# UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

TEMA:

**Sistema Modular para realizar prácticas de servomecanismos y control mediante deslizador lineal e interfaz con LabVIEW**

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO INGENIERO EN MECATRÓNICA

AUTOR: Ernesto Vladimir Palacios Merino

DIRECTOR: Ing. Diego Terán

Ibarra - Ecuador

2013

# DECLARACIÓN

Yo, Ernesto Vladimir Palacios Merino, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Técnica del Norte, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ernesto Palacios

# CERTIFICACIÓN

Certifico que el presente trabajo fue elaborado por Ernesto Palacios bajo mi supervisión.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ing. Diego Terán

DIRECTOR DEL TRABAJO

# AGRADECIMIENTOS

Le agradezco inmensamente a mi Yumicita por aguantarme siempre y darme ánimos cuando lo necesitaba ;)

# DEDICATORIA

# CONTENIDO

# 

# 

# RESUMEN

El presente trabajo de grado describe el diseño e implementación de un sistema modular para realizar prácticas de servomecanismos y control, este sistema permitirá a los estudiantes de la Facultad de Ingeniería en Ciencias Aplicadas tener mayor flexibilidad a la hora de realizar sus proyectos, ya que contarán con una interfaz que les permita controlar uno de los servomotores con los que cuenta el laboratorio de mecatrónica. Este sistema, al ser modular, permite diferentes medios de comunicación y diferentes tipos de control, con lo cual el estudiante tiene muchas más opciones al momento de realizar sus prácticas.

También se encuentra implementado un deslizador lineal, el cual puede ser utilizado por el estudiante, o si lo prefiere puede trabajar con el servomotor directamente, inclusive implementar otro tipo de mecanismo al servomotor pudiendo controlarlo utilizando el sistema micro controlado que en este trabajo se implementa.

Además del bloque electrónico, que permite controlar el servomotor, se ha implementado librerías de comunicación y control con el entorno de desarrollo LabVIEW[[1]](#footnote-1), esto permite que la comunicación con el controlador del motor se realice de una manera muy fácil, y aún más, estas librerías cuentan con tres niveles de abstracción, liberando al estudiante de los detalles de implementación del sistema, y de esta manera permitiéndole centrarse en el desarrollo de su aplicación.

Esto no quiere decir que el estudiante esté cerrado a utilizar únicamente este entorno de desarrollo, el sistema puede ser implementado en cualquier lenguaje de programación que permita comunicarse con un puerto serial, algunos de estos programas incluyen: Matlab, python, C/C++, java, entre otros. Con lo cual la flexibilidad a la hora de configurar y utilizar el sistema es muy grande, y las aplicaciones que se pueden realizar son muy variadas.

Esto es importante a la hora de realizar prácticas de control ya que se pueden utilizar diferentes tipos de programación, el control puede ser implementado utilizando LabVIEW, Matlab o un microcontrolador, y ser desplegado en el deslizador de una manera rápida. Es posible modificar el código del controlador y volverlo a ejecutar de una manera rápida, si el servomotor estuviera controlado por un PLC se necesitaría volver a programar el dispositivo con cada cambio del programa, en cambio al utilizar este sistema eso no es necesario ya que el control del sistema lo realiza el programa del estudiante directamente.

# ABSTRACT

This paper describes the design and implementation of a modular system for servo-mechanism and control projects, this system will allow students of the Engineering Faculty for Applied Science to have greater flexibility to carry out their projects, because they will have an interface that enables them to control the servo motors in the laboratory of mechatronics, and this system is modular, allowing different communication protocols and different types of control, which the student has many more options when making their practices.

Is also implements a linear slider, which can be used by the student, or they can work with the servomotor directly, or even implement other mechanism and can still control the servo motor using microcontroller system that is implemented in this work.

Besides the electronic block, which controls the actuator, it has been implemented libraries for communication and control with the LabVIEW development environment, this means communication with the motor controller is done very easily, moreover, these libraries have come in three levels of abstraction, freeing the student from the implementation details of the system, thus allowing you to focus on developing your application.

This does not mean that the student is locked to only use this development environment, the system can be implemented in any programming language that can communicate with a serial port, and some of these programs include: Matlab, Python, C/C++ and java, among others. Whereupon the flexibility to configure and use the system is very large, and applications that can be performed vary widely.

This is important when making control practices that can use different types of programming, the control can be implemented using LabVIEW, Matlab or a dedicated microcontroller, and be deployed quickly on the slider, you can modify the code driver and re-run it, if the actuator was controlled by a PLC it would be required to reprogram the device with each program change, however when using this system it is not necessary because the system control is performed by the student's program directly on the computer.

# PRESENTACIÓN

La carrera de ingeniería en mecatrónica es muy variada en sus campos de estudio, en ella se abarcan diferentes áreas de ingeniería, por lo tanto es importante que al realizar prácticas en los laboratorios, se cuente con el equipamiento necesario para el estudio de estas áreas.

La adición de un deslizador lineal servirá a este propósito brindando al estudiante la capacidad de realizar diversas aplicaciones utilizando el sistema modular que se ha implementado en el presente trabajo de grado. Las aplicaciones que puede realizar son muy variadas, ya que la modularidad del sistema permite utilizar sus distintas partes de forma aislada, o también, aumentar las capacidades del sistema aumentando sus módulos.

Todo se ha diseñado tomando en cuenta la participación del estudiante al realizar sus aplicaciones. La programación, la comunicación, los sensores, el tipo de control sobre el motor, todo es susceptible de personalización adecuándose a las necesidades y al nivel de complejidad del proyecto.

Para poder lograr lo aquí expuesto se ha aplicado todos los conocimientos adquiridos a lo largo de la carrera en mecatrónica, el diseño de las librerías, el sistema de control y el diseño del deslizador, todo esto y más es el testimonio de la culminación de años de estudio y dedicación.

# 

# 

CAPITULO I. ANTECEDENTES

1.1 INTRODUCCIÓN

Lo que este trabajo busca es ampliar las funcionalidades del laboratorio de mecatrónica, para que sus elementos puedan ser utilizados de mejor manera por un amplio grupo de estudiantes, permitiéndoles interactuar con diversos elementos, y brindarles las herramientas necesarias para poder desplegar sus proyectos de una manera rápida, y desarrollar sus aplicaciones utilizando ideas y conceptos con los que ya están familiarizados.

El deslizador lineal es de un diseño muy simple, pero a pesar de la sencillez de su diseño brinda la oportunidad a los estudiantes de estudiar la respuesta de un sistema servo controlado y analizar el comportamiento del sistema bajo distintos tipos de cargas mecánicas, estudiar la inercia producida por la carga y modelar un sistema de compensación, todo esto dentro de un mismo proyecto, desde la programación básica del deslizador, hasta la implementación de un sistema de control, integrado en un sólo sistema modular que permite un estudio completo y no como partes separadas.

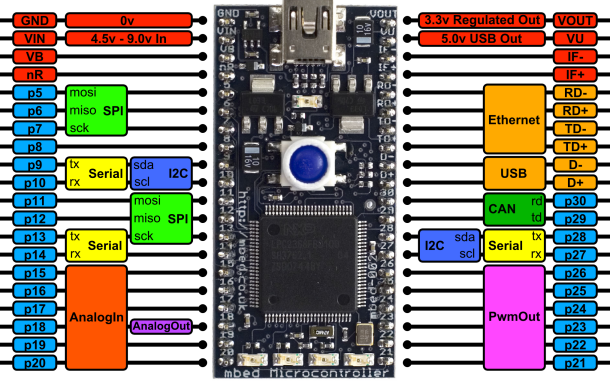
La programación actual del microcontrolador permite la utilización de todos sus puertos e interfaces de manera dinámica a través de la implementación del protocolo RPC[[2]](#footnote-2) sobre Ethernet. Esto posibilita que se le pueda dar muchas aplicaciones más allá del alcance que este trabajo de grado presenta, todo esto sin necesidad de reprogramar el microcontrolador, directamente desde las librerías de LabVIEW que se ofrecen junto con el microcontrolador en su sitio web[[3]](#footnote-3). Esto convierte al microcontrolador mbed en una muy poderosa tarjeta de adquisición en LabVIEW, sin necesidad de hardware adicional.

1.2 EL MICROCONTROLADOR ARM MBED

El microcontrolador es la parte más importante del sistema, posee las características necesarias para su implementación, y es el elemento que provee de modularidad y flexibilidad al momento de controlar el deslizador, por tal motivo se ha escogido el microcontrolador ARM-mbed el cual reúne todas las prestaciones que el sistema necesita.

Sin las herramientas adecuadas, los detalles de la implementación pueden rápidamente entorpecer el desarrollo de prototipos y la experimentación. El microcontrolador mbed resuelve este problema ya que cubre las bases sobre las cuales desarrollar diversos proyectos.

Fig. 1 - Esquema del microcontrolador ARM-mbed



Está basado en el chip NXP LPC1768, con un núcleo ARM Cortex-M3 de 32 bits corriendo a 96 MHz. Incluye 512KB de memoria FLASH, 32KB RAM y muchas interfaces incluyendo Ethernet incorporado, USB Huésped y dispositivo, CAN, SPI, I2C, ADC, DAC, PWM, y otras interfaces de E/S. El diagrama arriba muestra las interfaces más comunes y sus lugares. Note que todos los pines numerados (p6, p30) pueden también ser usados como interfaces de Entrada y Salida digital

Cuadro 1 – Especificaciones microcontrolador ARM mbed

Fuente: mbed.org

|  |  |
| --- | --- |
| Núcleo: | ARM Cortex-M3 LPC1768 |
| Frecuencia: | 96 MHz |
| Memoria Flash | 512 KB |
| Memoria RAM | 32KB |
| Consumo energético | 60-120 mA (Vin) |
| Conexión Ethernet | 1 |
| USB Host | 1 |
| USB Device | 1 |
| Módulo SPI | 2 |
| Módulo I2C | 2 |
| Módulo CAN | 1 |
| Entradas Analógicas | 6 |
| Salidas de PWM | 6 |
| Salidas Analógicas | 1 |

El microcontrolador ARM mbed está diseñado para desarrollar proyectos de una manera rápida, flexible de bajo costo y de una manera profesional. Para ello cuenta con un el microprocesador ARM Cortex-M3 antes mencionado montado sobre una placa de cuarenta pines, la cual se puede montar en una protoboard.

Se puede alimentar externamente aplicando un voltaje entre 4.5v a 9.0v a la entrada VIN o directamente utilizando el conector mini USB incorporado.

También se tiene soporte para un puerto serial virtual usando la misma conexión USB, permitiendo comunicación serial con un computador ya sea en una consola serial como Hiperterminal[[4]](#footnote-4), LabVIEW, Matlab, o cualquier otro tipo de lenguaje de programación que pueda comunicarse con un puerto COM.

Existen dos formas para realizar, una cuando se encuentre conectado a internet, usando el compilador en línea, o caso contrario se puede utilizar cualquiera de los siguientes compiladores fuera de línea: Keil uVision, Code Red, o GCC-ARM.

El resultado final de cualquiera de estos compiladores es un archivo en código binario (.bin) que debe ser programado al mbed. Cuando se conecta el ARM-mbed a un computador este aparece en el sistema como una unidad de almacenamiento masivo de 2MB, para programar el microcontrolador se debe copiar el archivo .bin dentro de esta memoria y resetear el dispositivo. Se puede tener varios programas (archivos .bin) cargados en la memoria, se ejecutará el último archivo en ser añadido y se programará el dispositivo con este archivo.

Para el deslizador se ha aprovechado de las capacidades de comunicación serial y ethernet, así como la interfaz I2C y DAC para comunicarse con el encoder y el control de velocidad respectivamente,

1.2.1 INTERFAZ MBED CON RPC

Hay muchas ocasiones en que es útil ser capaz de comunicarse con mbed desde un ordenador. Esto podría ser para controlar actuadores que utilizan las salidas de mbed, recopilar datos utilizando sensores o crear aplicaciones remotas a través de una red. Crear una interfaz entre mbed y un ordenador puede ser difícil ya que requiere que especifique un formato de comunicación y luego escribir el código tanto en mbed y en el equipo con el cual se desea conectar.

El microcontrolador mbed es capaz de recibir e interpretar comandos RPC esto puede ser usado para simplificar en gran medida la creación de un interfaz. Los comandos RPC se encuentran en un formato predefinido y se pueden enviar a través de cualquier mecanismo de transporte que pueda enviar una secuencia de texto. Ellos le permiten interactuar directamente con los objetos de mbed.

También hemos creado bibliotecas de varias lenguas populares que le permiten utilizar RPC a través de varios mecanismos de transporte sin necesidad de tener que hacer ningún trabajo para establecer el transporte mecanismo o el formato de sus mensajes. Las bibliotecas se han desarrollado para: MATLAB, LabVIEW, Python, Java y .NET.[[5]](#footnote-5)

El microcontrolador es capaz de levantar un servidor HTTP de mbed tiene un controlador RPC, así que al usar la librería HTTP Server se puede usar RPC sobre HTTP. Los comandos RPC son enviados al ir agregándolos al URL de mbed en un navegador.

http://<url de mbed>/rpc/<Nombre del Objeto>/<Nombre del método> <Argumentos separados por espacios>

El microcontrolador mbed enviará su dirección URL sobre serial USB.

LabVIEW RPC le permite controlar directamente a mbed mediante el protocolo RPC. Se ha creado algunos bloques de construcción de la interfaz RPC de mbed con LabVIEW, permitiendo que los programas de LabVIEW puedan interactuar con el mundo real. Esto podría ser usado para cosas como:

* + Adquisición de datos en LabVIEW a través de sensores conectados a mbed
  + Actuadores de control conectados a mbed desde LabVIEW
  + Programas de LabVIEW con hardware en el bucle, donde los sensores y actuadores se interconectan con mbed pero los cálculos y el control son en LabVIEW

La librería ‘RPC-Interface’ proporciona un mecanismo para agregar rápidamente la funcionalidad RPC a su propio código. Esta librería de LabVIEW incluye soporte para ‘RPCFunction’ y objetos ‘RPCVariable’. Usted puede atar una función ‘RPCFunction’ o una variable ‘RPCVariable’ al objeto correspondiente en mbed. Luego se puede ejecutar la función o leer y escribir a las variables que se adjuntan.

Utilizando estas herramientas se ha creardo una API para el sistema modular, controlando el deslizador desde LabVIEW mediante una red local.

## 

## 1.3 EL SERVOMOTOR

Para la implementación del sistema se utiliza un servomotor de corriente alterna, estos motores combinan la potencia de un motor de corriente alterna, con la precisión de un motor de pasos. Lo que nos permite tener una muy alta precisión y velocidades de rotación bajas manteniendo el torque del motor.

Un Servomotor podría definirse genéricamente como un motor utilizado para obtener una salida precisa y exacta en función del tiempo. Dicha salida está expresada habitualmente en términos de posición, velocidad y/o torque.

La aplicación industrial de dichos motores se está desarrollando significativamente por múltiples razones entre las que podemos mencionar: nuevos y más potentes componentes magnéticos para los motores como los imanes de tierras raras, reducción de costo de los motores y los equipos electrónicos necesarios para el control de los mismos, incorporación en dichos equipos electrónicos de nuevas funciones para un control preciso y confiable del movimiento que permiten utilizarlos eficientemente e incorporar nuevas áreas a su dominio de aplicación.

Esencialmente, un motor sin escobillas a imán permanente es una máquina sincrónica con la frecuencia de alimentación, capaz de desarrollar altos torques (hasta 3 o 4 veces su torque nominal) en forma transitoria para oponerse a todo esfuerzo que trate de sacarla de sincronismo. La denominación sin escobillas es una forma de diferenciarlo de sus predecesores, los servomotores a imán permanente alimentados con corriente continua.

En comparación con motores asíncronos de jaula de ardillas (que eroguen el mismo torque/velocidad en su eje) la inercia de un servomotor sin escobillas es sustancialmente menor. Ambas características: sobre torques importantes e inercias reducidas son características apreciadas y útiles para el control del movimiento pues permiten rápidas aceleraciones y deceleraciones así como control preciso de posición en altas velocidades.

Constructivamente el servomotor sin escobillas posee un estator parecido al de un motor de jaula con un núcleo laminado y un bobinado trifásico uniformemente distribuido. El rotor está constituido por un grupo de imanes permanentes fijados en el eje de rotación. La forma de los rotores a imanes varía de acuerdo al diseño y puede clasificarse en cilíndricos o de polos salientes.

La fijación de los imanes al rotor ha sido uno de los puntos críticos en la construcción de estos motores debido a las altas fuerzas centrífugas a las que se encuentran sometidos durante los procesos de aceleración y frenado. Actualmente se combinan fijaciones mecánicas de diferentes tipos (atadura con fibra de vidrio, chaveteado con diferentes materiales, etc.) con pegado utilizando adhesivos especiales.

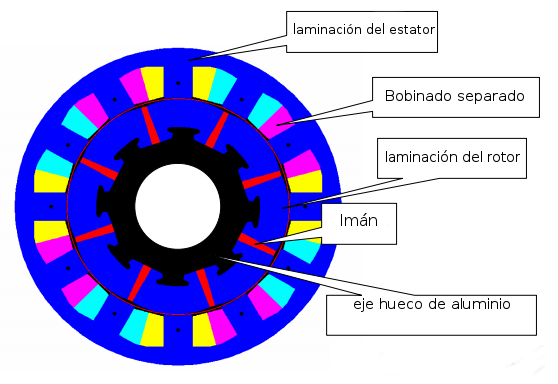


Fig. 2 - Esquema del Servomotor.

Fuente: Autor

Cuando circula corriente alterna en las fases del bobinado de estator se produce un campo magnético rotante en el entrehierro del motor. Si en cada instante el campo magnético generado en el estator intersecta con el ángulo correcto al campo magnético producido por los imanes del rotor generamos torque para lograr el movimiento del motor y la carga acoplada a él.

La utilización de un dispositivo electrónico denominado *servodrive* o driver para el servomotor para alimentar el estator con la tensión y frecuencia correcta, permite en cada instante, generar un campo magnético en el estator de magnitud y posición correctamente alineada con el campo magnético del rotor. De esta forma obtenemos el torque necesario para mantener la velocidad y posición deseada del eje del motor.

El proceso implica conocer en todo instante la posición del rotor para lo cual se equipan los servomotores con dispositivos tales como *resolvers*, *encoders* u otros. Los mismos rotan solidariamente con el eje del servomotor e informan al servodrive la posición del rotor. Dichos dispositivos de realimentación de posición se diferencian en la robustez, resolución, capacidad de retener la información de posición ante cortes de alimentación y número de conexiones necesarias entre otras. Por ejemplo en una servo-máquina de tracción directa que rota normalmente a una velocidad nominal de algunas centenas de RPM deberemos seleccionar dispositivos con un alto número de pulsos por revolución a fin de tener control de torque durante la partida y parada del ascensor.

Actualmente los servodrives operan por técnicas de modulación de ancho de pulso (PWM) con configuraciones de hardware (básicamente en la parte de potencia) parecidas a los inversores para el control de motores asincrónicos. De hecho existen en el mercado drivers que permiten controlar ambos tipos de motores.

Debe puntualizar que para la operación normal de un servomotor necesitamos un servodrive, el motor no puede ser operado directamente de la red de suministro.

La selección de un servomotor para una determinada aplicación requiere conocer el torque de pico necesario para acelerar y frenar la carga impulsada por el motor así como el torque eficaz requerido por la aplicación. Básicamente el conjunto servodrive - servomotor deben estar en condiciones de satisfacer los requerimientos de torque de pico solicitados por el sistema y el motor debe soportar sin deterioro el régimen térmico impuesto por manejar el torque eficaz requerido por la aplicación.

La utilización de servomotores se está popularizando en todas las ramas de la industria. En el transporte vertical vemos cada vez más frecuentemente aplicaciones que aprovechan la alta capacidad de sobre torque y la baja inercia del motor para lograr un perfecto control del viaje y nivelación aun en muy altas velocidades en máquinas de tracción o posicionamientos perfectos con alto control del torque en operadores de puerta.

La alta capacidad volumétrica de torque del motor permite obtener máquinas de dimensiones reducidas en comparación con las máquinas de otras tecnologías facilitando la instalación. Los servodrives incorporan mayor capacidad de control de movimiento y comunicación realizando en algunos casos funciones que eran anteriormente dominio exclusivo de la maniobra.

Los conjuntos son más eficientes desde el punto de vista rendimiento y consumen menos energía que algunas aplicaciones tradicionales. Por lo tanto es de esperar en un futuro cercano una mayor difusión de este tipo de soluciones acompañada por una baja de su costo, producto de la mayor cantidad de unidades manufacturadas y número de proveedores presentes en el mercado.

1.3.1 EL SERVOMOTOR 110SJT

El servomotor utilizado es el GSK 110SJT el cual está controlado por el driver GSK DA98D. Las características de este servomotor son las siguientes:

Cuadro 2 – Especificaciones Servomotor GSK 110SJT

|  |
| --- |
| Un = 220V |
| In = 8A |
| Tn = 2 N.m |
| Nn: 3000 RPM |
| Nmax = 3300 RPM |
| INS: CLASS B M: 2500 r/rev |

Fuente: Manual Driver GSK

La unidad DA98D AC servodrive es una nueva generación de la plena unidad de disco digital de AC servo producido por GSK. Este producto incluye dos modos de control de velocidad y posición. Se pueden combinar con diversos sistemas de control en lazo abierto y lazo cerrado y ha sido ampliamente aplicado a máquinas herramientas CNC y la industria de automatización.

Las conexiones eléctricas entre el servomotor y el driver están realizadas y listas en los laboratorios, por lo tanto es necesario solamente realizar la conexión entre el driver y el elemento de control. Este elemento de control tradicionalmente ha sido el PCL Siemens S7-200, pero ahora también existe la posibilidad de controlar el servomotor a través del microcontrolador ARM-mbed.

1.4 PROTOCOLO DE COMUNICACIÓN ETHERNET

En los años 40, la instrumentación de campo todavía se apoyaba en señales de presión para la monitorización de los procesos. En los 60 se introdujo la señal estándar 4-20 mA en las aplicaciones de instrumentación. A pesar de su éxito, señales de diferentes niveles se utilizaban en dispositivos no adecuados al estándar, defendidos por unos u otros fabricantes. El primer autómata programable aparece en 1969. A mediados de los 70, la empresa Honeywell anuncia el primer sistema de control de procesos distribuido (DCCS). En los años 80 aparecieron los sensores inteligentes basados en microprocesador, esto potenció la aparición de los buses de campo que comunicaran los distintos dispositivos de la instalación entre sí.

Desde entonces, tal como ocurrió con la señalización analógica, se realizaron grandes esfuerzos en el control de procesos para unificar tanto las comunicaciones entre dispositivos como los perfiles a los que estos debían responder para garantizar el comportamiento estandarizado. Los bocetos del estándar propuesto por el comité IEC/ISA SP50 se centraron en definir las siguientes funciones:

• Capa física. Especifica el medio de transmisión, sería el sustituto digital de la señal 4-20 mA en el entorno de proceso.

• Capa de enlace. Especifica las comunicaciones entre dispositivos de un mismo bus, el método de acceso a éste y chequea posibles errores.

• Capa de aplicación. Encargada de dar formato de mensaje a los datos, de forma que sean entendibles por el dispositivo receptor y emisor. También ofrece servicios a la capa de usuario.

• Capa de usuario. Ofrece a las aplicaciones finales funciones específicas de control e identificación automática de dispositivos.

Sin embargo, ante el retraso en la salida del estándar, cada fabricante abogó de nuevo por implementaciones propietarias. Es el momento de ISP (Interoperable Systems Project) y WorldFIP, que dieron lugar a la actual Fielbus Foundation, o de la Profibus User Organization (PNO). ModBus aparece en 1979; Interbus-S en 1984 y CAN (especifica capas 1 y 2) en 1986. FieldBus Foundation especificó el bus H1 en 1996; un año antes, PNO especifica Profibus PA. AS-Interface (1993) surge como bus especializado en señales todo-nada y posteriormente intentará mejorar sus prestaciones en transmisión de datos analógicos. DeviceNET aparece en 1994. Dentro del campo de la automatización de edificios aparecen BatiBUS, EIB (1990), LonWorks (1991) o BACNet (1995).

Así surgieron multitud de soluciones de comunicación industrial. Tanto el cliente final como integradores y fabricantes debían apostar por una solución u otra sin tener demasiado claras las perspectivas de futuro de dicha solución. La apuesta por cualquiera de ellas suponía además formar a personal especializado para su instalación, puesta en marcha y mantenimiento. Tiempos difíciles para el diseño de redes de control en las que, por ejemplo, la formación previa del personal propio condicionaba drásticamente las soluciones ofrecidas para los siguientes proyectos.

El estándar Ethernet a 10 Mbps es publicado por el IEEE (802.3) en 1985 y rápidamente conquistó el terreno de las comunicaciones de área local en el entorno ofimático. En 1993 aparecen los primeros conmutadores Full Duplex y Fast Ethernet (100 Mbps) se estandariza en 1995. Una vez asentada esa prevalencia en el sector no industrial, el estándar Ethernet empezó a ser visto como una posible solución a la falta de unificación práctica en las capas física y de enlace de los equipos de bus de campo provenientes de diferentes fabricantes. Los estándares de calidad de servicio en capa 2 no aparecen hasta 1997.

La implantación de Ethernet como soporte para los protocolos de nivel superior era clara a nivel de empresarial (nivel ERP en la estructura de producción) y rápidamente bajó al nivel de Sala de Información (niveles MES y SCADA). El salto al nivel de Control (comunicación entre DCSs, autómatas y sistemas HMI locales) se convirtió en una realidad a medida que la electrónica de red se implementó en las unidades de control de proceso. El paso a nivel de dispositivos de campo es claro en algunos campos de aplicación (Transport Automation) pero lento en procesos continuos y discretos. El motivo lo debemos buscar en la naturaleza de los dispositivos finales empleados

La interoperabilidad en capa 1 y 2 da a la electrónica de red Ethernet un impulso industrial notable. A los fabricantes les ofrece la posibilidad de ofrecer soluciones basadas en diferentes protocolos superiores y por lo tanto acceder a mayores mercados. A los instaladores y diseñadores les facilita la vida al permitir unificar el medio físico independientemente de la red que estén tirando (de oficina o de producción). La gestión del conocimiento también se facilita por no ser necesario personal extremadamente especializado en un sistema de comunicación propietario para poner en marcha o mantener instalaciones muy concretas. Incluso el personal encargado de las redes empresariales podría llegar a hacerse cargo de la red industrial. El personal de automatización también se beneficia del hecho de contar con un conocimiento estable en el tiempo y compatible con todos los protocolos superiores que se puedan plantear en cada aplicación. A nivel de mantenimiento la posibilidad de reducir dificultades para centralizar las islas de automatización de sistemas heredados supone sin duda una ventaja importante.

Según Smith (2005) El protocolo ethernet es la red que más se ha difundido en oficinas e industrias globalmente. Su infraestructura, interoperabilidad y escalabilidad aseguran un fácil desarrollo. Una vez que un equipo ha sido integrado a una red ethernet es posible su monitoreo a través de internet.

Las tecnologías Ethernet que existen se diferencian en estos conceptos:

**Velocidad de transmisión:**

- Velocidad a la que transmite la tecnología.

**Tipo de cable:**

- Tecnología del nivel físico que usa la tecnología.

**Longitud máxima**

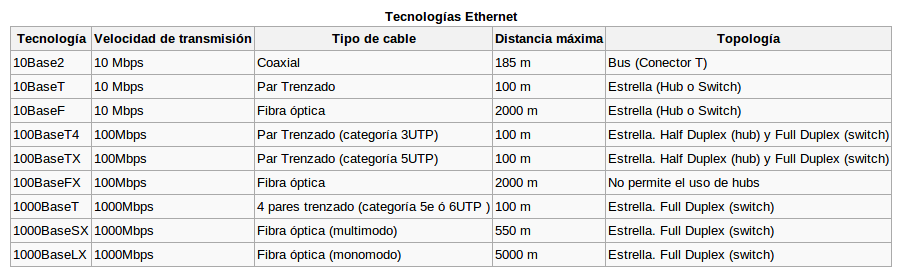
- Distancia máxima que puede haber entre dos nodos adyacentes (sin estaciones repetidoras).

**Topología**

- Determina la forma física de la red. Bus si se usan conectores T (hoy sólo usados con las tecnologías más antiguas) y estrella si se usan hubs (estrella de difusión) o [switches](http://es.wikipedia.org/w/index.php?title=Switche&action=edit&redlink=1) (estrella conmutada).

A continuación se especifican los anteriores conceptos en las tecnologías más importantes:

Cuadro – Tecnologías Ethernet



Fuente: Robin Sharp – Principles of Protocol Design

“En un sentido general, la comunicación entre ordenadores se lleva a cabo por el intercambio de datos -información codificada de alguna manera la cual depende del sistema. Podemos considerar este cambio como llevado a cabo en pasos discretos, que llamaremos comunicaciones elementales, en cada uno de los cuales un mensaje es transmitido”[[6]](#footnote-6)

Ethernet es un estándar de [redes de área local](http://es.wikipedia.org/wiki/Red_de_%C3%A1rea_local) para computadores con acceso al medio por contienda [CSMA/CD](http://es.wikipedia.org/wiki/CSMA/CD). CSMA/CD (Acceso Múltiple por Detección de Portadora con Detección de Colisiones), es una técnica usada en redes Ethernet para mejorar sus prestaciones.

El protocolo CSMA/CD incorpora dos mejoras que aumentan el rendimiento en una red: en primer lugar, no se transmite si hay otra estación hablando, y en segundo, si mientras se está transmitiendo detecta que otra estación transmite (es decir, se produce una colisión), la estación se calla, en lugar de seguir transmitiendo inútilmente al final de la trama.

Se produce una “colisión” cuando dos o más estaciones empiezan a transmitir simultáneamente o con una separación en el tiempo de propagación que las separa. Por ejemplo, en una red donde el tiempo de ida y vuelta es igual a 5.06 ms se producirá una colisión siempre que los dos nodos transmiten con una separación en el tiempo menor de 2.53 ms. Si la separación es mayor que 2.53 ms, no se producirá colisión.

La trama en una red Ethernet puede variar entre un mínimo de 72 bytes y un máximo de 1526. De este modo, la máxima tasa de colisiones para una trama Ethernet variará de acuerdo con el tamaño de la trama. Las operaciones en Ethernet necesitan un “tiempo muerto” entre tramas de 9,6 ms. El tiempo por bir para una Ethernet a 10 Mbps es 1/10-7 o 100 ns. Con base en lo anterior se puede calcular el máximo número de tramas por segundo para tramas de 1526 bytes:

9.6 ms + 1526 bytes \* 8 bits/bytes

9.6 ms + 12208 bits \* 100ns/bit

1.23 ms

Así en un segundo puede haber un máximo de 1/1.23 de tiempo por bit u 812 bytes de tamaño máximo de trama. Para un mínimo de tamaño de trama el tiempo por trama es:

9.6 ms +72 bytes \* 8 bits/bytes \* 100 ns/bit

67.2 x 10-6 s

De esta forma, en un segundo puede haber un máximo de 1/67.2 x 10-6 ms (en tiempo por bit ) o 14880 bytes y un tamaño mínimo de trama de 72 bytes.

Según IDC (International Data Corporation), a finales de 1997 más del 85% de las conexiones de red instaladas en el mundo eran Ethernet, lo cual representa unos 118 millones de ordenadores. El 17% restante está formado por Token Ring, FDDI, ATM y otras tecnologías. Todos los sistemas operativos y aplicaciones populares son compatibles con Ethernet, así como las pilas de protocolos de niveles superiores, tales como TCP/IP, IPX, NetBEUI y DECnet.

Así pues, la hegemonía en el mundo de las redes locales que Ethernet ha disfrutado desde su debut comercial en 1981 no sólo se mantiene sino que parece ir más lejos. Todos sus competidores han quedado en el camino. ATM, que durante algún tiempo parecía ser el futuro de las redes locales, no sólo no ha conquistado al usuario final sino que al parecer está desplazando rápidamente al backbone de campus por Gigabit Ethernet. Más aún, las últimas tendencias en redes de área extensa de muy alta velocidad basadas en DWDM (Dense Wavelenght Division Multiplexing) estudian la posibilidad de sustituir las tecnologías tradicionales ATM y SONET/SDH como medio de transporte de tráfico IP por una versión de Ethernet que funcione a 10 Gbps.[[7]](#footnote-7)

**Seguridad de red**

Las pegas a la inseguridad ofrecida por las redes apoyadas en Ethernet son fácilmente solventables mediante el uso de técnicas ya integradas en la electrónica estándar. Estas técnicas se pueden aplicar a distintos niveles:

• Nivel de puerto. Posibilidad de especificar qué equipos pueden comunicar a través de qué puertos. Se basan en las direcciones MAC o IP del equipo conectado.

• Nivel de nodo. La implementación de protección contra ataques de denegación de servicio (DoS) en base a limitadores de broadcast o de listas de control de acceso (ACL) para especificar con quién puede hablar un dispositivo concreto y con quién no.

• Nivel de diseño. Definición de reglas de acceso en los accesos remotos mediante dispositivos cortafuegos específicamente diseñados para esta función y fácilmente integrables en estructuras redundantes.

• Nivel de dispositivo final. Los equipos conectados a una red Ethernet deben ser capaces de reaccionar adecuadamente ante un ataque externo. Esto no afecta tanto a la electrónica de red como a la implementación del equipo final (PC, PLC, DCI...). Un ejemplo típico es un autómata con procesador único que sufre un ataque de denegación de servicio. La sobrecarga de comunicaciones podría llegar a afectar al ciclo de procesamiento. La mayoría de los controladores industriales actuales están preparados para este tipo de contingencia.

***Interfaz hombre-máquina.-***

Decíamos que una de las grandes ventajas de los buses de campo basados en Ethernet es ese conocimiento común en los niveles 1 y 2. En nivel 1, las particularidades de instalación de los enlaces serán los mismos para todos los proyectos. En el nivel 2 nos encontraremos con la configuración de la electrónica de red. Se ha realizado un esfuerzo en facilitar el manejo dichos equipos. El resultado es el interfaz gráfico apoyado en explorador http. Esto elimina la necesidad de herramientas especiales de configuración (basta con tener Internet Explorer en el equipo desde el que configuramos).

Una vez puesta en marcha la red, el personal de mantenimiento se enfrentará a la labor de conocer y diagnosticar el estado de la red. La solución tradicional para esto han sido las herramientas de gestión SNMP. Sin embargo ya existen herramientas software que, apoyándose en los mismos protocolos estándar, ofrecen una apariencia de sistema SCADA, haciendo transparente para el usuario el protocolo de gestión de red. Por último, las herramientas de integración apoyadas en OPC permiten incluir el estado de los dispositivos de red en cualquier sistema SCADA preexistente.

El avance de la tecnología Ethernet en el sector industrial es fuerte gracias a su adaptación a los retos que su nuevo ambiente plantea. La permanencia de Ethernet como solución en el tiempo está garantizada por el tremendo apoyo que recibe por parte de los fabricantes de equipos de control, integradores y clientes de sistemas industriales.

Sin duda el paso definitivo de Ethernet hacia el nivel de dispositivo final vendrá de la mano de la integración generalizada de interfaces Ethernet en éstos. A medida que el dispositivo va siendo más simple, la integración de un interfaz más complejo puede suponer un coste significativo añadido al equipo. En este sentido los módulos de entrada-salida distribuidos, tan habituales en las soluciones de control, agrupan conjuntos I/O que utilizan un único interfaz cuyo coste adicional está más justificado.

1.4.1 PROTOCOLO RPC

“El protocolo de Llamada de Procedimiento Remoto (RPC, por sus siglas en inglés, Remote Procedural Call) de Microsoft es una tecnología de gran alcance para crear programas distribuidos cliente/servidor. RPC es una técnica de comunicación entre procesos que permite que el software cliente y servidor se comuniquen.

Los sistemas operativos y programas se han ido volviendo cada vez más complejos a lo largo de los años. Con cada lanzamiento, hay más características. La complejidad creciente de los sistemas hace que sea más difícil para los desarrolladores evitar errores durante el proceso de desarrollo.

RPC está diseñado para mitigar estos problemas al proporcionar una interfaz común entre aplicaciones. RPC sirve como intermediario para hacer la interacción cliente/servidor más fácil y segura mediante la factorización de tareas comunes, como la seguridad, sincronización y manejo de flujo de datos, en una biblioteca común para que los desarrolladores no tengan que dedicar su tiempo y esfuerzo en desarrollar sus soluciones.

RPC de Microsoft es un mecanismo de comunicación entre procesos (IPC), que permite el intercambio de datos y la invocación de la funcionalidad que reside en un proceso diferente. Ese proceso puede estar en el mismo equipo, en la red de área local (LAN), o a través de internet. [...] Con RPC, la lógica esencial de la programación y el código de procedimiento relacionado pueden existir en diferentes equipos, lo cual es importante para las aplicaciones distribuidas”[[8]](#footnote-8)

1.5 PROGRAMACIÓN GRÁFICA EN LABVIEW

LabVIEW es un entorno de programación gráfica,

Hablar acerca de la Comunicación Serial y Ethernet

CAPITULO II. ARQUITECTURA DEL SISTEMA

2.1 DESCRIPCIÓN GENERAL

El sistema se plantea como una herramienta de fácil manejo para los estudiantes. Por tal motivo se ha diseñado un sistema en el cual las posibilidades de cometer errores sean minimizados, y que a la vez brinde flexibilidad para que los estudiantes puedan experimentar con el módulo de una manera sencilla.

Este sistema está pensado para que el usuario programe sus propias rutinas y elija el método de transporte de la comunicación, vía ethernet o serial, además que pueda hacer uso de los diferentes modos de funcionamiento del servomotor. Por lo tanto es importante definir las funciones básicas que permitirán controlar al motor, lo que es llamado la interfaz entre el programador, en este caso el estudiante, y la aplicación, las rutinas que ejecutará el módulo.

La interfaz programador/aplicación, (API[[9]](#footnote-9) por sus siglas en inglés) está implementada de tal manera que sea de fácil manejo para un amplio rango de estudiantes, con esto en mente se creó tres niveles de programación. El nivel bajo, con el cuál se entra en contacto directamente con el microcontrolador, el nivel medio, el cual sirve para controlar el motor utilizando las librerías del nivel medio, y el nivel alto, con el cual se puede controlar directamente el deslizador.

El microcontrolador recibe estos comandos, los procesa y ejecuta, al ejecutarlos lo que hace son las señales correspondientes al driver del servomotor, el cual controla el servomotor y finalmente el deslizador. Para que el microcontrolador envíe estas señales, necesita de circuitería que le permita interactuar con el driver.

La protección del deslizador es importante, ya que al ser utilizado por muchos estudiantes para realizar diversos tipos de prácticas es posible que se lo utilice de manera equivocada. La base del deslizador es la que se debe proteger contra una programación inadecuada, esto se logra a través de sensores que impiden que esta base choque contra los extremos del deslizador, enviando una señal de alerta al usuario y modificando el comportamiento del deslizador para evitar estas colisiones.

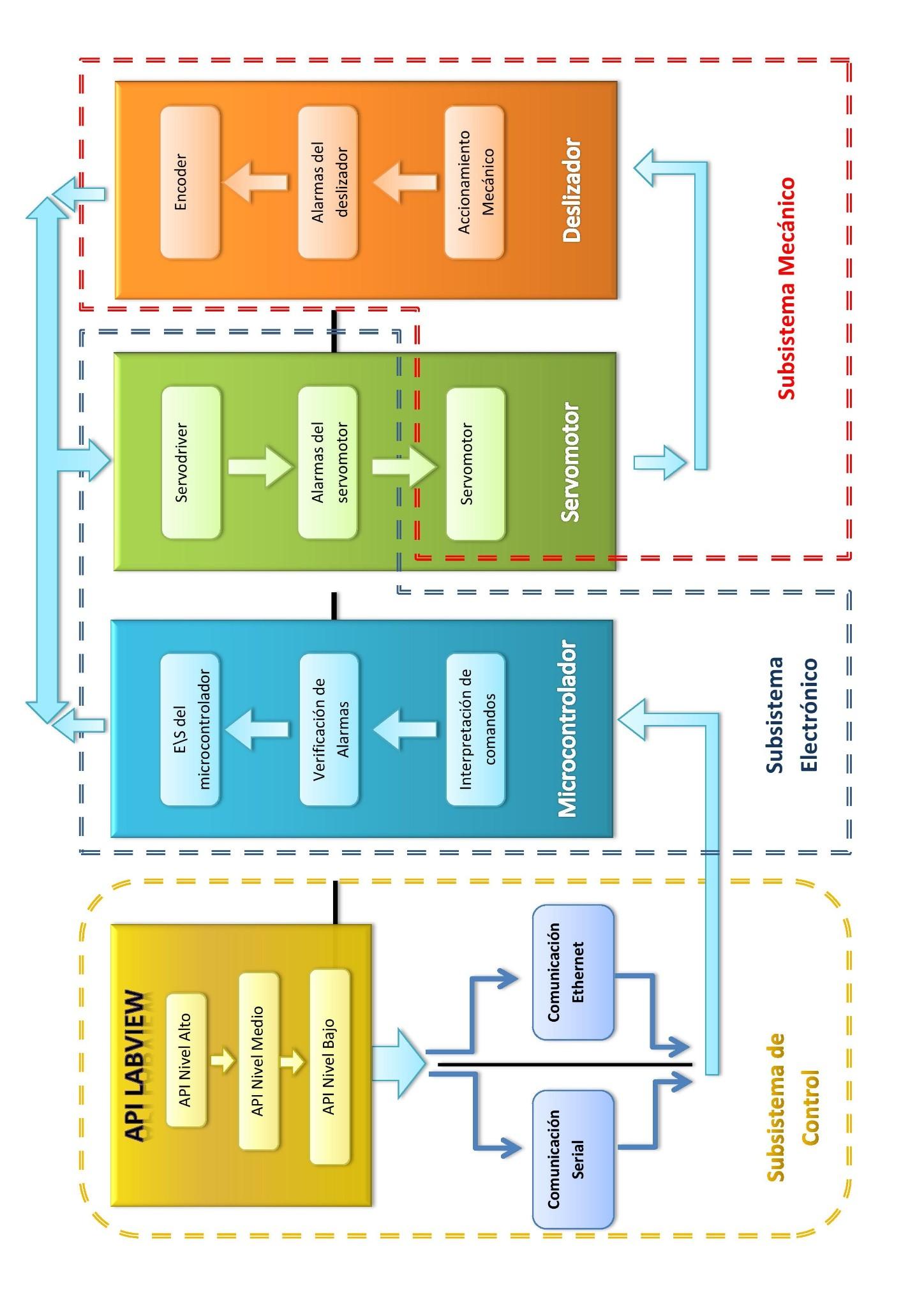
2.2 CONCEPCIÓN A BLOQUES

El sistema cuenta de tres partes definidas, las cuales son: el subsistema mecánico, el subsistema electrónico y el subsistema de control. El estudiante tendrá acceso al subsistema mecánico como al de control, es decir, es relativamente sencillo cambiar las piezas que componen estos subsistemas, sin embargo, el subsistema electrónico es fijo ya que de este depende el buen funcionamiento del sistema.

El subsistema electrónico, es el encargado de hacer llegar al motor los comandos enviados desde el computador o microcontrolador. Para eso necesita conectarse con el servodriver del motor, es importante que la conexión entre el microcontrolador y el driver se realice de manera correcta, pues es aquí donde existe mayor riesgo de dañar el equipo por su mal uso, por lo tanto se han tomado las medidas de seguridad necesarias para proteger el circuito tanto del microcontrolador como del servomotor etiquetando los conectores que son requeridas para el control del servomotor.

El sistema de control lo componen la interfaz de programación y el mecanismo de comunicación con el microcontrolador. El usuario puede elegir entre una conexión vía ethernet o serial, además puede elegir la velocidad en baudios de la conexión serial y si se utiliza el puerto CDC USB integrado[[10]](#footnote-10), o la conexión serial estándar. La interfaz de programación está desarrollada en LabVIEW, pero al ser este un sistema abierto, se tiene acceso al código fuente tanto de la interfaz como del microcontrolador, permitiendo comunicarse con el microcontrolador usando cualquier otro lenguaje de programación si el usuario así lo desee.

El sistema mecánico es la salida del sistema, sobre el cual realizaremos las diferentes prácticas. Siendo éste el deslizador, el servomotor, y el encoder encargado de leer la posición del deslizador. El encoder cuenta con un pequeño microcontrolador dedicado, el cual lee constantemente cualquier cambio de posición en el encoder y se comunica con el microcontrolador ARM-mbed de manera serial sincrónica con el protocolo I2C[[11]](#footnote-11).



2.3 DETERMINACIÓN DE SUBSISTEMAS

2.3.1 ESTRUCTURA DEL DESLIZADOR

El deslizador es el elemento mecánico del sistema, transforma el movimiento rotacional del motor en movimiento lineal mediante la utilización de un tornillo sin fin. Desde el punto de vista mecánico su diseño es muy sencillo, tiene un grado de libertad y cuenta con un encoder de posición relativa para su eje principal.

La transmisión de potencia se la realiza mediante un tornillo sin fin, el cual desliza la base que está apoyada a dos guías laterales, los apoyos usan rodamientos lineales para mejorar el deslizamiento y reducir la fricción. Cuenta además con cuatro sensores ópticos, dos a cada extremo del deslizador, los cuales detectan si la base del deslizador se acerca demasiado a los extremos, esto sirve para impedir que el deslizador colisione con sus extremos debido a una mala programación del sistema.

2.3.2 CONTROLADOR MICROPROCESADO

Este dispositivo permitirá reemplazar al PLC que actualmente se ocupa, aportará métodos de comunicación más flexibles al sistema, alterar las variables del deslizador de una manera más sencilla permitiendo crear programas y ejecutarlos directamente en el ordenador,

Las funciones que el controlador microprocesado debe desempeñar son: leer los valores del encoder del deslizador, controlar el driver del servomotor y establecer comunicación con el programa del usuario.

Existen dos tipos de control del servomotor, el control de posición y de velocidad, el microcontrolador se encarga de ambos tipos de control. En el control de posición el microcontrolador se encarga de generar un tren de pulsos de frecuencia variable, mientras que para el control de velocidad el microcontrolador genera una salida de voltaje variable para controlar la velocidad del servomotor.

Para la comunicación con el programa creado por el usuario se tienen dos mecanismos de transporte de los comandos, uno de manera serial, a través del protocolo RS-232 y otro mediante una red local Ethernet, a través del protocolo RPC.

El protocolo serial cuenta con varias configuraciones dependiendo del tipo de aplicación y la plataforma en la que se esté desarrollando la aplicación. Se puede utilizar el puerto USB CDC incluido en el microcontrolador, o se puede usar un puerto serial con lógica TTL, y finalmente se puede conectar este puerto TTL a una salida RS-232 estándar mediante el uso de un conversor MAX232.

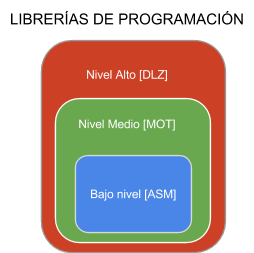
El encoder contará con un microcontrolador dedicado, el cual leerá constantemente los cambios de posición, éste se comunicará directamente con el microcontrolador del controlador a través del bus I2C, de esta manera se libera al microcontrolador principal de la lectura constante de la posición del encoder.

En el corazón del controlador tenemos al microcontrolador ARM-mbed el cual se encargará de implementar todo lo antes mencionado. Este microcontrolador es el principal elemento del sistema, las características con las que cuenta lo hacen ideal para este tipo de aplicaciones en las cuales se requiere de varias interfaces de comunicación y una ejecución rápida del código.

2.3.3 API DEL CONTROLADOR EN LABVIEW

Para controlar al deslizador se ha creado una interfaz de programación en LabVIEW. Esta permite controlar distintas variables del microcontrolador y consecuentemente del deslizador, para este propósito se las ha dividido en distintos niveles, dependiendo del control que se desee obtener del sistema.

De este modo existen tres niveles: bajo nivel o del microcontrolador, nivel medio o del motor, y nivel alto o del deslizador. Las librerías de nivel medio y nivel alto se construyen sobre la librería de bajo nivel, liberando al estudiante de los cálculos específicos del servomotor.



El nivel bajo es el nivel básico, el cual envía los comandos al microcontrolador. Es el nivel que habla directamente con el protocolo de comunicación sea este serial o ethernet, por tal motivo se le ha dado la etiqueta ASM. Las variables a controlar en este nivel son, el voltaje de salida, la frecuencia de salida, la dirección del motor, encenderlo o apagarlo, y leer la posición del encoder.

El nivel medio se construye sobre el nivel bajo, este se ocupa de las variables del motor, como su velocidad y posición, éstas variables dependen de la configuración de los parámetros del driver, por lo que esta librería toma en cuenta estos datos y calcula cuales son los valores que la debe tener el microcontrolador para que la salida del motor sea la deseada. Luego utiliza la librería de bajo nivel para comunicar estos valores al microcontrolador.

El nivel alto se encarga del control sobre las variables del deslizador, es decir, velocidad lineal, desplazarse una determinada distancia, ir a una determinada posición, este tipo de control requiere del manejo combinado del servomotor y el encoder, y utiliza las librerías de nivel medio y bajo para lograrlo de tal manera que el programador sólo debe asegurarse de que los parámetros del driver son los correctos.

2.3.4 RESPUESTA

El servomotor es nuestro elemento actuador, aquel que genera la respuesta del sistema, debido a su construcción y prestaciones ofrece un alto nivel de precisión en sus dos modos de operación, tanto para el control de velocidad como de velocidad. Especialmente para el control de posición ya que brinda una precisión de 1/10000Rev, pero una vez que se le aplica carga mecánica al sistema esta precisión se puede ver afectada.

El control de velocidad por otra parte, aunque preciso del lado del servomotor pierde precisión debido a que su voltaje es analógico y sujeto interferencias y ruidos al momento de llevar la señal de voltaje desde el controlador microprocesado hasta el servodrive. Se puede mitigar en algo estos efectos mediante una configuración adecuada de los parámetros del servodrive, y este es un aspecto importante para ser discutido en las clases de servo-mecanismos. Es en este tipo de control, en el control de velocidad donde la presencia de un encoder rotacional tiene gran importancia, ya que se puede controlar la velocidad del servomotor de una manera más precisa y aplicar diversos tipos de control que aseguren que la velocidad deseada se mantenga.

El control de velocidad o posición es configurado en el servodriver, estos no pueden ser cambiados remotamente desde el programa del usuario. Es importante que los parámetros sean los correctos de acuerdo a la operación que se va a realizar con el servomotor.

2.3.4.1 Control de Posición

El control de posición para del servomotor se lo realiza mediante un tren de pulsos, el servomotor cuenta con un encoder interno que utiliza para el posicionamiento. Este encoder tiene 10.000 divisiones por revolución, por lo que si se alimenta 10.000 pulsos al servodrive este rotará una revolución completa.

La velocidad con la que estos pulsos llegan al servomotor determina la velocidad con la que éste rota. Otro factor importante a considerar es el factor del encoder interno, ya que este puede ser modificado mediante dos parámetros del driver [PA:12 y PA:13], el cual, esencialmente, modifica electrónicamente este valor para que se pueda dar una revolución completa con un menor número de pulsos, lo cual aumenta la velocidad a la que gira el motor sin aumentar la velocidad del tren de pulsos, pero se pierde la precisión angular máxima con la que cuenta el servomotor.

2.3.4.2 Control de Velocidad

Cuando se encuentra en modo de operación de control de velocidad el servodriver espera recibir una señal de voltaje entre los terminales VCMD y AGND los cuales están destinados a este propósito,

# 

# 

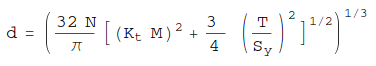
CAPITULO III. DISEÑO DEL SISTEMA

3.1 ESTRUCTURA DEL DESLIZADOR

3.1.1 SISTEMA MECÁNICO

Calculo del Eje.- Depende del torque del motor, de la carga máxima del deslizador, del acero del eje, y de los materiales disponibles.

Formula de: Diseño de elementos de máquinas; Mott, R; 4ta edición; Mexico; 2006.



3.1.2 DESLIZADOR

3.2 CONTROLADOR MICROPROCESADO

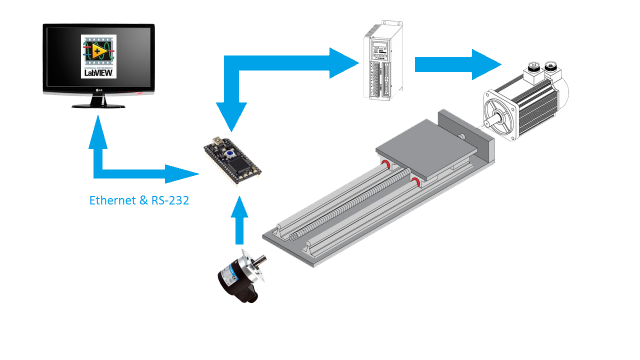
3.2.1 DIAGRAMA DE CONTROL

La forma en que se realiza el control es en la Programación de LabVIEW. En éste se debe crear los lazos de control o monitoreo del sistema, y se realizan las llamadas a hardware utilizando el API para el deslizador.

De esta manera el microcontrolador se vuelve el puente por el cual pasan todas las comunicaciones, este toma datos del encoder y realiza cambios en la salida al driver a petición del usuario, que es quién crea el control en su programación.

Sin embargo se deben tomar medidas de protección con el deslizador para evitar que una mala programación del usuario dañe el equipo. Para esto se han conectado cuatro sensores, dos a cada extremo del sensor, los cuales sirven para evitar una colisión de la base del deslizador. El microcontrolador automáticamente cambia la dirección de giro del servomotor cuando una de estas alarmas se activa y envía un mensaje de error al programa de control.

Fig. 3 – Representación del sistema modular



Fuente: Autor

El elemento de retroalimentación es el encoder, este puede enviar información de posición al programa principal para que actúe acorde a su programación.

Este equipo es modular, lo que quiere decir que lazo abierto o lazo cerrado depende del usuario en su programación. El sistema puede ser configurado con diversos elementos de entrada para su control y utilizar el encoder para monitorear el error de posición en caso de requerirse.

En lazo abierto:

Fig. 4 – Sistema en lazo abierto.

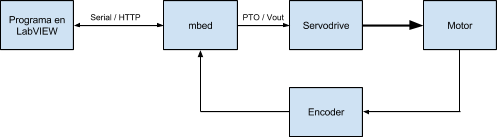


Fuente: Autor

En lazo abierto los comandos enviados desde el computador son ejecutados y aplicados al servomotor, las entradas del sistema son solamente, las alarmas del servodriver y del deslizador.

En lazo cerrado:

Fig. – Sistema en lazo cerrado



Fuente: Autor

En el lazo de control cerrado se utiliza el encoder del deslizador para obtener la posición angular relativa del servomotor. El encoder es rotativo y gira solidariamente al eje del motor, por tanto se puede conocer la posición actual y el sentido de rotación del servomotor.

El tipo de control, sea PID, Lógica difusa o cualquier otro es implementado por el usuario dentro de su programación.

3.2.2 ESQUEMA DEL CIRCUITO

El circuito que rodea al microcontrolador le sirve para ajustar los niveles de voltaje de sus entradas y salidas a las entradas y salidas del servodriver y del estándar Ethernet y Serial.

El circuito puede ser dividido en las siguientes partes:

**El microcontrolador:**

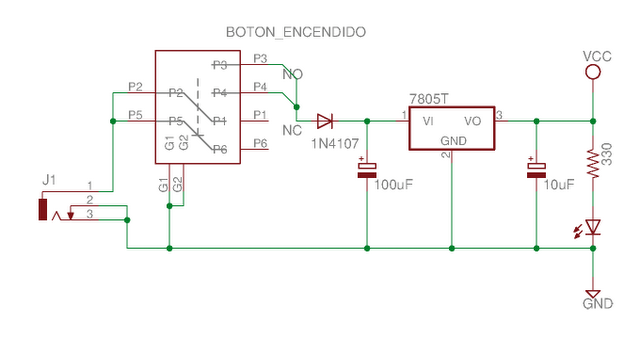
* + Se alimenta directamente desde el pin 2 (VIN) con 4,5v a 9,0v
  + Consume 200 mA de corriente

**El amplificador instrumental:**

* + Fuente partida de por lo menos +/- 10.0v
  + Masa virtual de referencia.
  + Ganancia de 6.06 aproximadamente.

Para alimentar los componentes digitales se utiliza una fuente de poder de 9.0 Vdc de entrada la cual es regulada a 5.0 Vdc la cual alimenta al microcontrolador. El esquema es el siguiente:

Fig. 6 – Esquemático del circuito de alimentación, placa principal



Fuente: Autor

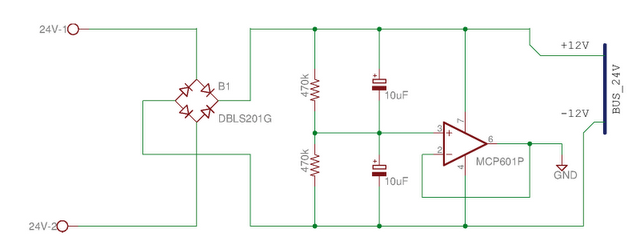
El voltaje de salida ofrece 5.0v con una corriente máxima de 1.0A. En los esquemas siguientes el terminal VCC se refiere a los 5.0v con referencia a masa GND. Se tiene además un botón para encender o apagar la placa del controlador, y un diodo de protección en caso de que la fuente se conecte de manera incorrecta.

# Salida analógica para el Driver

Para alimentar al amplificador instrumental tenemos un circuito que permite obtener +/- 12.0v a partir de 24.0v de entrada aplicados a los terminales dispuestos para este propósito.

Este circuito cuenta con un divisor de tensión compuesto por dos resistencias de valores iguales, también se ha dispuesto un amplificador operacional configurado como seguidor (buffer) en el punto que será la masa virtual del amplificador. Luego esta masa virtual se conecta con la masa del circuito de alimentación del microcontrolador, con lo que se logra una fuente simétrica para alimentar al amplificador instrumental y lograr una salida con voltajes positivos y negativos para el driver del servomotor. Una de las limitaciones de este circuito es que sólo puede entregar cantidades pequeñas de corriente, eso no es problema a que el amplificador no consume demasiada corriente, pero no es posible alimentar el resto del circuito directamente de este circuito.

Fig. 7 – Esquemático de fuente dividida para alimentación del amplificador instrumental



Fuente: Autor

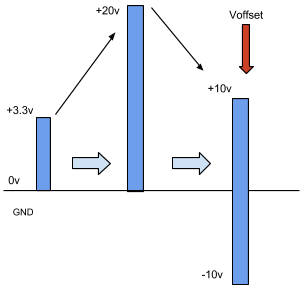
Para evitar una mala conexión el circuito cuenta con un puente de diodos, de esta manera la polaridad del voltaje aplicado en las borneras es indiferente, al amplificador siempre llegará la polaridad correcta.

El microcontrolador cuenta con una conversor DAC de 10 bits de resolución. Las librerías necesarias para el manejo de este recurso están implementadas como una variable de tipo flotante entre cero y uno (0.0 - 1.0), es decir, al valor de uno le corresponde una salida de 3.3V, a un valor de 0.5 le corresponde una salida de 1.65V en una relación lineal.

El Driver del servomotor, por otro lado, tiene una entrada analógica para el control de velocidad desde -10V a +10V, dependiendo del sentido de giro. Para poder acoplar la salida del microcontrolador con la entrada del driver se utiliza el amplificador instrumental AD620N.

Se configura el amplificador con una ganancia ‘G = 6.06’ aproximadamente, luego se le da un offset de -10V para poder obtener la variación de -10V+10V a partir de los 0V+3V del microcontrolador

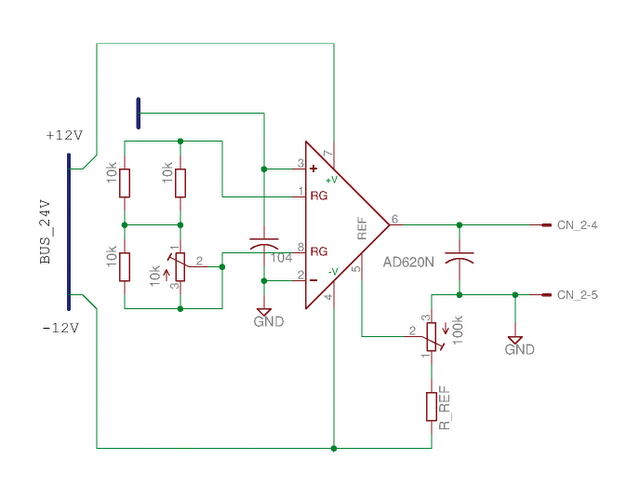
Fig. 8 – Niveles de voltaje para la salida analógica del microcontrolador



Fuente: Autor

El siguiente circuito muestra la implementación de la amplificación a la salida analógica del microcontrolador mbed (pin\_18) para controlar al servomotor en modo control de velocidad.

Fig. 9 – Esquemático del circuito de amplificación para la salida analógica

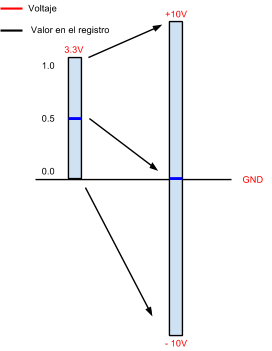


Fuente: Autor

El potenciómetro se encarga de regular la referencia del amplificador de manera que cuando el valor del microcontrolador se encuentra a la mitad (0.5 -1.65V) la salida al driver sea de cero voltios. De esta manera cualquier valor para la salida analógica mayor a cero punto cinco se dará un valor positivo hasta los diez voltios, y valores menores tendrán valores de voltaje negativos hasta menos diez voltios.

Por lo tanto se tiene la siguiente relación:

Fig. 10 – Niveles de voltaje resultantes



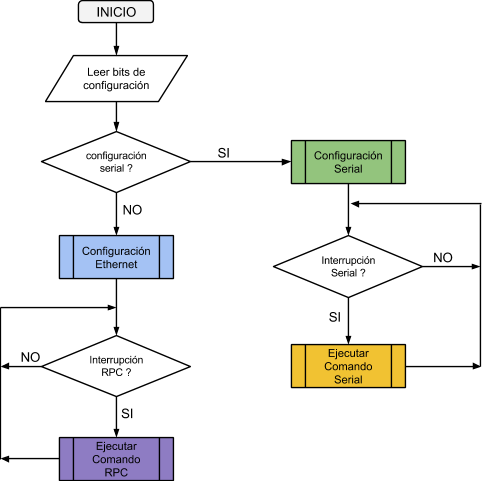
Fuente: Autor

3.2.3 PROGRAMACIÓN DEL MICROCONTROLADOR

La programación del microcontrolador se la ha realizado para que implemente una máquina de estados, estos estados son controlados por el usuario mediante el switch de configuración en la placa del microcontrolador. Estos estados configuran el mecanismo de transporte con el que se enviará los comandos al microcontrolador, estos deben ir acorde a la programación que se ha elegido para el microcontrolador

Al inicio del programa el microcontrolador lee los bits del Switch de configuración y actúa acorde a como han sido seteados.

Fig. – Lazo principal en la programación del microcontrolador

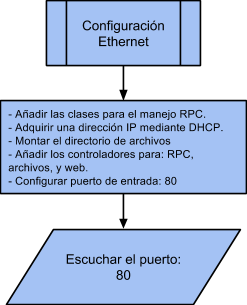


Fuente: Autor

Existen dos grandes divisiones: Configuración Ethernet y Configuración Serial. Se puede ejecutar solamente una a la vez, si se desea cambiar el modo de comunicación se debe cambiar la posición del bit uno del switch de configuración y luego resetear el microcontrolador.

Si se ha elegido la configuración ethernet el microcontrolador procede a inicializar las librerías y los objetos para recibir órdenes mediante el protocolo RPC. Si por el contrario se ha elegido la comunicación serial el microcontrolador configura el puerto de salida y velocidad de transmisión que se ha especificado en los bits de configuración.

Fig. 12 – Subrutina de Configuración para la comunicación vía Ethernet



Fuente: Autor

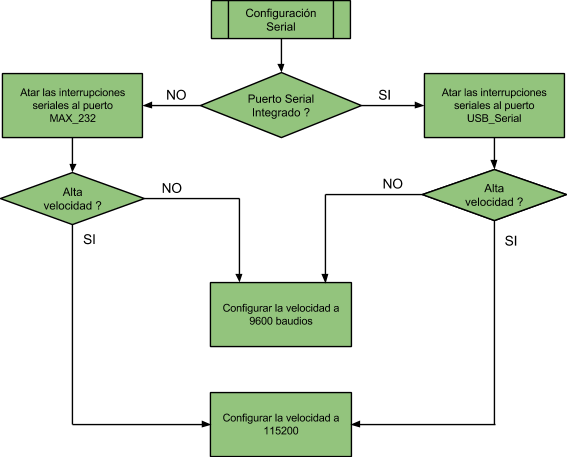
Para la configuración ethernet el microcontrolador añade las clases, funciones y objetos que van a ser controladas remotamente a través de RPC. Esto permite que se puedan realizar llamadas a funciones que ejecutan código en el microcontrolador desde el computador.

Estas funciones son las que controlan el servomotor, y pueden ser accesadas mediante el API de bajo nivel en LabVIEW. Sin embargo el control mediante protocolo RPC no sólo permite la ejecución de funciones sino también la creación de objetos de manera dinámica, esto quiere decir que se pueden acceder y utilizar las demás funciones del microcontrolador, como entradas y salidas digitales, salidas PWM y demás funciones que no han sido implementadas para el deslizador pero que el estudiante puede utilizar para el desarrollo de sus proyectos.

El uso de las funciones e interfaces del microcontrolador por medio del protocolo RPC queda fuera del alcance de este trabajo de grado, ya que su utilización no es necesaria para la implementación del sistema modular, pero es una función muy poderosa del microcontrolador la cual no requiere programación extra en el microcontrolador, más información acerca de su funcionamiento puede ser encontrado en la página web del microcontrolador[[12]](#footnote-12).

Fuente: Autor

Fig. 13 – Subrutina de configuración serial para la comunicación del microcontrolador



Fuente: Autor

Para la configuración serial se tiene dos puertos que pueden ser utilizados: El puerto serial integrado en el puerto USB, o el puerto serial con salida TTL.

Cualquiera de los dos puertos que se elija, la velocidad de transmisión puede cambiarse entre la genérica de 9600 baudios o una alternativa de alta velocidad, 115200 baudios. Además, para la salida a TTL se tiene un conversor MAX232 instalado y configurado en caso de que el usuario desee utilizarlo.

La configuración serial del sistema ha sido pensada para dar una gran variedad de posibilidades al momento de comunicarse con el microcontrolador, por lo tanto el usuario tiene la libertad de utilizar un microcontrolador, el computador o cualquier otro dispositivo con conexión RS-232 para el control del deslizador.

Una vez que el mecanismo de transporte de los comandos hacia el ARM-mbed ha sido establecido el microcontrolador entra en un estado de interrupciones, a la espera de que llegue un comando y procesarlo para nuevamente esperar la llegada de otro comando.

3.3 API PARA EL CONTROL Y MONITOREO EN LABVIEW

El sistema de control del deslizador está diseñado en el lenguaje de programación gráfica de LabVIEW, este cuenta con librerías en la forma de subrutinas (SubVI) las cuales actúan como una capa de abstracción entre el programador y el hardware del controlador.

Se tiene dos formas de hacer llegar los comandos al microcontrolador, de forma serial y mediante ethernet, por lo cual se tiene una división de librerías, las que están dedicadas a la comunicación serial y aquellas que sirven para comunicarse por medio de la red local. Sin embargo ambos tipos de librerías son simétricas, es decir, los mismos comandos se usan tanto para serial como para ethernet, la única diferencia es la forma de hacerlos llegar al microcontrolador.

Estas librerías están diseñadas para el paradigma de programación gráfico de LabVIEW, sin embargo se pueden utilizar otros lenguajes de programación siempre y cuando sean capaces de manejar los protocolos serial y RPC. Estos programas pueden ser: Matlab, python, java, C/C++, entre otros[[13]](#footnote-13), ejemplos de cómo implementar estos lenguajes pueden ser hallados en la página principal de la placa de prototipo ( <http://mbed.org> ), y de esta manera utilizar el API de LabVIEW como referencia para cualquier otro lenguaje de programación.

3.3.1 API DE BAJO NIVEL

Las librerías de bajo nivel establecen el mecanismo con el cual se pueden comunicar la aplicación del usuario con el microcontrolador. Forman la base de todas las demás librerías, por lo tanto deben ejecutarse rápidamente.

TIPOS DE CONTROL SOBRE EL DESLIZADOR:

* + 1. Encender el Servo ( SON, servo ON )
    2. Cambiar la frecuencia de salida. ( PULS+, tren de pulsos )
    3. Cambiar el sentido de giro ( SIGN+, salida digital )
    4. Cambiar el voltaje de salida ( VCMD, control de velocidad )
    5. Leer los datos del encoder.
    6. Leer las alertas del deslizador y del driver (ALM, alarmas)

Para cada uno de estas variables se ha creado una función en el microcontrolador que pueda manejar estos valores, ya sea de manera serial o ethernet.

**Protocolo serial.**

Se ha establecido una trama de datos los cuales son interpretados por el controlador cuando se reciben de manera serial:



La trama cuenta con dos parámetros separados por un guion, el primer parámetro es un número entero de 32 bits, el segundo parámetro es un código para la función que se va a ejecutar.

|  |  |
| --- | --- |
| S | Encender o apagar el servomotor |
| H | Cambiar la frecuencia de salida en hertzios |
| K | Cambiar la frecuencia de salidas en kilohertzios |
| D | Cambiar la dirección de giro |
| A | Cambiar el voltaje de salida. |
| G | Cambiar la posición angular del servomotor |
| V | Cambiar la velocidad de Posicionamiento Angular |
| E | Leer la posición del encoder |
| Z | Limpiar el contador del encoder |

El valor del primer parámetro (int\_32[[14]](#footnote-14)) desempeña diferentes funciones dependiendo del comando que lo acompaña, de la siguiente manera.

|  |  |  |  |
| --- | --- | --- | --- |
| Comando | Valores aceptables | Valor por defecto | Detalle |
| S | 0 - 1 | 0 | 0 Apagar el servomotor,  1 Encender servomotor. |
| H | 0 - 500000 | 0 | Frecuencia de salida del tren de pulsos, en hertzios. |
| K | 0 - 500 | 0 | Frecuencia de salida del tren de pulsos, en kilohertzios. |
| D | 0-1 | 0 | Sentido de manecillas de reloj (CW). 1 Contrario a las manecillas del reloj (CCW). |
| A | -10000 a +10000 | 0 | Voltaje de salida de +-10V con una resolución de 4.8mV. |
| G | 0 - 360 | 0 | Cambiar la posición angular del motor en grados sexagesimales. |
| V | 10 - 500000 | 1000 Hz | La velocidad en tren de pulsos a la cual se va a llegar al ángulo deseado para el eje del motor. |
| E | 0 - 1 | 0 | 0 Leer posición actual. 1 Leer velocidad actual. |

La salida analógica del microcontrolador se modifica para tener una mejor resolución sobre el voltaje de salida. Por esta razón el voltaje que se desee a la salida se debe lo debe ingresar en mili voltios, por ejemplo:

|  |  |
| --- | --- |
| Voltaje deseado | Comando enviado |
| + 5.000 V | 5000-A |
| - 5.000 V | -5000-A |
| + 9.874 V | 9874-A |
| - 8.750 V | -8750-A |
| +10.000 V | 10000-A |

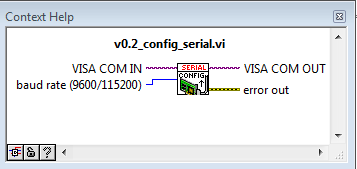
El microcontrolador ejecutará estos comandos sin importar cuál es la fuente de su procedencia, siempre y cuando arriben de manera serial y tengan el formato de trama correcto. Esto significa que se puede utilizar cualquier lenguaje de programación, o un microcontrolador inclusive, para controlar el deslizador.

**Librería en LabVIEW**

En LabVIEW las librerías que envían estos comandos al microcontrolador son las librerías del API de bajo nivel, en las que se simplifica y se automatiza la comprobación de errores y el envío de comandos al microcontrolador. Además de los comandos expuestos se tiene subrutinas que aceleran la salida de tren de pulsos de manera serial.

**API Serial - Configuración**

Para poder utilizar comunicación serial se debe primero configurar la velocidad y el puerto en el que se va a conectar el microcontrolador, para esta configuración se utiliza la subrutina config\_serial.vi



Entradas:

VISA COM IN: Puerto de entrada serial: COM3, COM5, ... etc

baud rate: Velocidad de comunicación, depende de la configuración del microcontrolador

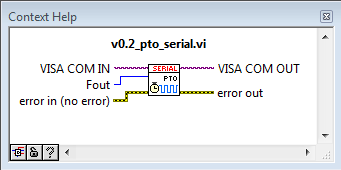
Salidas:

VISA COM OUT; Este es el puerto de comunicación listo para comunicarse con el microcontrolador.

error out: Salida de error.

**API Serial - Tren de pulsos**

Para controlar el tren de pulsos en el control de posición del servomotor se utiliza la siguiente subrutina pto\_serial.vi



Entradas:

VISA COM IN: Entrada del puerto de comunicación luego de haberlo configurado.

Fout: Frecuencia deseada para el tren de pulsos.

error in: Error de entrada

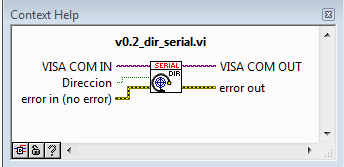
Salidas:

VISA COM OUT: El mismo puerto de entrada para conectar cualquier otra subrutina.

error out: Error de salida.

**API Serial - Dirección del Motor**

Para cambiar la dirección se utiliza la dirección del motor con una variable booleana, es decir, verdadero falso, se utiliza la subrutina dir\_serial.vi



Entradas:

VISA COM IN: Entrada del puerto de comunicación luego de haberlo configurado.

Dirección: Control de dirección.

Verdadero ( TRUE ) = CW

Falso ( FALSE ) = CCW

error in: Error de entrada

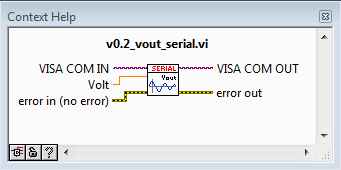
Salidas:

VISA COM OUT: El mismo puerto de entrada para conectar cualquier otra subrutina.

error out: Error de salida.

API Serial - Voltaje de salida

Para el control de velocidad del servomotor se tiene que controlar el voltaje de salida del microcontrolador con la subrutina: vout\_serial.vi



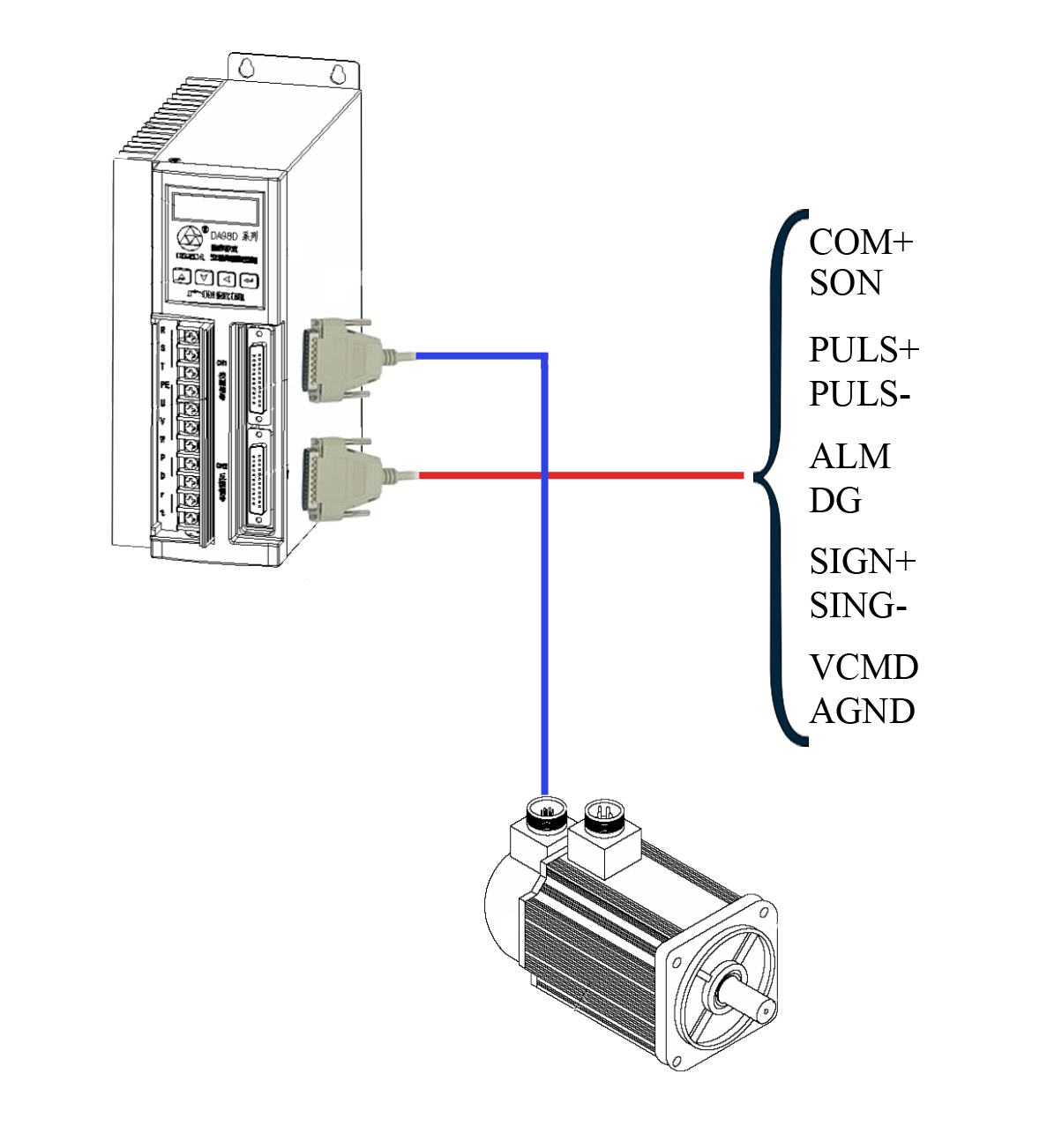
### 3.3.2 API DE NIVEL MEDIO

### 3.3.3 API DE ALTO NIVEL

## 3.4 RESPUESTA

### 3.4.1 MOTOR

### 3.4.2 DRIVER



# CAPITULO IV. IMPLEMENTACIÓN Y PRUEBAS

## 4.1 IMPLEMENTACIÓN FÍSICA DEL MÓDULO

## 4.2 CALIBRACIÓN DEL MÓDULO

## 4.3 PRUEBAS DE FUNCIONAMIENTO

## CONCLUSIONES

## 

## RECOMENDACIONES

## 

## ANEXOS

## Manual de operación del módulo

## Documentación de la interfaz de programación

## Manual de las prácticas a desarrollarse con el módulo

## Manual de mantenimiento del módulo.

1. LabVIEW es un entorno de desarrollo altamente productivo para la creación de aplicaciones a medida que interactúan con datos del mundo real o señales en campos como la ciencia y la ingeniería. [<http://www.ni.com/newsletter/51141/en/>] [↑](#footnote-ref-1)
2. RPC [Remote Procedural Call]. Es un protocolo que un programa puede utilizar para solicitar un servicio de un programa ubicado en otro equipo de una red sin necesidad de entender detalles de la red [↑](#footnote-ref-2)
3. Dirección del sitio web: <http://mbed.org/cookbook/Interfacing-with-LabVIEW> [↑](#footnote-ref-3)
4. Hiperterminal es un programa de comunicaciones y de emulación de terminal que viene con el sistema operativo Windows, a partir de Windows 98. [↑](#footnote-ref-4)
5. <http://mbed.org/cookbook/Interfacing-Using-RPC> [↑](#footnote-ref-5)
6. Principles of Protocol Design - Robin Sharp [↑](#footnote-ref-6)
7. ETHERNET: Su origen, funcionamiento y rendimiento - José Márquez Días y otros [↑](#footnote-ref-7)
8. Microsoft Library

   <http://technet.microsoft.com/en-us/library/cc759499(v=ws.10)> [↑](#footnote-ref-8)
9. API Aplication Programmer Interface [↑](#footnote-ref-9)
10. Se puede utilizar el mismo puerto de programación para comunicarse con el dispositivo [↑](#footnote-ref-10)
11. Circuitos inter integrados (Inter Integrated Circuit). [↑](#footnote-ref-11)
12. <http://mbed.org/cookbook/Interfacing-with-LabVIEW> [↑](#footnote-ref-12)
13. <http://mbed.org/cookbook/Homepage> [↑](#footnote-ref-13)
14. Número entero de 32 bits entre + 2 147 483 647 y - 2 147 483 647 [↑](#footnote-ref-14)