES TODO O NADA DE CC:

Se trata de detectores cuya salida suele ser un transistor PNP o NPN. Precisamente el tipo de transistor determina la forma de conexión a la carga.

DETECTORES DE SALIDA ANÁLOGA:

Los transductores con salida análoga dan una corriente proporcional a la distancia entre el cabezal detector y el objeto a detectar.

1.1.4 ALGUNAS CLASES DE SENSORES

DETECTORES INDUCTIVOS:

Este tipo de sensor sirve para detectar la proximidad de piezas metálicas en un rango de distancias que va desde 1mm a unos 30mm con una posible resolución del orden de décimas de milímetro. Están formados por un circuito L-C (Bobina - condensador) con alta frecuencia de resonancia. La bobina está construida sobre un núcleo de ferrita abierto, de forma que el flujo se cierra en la parte frontal a través de la zona sensible. La presencia de metal dentro de dicha zona sensible altera la reluctancia del circuito magnético, atenúa el circuito oscilante y hace variar la amplitud de oscilación. La detección de dicha amplitud permite obtener una señal de salida de conmutación entre alto y bajo.

DETECTORES CAPACITIVOS:

El principio de funcionamiento y las características son análogos a las descritas para los detectores inductivos pero en este caso el elemento sensible es el condensador del circuito oscilante, formado por dos aros metálicos concéntricos situados en la cara sensible y cuyo dieléctrico es el material de la zona sensible.

Este tipo de sensores permiten detectar materiales metálicos o no metálicos, pero su sensibilidad se ve afectada por el tipo de material y por el grado de humedad ambiental y del cuerpo a detectar.

DETECTORES ÓPTICOS:

Los detectores ópticos emplean fotocélulas como elementos de detección. Algunos tipos disponen de un cabezal que incorpora un emisor de luz y la fotocélula de detección del haz de luz reflejado sobre el objeto que se pretende detectar. Otros tipos trabajan a modo de barrera y están previstos para detección a mayores distancias con fuentes luminosas independientes del cabezal detector. Ambos tipos suelen trabajar con frecuencias luminosas en la gama de los infrarrojos.

DETECTORES ULTRASÓNICOS:

Estos detectores están basados en la emisión-recepción de ondas ultrasónicas. Cuando un objeto interrumpe el haz, el nivel de recepción varía y el receptor lo detecta. Como ventaja frente a los sensores ópticos, los detectores ultrasónicos pueden detectar con facilidad objetos transparentes como cristal y plásticos, materiales que ofrecen dificultades para la detección óptica.

Sin embargo y dado que estos detectores utilizan ondas ultrasónicas que se mueven por el aire, no podrán ser utilizados en lugares donde éste circule con violencia (bocas de aire acondicionado, cercanías de puertas, entre otros).

MEDIDORES DE POSICIÓN O DISTANCIA

POTENCIÓMETROS:

El potenciómetro es un transductor de posición angular de tipo lineal y con salida de tipo analógico. Consiste en una resistencia de hilo bobinado o en una pista de material conductor, distribuida a lo largo de un soporte en forma de arco y un cursor solidario a un eje de salida, que puede deslizar sobre dicho conductor.

ENCODERS:

Los encoders son dispositivos formados por un rotor con uno o varios grupos de bandas opacas y translúcidas alternadas y por una serie de captadores ópticos alojados en el estator, que detectan la presencia o no, de una banda opaca frente a ellos.

TRANSDUCTORES DE TEMPERATURA

TERMOSTATOS:

Los termostatos son sensores con salida tipo alto o bajo que conmuta a un cierto valor de la temperatura. Los más simples están basados generalmente en la diferencia de dilatación de dos metales.

TERMOPARES:

Los termopares son sensores activos (Producen un voltaje) de tipo analógico basados en el efecto Seebeck, dicho efecto consiste en la aparición de una tensión eléctrica entre dos piezas de distintos metales unidas o soldadas por un extremo, cuando éste se calienta (unión caliente) y se mantienen los otros dos extremos a una misma temperatura inferior (unión fría), se genera un milivoltaje en los terminales de la unión fría que son directamente proporcionales a la temperatura de la unión caliente.

PT100:

Son sensores de temperatura de tipo resistivo, esto es, cambian la resistencia con la temperatura; la resistencia entre sus terminales suelen tener un valor nominal de 100Ω a 0° C, de donde se deriva el nombre pt100.

1.2 PROTOCOLO DE COMUNICACIÓN USB[1]

Los puertos USB (Bus Universal en Serie) nacieron para facilitar la comunicación entre la PC y dispositivos de todo tipo: antes de su existencia era muy caro o molesto poseer memorias de almacenamiento masivo externas, como por ejemplo discos rígidos, y por otro lado si la computadora se había quedado sin slots de expansión (ISA o PCI) libres no podía conectar más periféricos.

Es necesario aclarar que este texto se referirá al estándar 1.1 de USB porque es el usado por la mayoría de los microcontroladores.

El USB versión 1.1 soporta dos velocidades, una alta a 12Mbits/segundo y una baja a 1.5Mbits/segundo. La baja es menos susceptible a las interferencias, reduciendo así el costo de componentes especializados para disminuir las mismas.

El *Universal Serial Bus* (USB) es controlado por un *Host* y solo puede haber un *Host* por bus; la especificación no define ningún diseño con múltiples maestros, pero en la especificación 2.0 se introduce un protocolo de negociación de *Host*, el cual permite a dos dispositivos negociar el papel de *Host* y está enfocado a conexiones punto a punto.

El USB usa una topología de estrella parecida a la de la red *Ethernet 10baseT*, lo que impone el uso de un concentrador; esta topología permite monitorear el poder de cada dispositivo y puede ser hasta apagado si ocurre un sobre voltaje, esto sin

[1] Traducido de www.beyondlogic.org

interrumpir el desempeño de los demás dispositivos, puede soportar dispositivos de velocidades diferentes, el *hub o concentrador* filtra las transacciones de máxima velocidad para que los dispositivos lentos no las reciban.

USB como lo sugiere su nombre es un bus serial. Usa 4 cables blindados de los cuales dos son alimentación (+5V y GND) los dos restantes son un par trenzado de señales diferenciales de datos. USB usa un sistema de codificación NRZI (*Non Return To Zero Invert*) para enviar los datos con un campo de sincronización para sincronizar los relojes del *host* y del dispositivo.

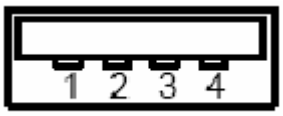
USB soporta plug and play con controladores que se cargan y descargan automáticamente. El usuario simplemente conecta el dispositivo al bus, el *host* detecta esta adición, interroga al nuevo dispositivo y carga el *driver* apropiado, el usuario no tiene que preocuparse por ningún dato de configuración, simplemente usa su dispositivo y lo desconecta cuando termine, una vez el host detecta la ausencia descarga el *driver* automáticamente.

La carga del *driver* apropiado se hace usando la combinación PID/VID (Product ID / Vendor ID) donde los VID son proporcionados por el foro de desarrolladores de USB por un precio y esto es considerado como un punto crítico para USB.

1.2.1 Conectores

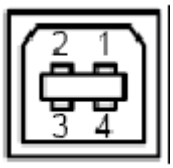
Todos los dispositivos tienen una conexión hacia el Host y el Host una conexión hacia los dispositivos. Los conectores en el Host y en el dispositivo no son iguales, hay dos tipos de conectores llamados A y B como se muestra en las figuras 1 y 2.

Figura 1. Pines del conector USB Tipo A



Fuente www.beyondlogic.com

Figura 2. Pines del conector USB Tipo B



Fuente www.beyondlogic.com

Conectores del tipo A se encuentran en los *Host* y en los *hub*, los conectores tipo B se encuentran en los dispositivos.

Los cables de USB usan alambres internos de color como se muestra en la tabla 1.

Tabla 1. Estándar Cables internos del USB

Número de Pin	Color del Cable	Función
1	Rojo	VBUS (5 voltios)
2	Blanco	D-
3	Verde	D+
4	Negro	Tierra

Fuente www.beyondlogic.com

1.2.2 Parte Eléctrica

USB usa un par trenzado de transmisión diferencial para los datos codificado usando NRZI y está empacado para garantizar la adecuada transmisión en la corriente de datos. Para transmitir un uno diferencial D+ 2.8V con una resistencia de 15K conectada a tierra y D- 0.3V con una resistencia de 1.5K conectada a 3.6V y un cero diferencial es el contrario en los terminales con las mismas resistencias conectadas a positivo y negativo.

El receptor define un uno (1) diferencial como D+ 200 mV más alto que D- y un cero diferencial D+ 200 mV menor que D-. La polaridad de la señal se invierte dependiendo de la velocidad del bus, por ende se usan los términos ‘J’ y ‘K’ para representar los niveles lógicos, en baja velocidad ‘J’ es un cero diferencial y en máxima velocidad ‘J’ es un uno diferencial.

1.2.3 Descriptores de USB

Todos los dispositivos de USB tienen una jerarquía de descriptores que describe al Host información como qué es el dispositivo, quién lo hace, qué versión de USB usa, de cuántas maneras puede ser configurado, cuántos caracteres de finalización (endpoint) tiene y de qué tipo son.

Los descriptores de USB más comunes son:

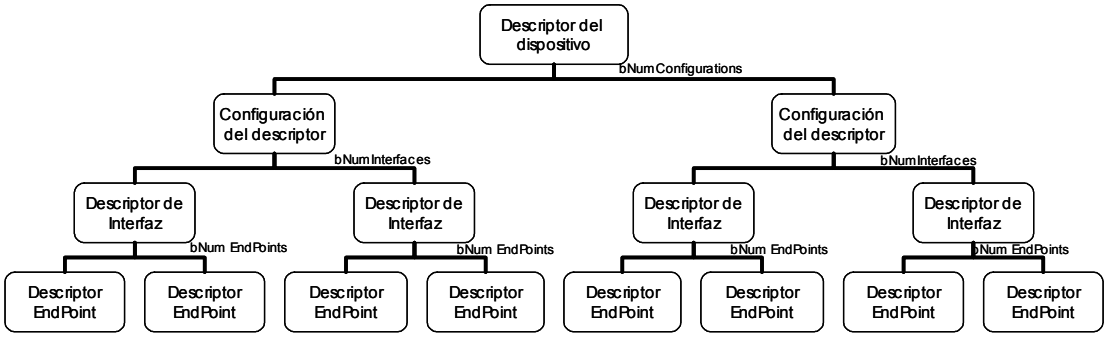
- Descriptor de dispositivo.
- Descriptor de configuración.
- Descriptor de interfaz.
- Descriptor de *endpoint*.
- Descriptor de cadena.

Los dispositivos USB solo pueden tener un descriptor de dispositivo. El descriptor de dispositivo incluye información como cuál es la versión de USB. La identificación de producto y vendedor es usada para cargar los *driver* apropiados y el número posible de configuraciones que un dispositivo puede tener. El número de configuraciones indica cuántas ramas de descriptores de configuración siguen.

El descriptor de configuración especifica valores tales como la cantidad de voltaje que la configuración usa, si el dispositivo usa alimentación propia o es energizado por el bus y el número de interfaces que tiene. Cuando el dispositivo es enumerado, el *Host* lee el descriptor del dispositivo y qué configuración puede usar; solo puede usar una configuración al tiempo.

El descriptor de interfaz puede ser visto como una cabecera o un grupo funcional de *endpoint* desempeñando una sola característica del dispositivo. A diferencia de los descriptores de configuración, no hay limitación para tener un solo descriptor de interfaz activo al tiempo. Un dispositivo puede tener uno o varios descriptores de interfaz activos al tiempo como se ve en la figura 3.

Figura 3. Descriptores USB



Fuente www.beyonlogic.com

Composición de los Descriptores de USB:

Todos los descriptores están hechos con un formato en común. El primer byte especifica el tamaño del descriptor, mientras que el segundo indica el tipo de descriptor. Si el tamaño del descriptor es menor al especificado, el Host deberá

ignorarlo, pero si es mayor al tamaño especificado el Host ignorará los bytes extra y buscará el siguiente descriptor al final del tamaño actual devuelto. Ver Tabla 2

Tabla 2. Composición de los descriptores USB

Offset	Campo	Tamaño	Valor	Descripción
0	bLength	1	Número	Tamaño de la descripción en bytes
1	bDescriptorType	1	Constante	Tipo de descripción
2	bcdUSB	2	BCD	Inicio de parámetros para el descriptores USB

Fuente www.beyondlogic.com

El descriptor de dispositivo para USB solo puede tener una descripción. Especifica una información básica e importante tal como la versión de USB, el tamaño máximo de paquete, la identificación del producto y del vendedor y el número de configuraciones posibles para el dispositivo. El formato del descriptor del dispositivo se muestra a continuación. Ver Tabla 3.

Tabla 3. Formato de los descriptores USB

Offset	Campo	Tamaño	Valor	Descripción
0	bLength	1	Número	Tamaño del descriptor en Bytes (18 bytes)
1	bDescriptorType	1	Constante	Descriptor del dispositivo (0x01)
2	bcdUSB	2	BCD	Número de especificación del dispositivo que cumple.
4	bDeviceClass	1	Clase	Código de clase Si es igual a cero cada interfaz específica su propio código de clase. Si es igual a 0xFF, el código de clase lo especifica el vendedor. De otro modo el campo es un código de clase válido.
5	bDeviceSubClass	1	SubClase	Código de subclase (Asignado por USB Org)
6	bDeviceProtocol	1	Protocolo	Código de Protocolo (Asignado por USB Org)
7	bMaxPacketSize	1	Número	Tamaño máximo del

				paquete para *Zero Endpoint*. Tamaños válidos 8, 16, 32, 64
8	idVendor	2	ID Vendedor	ID (Asignado por USB Org)
10	idProduct	2	ID Producto	ID (Asignado por el fabricante)
12	bcdDevice	2	BCD	Número de liberación del dispositivo.
14	iManufacturer	1	Index	Índice de manufactura. Cadena de descriptor.
15	iProduct	1	Index	Índice de Producto. Cadena de descriptor.
16	iSerialNumber	1	Index	Índice del número serial. Cadena de descriptor.
17	bNumConfigurations	1	Entero	Número de posible configuración.

Fuente www.beyondlogic.com

Descriptor de Configuración:

Un dispositivo USB puede tener muchas configuraciones diferentes pero la mayoría de los dispositivos son simples y tienen solo una. El descriptor de configuración especifica cómo es energizado el dispositivo, cuál es el consumo máximo de energía y el número de interfaces que tiene. Es posible tener dos configuraciones, una cuando el dispositivo es energizado por el bus y otra cuando tiene fuente propia. Como esta es una cabecera para los descriptores de interfaz, también es posible tener una configuración usando un modo de transferencia diferente al de otra configuración. Ver Tabla 4 y figura 4.

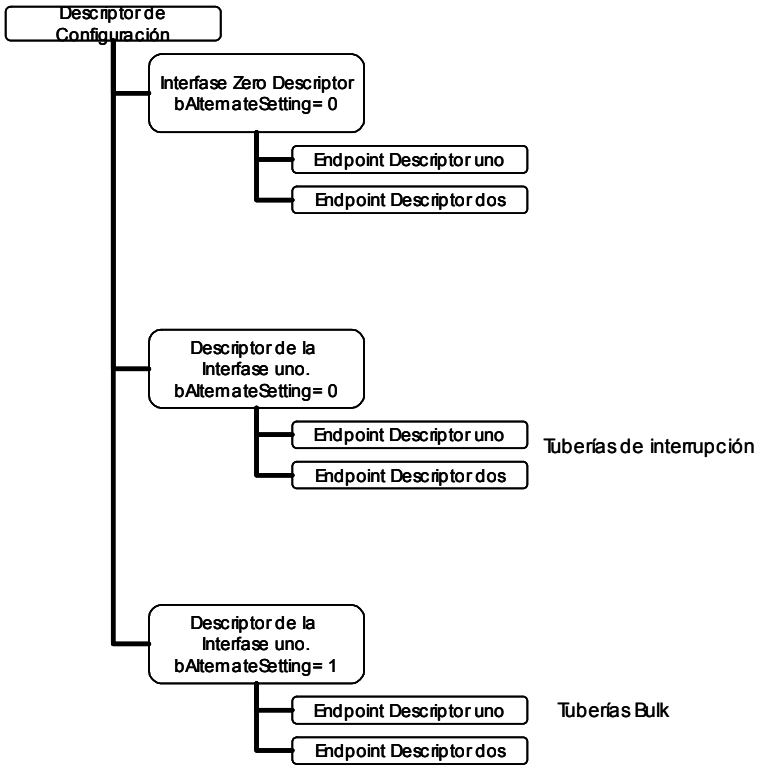
Tabla 4. Descriptor de configuración

Offset	Campo	Tamaño	Valor	Descripción
0	bLength	1	Número	Tamaño del descriptor en bytes.
1	bDescriptorType	1	Constante	Descriptor de configuración (0x02)
2	wTotalLenght	2	Número	Longitud total del dato de retorno.
4	bNumInterfaces	1	Número	Número de interfaz.
5	bConfigurationValue	1	Número	Valor para usar como

				un argumento para seleccionar esta configuración.
6	iConfiguration	1	Index	Índice de la cadena del descriptor. Para describir la configuración.
7	bmAttributes	1	Bitmap	D7 Poder del Bus D6 Poder propio. D5 Alarma remota D4 Reservado. (0)
8	bMaxPower	1	mA	Máximo consumo de potencia.

Fuente www.beyondlogic.com

Figura 4. Jerarquía de Descriptores



Fuente www.beyondlogic.com

Descriptor de Interfaz:

El descriptor de interfaz puede ser visto como una cabecera o un grupo funcional de *endpoint* desempeñando una sola característica del dispositivo. El descriptor de interfaz contiene el siguiente formato. Ver Tabla 5.

Tabla 5. Descriptor de interfaz

Offset	Campo	Tamaño	Valar	Descripción
0	bLength	1	Número	Tamaño del descriptor en bytes.
1	bDescriptorType	1	Constante	Descriptor de interfaz (0x04)
2	bInterfaceNumber	1	Número	Número de interfaz.
3	bAlternateSetting	1	Número	Valor usado para seleccionar configuraciones alternativas.
4	bNumEndpoints	1	Número	Número de *endpoints* usados para esta interfaz.
5	bInterfaceClass	1	Clase	Código de clase. (Asignado por USB Org)
6	bInterfaceSubClass	1	SubClase	Código de subclase. (Asignado por USB Org)
7	bInterfaceProtocol	1	Protocolo	Código de Protocolo
8	iInterface	1	Index	Índice de la cadena del descriptor que describe esta interfaz.

Fuente www.beyondlogic.com

Descriptor de Endpoint:

El descriptor de *endpoint* es usado para describir *endpoint* diferentes al *endpoint* 0. El *endpoint* 0 siempre está definido para ser un *endpoint* de control y es configurado antes de que los descriptores sean pedidos. El bus va a usar la información devuelta por estos descriptores para determinar los requerimientos de ancho de banda en el bus. Ver Tabla 6.

Tabla 6 Descriptor End Points

FOCET	Campo	Tamaño	Valor	Descripción
0	bLength	1	Número	Tamaño del descriptor en Bytes (7 bytes)

1	bDescriptionType	1	Constante	Descriptor Endpoint (0x05)
2	bEndpointAddress	1	Endpoint	Dirección Endpoint decodificada como sigue: 0..3b: Número Endpoint 4..6b:Reservado Set to Zero 7b: Dirección (Ignorada por el control de Endpoints) 0= Salida Endpoint 1 = Entrada Endpoint
3	bmAttributes	1	Bitmap	Bits 0..1 Tipo de transferencia 00 = Control 01 = Asincrónica 10 = Bulk 11 = Interrupción Bits 2..7 están reservados si el endpoint es asincrónico. Bits 3..2 = Tipo sincrónico (Modo Iso) 00 = No Sincronización 01 = Asincrónico 10 = Adaptativo 11 = Sincrónico Bits 5..4 = tipo de uso (Modelos Iso) 00 = Endpoint de dato 01 = Endpoint de Feedback 10 = Endpoint explícito de Feedback de datos 11 = Reservado
4	wMaxPacketSize	2	Número	Tamaño máximo del paquete. Este endpoint es capaz de enviar o recibir.
6	bInterval	1	Número	Intervalos para conteo de transferencias de datos del Endpoint. Valor en cuadro. Conteos ignorados para los endpoint de bulk y control. Iso debe ser igual a 1 y el campo debe tener un rango de 1 a 255 para interrupción

				de los endpoint.

Fuente www.beyondlogic.com

Descriptor de cadena:

Los descriptores de cadena proveen información que puede ser leída por el usuario y son opcionales. Si no se usan todos los índices de descriptores de cadena, éstos deben ser puestos en 0 indicando que no hay ningún descriptor de cadena.

Las cadenas están codificadas en formato *unicode* y puede haber productos que soporten múltiples lenguajes. El índice de cadena 0 debe devolver una lista de los lenguajes soportados. Ver Tabla 7.

Tabla 7. Descriptor de cadena

Offset	Campo	Tamaño	Valor	Descripción
0	bLength	1	Número	Tamaño del descriptor en Bytes
1	bDescriptorType	1	Constante	Cadena del descriptor(0x03)
2	wLANGIS[0]	2	Número	Lenguaje soportado código Zero (Por ejemplo 0x0409 Inglés-Estados Unidos)
3	wLANGIS[1]	2	Número	Lenguaje soportado código Zero (Por ejemplo 0x0c09 Inglés-Australia)
4	wLANGIS[2]	2	Número	Lenguaje soportado código Zero (Por ejemplo 0x0407 Alemán – estándar)

Fuente www.beyondlogic.com

1.3 PROCESAMIENTO DE TIEMPO REAL

Un Sistema en Tiempo Real (STR) es aquel sistema que interactúa activamente con un entorno de dinámica conocida en relación con sus entradas, salidas y restricciones temporales, para darle un correcto funcionamiento de acuerdo con los conceptos de estabilidad (que permanezca funcional), controlabilidad (que se puedan modificar los parámetros) y alcanzabilidad (que sea accesible).

Los Sistemas en Tiempo Real (STR) están presentes en nuestra vida diaria, prácticamente en todo lo que nos rodea; en los aviones, trenes y automóviles; en el televisor, la lavadora o el horno de microondas, en los teléfonos celulares y en las centrales telefónicas digitales. Son un elemento imprescindible para garantizar la generación, transmisión y distribución de la energía eléctrica y para asegurar la calidad y la seguridad de incontables procesos industriales.

La principal característica que distingue a los STR de otros tipos de sistemas es el tiempo. Sin embargo, antes de continuar es necesario decir qué significan las palabras tiempo y real.

La palabra tiempo significa que el correcto funcionamiento de un sistema depende no sólo del resultado lógico que devuelve la computadora, también depende del retardo en que se produce ese resultado.

La palabra real quiere decir que la reacción de un sistema a eventos externos debe ocurrir durante su evolución. Como una consecuencia, el tiempo del sistema (tiempo interno) debe ser medido usando la misma escala con que se mide el tiempo del ambiente controlado (tiempo externo).

Un STR tiene tres condiciones básicas:

a) Interactúa con el mundo real (proceso físico),
b) Emite respuestas correctas y
c) Cumple restricciones temporales (Tiempos de respuesta de las partes).

En contraste con la definición de STR, un sistema rápido produce su salida sin considerar las restricciones de tiempo del ambiente con el que interactúa. Para esa clase de sistemas no es importante el tiempo en el cual los datos llegan e él sino el tiempo en que la salida es producida.

De igual manera, tiende a confundirse el concepto de STR con Sistema en Línea: Un “Sistema en Línea es aquel que siempre debe estar encendido, disponible y generalmente conectado a una red de computadoras y depende de la capacidad del hardware para atender peticiones de servicio” y en ningún momento está en sincronía con el mundo real ni tiene restricciones temporales.

En adición a esto, un Sistema Fuera de Línea es aquel que no siempre está disponible para recibir y enviar información y que depende de una base de datos previamente establecida para ejecutar su cometido. Como ejemplos de sistema en línea se tienen las aplicaciones de Internet como los navegadores de páginas Web o la adquisición de datos a través de una tarjeta especializada en un ambiente de tiempo compartido como Windows.

1.4 HARDWARE

1.4.1 Microcontroladores[2]

Un microcontrolador es un circuito integrado de alta escala de integración que incorpora la mayor parte de los elementos que configuran un computador. Un microcontrolador dispone normalmente de los siguientes componentes:

- Procesador o UCP (Unidad Central de Proceso).
- Memoria RAM para almacenar los datos.
- Memoria de solo lectura para el programa, tipo ROM/PROM/EPROM.
- Líneas de E/S para comunicarse con el exterior.
- Diversos módulos para el control de periféricos (temporizadores, Puertos Serie y Paralelo, CAD: Conversores Analógico/Digital, CDA: Conversores Digital/Analógico, etc.).
- Generador de impulsos de reloj que sincronizan el funcionamiento de todo el sistema.

Los productos que para su regulación incorporan un microcontrolador disponen de las siguientes ventajas:

- Aumento de prestaciones: un mayor control sobre un determinado elemento representa una mejora considerable en el mismo.
- Aumento de la fiabilidad: al reemplazar el microcontrolador por un elevado número de elementos disminuye el riesgo de averías y se evitan ajustes.
- Reducción del tamaño en el producto acabado: La integración del microcontrolador en un chip disminuye el volumen.
- Mayor flexibilidad: las características de control están programadas, por lo que su modificación sólo necesita cambios en el programa de instrucciones.

El microcontrolador es en definitiva un circuito integrado que incluye todos los componentes de un computador. Debido a su reducido tamaño es posible montar

---

[2] Tomado de www.monografias.com

el controlador en el propio dispositivo al que gobierna. En este caso el controlador recibe el nombre de controlador embebido.

Los microcontroladores están siendo empleados en multitud de sistemas presentes en la vida diaria, como pueden ser juguetes, electrodomésticos, computadoras, impresoras, módems, vehículos, instrumentación electrónica, control de sistemas. Una aplicación típica es emplear varios microcontroladores para controlar pequeñas partes de un sistema. Estos pequeños controladores pueden comunicarse entre ellos y con un procesador central (sistema maestro - esclavo), probablemente más potente, para compartir la información y coordinar sus acciones.

A la hora de escoger el microcontrolador a emplear en un diseño concreto hay que tener en cuenta multitud de factores, como la documentación y herramientas de desarrollo disponibles, su precio, la cantidad de fabricantes que lo producen y por supuesto las características del microcontrolador (tipo de memoria de programa, número de temporizadores, interrupciones, etc.). Los factores más importantes son:

- Costo: como es lógico, los fabricantes de microcontroladores compiten intensamente para vender sus productos.
- Aplicación: antes de seleccionar un microcontrolador es imprescindible analizar los requisitos de la aplicación:
- Procesamiento de datos: puede ser necesario que el microcontrolador realice cálculos críticos en un tiempo limitado. En ese caso se debe asegurar seleccionar un dispositivo suficientemente rápido para ello. Por otro lado, habrá que tener en cuenta la precisión de los datos a manejar: si no es suficiente con un microcontrolador de 8 bits, puede ser necesario acudir a microcontroladores de 16 ó 32 bits, o incluso a hardware de punto flotante. Una alternativa más económica y quizá suficiente es usar librerías para manejar los datos de alta precisión que se cargan en la memoria de éste.
- Entrada/Salida: para determinar las necesidades de Entrada/Salida del sistema es conveniente dibujar un diagrama de bloques del mismo, de tal forma que sea sencillo identificar la cantidad y tipo de señales a controlar. Una vez realizado este análisis puede ser necesario añadir periféricos o hardware externo o cambiar a otro microcontrolador más adecuado a ese sistema.
- Consumo: algunos productos que incorporan microcontroladores están alimentados con baterías y su funcionamiento puede ser tan vital como activar una alarma antirrobo. Lo más conveniente en un caso como éste puede ser que el microcontrolador esté en estado de bajo consumo pero que despierte ante la activación de una señal (una interrupción) y ejecute el programa adecuado para procesarla.
- Memoria: para detectar las necesidades de memoria de nuestra aplicación debemos separarla en memoria volátil (RAM), memoria no volátil (ROM,

EPROM, etc.), la memoria Flash y la memoria no volátil modificable (EEPROM). Este último tipo de memoria puede ser útil para incluir información específica de la aplicación como un número de serie o parámetros de calibración. El tipo de memoria a emplear vendrá determinado por el volumen de ventas previsto del producto: de menor a mayor volumen será conveniente emplear EPROM, OTP (*One-Time Programmable* solo una programación) y ROM. En cuanto a la cantidad de memoria necesaria puede ser imprescindible realizar una versión preliminar (aunque sea en pseudo-código) de la aplicación y a partir de ella hacer una estimación de cuánta memoria volátil y no volátil es necesaria y si es conveniente disponer de memoria no volátil modificable.

- Tamaño de palabra: el criterio de diseño debe ser seleccionar el microcontrolador de menor tamaño de palabra que satisfaga los requerimientos de la aplicación. Usar un microcontrolador de 4 bits supondrá una reducción en los costes importante, mientras que uno de 8 bits puede ser el más adecuado si el tamaño de los datos es de un byte. Los microcontroladores de 16 y 32 bits, debido a su elevado costo, deben reservarse para aplicaciones que requieran sus altas prestaciones (Entrada/Salida potente o espacio de direccionamiento muy elevado).
- Diseño de la tarjeta: la selección de un microcontrolador específico condicionará el diseño de la tarjeta de circuito. Debe tenerse en cuenta que quizá usar un microcontrolador de bajo costo puede encarecer el resto de componentes del diseño.

1.4.2 DISEÑO CIRCUITOS

Para construir la tarjeta que evita fallas de conexión y poder replicarla fácilmente se tuvieron en cuenta los siguientes estándares.

1.4.3 Estándares para los circuitos impresos[3]

El diseño de circuitos impresos debe seguir una serie de estándares que pretenden mantener la integridad de las señales, evitando la producción de interferencias.

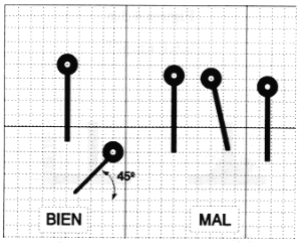
Estos estándares son unas reglas simples definidas por la IEEE P1801, que básicamente hablan de la disposición de los componentes, de la distancia entre estos y el ancho de las pistas conductoras.
Estas reglas son:

- Se diseñará sobre una hoja cuadriculada en décimas de pulgada, de modo que se hagan coincidir las pistas con las líneas de la cuadrícula o formando un

[3] Tomado de www.lu1dma.com.ar

ángulo de 45º con éstas, y los puntos de soldadura con las intersecciones de las líneas, como se ve en la figura 5.

Figura 5. Ubicación de los componentes

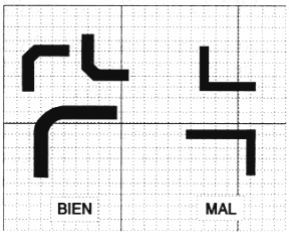


Fuente www.lu1dma.com.ar

- El diseño debe ser lo más sencillo posible, entre más cortas sean las pistas y más simple la distribución de componentes, mejor será el diseño.

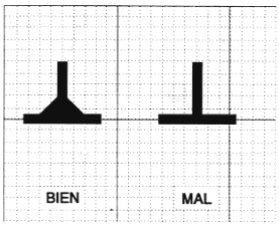
- No deben haber pistas con ángulos de 90º; cuando sea preciso efectuar un giro en una pista, se hará con dos ángulos de 135º (figura 6); si es necesario ejecutar una bifurcación en una pista, se hará suavizando los ángulos con sendos triángulos a cada lado, figura 7.

Figura 6. Ángulos en las pistas



Fuente www.lu1dma.com.ar

Figura 7. Bifurcación



Fuente www.lu1dma.com.ar

- Los puntos de soldadura deben ser círculos cuyo diámetro será, al menos, el doble del ancho de la pista que en él termina.

- El ancho de las pistas dependerá de la intensidad que vaya a circular por ellas. Para esto se tiene los datos mostrados en la Tabla 8:

Tabla 8. Ancho de Pistas

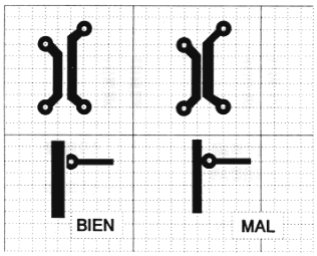
Ancho (milímetros)	Corriente (Amperios)
0.8 mm	2 A
2 mm	5 A
4.5 mm	10 A

Fuente www.lu1dma.com.ar

- En general, se realizan pistas de unos 2 mm aproximadamente.

- Entre pistas próximas y entre pistas y puntos de soldadura, se observará una distancia que dependerá de la tensión eléctrica que se prevea existirá entre ellas; como norma general, se dejará una distancia mínima de unos 0,8 mm.; en casos de diseños complejos, se podrá disminuir los 0,8 mm hasta 0,4 mm. En algunas ocasiones será preciso cortar una porción de ciertos puntos de soldadura para que se cumpla esta norma. Ver figura 8.

Figura 8. Distancia entre pistas

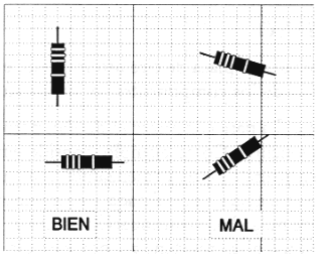


Fuente www.lu1dma.com.ar

- La distancia mínima entre pistas y los bordes de la placa será de dos décimas de pulgada, aproximadamente unos 5 mm.

- Todos los componentes se colocarán paralelos a los bordes de la placa. Ver figura 9.

Figura 9. Posición de los componentes



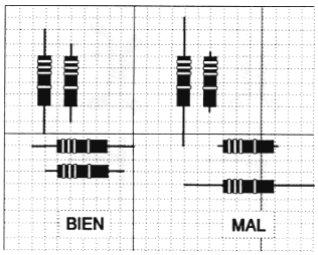
Fuente wwwlu1dma.com.ar

- No se podrán colocar pistas entre los bordes de la placa y los puntos de soldadura de terminales de entrada, salida o alimentación, exceptuando la pista de masa.

- No se pasarán pistas entre dos terminales de componentes activos (transistores, tiristores, etc.).

- Como norma general, se debe dejar, una o dos décimas de pulgada de patilla entre el cuerpo de los componentes y el punto de soldadura correspondiente. Ver figura 10.

Figura 10. Soldadura de los componentes



Fuente www.u1dma.com.ar

Es importante indicar que para diseñar un circuito impreso, es necesario conocer el tamaño y la forma física de los componentes, o, mejor aún, disponer de ellos.

1.4.4 HERRAMIENTAS PARA EL DISEÑO DE CIRCUITOS ELECTRÓNICOS[4]

Los grandes cambios que ha sufrido la industria electrónica en los últimos años, se han dado en parte por el uso de herramientas de diseño asistido por computadora (CAD) y en forma especifica a herramientas de diseño automático tipo EDA.

El uso de éste tipo de herramientas además de simplificar el trabajo de diseño ha acelerado los procesos de este mismo. Esto se tradujo en un cambio de metodología para el diseño y evolución de los circuitos electrónicos.

Existen gran número de herramientas que apoyan el diseño de circuitos electrónicos, las cuales poseen entre si características que las diferencian las unas de las otras. En ésta sección se hace una breve descripción de la herramienta *CircuitMaker*, la cual se encuentran disponible en la Universidad.

CircuitMaker

Actualmente es el programa para el diseño de circuitos que más se está empleando en la universidad ya que su uso es sencillo y posee características que optimizan el trabajo y por eso se adoptará esta herramienta para el diseño de la tarjeta controladora del proyecto. Adicionalmente las empresas dedicadas a la fabricación de circuitos impresos utilizan entre otros (*Eagle, Proteus, MultiSim)* este software lo que facilita su elaboración. A continuación se describe la herramienta con mayor detalle.

---

[4] Apartes tomados de www.forosdeelectronica.com

- Diseño de esquemas: Permite crear esquemas con una gran simplicidad. Después de haber insertado los componentes en el espacio de trabajo, la conexión entre patillas del circuito se simplifica gracias al software. Éste permite conectar las patillas de un componente a las patillas del circuito al cual se quiere conectar encargándose del trazado de la conexión. También se pueden trazar pistas manualmente y cuando se desplaza un circuito, los trazados de pistas se vuelven a editar automáticamente.

- Soporta la simulación lógica, analógica o mixta del circuito: Gracias al motor de simulación, *Circuitmaker* permite simular realmente los esquemas teniendo en cuenta las características de los componentes (retrasos de propagación, tiempo de espera, etc.). De esta manera se puede simular cualquier combinación de componentes lógicos y analógicos sin insertar convertidores D/A o A/D. Posee también instrumentos virtuales para la comprobación y análisis del circuito.

- Biblioteca de componentes: Gracias al editor de símbolos, *Circuitmaker* permite crear nuevos símbolos e incluirlos en las librerías. No existen limitaciones de patillas, de esta forma se podrán crear símbolos para microprocesadores y otros tipos de componentes más complejos. También tiene la posibilidad de editar los parámetros de los componentes y de modificarlos en todo momento. Permite además la importación y exportación de otras librerías.

1.5 SOFTWARE

1.5.1 LabVIEW

Es un software de programación gráfico, que posee dos ventanas: el panel de control y el diagrama de conexiones también llamado diagrama de bloques.

En el panel de control se ubican todos los controles e indicadores que se requiera; y en el diagrama de conexiones se alambran estos últimos con los operadores del programa.

1.5.2 PICC

Es un software para las personas que prefieren programar en bajo nivel con los recursos de un lenguaje de alto nivel como el C. Por tal razón el desarrollo se realiza a través de la sintaxis de C.

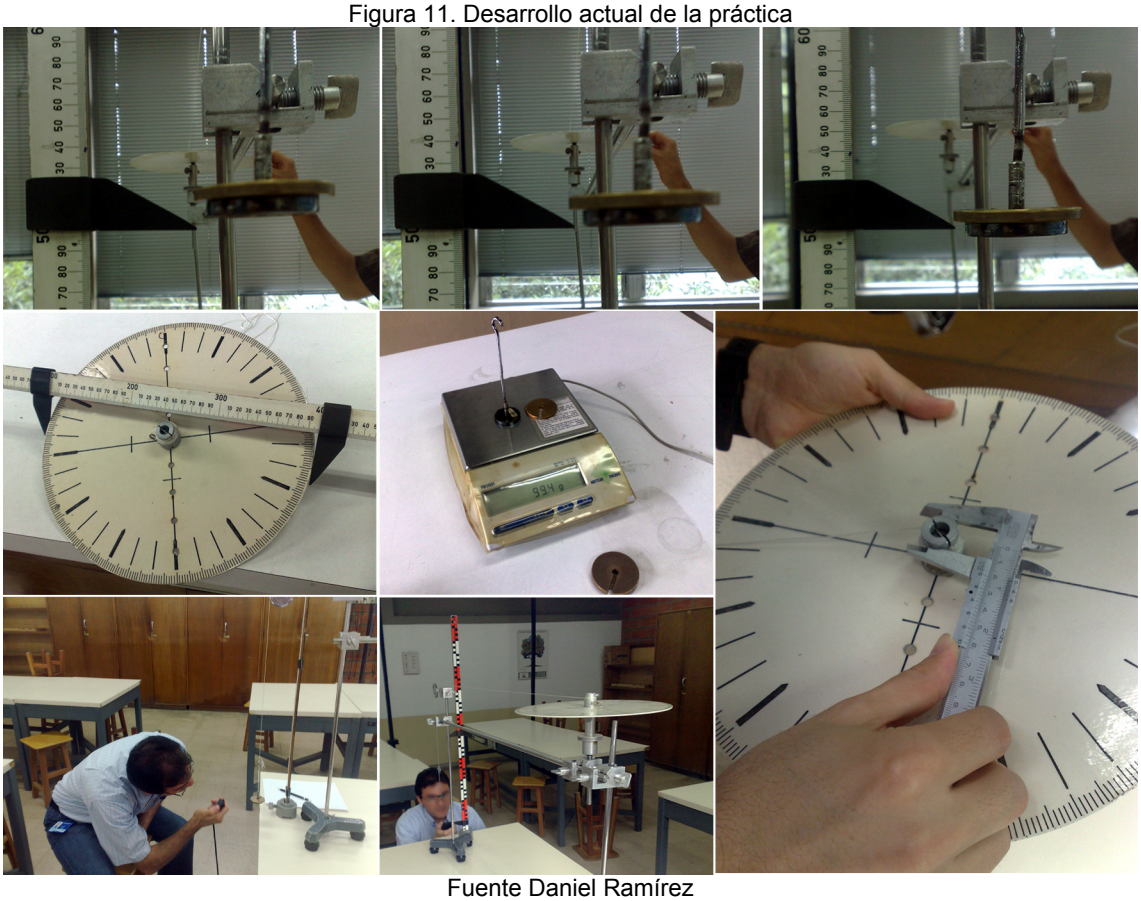
Fue desarrollado para cumplir con las especificaciones de lenguaje ANSI C. El compilador produce tres tipos de archivos. Archivos con extensión **.hex** que es el

código que se programa en el PIC por medio de un programador. El archivo con extensión **.asm** contiene el listado del código en (lenguaje ensamblador con el mapeo de la memoria) y la extensión **.pre** que contiene la información preprocesada del programa, #defines, #includes, etc.

2 METODOLOGÍA ACTUAL PARA LA ADQUISICIÓN DE DATOS

En la actualidad en las prácticas de los laboratorios de física de la Universidad EAFIT (Física I código CB0237 y Física II código CB0237), la toma de datos se realiza de forma manual en la mayoría de los casos, lo que conlleva a introducir diversos tipos de errores en las medidas, debidos entre otras a la demora en la reacción humana, el uso de diferentes instrumentos de medida los cuales pueden presentar inconsistencias al compararlo con el valor arrojado por otro que pueda estar descalibrado, por otro lado en algunas prácticas se emplean varios equipos lo que implica que la conexión sea más compleja. Otra de las dificultades tiene que ver con la construcción de las gráficas, las cuales se realizan introduciendo manualmente los datos en la hoja de cálculo Excel. El ingreso manual de los datos experimentales supone un gasto considerable de tiempo que podría ser invertido en el análisis del gráfico mismo, en el significado de cada uno de sus parámetros y en la comprensión de la relación funcional existente entre las variables graficadas, esto es, en el análisis del fenómeno físico.
A continuación se presentan unos registros fotográficos sobre la forma como se realiza la práctica de momento de inercia., figura 11.

Figura 11. Desarrollo actual de la práctica



Fuente Daniel Ramírez

Es por esto que, buscando minimizar los errores en la toma de datos y optimizar el tiempo, calidad y precisión de los datos, se desarrolló un sistema confiable asistido por computador, de manera que se habilite el manejo sistemático y la toma de datos de manera automatizada, con el fin de facilitar el desarrollo de las prácticas de laboratorio. Esto se hará por medio de la creación de tablas y gráficos que permitan al estudiante hacer mayor énfasis en el análisis de los resultados que en la adquisición de los mismos, y por otra parte reducir el margen de error instrumental obtenido en el proceso manual de toma de datos.

3 DESCRIPCIÓN TÉCNICA DEL PROYECTO

3.1 Referentes de algunos sistemas comerciales de adquisición de datos

3.1.1 MultiLogPRO:

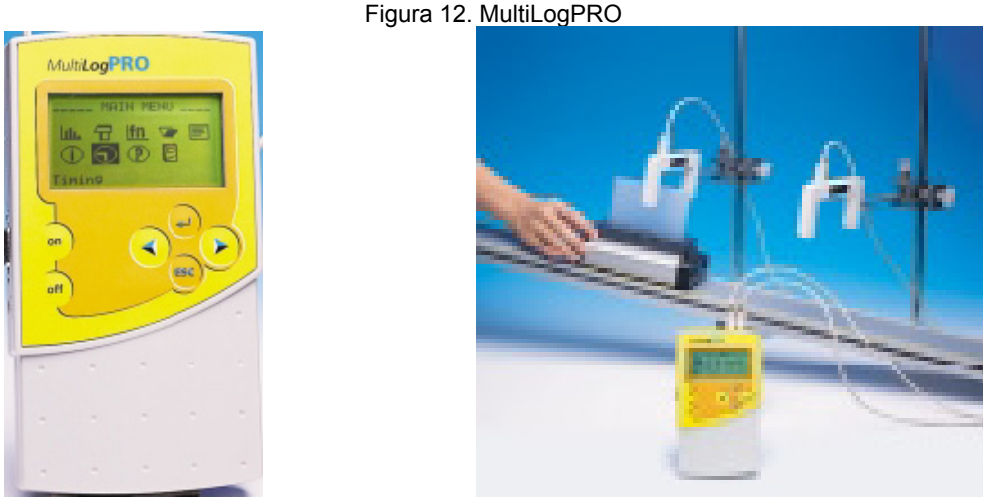
Solo soporta sensores ITPsoft. Máximo 2 entradas digitales simultaneas. Máximo 2 salidas de control digital. Solo 4 puertos (expandibles a 8) para entradas/salidas, si se utiliza un control o una entrada digital ocupa uno de los 4 puertos (o 2 de los 8) y solo quedan 3 puertos disponibles para entradas análogas, ver figura 12.

Capacidad para almacenar 104.000 datos, divididos entre la cantidad de sensores.

Hasta 20.800 muestras por segundo, pero no las transmite al PC sino hasta después de haberlas tomado. En la muestra del equipo, en 2 minutos sólo había transmitido un 22% de los datos.

Trae un software (MultiLab) y protocolo propietario. Solo puede ser utilizado con este software. Para ampliar la información sobre datos técnicos remítase al anexo C.

Figura 12. MultiLogPRO



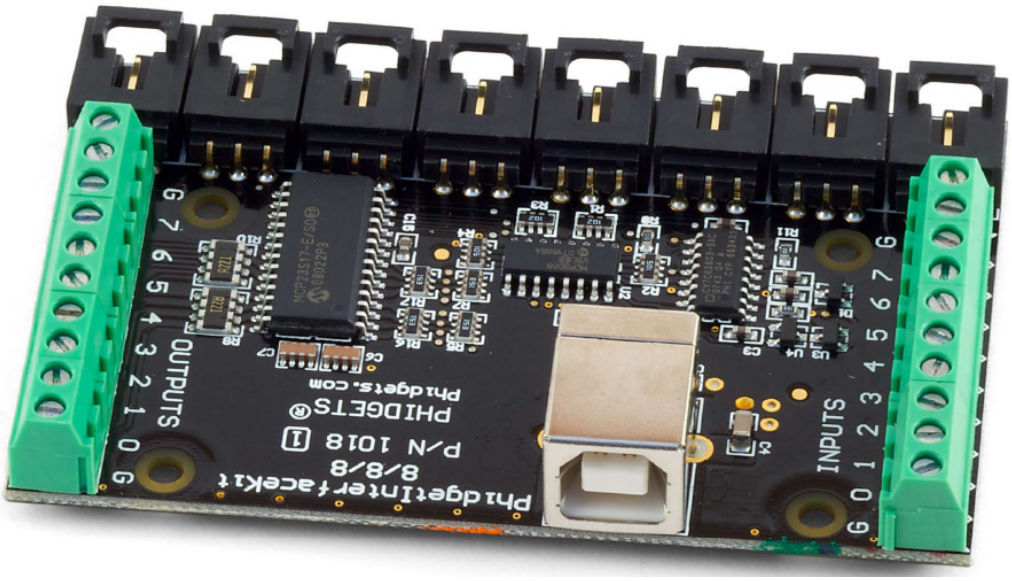
Fuente www.electrocomponentes.com.ar

3.1.2 PhidgetInterfaceKit 8/8/8

Esta tarjeta es para 8 entradas análogas y 8 digitales y 8 salidas digitales, solo soporta los sensores construidos por el fabricante. El Hardware es inalterable y no trae ningún software de control. El porcentaje de muestreo es de aprox. 65 muestras por segundo. El programa del Hardware (Microcontrolador) es

inalterable, mostrado en la figura 13. Para ampliar la información sobre datos técnicos remítase al anexo C.

Figura 13. PhidgetInferfaceKit 8/8/8



Fuente Phidgets

3.1.3 ScienceWorkshop 750 Interface de Pasco

Es una interfaz con 4 puertos digitales, 3 análogos y una salida digital, la conexión se hace a través del puerto USB, con captura simultánea de análogos y digitales Cuando las muestras por segundo son más de 100 estas no pueden ser graficadas, además posee función de generador de ondas y tiene interfaz con LabVIEW, este se muestra en la figura 14. Para ampliar la información sobre datos técnicos remítase al anexo C.

Figura 14. ScienceWorkshop 750 Interface de Pasco

Fuente Pasco

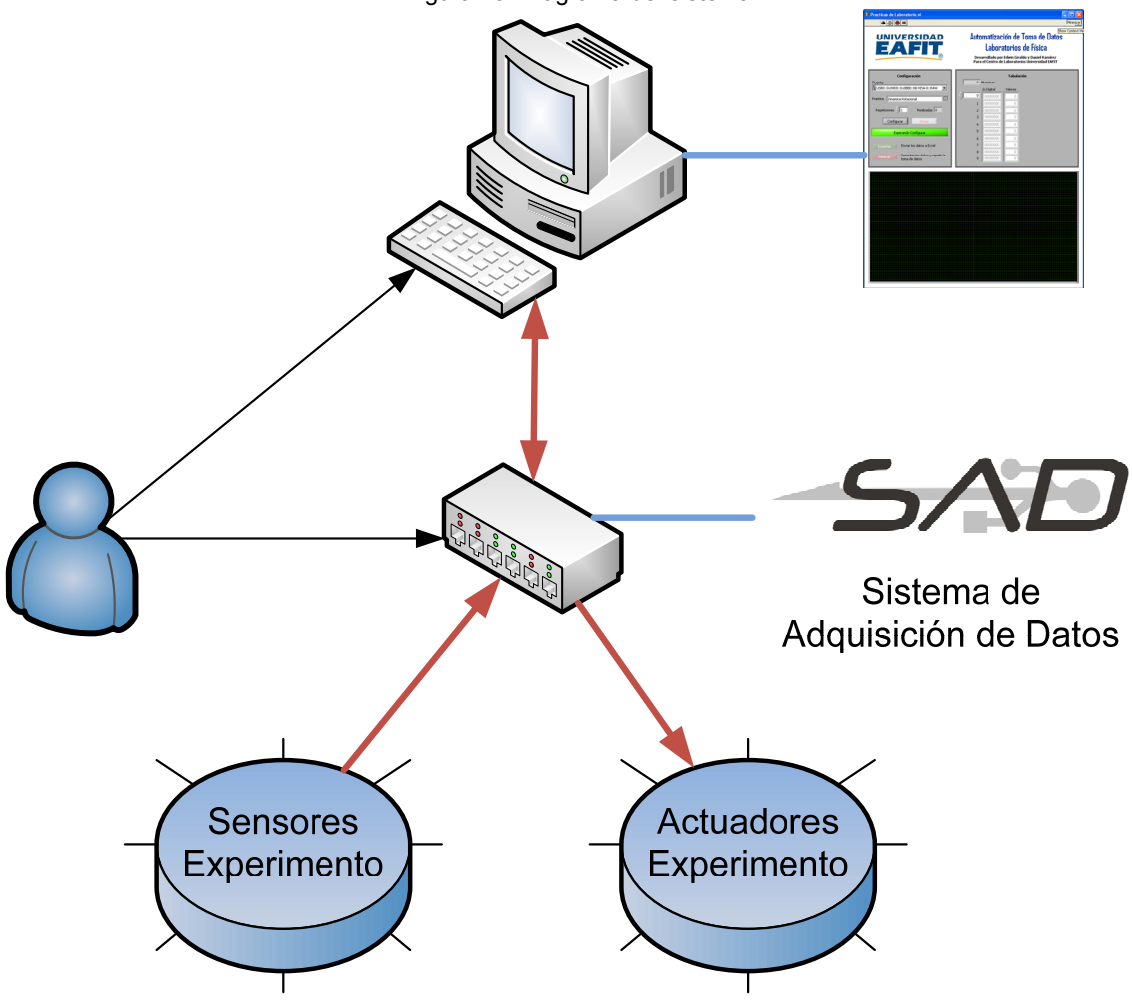
3.2 Desventajas de algunos sistemas existentes

- Los costos son muy elevados.
- La funcionalidad está sujeta a lo que el fabricante tenga preestablecido.
- No es posible modificar ni el software, ni el hardware, haciendo que no se adecue a las prácticas sino que las prácticas se deben adecuar a ellos.
- En la mayoría de los casos solo funciona con los sensores propios del fabricante.

4 DESARROLLO DEL PROYECTO

El sistema de automatización de captura y adquisición de datos esta conformado por tres subsistemas como se muestra en la figura 15, estos son aplicación, interfaz y sensores.
En el PC la aplicación se desarrollado en LabVIEW, el cual permite la comunicación vía USB con la interfaz. Por su parte la interfaz posee componentes hardware y software que le permiten la comunicación a través de USB con el PC e interacción directa con los sensores y actuadores.

Figura 15: Diagrama del sistema



Fuente Daniel Ramírez V

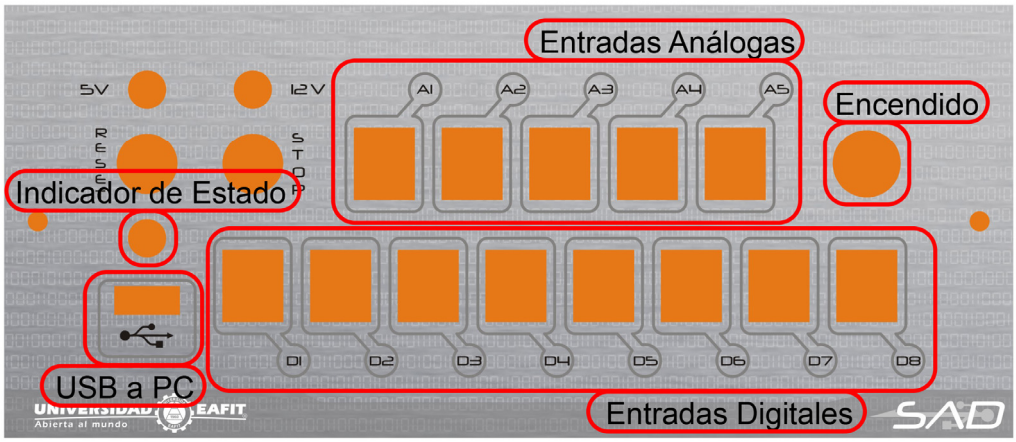
Seguidamente se describirá de una manera amplia cada unos de los subsistemas antes mencionados.

4.1 Hardware

A continuación se describen las partes y la función de los componentes del hardware. Es importante aclarar que para llegar al producto final fue necesario realizar diferentes pruebas hasta llegar a esta solución; una de las partes que más tiempo consumió fueron las pruebas con diferentes formas de comunicación a USB, siempre buscando aumentar la velocidad de comunicación para así minimizar las pérdidas de datos en el momento de las medidas.

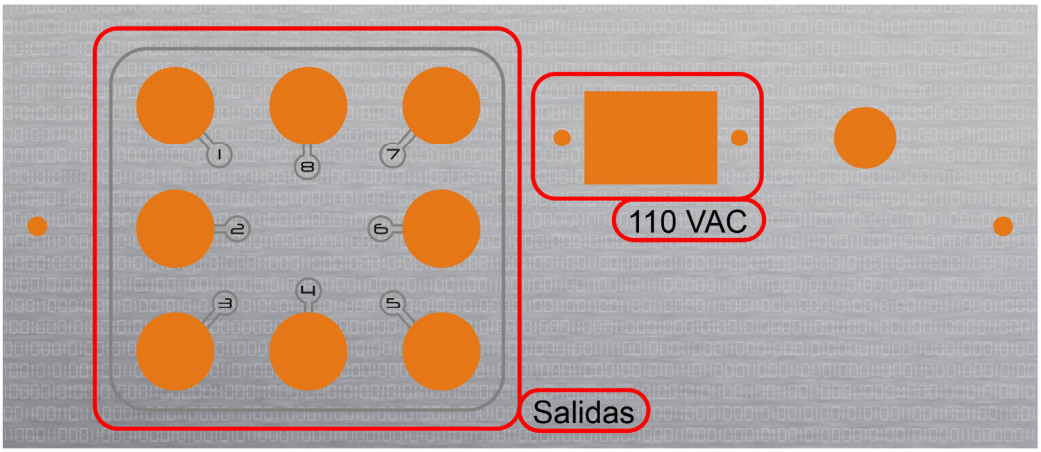
En la figura 16 se muestra el panel frontal de la carcasa y en la figura 17 se muestra el panel posterior de la misma, en la cual están contenidos los elementos descritos a continuación.

Figura 16: Diagrama de conexiones parte frontal



Fuente Daniel Ramírez V

Figura 17: Diagrama de conexiones parte posterior

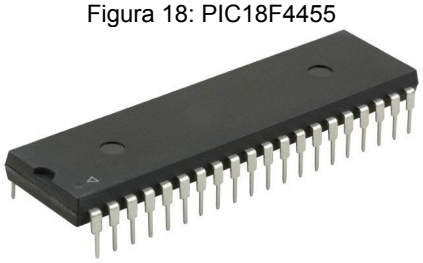


Fuente Daniel Ramírez V

4.1.1 Microcontrolador

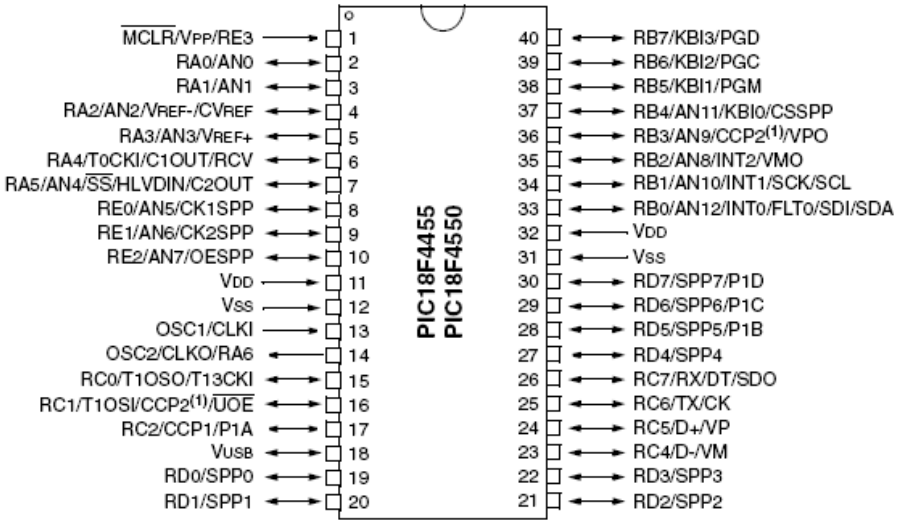
En un inicio se pensó en la posibilidad de emplear un microcontrolador pic18f2255 de la familia Microchip, pero debido a que el número de entradas y salidas no era suficiente, se cambió por un microcontrolador con un número mayor de conectores de entrada y salida. Era también indispensable que manejara comunicación USB, por lo que se eligió el pic18f4455 (figura 18 y 19). Por sus características es un microcontrolador óptimo para manejar del sistema que se construyó y a su vez implementar la comunicación vía puerto USB.

Figura 18: PIC18F4455



Fuente Daniel Ramírez V

Figura 19: Terminales del PIC 18f4455



Fuente Microchip

Entre sus funciones se encuentran la comunicación a través de USB (versión 1.1. a alta velocidad es decir hasta 12Mbps) con el PC, de donde recibe la información de la configuración con las salidas que deben generarse durante el proceso, además de realizar la lectura de las entradas análogas y entradas digitales y enviarlas a través del USB al computador. También maneja un indicador luminoso que muestra el estado en el que se encuentra, sea de desconexión, espera o toma y transmisión de datos. La forma como se usa cada uno de los terminales del PIC en el sistema SAD se puede ver anexo B.

A continuación se definen algunos elementos incluidos en el desarrollo del sistema de adquisición de datos y su funcionalidad, en la figura 20 donde se muestra el esquema de la tarjeta y la figura 21 la imagen de la misma ya terminada.
.

4.1.2 Transistores

Estos se tienen tanto para las entradas como para las salidas. En las salidas se usan porque, una vez la señal sale del microcontrolador ésta no es lo suficientemente potente para activar los relés. La función de los transistores es la de gobernar la corriente necesaria para activarlos.
En las entradas se usa para que el voltaje de entrada pueda ser desde 3V a 35V sin producir daños en el microcontrolador que recibe señales de entrada entre 0 y 5V.

4.1.3 Resistencias

Estas se encargan de limitar el paso de corriente a través del circuito, evitando que los componentes sufran daño, especialmente el microcontrolador. Entre más alto sea el valor nominal de éstas, mayor será la resistencia al paso de corriente.

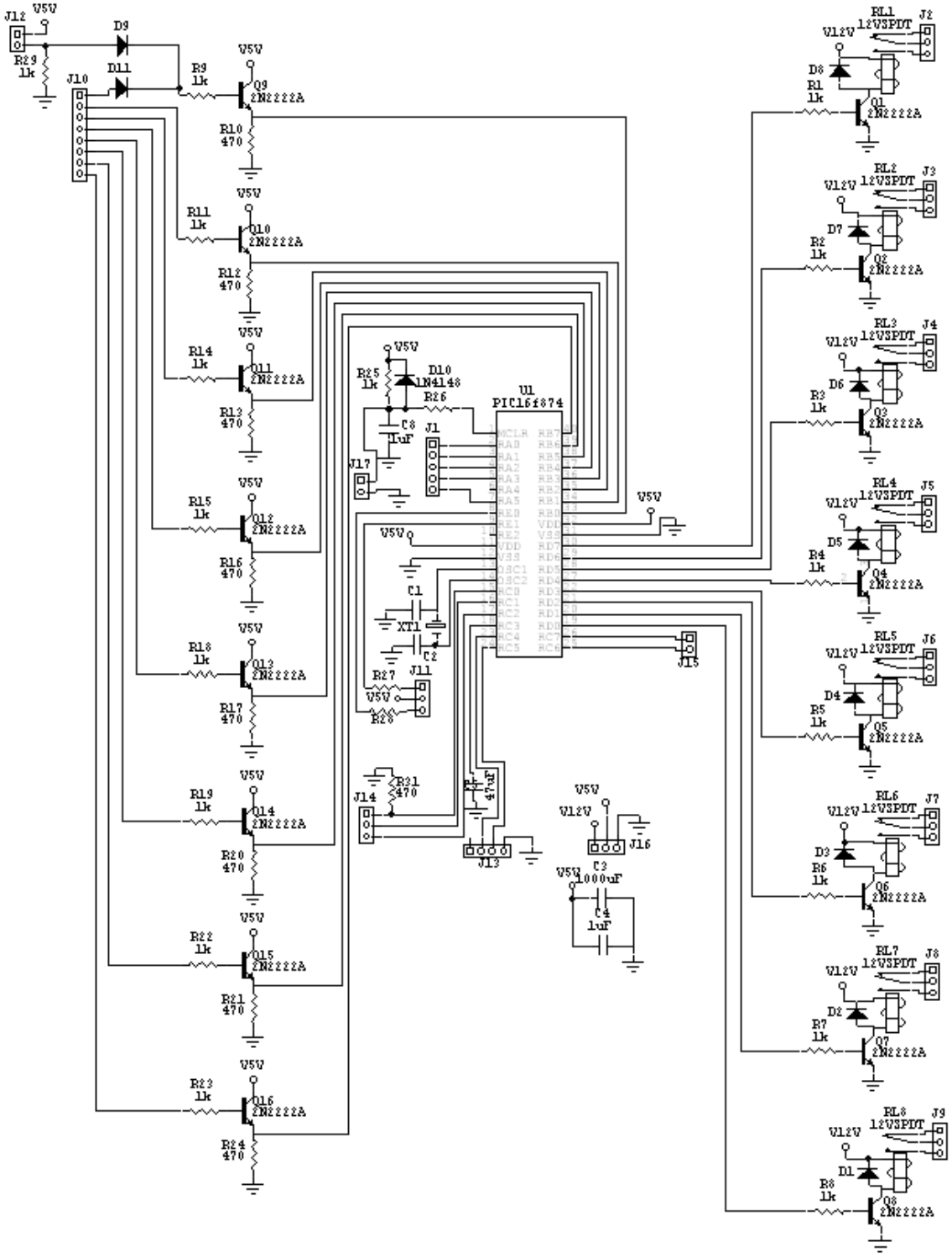
4.1.4 Condensadores

Tienen dos funciones una es filtrar o atenuar el ruido dentro del circuito, permitiendo un buen funcionamiento del control, y la otra ayudar a la oscilación del cristal.

4.1.5 Diodos

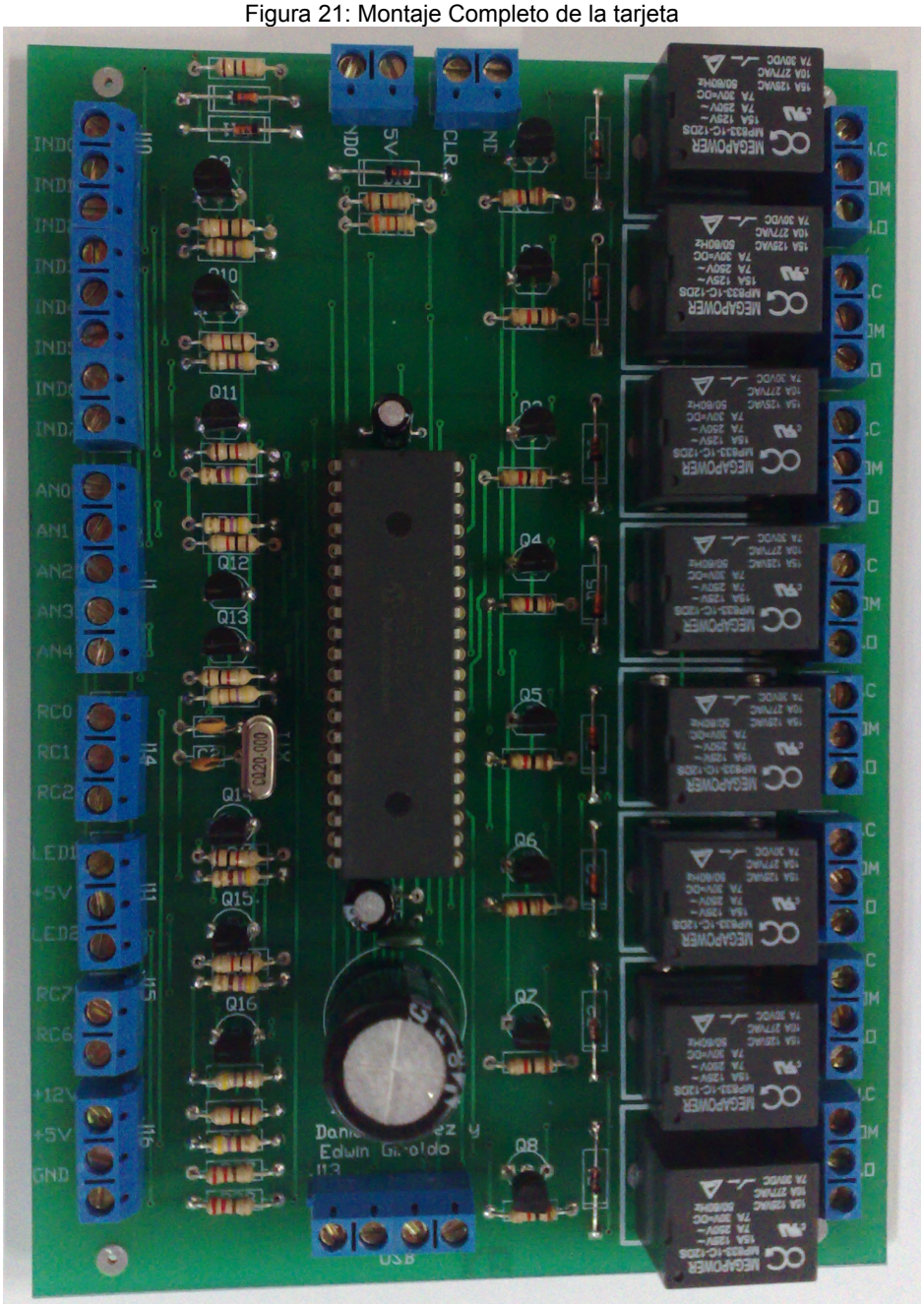
Estos están ubicados de tal manera dentro del circuito que evitan que la corriente inversa por la inductancia de la bobina del relé produzca daños en los circuitos de salida del microcontrolador que envían la señal a éstos.

Figura 20: Esquemático de Control



Fuente Edwin Giraldo A.

Figura 21: Montaje Completo de la tarjeta



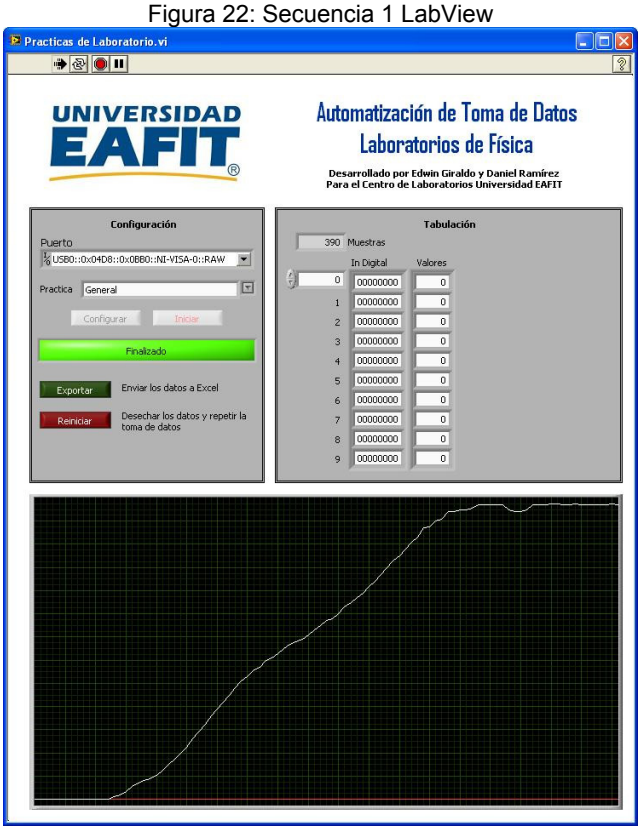
Fuente Edwin Giraldo A

Esta tarjeta va conectada a una fuente (Anexo A) y a los conectores del chasis.

4.2 Software

En la figura 22 se presenta el panel frontal del programa implementado en LabVIEW para la utilización del sistema SAD.

Figura 22: Secuencia 1 LabView

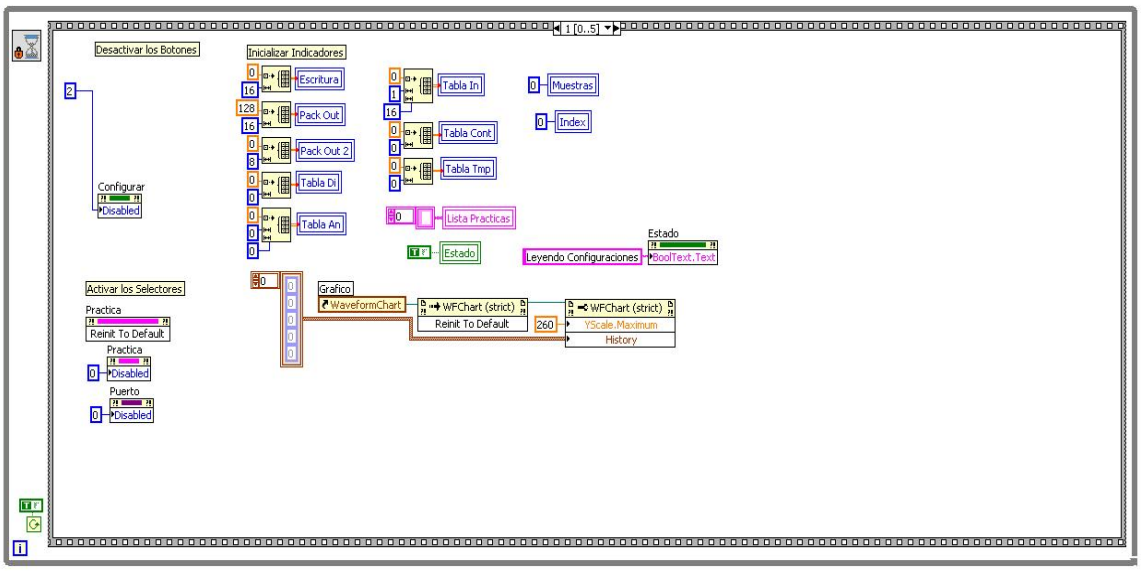


Fuente Daniel Ramírez

El usuario debe interactuar con selección de puerto, selección de práctica, configurar e iniciar. El indicador que está en verde muestra el estado del sistema y sirve al usuario como guía durante el proceso. Para ampliar el uso de estos remítase al Manual de Usuario, Anexo E.

A continuación se presentan las secuencias más importantes del programa, con una breve descripción a fin de ubicar al lector.
En la secuencia uno (1) se inicializan las variables en ceros, inicializan los selectores para que aparezcan sin texto, se ponen los botones inactivos y se activan el selector de puerto y práctica. A continuación se muestra la imagen de dicha secuencia, figura 23.

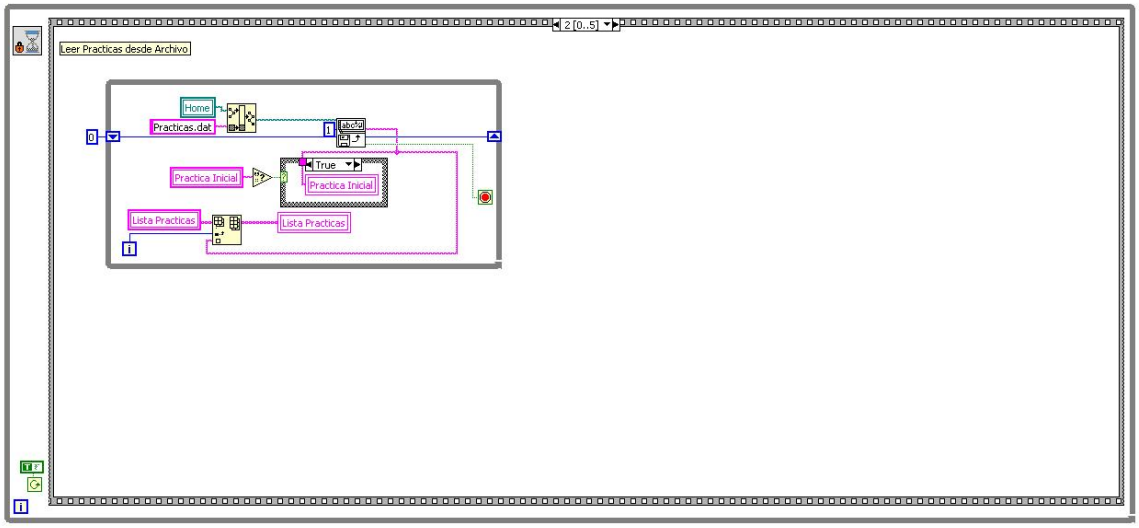
Figura 23: Secuencia 1 LabView



Fuente Daniel Ramírez

En la secuencia dos (2), se lee la lista de prácticas desde el archivo *Practicas.dat*, el cual es un archivo plano en el que está la lista de las prácticas, por ahora se encuentra General, Carga y descarga de condensadores y Dinámica de rotación, para agregar una nueva práctica solo se necesita incluirla en la lista dentro del archivo, tomar la plantilla (General.vi) y realizar los cambios respetando las salidas acorde a la definición de paquetes, como se muestra en la figura 24.

Figura 24: Secuencia 2 LabVIEW

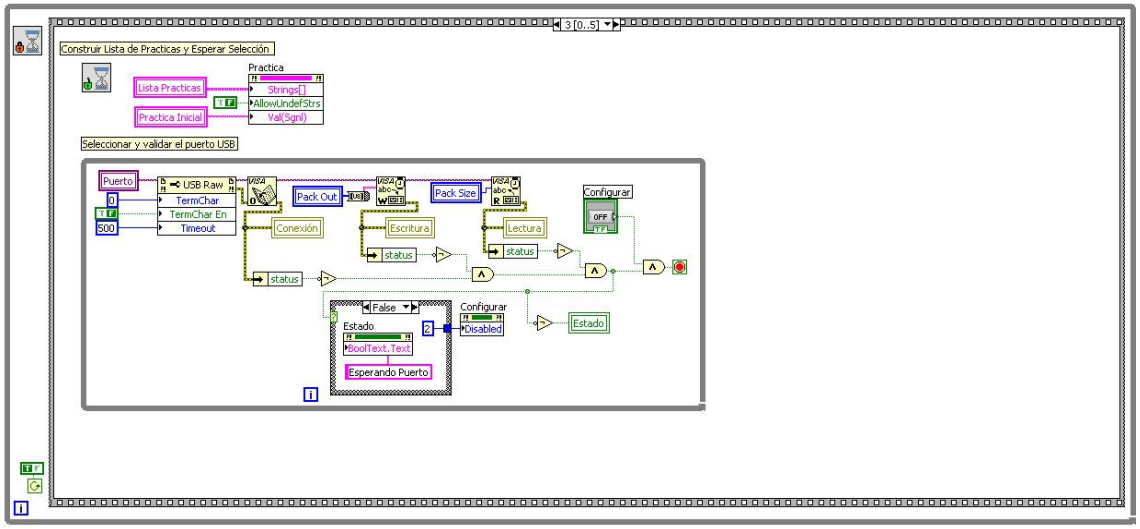


Fuente Daniel Ramírez

Continuando con el programa en LabVIEW la secuencia tres (3) pone los datos encontrados en el archivo leído por la secuencia anterior en el selector de práctica

(por defecto pone general) y espera la selección del usuario. Se queda esperando hasta que se realice la comunicación para poder continuar. En este momento activa el botón de configurar, ver figura 25.

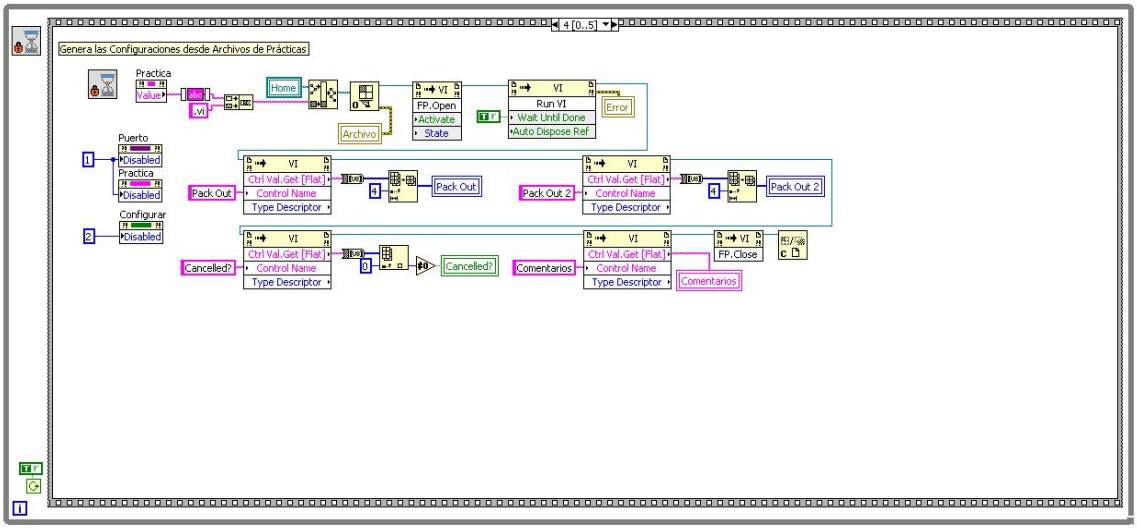
Figura 25: Secuencia 3 LabVIEW



Fuente Daniel Ramírez

La secuencia cuatro (4) despliega la ventana de opciones de configuración y toma los datos configurados por el usuario una vez son aceptados, además bloquea el selector de puerto, de práctica y configuración; además de activar el botón de inicio, ver figura 26.

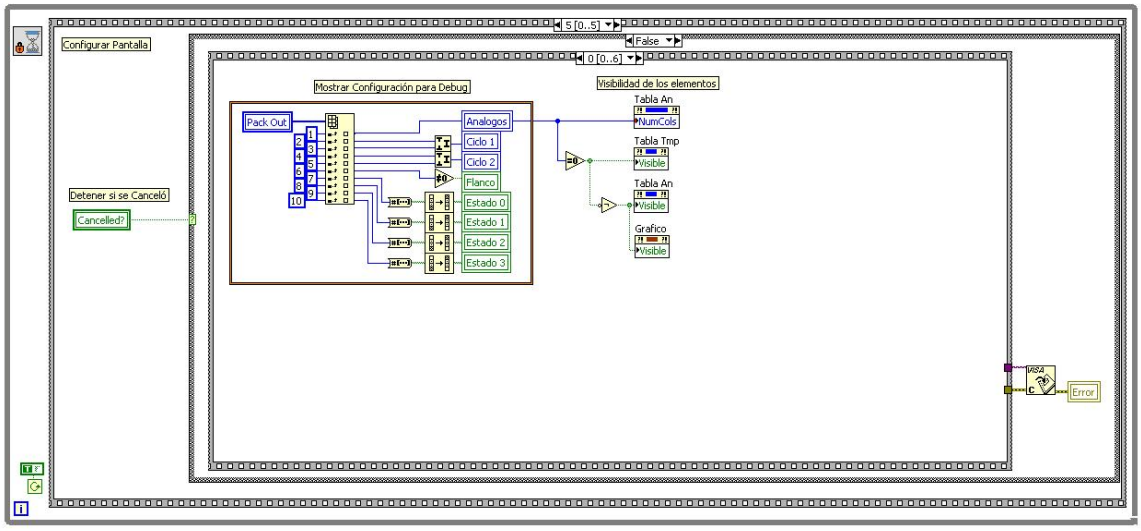
Figura 26: Secuencia 4 LabView



Fuente Daniel Ramírez

Dentro de la secuencia cinco (5), existe otra secuencia que va desde cero (0) hasta seis (6), en la secuencia 5-0 se generan la tabla de tabulación y la gráfica de acuerdo con la configuración seleccionada por el usuario en la secuencia anterior (la gráfica se muestra cuando hay al menos una entrada análoga), ver figura 27.

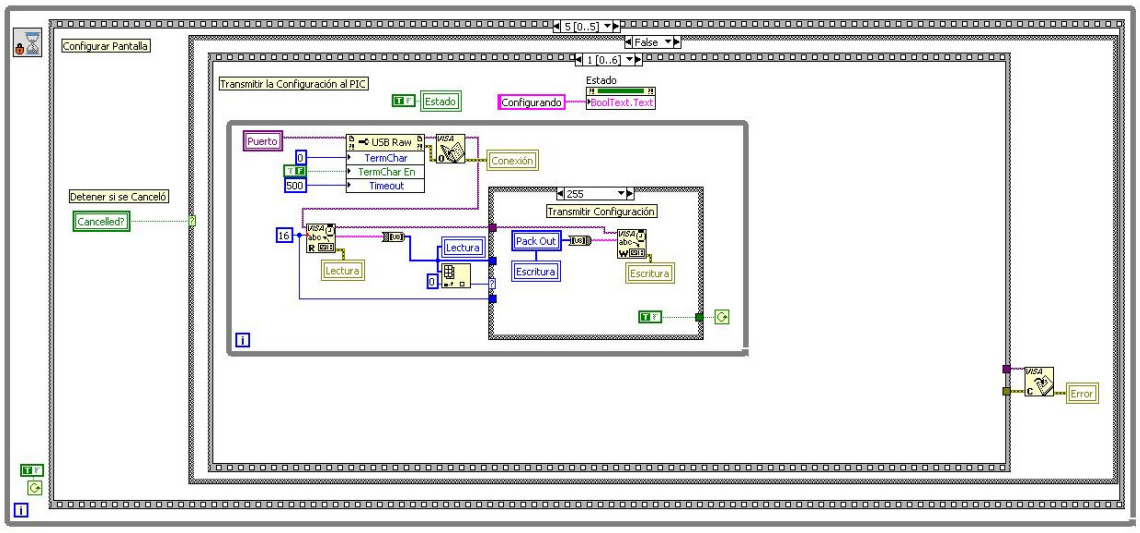
Figura 27: Secuencia 5 - 0 LabVIEW



Fuente Daniel Ramírez

La secuencia 5-1 transmite la configuración al microcontrolador PIC y se queda esperando la confirmación de este, ver figura 28.

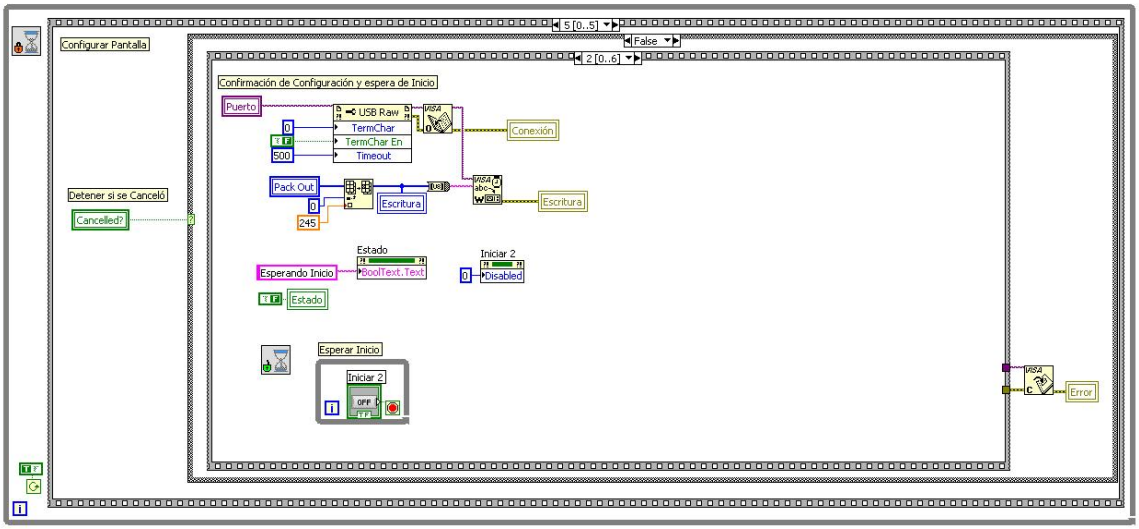
Figura 28: Secuencia 5 - 1 LabView



Fuente Daniel Ramírez

La secuencia 5-2 activa el botón iniciar y espera que el usuario lo seleccione, ver figura 29.

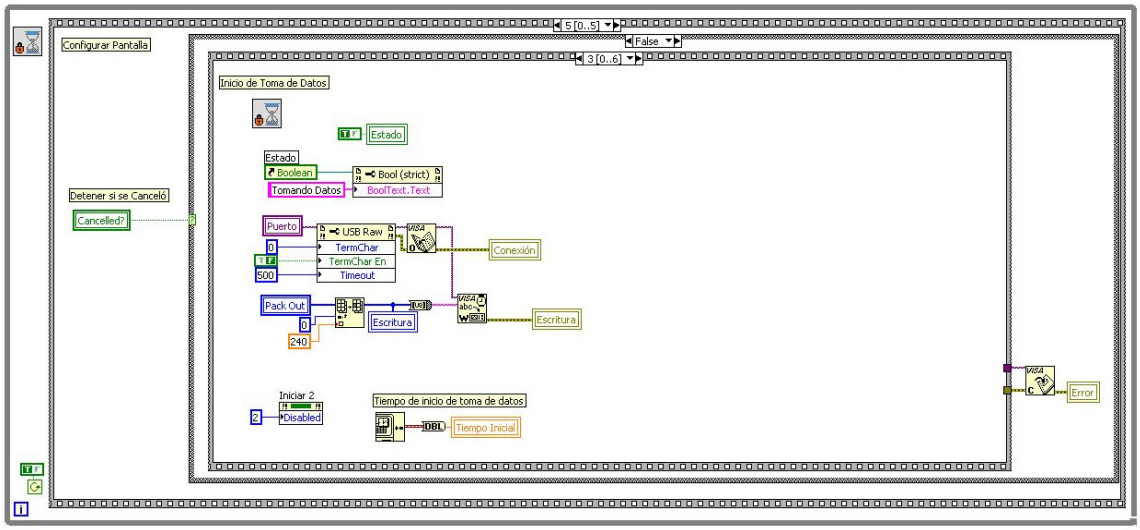
Figura 29: Secuencia 5 - 2 LabVIEW



Fuente Daniel Ramírez

En la secuencia 5-3 se deshabilita el botón iniciar, se envía la señal de inicio al microcontrolador PIC y toma el tiempo inicial en que comienza la toma de datos, ver figura 30.

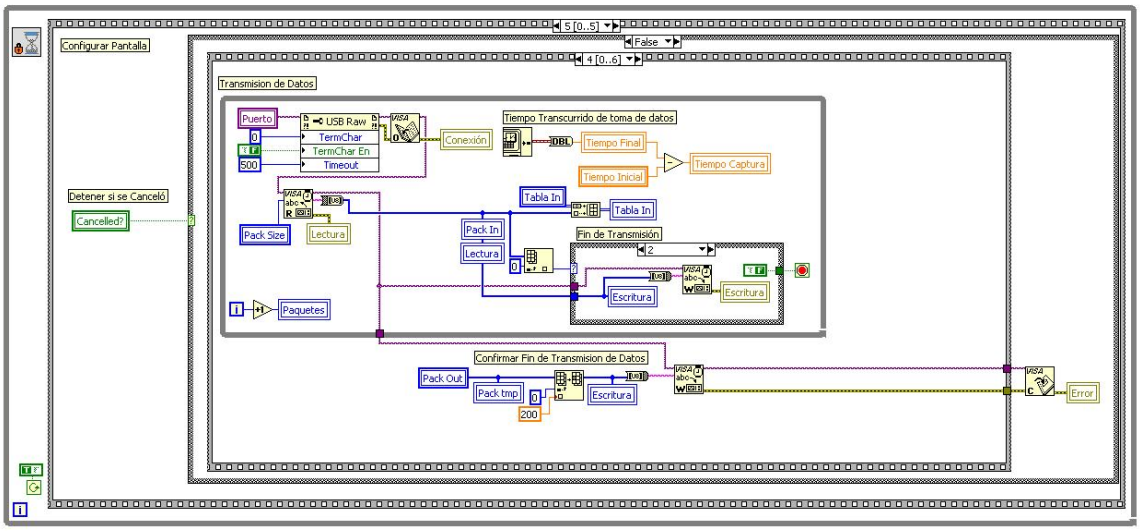
Figura 30: Secuencia 5 - 3 LabView



Fuente Daniel Ramírez

La secuencia 5-4 lee los datos y los almacena en una tabla hasta que el microcontrolador le envíe la señal de finalización (Id 200), ver figura 31.

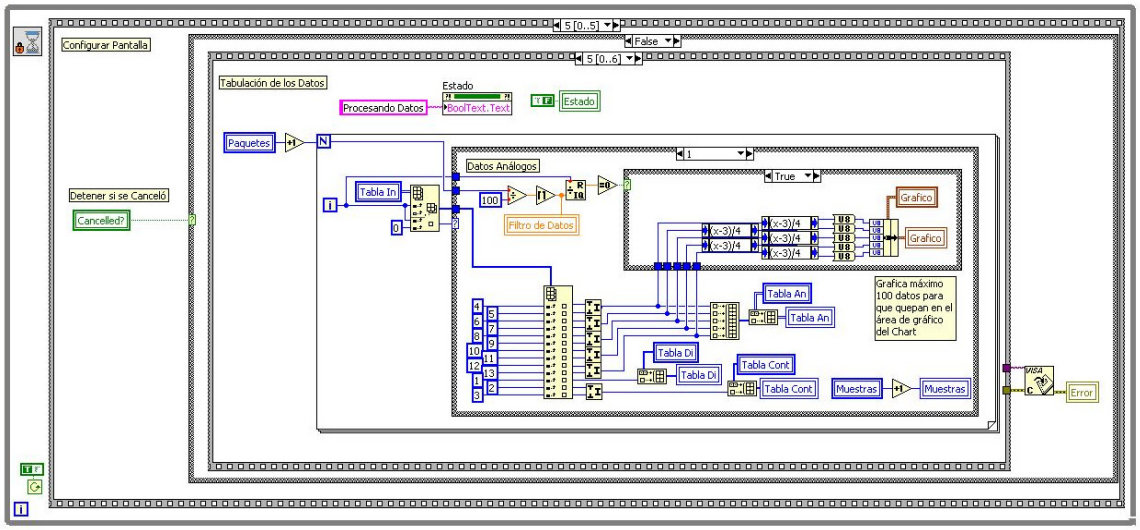
Figura 31: Secuencia 5 - 4 LabVIEW



Fuente Daniel Ramírez

La secuencia 5-5 construye la tabulación de datos en la tabla y construye el gráfico, ver figura 32.

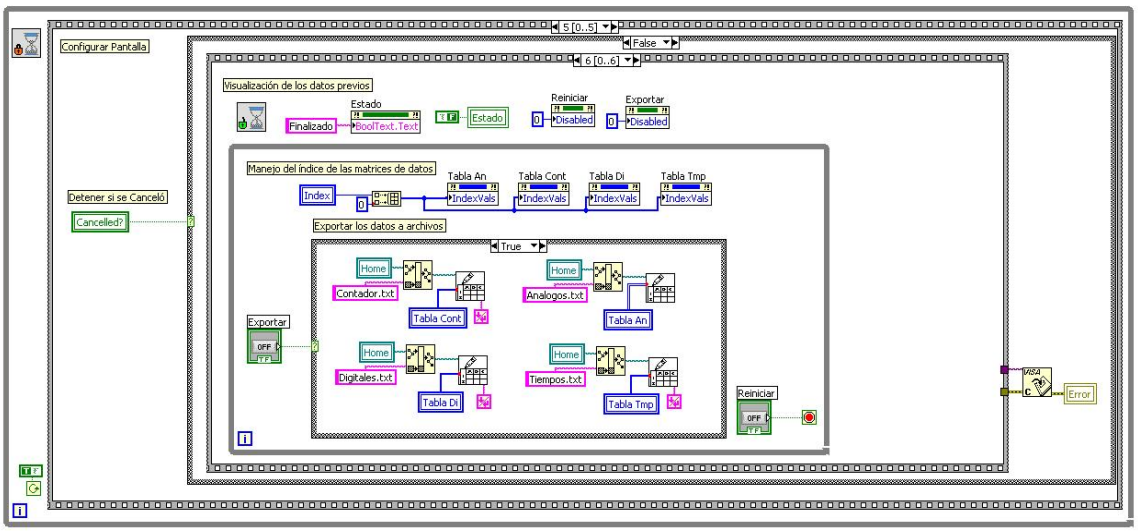
Figura 32: Secuencia 5 - 5 LabVIEW



Fuente Daniel Ramírez

La secuencia 5-6 activa los botones reiniciar y exportar, además permite la visualización de la tabla y exportar los datos a archivos planos; abre Excel y muestra datos en Excel, ver figura 33.

Figura 33: Secuencia 5 - 6 LabVIEW



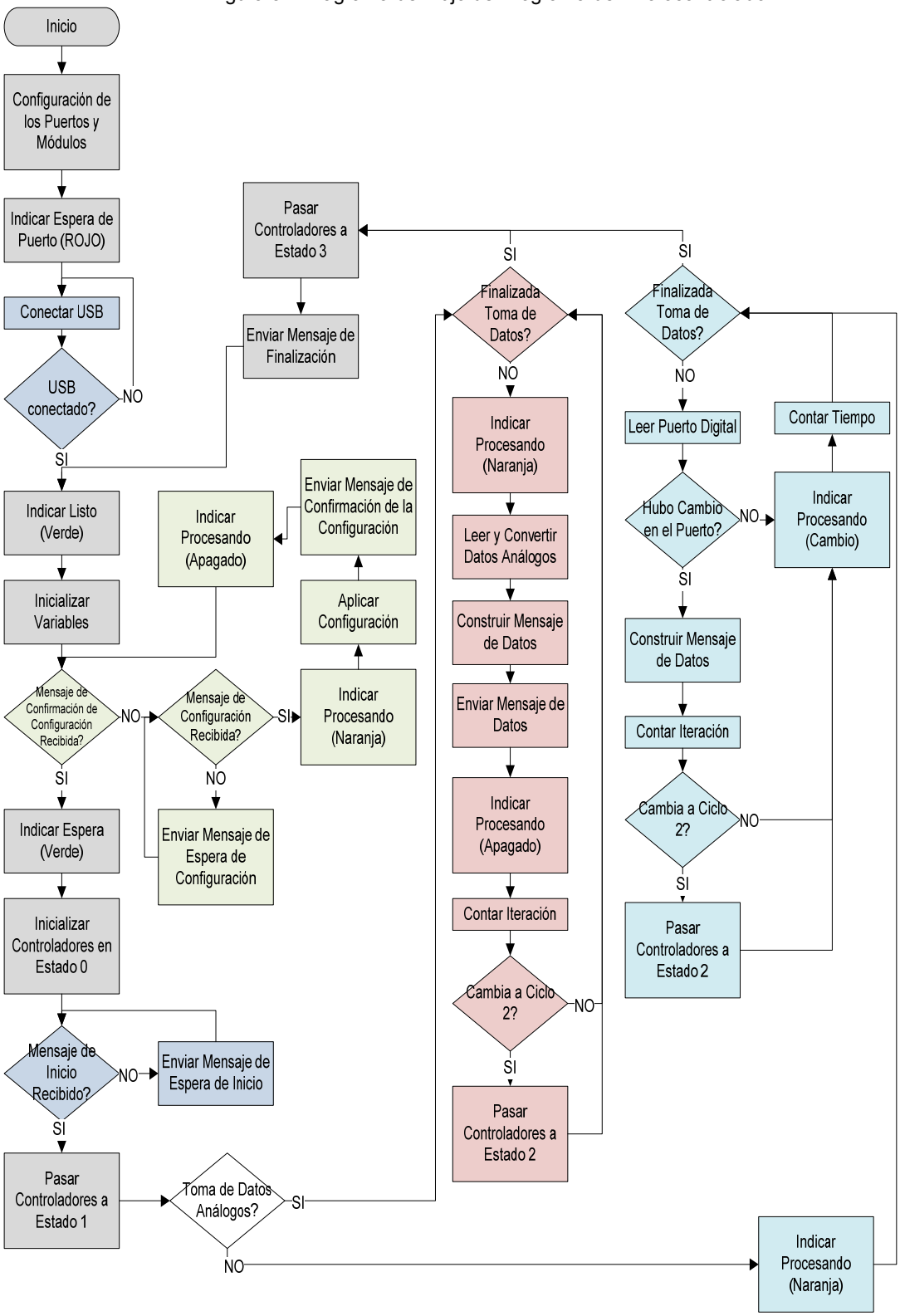
Fuente Daniel Ramírez

4.3 PICC

Para la programación del microcontrolador PIC, se utilizaron las librerías incluidas en el software PICC, en especial las de configuración, control y transmisión de USB. Estas facilitaron en parte la programación y comunicación.

La figura 34 muestra el diagrama de flujo del programa que controla el PIC, en el cual se observan en color gris las configuraciones, en azul oscuro se aprecia el ciclo de inicialización de comunicación por el protocolo USB con el PC, en verde se encuentra el ciclo de lectura y confirmación entre el PIC y el PC de configuración interna para la práctica específica que se procesará y, dependiendo de esta configuración, se ejecuta la línea roja, en la cual se realizan lecturas análogas y digitales, o la línea azul para lecturas solo digitales. El código completo en C, se puede consultar en el anexo F

Figura 34: Diagrama de Flujo de Programa del Microcontrolador



Fuente Daniel Ramírez

5 CONCLUSIONES

Desde los objetivos

Se desarrolló un sistema de adquisición automatizada de datos adaptado a las necesidades de los laboratorios de Física de la Universidad EAFIT.

La construcción del sistema de automatización de toma de datos para las prácticas de laboratorio de física permite que los estudiantes centren sus esfuerzos en el análisis de los fenómenos estudiados. Adicionalmente el sistema disminuye los errores que pudieran ser introducidos por la manipulación humana en la toma de datos.

Para el desarrollo de las prácticas fue necesario identificar las variables físicas relevantes de tal forma que las gráficas puedan ser construidas a partir de ellas.

Se emplearon sensores para la captura de los datos y fueron llevados a procesamiento usando protocolos de transmisión por medio de USB.

El sistema fue inicialmente creado con las prácticas de carga y descarga del condensador, y dinámica rotacional. Estas fueron seleccionadas para poder modelar sistemas digitales y análogos.

Desde los hallazgos

Se inició un trabajo con el protocolo USB a baja velocidad tomando como referencia un proyecto de grado *Diseño E Implementación De Un Control Automático Para Una Máquina Cortadora De Piezas De Madera*. Durante el desarrollo de este proyecto se cambió de controlador lo que permitió el aumento de la velocidad transmisión de los datos al PC, además que es mucho más fácil de manejar y explicar.

El uso del nuevo controlador de USB permitirá la actualización eficiente del software de control LabVIEW usado en la universidad sin importar la versión ya que el controlador viene incluido en el software LabVIEW a partir de la versión 7.0

Se encontró que la velocidad de transmisión de datos de la interfaz USB es mucho más alta que la recepción de datos desde LabVIEW, por lo cual esta tuvo que ser restringida a la más lenta, que es de 4milisegundos entre ciclos de transmision sin que se presente pérdida de datos.

El desarrollo de este proyecto permitirá a la universidad EAFIT disponer de un sistema de excelentes prestaciones y menor costo comparado con los sistemas MultiLogPRO, PhidgetInterfaceKit 8/8/8, ScienceWorkshop 750 de pasco, además

posibilita el uso de cualquier sensor solo adicionando un acondicionador de señal, facilita el mantenimiento del sistema y es un equipo construido pensando en las necesidades de nuestros laboratorios, pero con una arquitectura lo suficientemente robusta que le permite ajustarse a prácticas de laboratorio diferentes a las aquí descritas.

6 TRABAJOS FUTUROS

El trabajo que se hizo con el sistema SAD es el inicio de numerosas oportunidades para el crecimiento del mismo. El sistema puede crecer de muchas maneras diferentes dependiendo al área que quiera enfocarse. En el área de programación, se puede hacer una aplicación más robusta en el computador.

Reproducción del módulo para realizar las prácticas automatizadas del laboratorio de física.

Implementación de las prácticas restantes.

Adecuación e Implementación de hardware para acondicionar los sensores y equipos de prácticas de laboratorios restantes.

Desarrollo de una interfaz más rápida que puede servir de adquisición de datos para proyectos de investigación donde velocidad de captura debe ser mayor.

7 BIBLIOGRAFÍA

- Diccionario RAE. Real Academia Española. Madrid, 2005
- Enciclopedia Encarta 2008. Microsoft.
- INSTITUTO COLOMBIANO DE NORMAS TÉCNICAS Y CERTIFICACIÓN. TRABAJOS ESCRITOS:presentacion y referencias bibliograficas.2008. Colombia.
- Microchip Technology. “Embedded Control Handbook Volume 1” 1997, USA
- Microchip Technology. Datasheet PIC16C745/765. 2000. USA.
- Microchip Technology. DataSheet PIC18F2455/2550/4455/4550. 2004. USA
- Mindshare,Inc. ANDERSON Don. ”Universal Serial Bus System Architecture” Ed Addison- Wesley Longman, Inc 1998 . USA
- PALLÁS ARENY, Ramón. “Diafonía En Circuitos Impresos. Criterios De Diseño”. Revista Mundo Electrónico Diciembre 1993. Boixareu Editores. Barcelona, España.
- RENDON, Elizabeth. Diseño y Construcción de una cortadora CNC de patrones en balsa.. Medellín 2001. Trabajo de grados Ingeniería Mecánica. Universidad EAFIT. Departamento de Ingeniería Mecánica.
- RODRÍGUEZ, LORA, Vanessa y VILLEGAS, Nicolás. Diseño E Implementación De Un Control Automático Para Una Máquina Cortadora De Piezas De Madera. Medellín 2001. Trabajo de grados Ingeniería de Sistemas. Universidad EAFIT. Departamento de Ingeniería de Informática y Sistemas.
- SÁNCHEZ MILLARES, Álvaro. Sensores y actuadotes. Instituto de investigación técnica. 2004
- MURILLO, Hoyos, Hugo. Guía de Hardware y conexiones. Universidad EAFIT. 2007.
- MURILLO, Hoyos, Hugo. Guía de Laboratorio de Control. Universidad EAFIT. 2001.
- CCS. Manual Reference PICC Compiler. 2005, USA.

Páginas Web

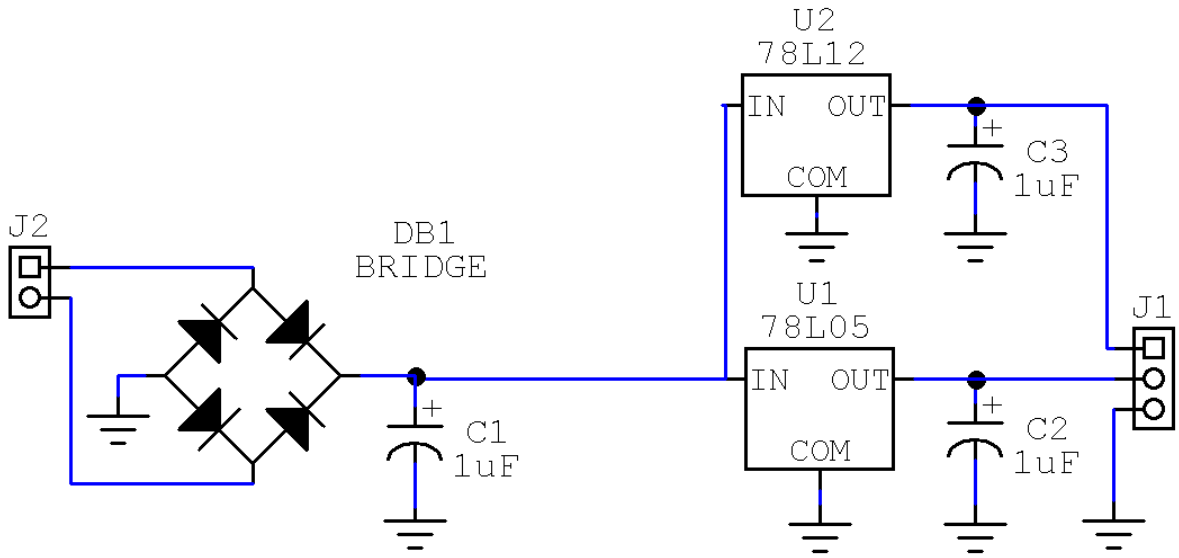
- http://www.phidgets.com/. Pagina visitada el 20 de diciembre de 2008
- http://www.electrocomponentes.com.ar/. Pagina visitada el 3 de enero de 2009.
- http://www.pasco.com/. Pagina visitada el 8 de enero de 2009.
- http://www.todorobot.com.ar. Página Web visitada el 2 de febrero de 2009
- http://www.ccsinfo.com/picc. Pagina visitada 15 de enero de 2009.
- www.beyondlogic.org. Página visitada el 28 de enero de 2009.
- http://www.semiconductors.philips.com. Página Visitada el 17 de marzo de 2009.
- http://www.monografias.com. Página Visita el 30 de abril 2009
- http://www.electronica2000.250x.com. Página visitada el 2 de mayo de 2009.
- http://www.forosdeelectronica.com. Página visitada el 10 de mayo de 2009

8 ANEXOS

8.1 Anexo A. Tarjeta reguladora de voltaje

Esta tarjeta se encarga de mantener constante el voltaje, entrega dos potenciales 5V para la tarjeta de control del microncotrolador y 12V para los relés, ambos voltajes están en los conectores de salida para ser usados por el usuario, figura 35

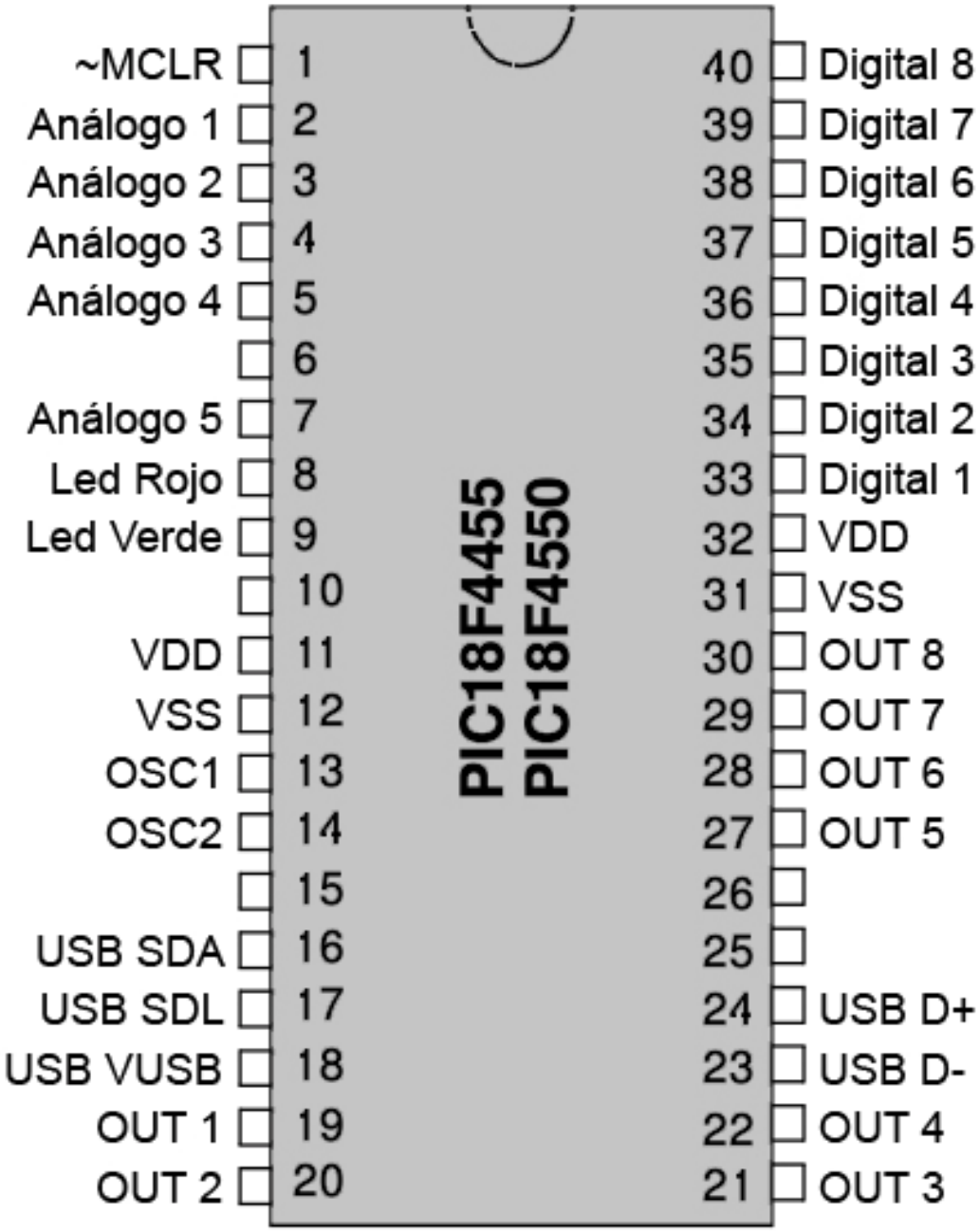
Figura 35: Diagrama Esquemático de Fuente de 12VDC y 5VDC



Fuente Edwin Giraldo

8.2 Anexo B. Distribución de conexiones del PIC 18F4455

Figura 36: Distribución de Pines del PIC 18F2550



Fuente Daniel Ramírez

8.3 Anexo C. Especificación técnica proyectos similares de terceros

Figura 37: Especificaciones técnicas del MultiLogPRO

Especificaciones del MultiLogPRO	
Entradas	Hasta 8 entradas simultáneas analógas
	Hasta 2 entradas simultáneas digitales
	65 sensores externos disponibles
Salidas	Interfase PC RS-232 a 19200 bps
	Comunicación USB
	2 salidas de Control Digitales
Muestreo	Capacidad: 104,000 muestras
	Ritmo: Variable, desde 1 Muestra/Hora hasta 20,800 Muestras/Seg
	Resolución: 12 bit (4096 niveles)
Características	Teclado: permite la programación manual
	Pantalla gráfica LCD: 64 x 128 pixels, vista de especificaciones y datos medidos
	Operación independiente: Trabaja y toma muestras sin estar conectado al computador
	Reconocimiento automático de sensores (para 4 entradas)
	Auto comprobación automática: Informa sobre el estado del sistema
	Activador: Programable o manual
	Calibración: Calibración automática del desbalance de los sensores
	Cronómetro incorporado para recolecciones postergadas
	Función de cronometraje por medio de uno o dos fotopuentes
	Batería de respaldo de memoria: Guarda los datos hasta 5 años
	Reloj y calendario incorporados: Registra fecha y hora de los datos almacenados
	Batería recargable 7.2V incorporada
	Grabación de eventos
Suministro de Energía	Voltaje: Batería interna 7.2 V NiCA o entrada externa de 9V a 12V CD
	Vida útil de la batería (sin sensores conectados): 50 horas
Software	MultiLab
Rango de Temp. de funcionamiento	0°C a 50°C
Dimensiones	185 x 100 x 32 mm
Peso	~450gr
Conformidad con estándares	CE, FCC

Fuente www.electrocomponentes.com.ar

Tabla 9: Especificaciones técnicas ScienceWorkshop 750 Interface de Pasco

Power:	• 12 VDC to 20 VDC at 2 A, 2.1 mm jack
Digital Channels:	• 4 identical channels, TTL compatible (8 mA max. drive current) • Maximum input logic transition time: 500 ns • Edge sensitive-sampled at 10 KHz. (1 µs res. for Motion Sensor)
Analog Input Channels:	• 3 identical channels with differential inputs and 1 MOhm impedance • ±10 V maximum usable input voltage range (±12 V absolute input voltage range) • 3 voltage gain settings on each analog channel: 1, 10, and 100 • Small signal bandwidth up to the ADC: 1 MHz for a gain of 1, 800 KHz for a gain of 10, and 120 KHz for a gain of 100; input amplifier slew rate: 1.2 V/ µs
Electrostatic Discharge (ESD) protected:	• Both digital and analog inputs have ESD protection
12-Bit Analog to Digital Conversion:	• 5 inputs: channels A-C, analog output voltage and current • Voltage resolution at ADC input: 4.88 mV (.488 mV at a gain of 10, 0.049 mV at a gain of 100) • Current measurement resolution: 244 µA, (1 V = 50mA) mA • Offset voltage accuracy < ±3 mV. (For measuring full-scale voltages the total error is less than ±15 mV, accounting for the gain error in the input amplifier.) • Sample rate range: once every 3,600 seconds (250 KHz) (Conversion time for consecutive channels in a burst is 2.9 µs.) • 8X oversampling for better accuracy at sample rates less than or equal to 100 Hz.
Analog Output:	• DC value ranges: -4.9976 V to +5.0000 V in steps of 2.44 mV • Accuracy at the DIN connector: (±3.6 mV ±0.1% full scale) • Peak-to peak amplitude adjustment ranges for AC waveform: 0 V to ±5 V in steps of 2.44 mV • AC waveform frequency ranges: 0.001 Hz-50 KHz, ±0.01% • Maximum amplified output at the banana jacks: about 300 mA at ±5 V, current limited at 300 mA ±12 mA

Fuente Pasco

Tabla 10: Especificaciones técnicas de PhidgetInterfacekit 8/8/8

Device Specifications	
Analog Input Impedance	900K ohms
Analog Input 5V Reference Error	Max 0.5%
Digital Output Series Resistance	300 ohms
Digital Input Pull-Up Resistance	15K ohms
Analog Input Update Rate	~65 samples / second
Digital Output Update Rate	~125 samples / second
Digital Input Update Rate	~125 samples / second
Digital Input Recommended Wire Size	16 - 26 AWG
Digital Output Recommended Wire Size	16 - 26 AWG
Digital Input Wire Stripping	5-6mm strip
USB-Power Current Specification	Max 500mA
Quiescent Current Consumption	13mA
Available External Current (source)	487mA

Fuente Phidget

8.4 Anexo D. Manual del sistema

Manual del sistema

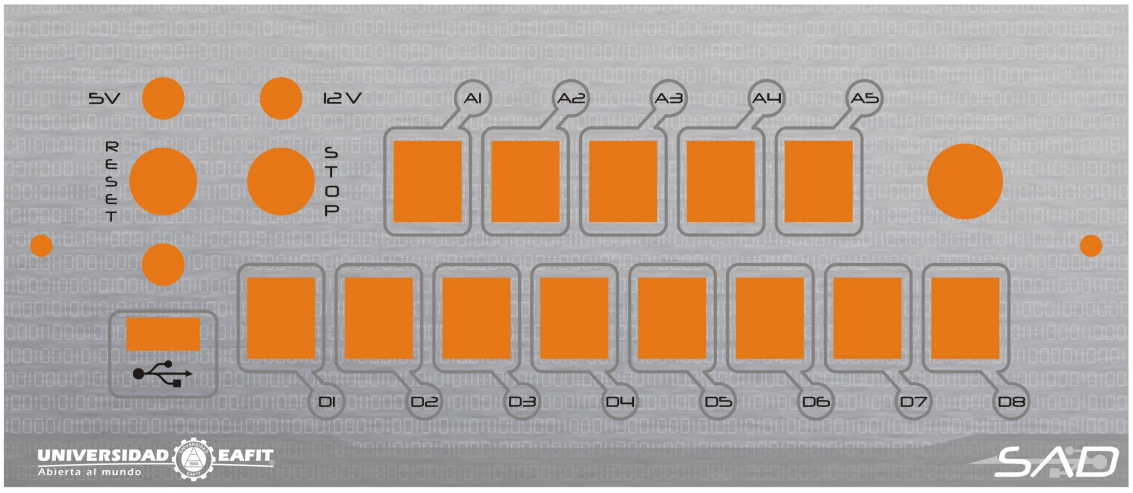
Requisitos del sistema
Sistema Operativo Windows XP o Vista.
Procesador Pentium Core.
512 MB de memoria RAM.
Espacio en disco 5 MB.

Software
LabView 8.5 o superior.

Otros
Hardware de Automatización de Toma de Datos.
Sensores y actuadores correspondientes a la practica a realizar.
Debe estar configurada la conexión de USB como conexión NI-VISA. Para esto se copia el archivo “ADAQ PIC18FF4455.inf” para Windows XP, o el archivo “ADAQ PIC18FF4455_vista.inf” para Windows Vista, en la ruta “c:/Windows/system/”; una vez copiado se debe dar click derecho sobre el archivo y seleccionar “instalar”.

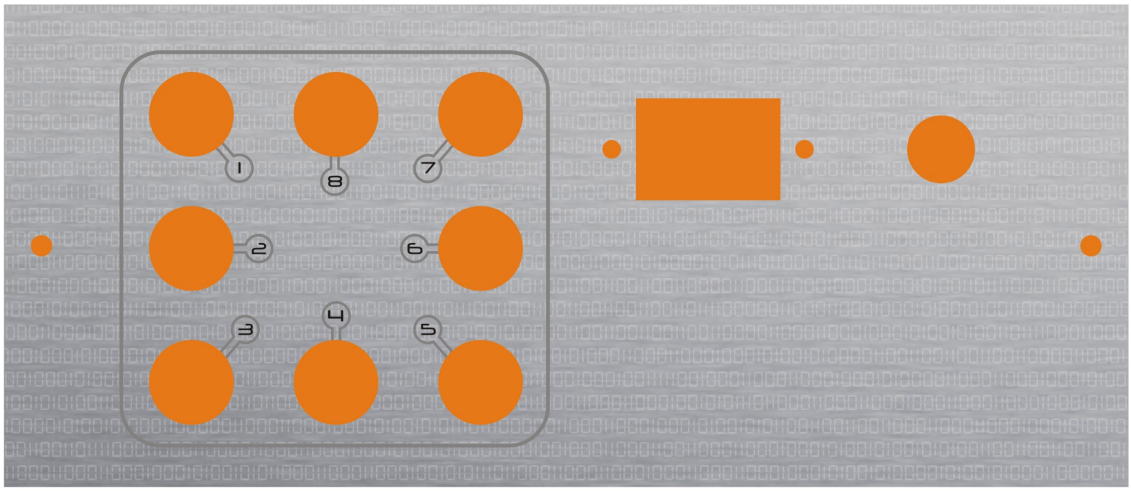
Conexión
Con el objeto de evitar daños en los componentes del control y por ende problemas en su funcionamiento, las terminales de las conexiones se diseñaron para que fueran diferentes. Las conexiones deben ser de la siguiente manera, ver figura 38.

Figura 38: Vista Frontal de la Interfaz



Fuente Daniel Giraldo

Figura 39: Vista Posterior de la Interfaz



Fuente Daniel Giraldo

Las entradas se deben conectar por la parte frontal del equipo; los sensores análogos se deben acoplar en los conectores de la parte superior, es importante resaltar que la señal debe tener una variación de voltaje entre 0 y 5VDC; por su parte los conectores de la parte inferior son para acoplar los sensores digitales, estas entrada están diseñadas para aceptar sensores que entreguen voltajes para señal alta entre 5VDC y 30VDC, y como señal baja 0VDC.

Las salidas se conectan por la parte posterior del equipo de captura y control. Estos conectores tienen en sus terminales un contacto normalmente cerrado, normalmente abierto y común, además de 5V, 12V y GND, para ser usados de acuerdo al tipo de salida que sea necesario activar para la captura de datos de las practicas. Adicionalmente, estos conectores pueden controlar conexiones de 110VAC y 220VAC, tomados desde alguna fuente externa.

El equipo tiene tres indicadores que pueden ayudar al usuario a detectar errores, un indicador de correcto funcionamiento de la fuente de 12VDC, otro de 5VDC, además hay otro indicador que muestra la fase del proceso en el que se encuentra el equipo; el color rojo indica que se encuentra esperando conexión del USB con el PC, el verde cuando está conectado y en espera de la configuración o la orden de inicio de captura, y en color naranja cuando está en proceso de captura y transmisión de datos.

En lo correspondiente al software: para agregar una práctica nueva, es necesario tomar una copia del archivo de práctica general “General.vi” y modificar la configuración de esta para adecuarla a la nueva práctica, por medio del cambio de la pantalla visible, eliminando los campos de visualización que no sean necesarios y adicionar las formulas de cálculo de los valores de configuración de acuerdo a la práctica. Esta labor debe ser realizada por una persona cualificada en el manejo y

programación de LabView, y del Sistema de Automatización de Toma de Datos; después de generar el archivo de configuración, se debe adicionar en el archivo “Practicas.dat” el nombre exacto del nuevo archivo, excluyendo únicamente la extensión del mismo (“.vi”), pues este será el nombre que se mostrará la lista de selección de prácticas. Por ejemplo, si la nueva práctica se llama “Movimiento Lineal Uniformemente Acelerado”, el archivo deberá llamarse “Movimiento Lineal Uniformemente Acelerado.vi”, y la nueva línea en el archivo de configuración “Practicas.dat” será “Movimiento Lineal Uniformemente Acelerado”.

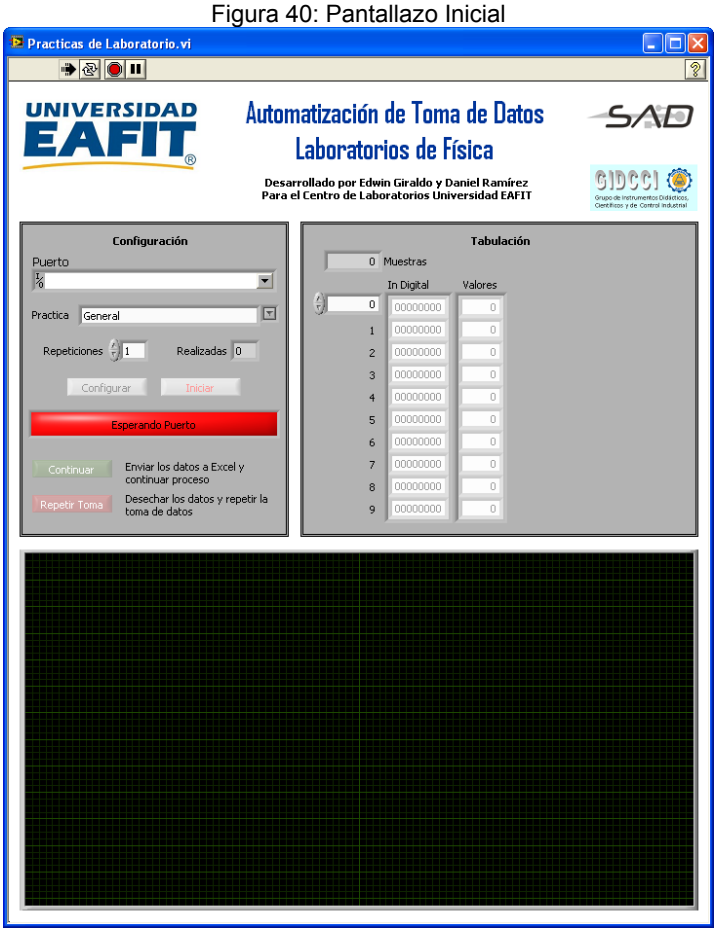
En caso de presentarse algún daño en el equipo, esté solo debe ser reparado por personal cualificado.

8.5 Anexo E. Manual de Usuario

Manual de Usuario

Cómo usar el programa
Iniciar ejecutando el archivo “Practicas de Laboratorio.vi”. Se recomienda antes de iniciar el programa conectar la interfaz de captura a través de un puerto USB. Aparecerá la pantalla que se muestra a continuación.

Figura 40: Pantallazo Inicial



Fuente Daniel Ramírez y Edwin Giraldo

Para poder seleccionar y ejecutar una práctica, el indicador de estado de la interfaz debe encontrarse de color Verde. En caso de que esta permanezca en color Rojo, se debe pulsar el botón “Reset” ubicado en la parte frontal superior izquierda del dispositivo.

En la pantalla aparece la barra en color rojo con el mensaje “Esperando puerto”. Se debe seleccionar en el campo “Puerto” y elegir “USB0::0x4DB8::NIVISA-#::RAW”, si este no aparece cerciórese que el equipo de captura esté conectado en un puerto USB y seleccione “Refresh” en el campo “Puerto”. Una vez realizada la comunicación con el equipo de captura y control, la barra de estado que estaba

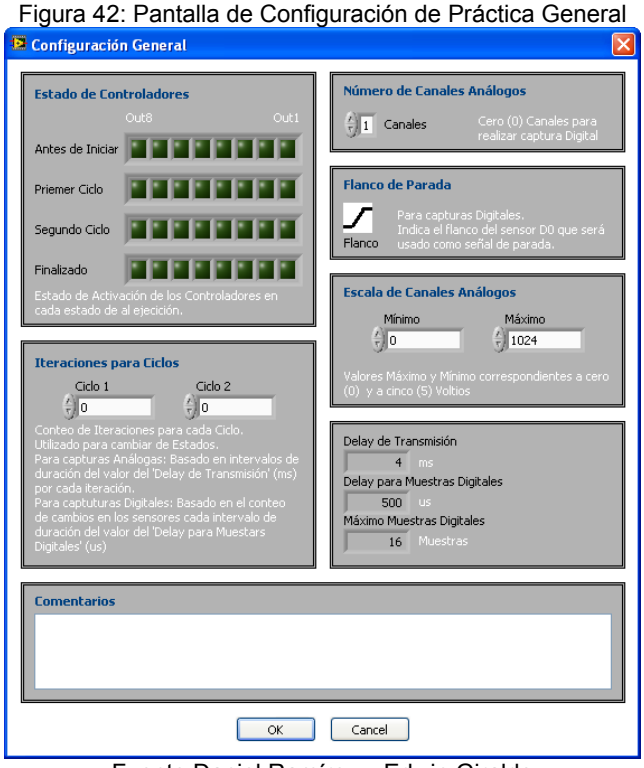
en color rojo cambiara a verde y el nuevo mensaje será “Esperando configurar”, como se ve en la siguiente figura.

Figura 41: Configuración de Puerto y Práctica

Fuente Daniel Ramírez y Edwin Giraldo

En el campo “Páctica” debe seleccionarse el nombre de la práctica de laboratorio a realizar, por defecto está seleccionada la práctica “General”. Una vez seleccionada ésta, el usuario debe dar click en el botón “Configurar”; al presionar éste aparecerá la pantalla de configuración de la práctica seleccionada.

Figura 42: Pantalla de Configuración de Práctica General



Fuente Daniel Ramírez y Edwin Giraldo

Esta ventana cambia de acuerdo a la práctica seleccionada; en este caso se muestra la de configuración general ya que es la que mas parámetros permite modificar (en las practicas especificas solo es necesario cambiar los campos que aparezcan en cada una de las diferentes pantallas). Para el caso de la práctica de “Carga y Descarga de Condensador” se ingresan el valor del Capacitor en micro Faradios y el valor de las Resistencias en ohmios y en el caso de la práctica de “Dinámica Rotacional” se ingresan los valores de las masas, los radios y la altura en unidades del sistema internacional.

En la práctica general se configuran la cantidad de puertos análogos que se van a capturar (0 para capturas Digitales), numero de ciclos, flanco de finalización si es por nivel alto o bajo y las salidas de acuerdo a los estados.

Cuando el usuario acepta la configuración se bloquean los campos “Puerto” y “Práctica” y el botón de “Configuración”; acto seguido se activa el botón de “Inicio” y la barra de estado muestra el mensaje “Esperando Inicio” como se ve en al siguiente figura.

Figura 43: Espera de Inicio

Fuente Daniel Ramírez y Edwin Giraldo

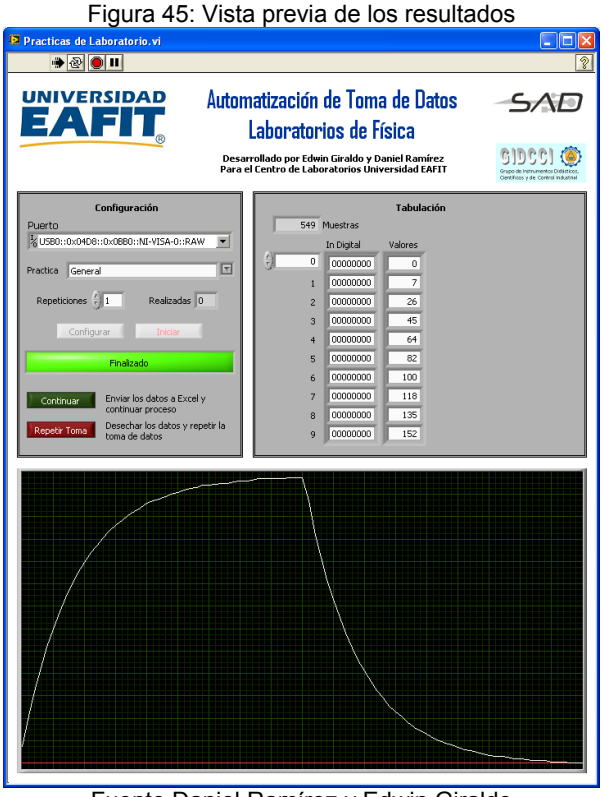
El usuario debe dar click en “Iniciar” para empezar la captura de datos. Una vez hecho esto , la barra de estado muestra el mensaje “Tomando Datos” y comienza la adquisición y transmisión de datos del fenómeno bajo estudio. El indicador de estado de la interfaz comienza a titilar de color Naranja.

Figura 44: Estado de Transmisión de Datos

Fuente Daniel Ramírez y Edwin Giraldo

Cuando llega la señal de finalización se despliega una vista del resultado de la captura.

Figura 45: Vista previa de los resultados



Fuente Daniel Ramírez y Edwin Giraldo

En la pantalla se muestra un previo de los datos adquiridos; en esta grafica previa pueden no mostrarse la totalidad de las muestras tomadas, pero los datos estarán todos disponibles en el campo de “Tabulación”. La grafica se verá solo en caso que allá seleccionado al menos una entrada análoga.

El botón de “Exportar” permite enviar los datos a Excel, y el botón de “Reiniciar” ignora los datos de la captura y regresa al inicio del programa para realizar una nueva captura.

8.6 Anexo F. Código programa en PICC

```
#include <18F4455.h>
#device adc = 10
#fuses NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,HSPLL,PLL5,CPUDIV3,VREGEN
/* USBDIV   Reloj del USB resulta de 96 MHz PLL dividido 2 (Para USB FULL Speed)
   HSPLL    Manejo del PLL a alta velocidad
   PLL5     Divide entre 5 con un cristal de 20 MHz (Prescaler)
   CPUDIV3  96 MHz PLL Dividido 4 para el reloj interno
*/
#use delay(clock = 24000000)
// El valor del clock depende de la division del PLL (96 MHz dividido 4)
#define USB_USE_FULL_SPEED FALSE // Velocidad de transferencia por USB (Hi/Lo)

// Definicion de puertos
#byte PORT_A = 3968
#byte PORT_B = 3969
#byte PORT_C = 3970
#byte PORT_D = 3971
#byte PORT_E = 3972

// Configuracion USB
#DEFINE USB_HID_DEVICE        FALSE
#define USB_EP1_TX_ENABLE     USB_ENABLE_BULK
#define USB_EP1_TX_SIZE       31
#define USB_EP1_RX_ENABLE     USB_ENABLE_BULK
#define USB_EP1_RX_SIZE       31

// Controladores USB
#include <pic18F4455_USB.c>
#include <descriptores_USB.h>
#include <usb.c>

// Definicion de Variables ============================================================

#BIT _STOP        = PORT_B.0
#BIT _LED_ROJO    = PORT_E.0
#BIT _LED_VERDE   = PORT_E.1
#define _CONFIG         255
#define _INICIO         240
#define _FIN            200
#define _DELAY          4
#define _PACK_SIZE      16

#define _APAGAR         0
#define _ROJO           1
#define _VERDE          2
#define _NARANJA        3
#define _TOGGLE         4
#define _ON             0
#define _OFF            1

#define _MAX_DIGITAL    16
#define _DELAY_DIGITAL  500

// Paquetes para envio y recepcion de datos
int8  paquete_0[_PACK_SIZE], paquete_1[_PACK_SIZE], paquete_2[_PACK_SIZE],
      paquete_in[_PACK_SIZE];
// Arreglo de valores análogos (ADC: 1 Byte por canal)
int16  analog[5];
// Arreglo de valores digitales
```

```
int8  digital[3*255];
// Contadores
int16  j, cont, cont2;
int8 i;
// Auxiliares
int16 ciclos1, ciclos2, max_digital, delay_digital;
int8  canales, estado_0, estado_1, estado_2, estado_3, ciclo, anterior,
      actual, flanco, delay;
int1  continuar;
int32 contaUSB;

// Funciones ======================================================================

// Envio de paquete ---------------------------------------------------------------
void envio(int paquete) {
   switch(paquete) {
      case 0:
         usb_put_packet(1, paquete_0, _PACK_SIZE, USB_DTS_TOGGLE);
         break;
      case 1:
         usb_put_packet(1, paquete_1, _PACK_SIZE, USB_DTS_TOGGLE);
         break;
      case 2:
         usb_put_packet(1, paquete_2, _PACK_SIZE, USB_DTS_TOGGLE);
         break;
   }
}

// Recepcion de datos -------------------------------------------------------------
void recepcion(void) {
   if (usb_kbhit(1)) {
      usb_get_packet(1, paquete_in, _PACK_SIZE);
   }
}

// Indicador de transmision -------------------------------------------------------
void indica(int color) {
   switch(color) {
      case _NARANJA:
         _LED_ROJO = _ON;
         _LED_VERDE = _ON;
         break;
      case _ROJO:
         _LED_ROJO = _ON;
         _LED_VERDE = _OFF;
         break;
      case _VERDE:
         _LED_ROJO = _OFF;
         _LED_VERDE = _ON;
         break;
      case _TOGGLE:
         _LED_ROJO = !_LED_ROJO;
         _LED_VERDE = !_LED_VERDE;
         break;
      case _APAGAR:
         _LED_ROJO = _OFF;
         _LED_VERDE = _OFF;
         break;
   }
}
```

```
// Main ============================================================
void main() {
   // Configuraciones ------------------------------------------------------
   // Configuracion de los puertos
   SET_TRIS_A(0b11111111);
   SET_TRIS_B(0b11111111);
   SET_TRIS_C(0b11111111);
   SET_TRIS_D(0b00000000);
   SET_TRIS_E(0b00000000);

   // inicializar puerto en ceros
   PORT_B = 0;
   PORT_D = 0;
   PORT_E = 1;

   // Desactivar WDT
   setup_wdt(WDT_OFF);

   // Configuracion modulo ADC
   setup_adc_ports(AN0_TO_AN2 | VSS_VDD);
   setup_adc(ADC_CLOCK_DIV_16 | ADC_TAD_MUL_20);

   // iniciar en cero
   set_rtcc(0);
   indica(_ROJO);

   do {
      // Inicializar USB
      usb_init();
      delay_ms(20);

      delay_ms(1000);
   } while (!usb_enumerated());

   // Habilitar interrupciones USB
   usb_task();

   // Ciclo principal (Infinito) -------------------------------------------
   while (true) {
      // Indicar inicio
      indica(_VERDE);
      delay_ms(1000);

      // Limpiar los datos de entrada y salida
      for (i = 0; i < _PACK_SIZE; i++){
         paquete_in[i] = 0;
         paquete_0[i] = 0;
         paquete_1[i] = 0;
         paquete_2[i] = 0;
      }

      // ID de paquetes
      paquete_1[0] = 1;
      paquete_2[0] = 2;

      // Loop configuracion 255 -----------------------------------------------
      indica(_VERDE);

      delay = _DELAY;
```

```
delay_digital = _DELAY_DIGITAL;
max_digital = _MAX_DIGITAL;

// Construccion de paquete 0 CONFIG
paquete_0[0] = _CONFIG;

while (paquete_in[0] != _CONFIG - 10) {
   // Loop hasta recibir configuracion
   while (paquete_in[0] != _CONFIG) {
      // Envio de paquete 0
      envio(0);
      // Recepcion de datos
      recepcion();
   }

   // Aplicar configuracion ----------------------------------------------
   indica(_NARANJA);
   canales = paquete_in[1];
   ciclos1 = make16(paquete_in[2], paquete_in[3]);
   ciclos2 = make16(paquete_in[4], paquete_in[5]);
   flanco = paquete_in[6];
   estado_0 = paquete_in[7];
   estado_1 = paquete_in[8];
   estado_2 = paquete_in[9];
   estado_3 = paquete_in[10];
   if (paquete_in[11] != 0) {
      delay = paquete_in[11];
   }
   if (paquete_in[12] != 0 && paquete_in[13] != 0) {
      delay_digital = make16(paquete_in[12], paquete_in[13]);
   }
   if (paquete_in[14] != 0) {
      max_digital = paquete_in[14];
   }

   // Confirmar configuracion
   paquete_0[0] = _CONFIG - 5;
   // Enviar los datos recibidos
   for (i = 1; i < _PACK_SIZE; i++){
      paquete_0[i] = paquete_in[i];
   }
   // Envio de paquete 0
   envio(0);
   // Recepcion de datos
   recepcion();

   indica(_TOGGLE);
}

indica(_VERDE);

// Estado 0: Inicial
PORT_D = estado_0;

// Construccion de paquete 0 INICIO
paquete_0[0] = _INICIO;

// Esperar Inicio -----------------------------------------------------
while (paquete_in[0] != _INICIO) {
   // Envio de paquete 0
```

```
   envio(0);
   // Recepcion de datos
   recepcion();
}

cont = 0;
cont2 = 0;
continuar = 1;
ciclo = 1;

// Estado 1: Ciclo 1
PORT_D = estado_1;

if (canales > 0) {
   // Inicio de toma de datos con canales analogos ------------------------
   while (!_STOP && continuar) {
      indica(_NARANJA);

      // Lectura de los canales Analogos
      for (i = canales; i > 0 ; i--) {
         set_adc_channel(i - 1);
         delay_us(10);
         analog[i - 1] = read_adc();
      }

      // Construccion de paquetes
      switch(canales) {
         case 5:
            paquete_1[13] = make8(analog[4],0);
            paquete_1[12] = make8(analog[4],1);
         case 4:
            paquete_1[11] = make8(analog[3],0);
            paquete_1[10] = make8(analog[3],1);
         case 3:
            paquete_1[9] = make8(analog[2],0);
            paquete_1[8] = make8(analog[2],1);
         case 2:
            paquete_1[7] = make8(analog[1],0);
            paquete_1[6] = make8(analog[1],1);
         case 1:
            paquete_1[5] = make8(analog[0],0);
            paquete_1[4] = make8(analog[0],1);
         case 0:
            paquete_1[3] = make8(cont,0);
            paquete_1[2] = make8(cont,1);
            paquete_1[1] = PORT_B;
      }
      // Envio de paquete 1
      envio(1);

      delay_ms(delay / 2);
      indica(_APAGAR);
      delay_ms(delay / 2);

      // Recepcion de datos
      recepcion();

      cont++;
      switch (ciclo) {
         case 1:
```

```
            if (ciclos1 != 0 && cont >= ciclos1) {
               ciclo++;
               // Estado 2: Ciclo 2
               PORT_D = estado_2;
            }
            break;
         case 2:
            if (ciclos1 != 0 && cont >= ciclos2) {
               continuar = 0;
            }
            break;
      }
   }  // siguiente ciclo
} else {
   // Inicio de toma de datos sin canales analogos ----------------------
   if (ciclos1 == 0) {
      ciclos1 = max_digital;
   }
   if (ciclos1 > max_digital) {
      ciclos1 = max_digital;
   }
   if (ciclos2 > max_digital) {
      ciclos2 = max_digital;
   }
   anterior = PORT_B;
   indica(_NARANJA);

   while (continuar) {
      delay_us(delay_digital);
      actual = PORT_B;

      if (actual != anterior) {
         // Construccion de paquetes
         // Estado Digitales
         digital[3*cont] = actual;
         // Conteo de ciclos de PIC
         digital[3*cont+1] = make8(cont2,1);
         digital[3*cont+2] = make8(cont2,0);

         cont++;
         switch (ciclo) {
            case 1:
               if (cont >= ciclos1) {
                  ciclo++;
                  // Estado 2: Ciclo 2
                  PORT_D = estado_2;
               }
               break;
            case 2:
               if (cont >= ciclos2) {
                  continuar = 0;
               }
               break;
         }
         if (anterior % 2 == 0 && actual % 2 == 1 && flanco) {
            continuar = 0;
         }
         if (anterior % 2 == 1 && actual % 2 == 0 && !flanco) {
            continuar = 0;
         }
```

```
            }

            anterior = actual;
            cont2++;

            indica(_TOGGLE);

         }  // siguiente ciclo

         // Estado 3: Final
         PORT_D = estado_3;
         indica(_NARANJA);

         for (j = 0; j < cont; j++) {
            // Estado Digitales
            paquete_2[1] = digital[3*j];
            // Conteo de paquetes enviados
            paquete_2[3] = j;
            // Conteo de ciclos de PIC
            paquete_2[4] = digital[3*j+1];
            paquete_2[5] = digital[3*j+2];

            // Envio de paquete 1
            envio(2);

            delay_ms(delay / 2);
            indica(_TOGGLE);
            delay_ms(delay / 2);

            // Recepcion de datos
            recepcion();

            for (i = 0; i < _PACK_SIZE; i++){
               if(paquete_in[i] != paquete_2[i]){
                  i--;
                  break;
               }
            }
         }
      }
      // Finalizar ----------------------------------------------------------------
      // Estado 3: Final
      PORT_D = estado_3;
      indica(_VERDE);

      // ID de finalizacion
      paquete_0[0] = _FIN;
      while (paquete_in[0] != _FIN) {
         // Envio de paquete 0
         envio(0);
         delay_ms(delay);

         // Recepcion de datos
         recepcion();
         indica(_TOGGLE);
      }
      indica(_APAGAR);
      delay_ms(1000);
   }
}
```

8.7 Protocolo de comunicación entre el sistema de adquisición y el software

Lab View PIC		Espera Paquete In paquete_0	Configuración Paquete Out paquete_in	Confirmación Paquete In paquete_0	Paquete Out paquete_in
ID	0	255 <PIC>	255 <LV>	250 <PIC>	245 <LV>
	1	-	An	An	-
	2	-	Ciclos1H	Ciclos1H	-
	3	-	Ciclos1L	Ciclos1L	-
	4	-	Ciclos2H	Ciclos2H	-
	5	-	Ciclos2L	Ciclos2L	-
	6	-	Flanco_Final	Flanco_Final	-
	7	-	E0	E0	-
	8	-	E1	E1	-
	9	-	E2	E2	-
	10	-	E3	E3	-
	11	-	Delay	Delay	-
	12	-	Delay_DigitalH	Delay_DigitalH	-
	13	-	Delay_DigitalL	Delay_DigitalL	-
	14	-	Max_Digital	Max_Digital	-
	15	-	-	-	-

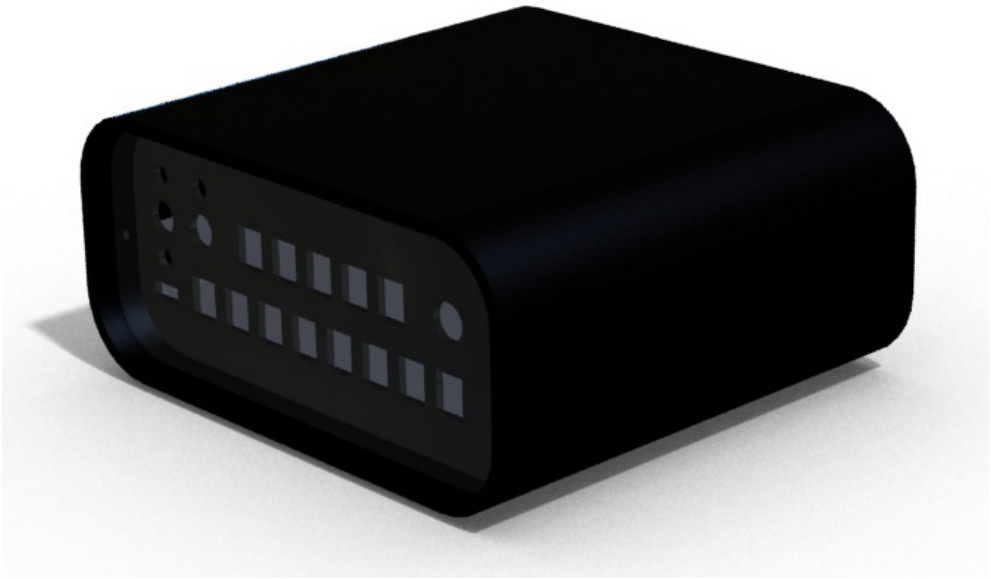
Lab View PIC		Espera Configuracion paquete_0	Inicio Configuracion paquete_0	Lectura An Paquete In paquete_1
ID	0	240 <PIC>	240 <LV>	1 <PIC>
	1	-	-	Digital
	2	-	-	ContadorH
	3	-	-	ContadorL
	4	-	-	A0H
	5	-	-	A0L
	6	-	-	A1H
	7	-	-	A1L
	8	-	-	A2H
	9	-	-	A2L
	10	-	-	A3
	11	-	-	A3L
	12	-	-	A4H
	13	-	-	A4L
	14	-	-	-
	15	-	-	-

Lab View PIC		Lectura Di Paquete In paquete_2	Confirmación Paquete In paquete_in	Final Paquete In paquete_0
ID	0	2 <PIC>	2 <LV>	0 200 <PIC>
	1	Digital	Digital	1 -
	2	-	-	2 -
	3	Contador	Contador	3 -
	4	TimerH	TimerH	4 -
	5	TimerL	TimerL	5 -
	6	-	-	6 -
	7	-	-	7 -
	8	-	-	8 -
	9	-	-	9 -
	10	-	-	10 -
	11	-	-	11 -
	12	-	-	12 -
	13	-	-	13 -
	14	-	-	14 -
	15	-	-	15 -

	General	Condensador	Rotacional
0	Repeticiones		
1	Análogos		
2	Ciclos1		
3	Ciclos2		
4	Delay		
5	Delay_Digital		
6	Mínimo		
7	Máximo		
8	Muestras		
9	Tiempo Inicial		
10	Tiempo Final		
11	Transcurrido		
12	<Interno>	-	R M
13	<Interno>	-	C m
14	<Interno>	-	V R
15	<Interno>	-	- r
16	<Interno>	-	- y
17	<Interno>	-	- -

8.8 Anexo G. Planos de diseño de la carcasa de la Interfaz

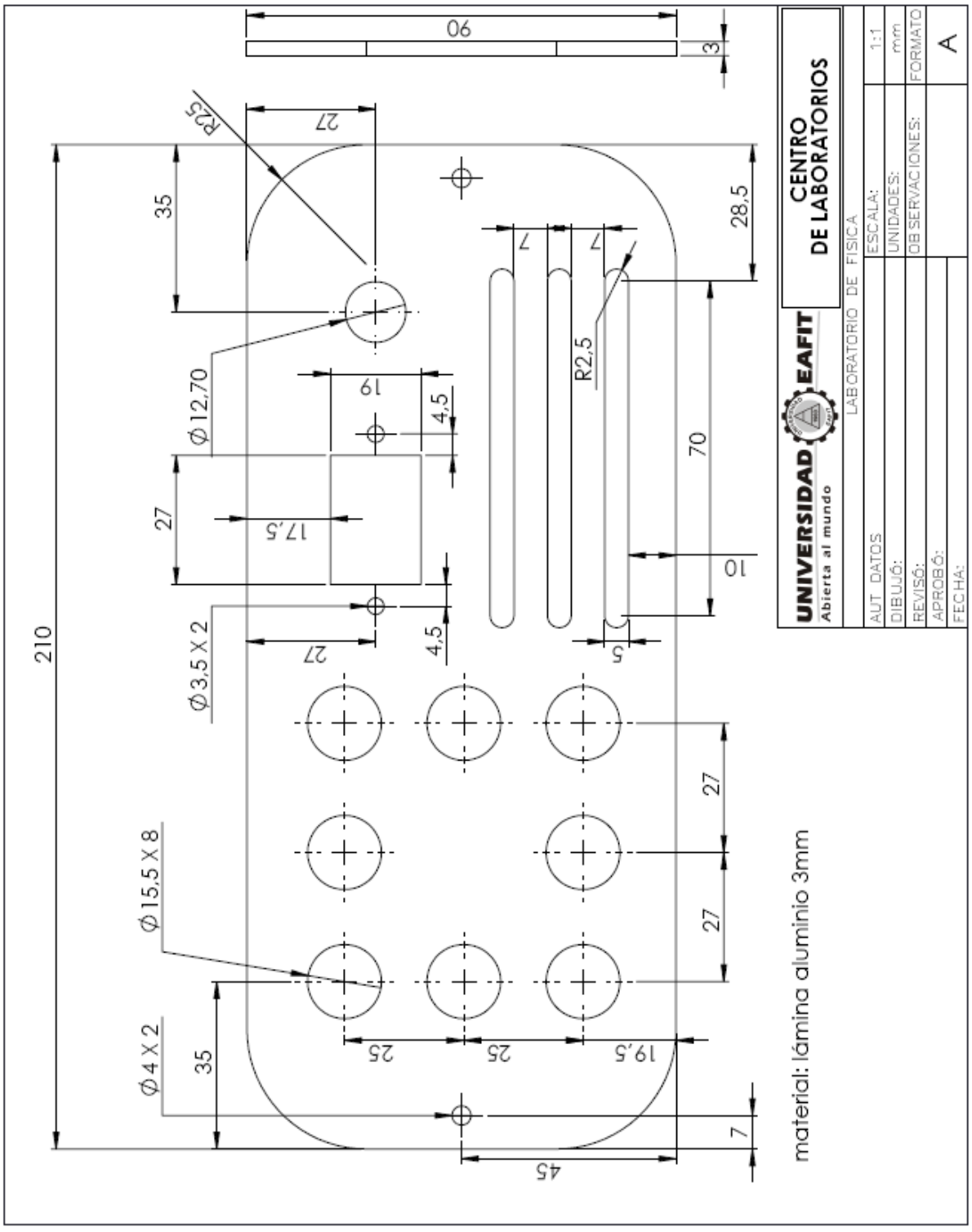
Figura 46: Vista Frontal de la carcasa

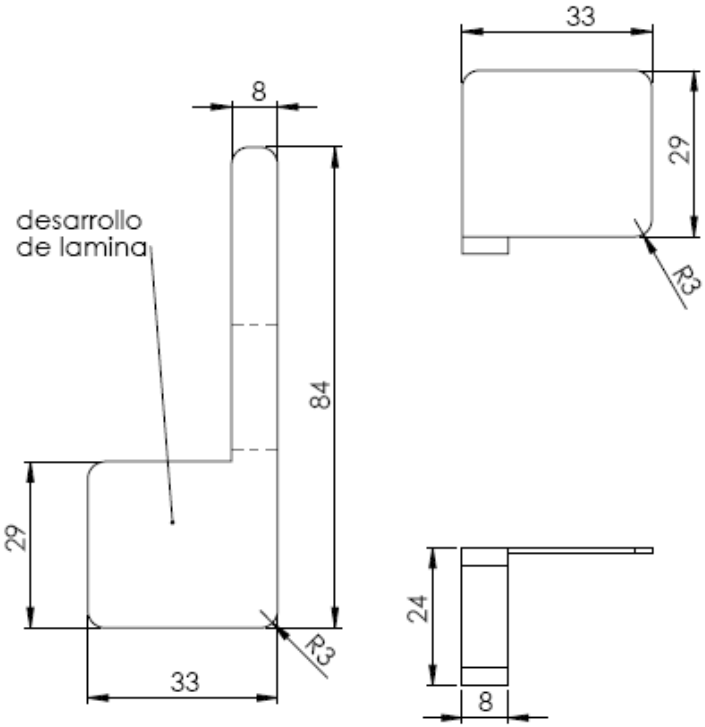


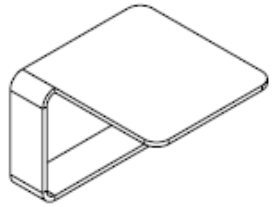
Fuente Daniel Giraldo

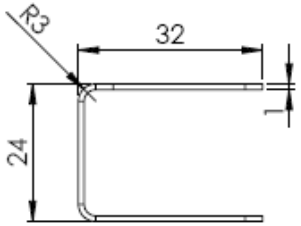
Figura 47: Vista Posterior de la carcasa

Fuente Daniel Giraldo


90
3
R23
27
35
28,5
7
7
R2,5
70
10
5
Ø12,70
19
4,5
27
17,5
Ø3,5 X 2
27
4,5
Ø15,5 X 8
27
27
Ø4 X 2
35
25
25
19,5
45
7
210
material: lámina aluminio 3mm
CENTRO
DE LABORATORIOS
UNIVERSIDAD EAFIT
Abierta al mundo
LABORATORIO DE FISICA
AUT DATOS
DIBUJÓ:
REVISÓ:
APROBÓ:
FECHA:
ESCALA: 1:1
UNIDADES: mm
OBSERVACIONES:
FORMATO A

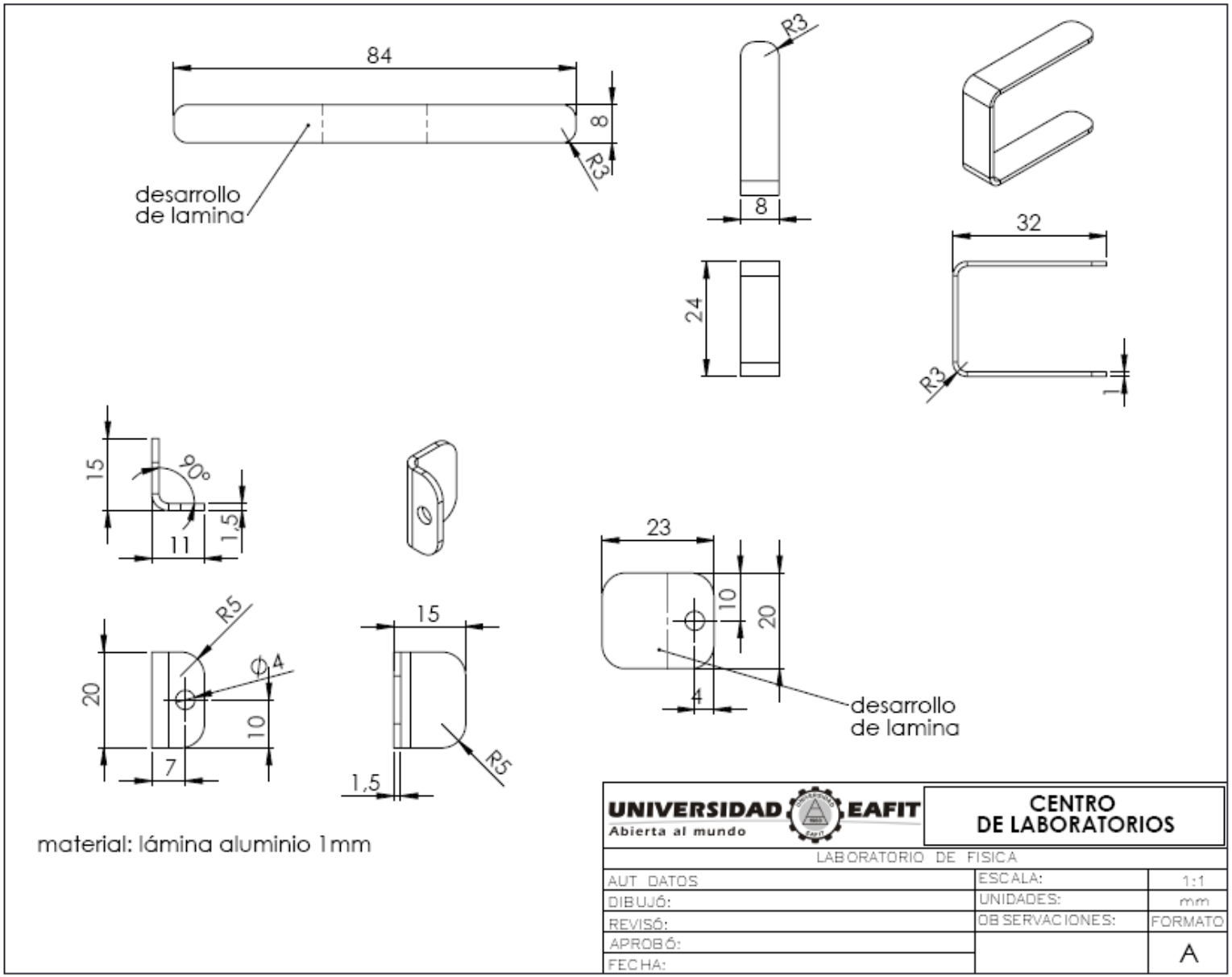


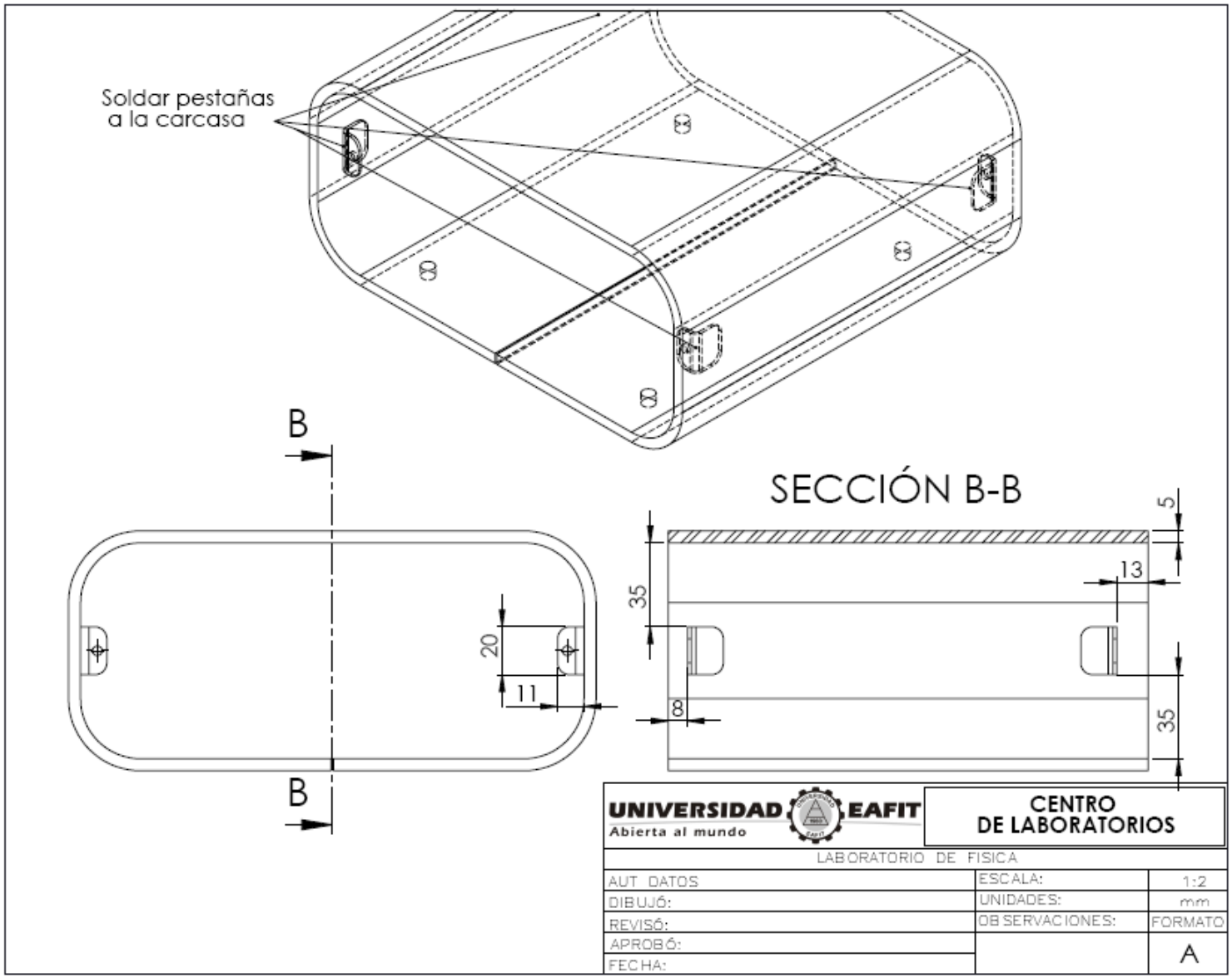




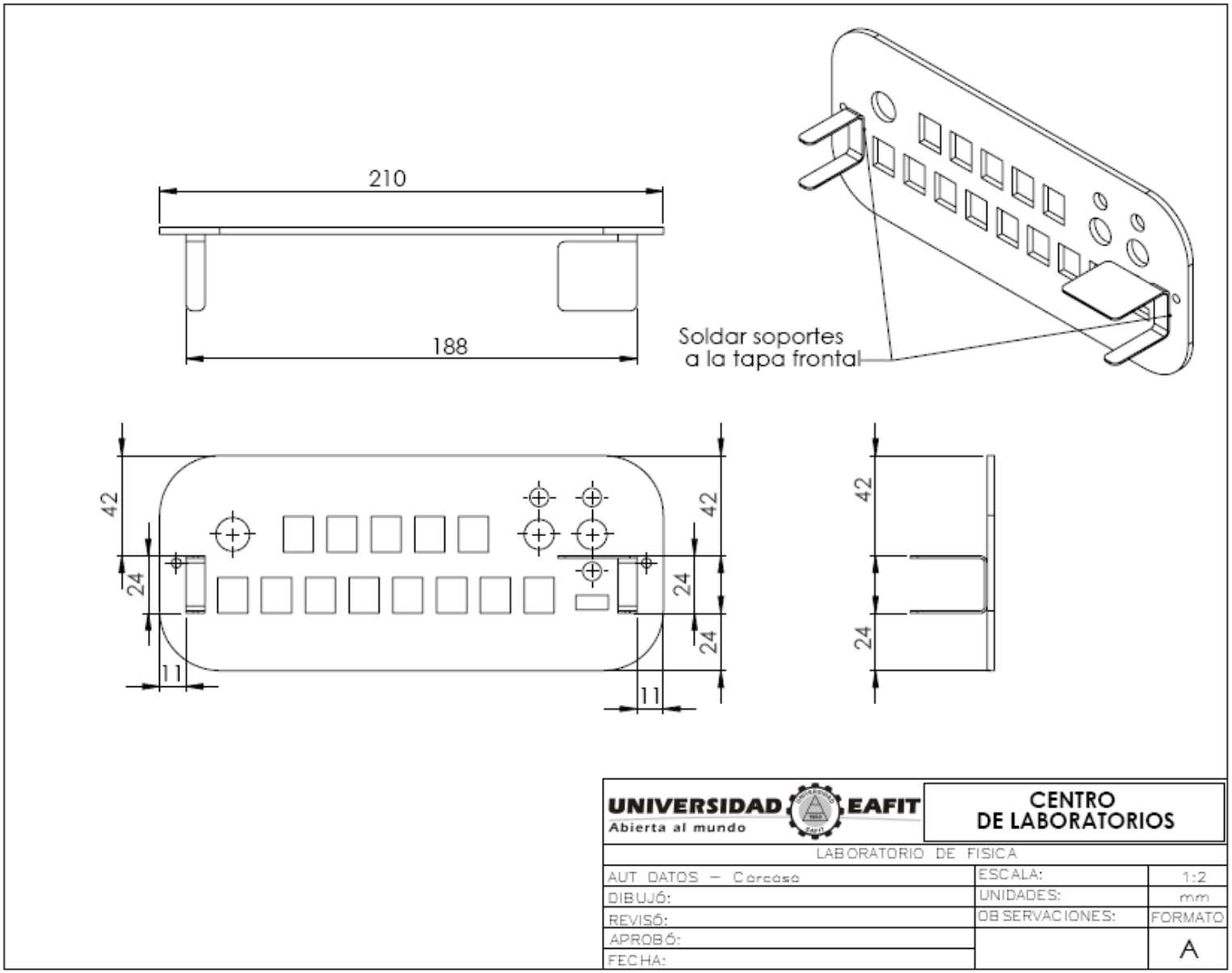
material: lámina aluminio 1mm

UNIVERSIDAD EAFIT Abierta al mundo	CENTRO DE LABORATORIOS	
LABORATORIO DE FISICA		
AUT DATOS	ESCALA:	1:1
DIBUJÓ:	UNIDADES:	mm
REVISÓ:	OBSERVACIONES:	FORMATO
APROBÓ:		A
FECHA:		


84
8
R3
desarrollo
de lamina
R3
8
24
32
R3
1
15
90°
11
1,5
R5
20
Ø4
10
7
15
1,5
R5
23
10
20
4
desarrollo
de lamina
material: lámina aluminio 1mm
UNIVERSIDAD EAFIT
Abierta al mundo
CENTRO
DE LABORATORIOS
LABORATORIO DE FISICA
AUT DATOS
DIBUJÓ:
REVISÓ:
APROBÓ:
FECHA:
ESCALA:
1:1
UNIDADES:
mm
OBSERVACIONES:
FORMATO
A



UNIVERSIDAD EAFIT Abierta al mundo	CENTRO DE LABORATORIOS	
LABORATORIO DE FISICA		
AUT DATOS	ESCALA:	1:2
DIBUJÓ:	UNIDADES:	mm
REVISÓ:	OBSERVACIONES:	FORMATO
APROBÓ:		A
FECHA:		



UNIVERSIDAD EAFIT Abierta al mundo	CENTRO DE LABORATORIOS	
LABORATORIO DE FISICA		
AUT DATOS – Carcasa	ESCALA:	1:2
DIBUJÓ:	UNIDADES:	mm
REVISÓ:	OBSERVACIONES:	FORMATO
APROBÓ:		A
FECHA:		