**Protocolos de comunicación**

Los protocolos de comunicación de campo son los que me permiten desarrollar mucha distancia.

A diferencia de los protocolos utilizados o que se implementan en los sistemas embebidos para comunicar los integrados entre sí. Ya hemos conocidos algunos como el UART, SPI, IC2.

Los protocolos empleados en los sistemas embebidos son protocolos que fueron obtenidos a partir de protocolos ya existentes en las computadoras y sistemas mayores, otras veces fueron adoptados a partir de protocolos utilizados en electrónica de consumo u industrial. Con el correr de los años estos mismos protocolos fueron sufriendo adaptaciones, dejando algunos en desuso y otros en plena vigencia con características que los hacen robustos incluso a las condiciones más adversas

**Protocolos Paralelos.**

La comunicación paralela transmite todos los bits de un dato de manera simultánea, por lo tanto la velocidad de transferencia es rápida, sin embargo tiene la desventaja de utilizar una gran cantidad de líneas, que lo hace más costoso y receptivo al ruido eléctrico. Esto es justamente es lo que los hace casi inexistentes en los ámbitos relacionados con microcontroladores, dado que lleva a un incremento de la complejidad de los circuitos periféricos al microcontrolador, y lleva a sacrificar una gran cantidad de pines en pos de mantener el bus de comunicación. Estas desventajas son las que abrieron lugar a la adopción de los protocolos seriales

**Protocolos Seriales.**

Existen dos modos básicos para realizar la transmisión de datos y son:

● Modo asíncrono.

● Modo síncrono.

***Comunicación Sincrónica****:* es una técnica que consiste en el envío de una trama de datos (conjunto de caracteres) que configura un bloque de información comenzando con un conjunto de bits de sincronismo (SYN) y terminando con otro conjunto de bits de final de bloque (ETB). En este caso, los bits de sincronismo tienen la función de sincronizar los relojes existentes tanto en el emisor como en el receptor, de tal forma que estos controlan la duración de cada bit y carácter.

***Comunicación Asíncrona*:** En esta transmisión el emisor decide cuándo va a enviar el mensaje por la red, mientras que el receptor no sabe en qué momento le puede llegar dicho mensaje, para esto se utiliza un bit de cabecera que va al inicio de cada carácter y uno o dos bits de parada que van al final de ese mismo carácter, esto se hace con la finalidad que tanto el emisor como el receptor puedan sincronizar sus relojes y poder decodificar el mensaje. En este tipo de transmisión no se maneja mucha velocidad ya que cada carácter es transmitido de uno en uno y por lo tanto puede ser un poco lenta.

**RS-232**

El protocolo RS-232 fue introducido por primera vez en 1962 por el sector de radio de la Alianza de Industrias Electrónicas (EIA). (ARC Electronics, 2010). Este protocolo en sus comienzos se utilizó para la comunicación entre dispositivos, conocidos en la jerga como dispositivos DTE (data terminal equipment) y dispositivos DCE (data communication equipment). Un equipo DTE es un equipo que convierte la información del usuario en señales, o convierte las señales recibidas. Los dispositivos DTE originales eran teletipos, y los dispositivos DCE originales eran usualmente módems, que a su vez transmitían los datos por la línea telefónica o por transmisores de radio para hacer packet. La versión actual de este protocolo es la TIA-232-F “Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange” lanzada en 1997. (CAMI Research)

**RS-485**

RS-485 o también conocido como EIA-485, que lleva el nombre del comité que lo convirtió en estándar en 1983. Es un estándar de comunicaciones en bus de la capa física del modelo OSI. (Texas Instruments, 2010) Está definido como un sistema en bus de transmisión multipunto diferencial, es ideal para transmitir a altas velocidades sobre largas distancias (35 Mbit/s hasta 10 metros y 100 kbit/s en 1200 metros) y a través de canales ruidosos, ya que reduce los ruidos que aparecen en los voltajes producidos en la línea de transmisión. El medio físico de transmisión es un par entrelazado que admite hasta 32 estaciones en 1 solo hilo, con una longitud máxima de 1200 metros operando entre 300 y 19 200 bit/s y la comunicación half-duplex(semi duplex) soportando hasta 32 transmisiones y 32 receptores. (Texas Instruments, 2010) 21 La transmisión diferencial permite múltiples drivers dando la posibilidad de una configuración multipunto. Al tratarse de un estándar bastante abierto permite muchas y muy diferentes configuraciones y utilizaciones, y su resistencia al ruido lo hace ideal en ambientes sensibles o industriales.

**Bus I2C**

I2C es el acrónimo de Bus Inter-IC. El bus I2C fue desarrollado en los comienzos de los años ochenta, por Phillips Semiconductors. El propósito original de este protocolo era proveer una manera sencilla de conectar la CPU con los chips periféricos en los televisores y demás sistemas de complejidad media-alta. Hoy por hoy, este bus es aceptado por la industria como un estándar de-facto. Asimismo, fue adoptado por los fabricantes líderes como ST Microelectronics, Atmel, Texas Instruments entre otros. (Embedded Systems Academy, 2015). Actualmente se encuentra en la revisión 4, del 13 de Febrero del 2012. (NXP, 2014)

**CAN Bus**

CAN (Controller Area Network) es un protocolo serial creado a mediados de los ochenta por la compañía Alemana Bosch. Está optimizado para enviar pequeñas cantidades de datos entre múltiples nodos. CAN tiene una tasa máxima de transmisión de 1MB/s, sin embargo su operación a bajas velocidades lo hace robusto al ruido y le permite cubrir largas distancias. El bus CAN fue originalmente diseñado para su uso en automóviles, pero fue haciéndose popular en otros ámbitos como control de líneas de ensamble industriales, como protocolo de transmisión dentro de barcos, etc. Pese a este éxito en crecimiento, la especificación de Bosch no define un estándar respecto a los voltajes o a los conectores o cables: cada organización define múltiples estándares a nivel físico. La capa física más común y utilizada es el estándar ISO 11898-1, pero puede ser implementado de otras maneras. (Bosch, 1991)

**USB**

El Universal Serial Bus más conocido por la sigla USB, es un bus estándar industrial que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre computadoras, periféricos y dispositivos electrónicos. Su desarrollo partió de un grupo de empresas del sector que buscaban unificar la forma de conectar periféricos a sus equipos, por aquella época poco compatibles entre sí, entre las que estaban Intel, Microsoft, IBM, Compaq, DEC, NEC y Nortel. La primera especificación completa 1.0 se publicó en 1996, pero en 1998 con la especificación 1.1 comenzó a usarse de forma masiva. (Allusb.com) Desde 2004, aproximadamente 6 mil millones de dispositivos se encuentran actualmente en el mercado global, y alrededor de 2 mil millones se venden cada año.

**SPI**

El Bus SPI (del inglés Serial Peripheral Interface) es un estándar de comunicaciones, usado principalmente para la transferencia de información entre circuitos integrados en equipos electrónicos desarrollados en un principio por Motorola. El bus de interfaz de periféricos serie o bus SPI es un estándar de facto para controlar casi cualquier dispositivo electrónico digital que acepte un flujo de bits serie regulado por un reloj. Es un protocolo full-duplex que funciona bajo un paradigma maestro-esclavo. Incluye una línea de reloj, dato entrante, dato saliente y un pin de chip select, que conecta o desconecta la operación del dispositivo con el que uno desea comunicarse. De esta forma, este estándar permite multiplexar las líneas de reloj. (Microchip Technology, 2014).

**PROTOCOLOS**

1. SPI.
2. I2C.
3. 802.15.4.
4. USB Device.
5. USB Host
6. Ethernet.

No ahondaremos en detalles en los protocolos SPI y I2C ya que fueron vistos en capítulos anteriores y desarrolladas sus características, trataremos de mencionar algunas y funcionalidades de los demás.

**IEEE 802.15.4 Redes inalámbricas de área**

**IEEE 802.15.4** es un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal con tasas bajas de transmisión de datos (low-rate wireless personal area network, LR-WPAN).

Especialmente diseñado para la comunicación de **sistemas con baja potencia computacional y limitaciones** de energía, el estándar 802.15.4 fue desarrollado por el Instituto de Ingenieros Eléctricos y Electrónicos y cuenta con un servidor base para otros protocolos de comunicación muy populares como Zigbee.

El estándar **WPAN 802.15** define el estándar de red de área personal inalámbrica. Aquí es donde encontramos las redes 802.15.4, que están compuestas por dispositivos de baja potencia computacional, baja potencia de señal y por lo tanto un rango reducido y normalmente alimentados por baterías. El estándar 802.15.4 define las capas físicas y de enlace de datos en el modelo OSI.

**IEEE 802.15.4: WPANs de baja tasa de**

**transferencia de datos (LR-WPAN)**

***Características:***

Una característica del IEEE 802.15.4 es la posibilidad de que en la capa física sea posible **evaluar el nivel de interferencias** en un determinado instante y así ajustar el canal de comunicación. Esta norma permite además el uso de direcciones de red de 16 bits o 64 bits, siendo que el uso de 64 bits es útil cuando tenemos redes de gran tamaño. En cuanto a la topología, esta norma es compatible con topologías de estrella y peer-to-peer

Comunicaciones simples de bajo costo.

Bajas tasas de transferencia (throughput).

Para aplicaciones con limitaciones de potencia.

Confiabilidad en la transferencia de datos.

Opera en una banda de frecuencia sin licencia.

Uso doméstico e industrial.

Dispositivos con fuente de alimentación autónoma.

Batería.

Panel solar.

Extremadamente bajo consumo de potencia (Ciclo de Trabajo).

***Principales áreas***:

Domótica y seguridad.

Redes de sensores inal ámbricos

Productos electónicos de consumo.

Electromedicina.

Control y monitoreo de veh ículos.

Agricultura inteligente.

**USB Device**

USB significa “Universal Serial Bus”

Controlado por “Host” (solamente uno por bus)

On‐the‐Go (Protocolo de negociación de host) – permite a dos dispositivos negociar el rol de host

Topología estrella

Se pueden utilizar hubs para dividir alta y baja velocidad

Hasta 127 dispositivos pueden ser conectados a un bus USB en cualquier momento

Utiliza 4 líneas malladas: 2 son de alimentación (+5v & GND) l d d d l ñ l 10 GND) y los otros 2, un par trenzado donde las señales se transmiten en modo diferencial

Suporta plug’n’plug con drivers que son cargados dinámicamente ‐ PID/VID (Product ID/Vendor ID)

USB soporta diferentes modos de transf i erenc a: Control, Interrupción (Interrupt), Masiva (Bulk) e Isócrona (Isochronous)

La alimentación se transporta por el Bus

USB distribuye la alimentación a todos los dispositivos conectados, eliminando la necesidad de una fuente externa para dispositivos de bajo consumo

**Codificación**

Utiliza codificación NRZI para enviar los datos con un campo de sincronización para sincronizar el clock del host y el receptor receptor

NRZI define un 0 lógico como una transición en el valor de tensión tensión, y un 1 manteniendo manteniendo el nivel

Se necesita Bit stuffing porque los receptores sincronizan transiciones. Si se envían muchos 1s entonces el receptor puede perder sincronismo

**Velocidad**

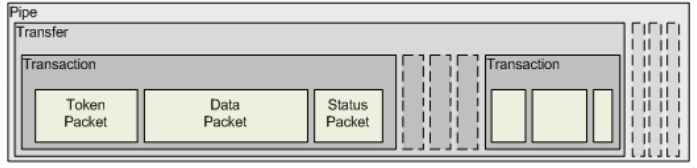
Un dispositivo USB debe indicar su velocidad llevando D+ o D‐ a 3.3 volts.

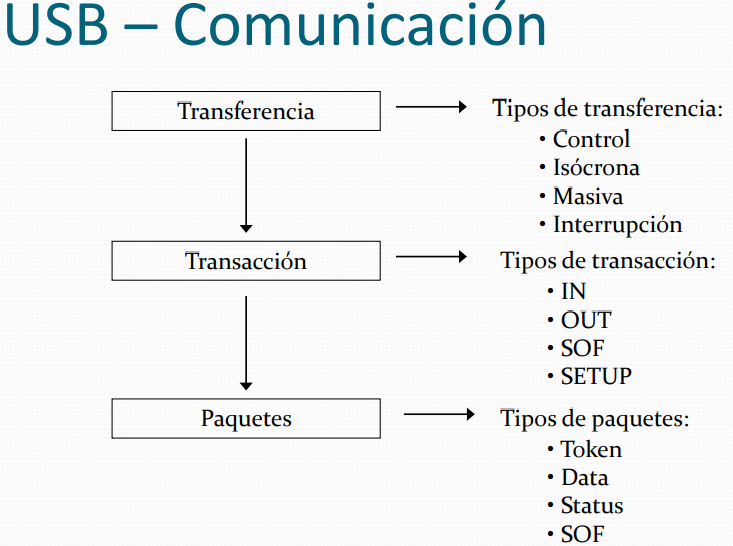
Sin resi i stenc a de pull‐up, USB asume que no hay nada conectado al Bus.

En el modo “high speed” el dispositivo primero se conecta en modo “full speed”, luego se remueve el resistor de pull‐up para balancear la línea

**Comunicación**

A diferencia de RS‐232 o interfaces serie similares donde el formato de los datos a ser enviados no está definido, USB posee varias capas de protocolos





**PAQUETES**

Cada transacción USB consiste en:

Paquete Token (encabezado que define lo que se espera a continuación)

Paquete de datos (opcional ‐ contiene el payload)

Paquete de Status (Usado como acknowledge en las transacciones y como una forma de corregir errores)

**Analizador de protocolo**

Las transferencias consisten en una o más transacciones. Un pipe solamente soporta un tipo de transferencia

En una transacción , transferencia desde “host” a dispositivo o viceversa. La dirección se define en el paquete “token”

En general, el destinatario responde con un paquete de status i di d n can o si fue exitosa

**USB Host**

• USB 3.1 – Especificación Actual – Poca aplicación en embedded por requerimientos de hardware, software, consumo y costo

• USB 2.0 – Mayor difusión en embedded – Es común encontrar dispositivos y hosts que la cumplan

**USB 2.0**

• Agrega una nueva velocidad de transferencia – Low speed (USB 1.x): 1.5 MBit – Full speed (USB 1.x): 12 MBit – High speed (USB 2.0): 480 Mbit

• Agrega un nuevo tipo de Host

– UHCI (Universal Host Controller Interface) USB 1.x: Mas carga en el software, hardware más simple (más barato y consume menos)

– OHCI (Open Host Controller Inteface): USB 1.x: Mas carga en el hardware, software más simple

– EHCI (Enhaced Host Controller Interface): USB 2.0, unifica el host para la velocidad high speed

HOST EMBEBIDO

• Cuando el dispositivo no necesita actuar como usb device, o necesita actuar simultáneamente como host y device, se emplea la especificación “Requirements and Recomendations for USB Products with embedded hosts”

• Es similar a OtG

• También tienen una Targeted Peripheral List. La lista puede incluir una clase completa de dispositivos (ej. Mass storage)

FUNCIONES

1. DETECTAR DISPOSITIVOS

Gestionar la enumeración • Fijar la velocidad de transmisión de datos • Asignar direcciones • Solicitar información adicional para que el S.O. determine el controlador a utilizar • Detectar la remoción del dispositivo e informar al S.O

1. GESTIONAR EL FLUJO DE DATOS

Asignar el tiempo de bus a cada dispositivo de acuerdo a su velocidad y tipo de transferencia. • Si al enumerar el dispositivo pide más ancho de bus que el disponible, negociar una porción menor

1. CHEQUEAR ERRORES DE TRANSMISION

En las transferencias, se agrega redundancia para detectar errores, si el dispositivo no reconoce la recepción, el host debe reenviar el paquete de datos • En caso de errores reiterados, el host informa al controlador del dispositivo

1. PROVEER Y GESTIONAR LA ALIMENTACION

Durante la enumeración el dispositivo plantea sus necesidades de corriente, el host debe verificar si puede proveerlo e informar al dispositivo si la corriente está disponible o negociar un consumo menor. • El host puede comandar a los dispositivos para que entren en bajo consumo para reducir la corriente del bus.

1. INTERCAMBIAR INFORMACION CON LOS DISPOSITIVOS

• El host inicia las transmisiones hacia los dispositivos, de acuerdo al tipo de transferencia y velocidad • Para las transferencias control e interrupt, el host debe realizar comunicaciones periódicas con el dispositivo • Para las transferencias bulk e isochronous, la transferencia empieza a ser dependiente del ancho de banda de bus disponible.

**ETHERNET**

En la actualidad se realiza una transición a la nueva versión del protocolo IP, denominada IPv6, que permite mayor cantidad de estas direcciones disponibles para ser asignadas a usuarios, mejor soporte y prioridad para entrega de datos en tiempo real, soporte para gran variedad de dispositivos además de computadores, buen manejo de tráfico a pesar del crecimiento de la red, entre otras características.

Por otro lado, en la actualidad se está incrementando el uso de los llamados sistemas embebidos en distintas áreas por razones de portabilidad y comodidad, entre otras. Sus aplicaciones van desde el uso doméstico hasta el industrial, caracterizándose por su enfoque en una determinada tarea, menor consumo de energía y fácil fabricación, permitiendo el desarrollo de aplicaciones como celulares, lavadoras, sistema de frenos, etc. En este sentido se puede argumentar que existe la necesidad de generar propuestas de investigación y desarrollo que permitan la integración de IPv6 en sistemas embebidos. Por tal razón, el presente paper pone a disposición de la comunidad académica una propuesta para incluir en los sistemas embebidos el soporte para la navegación en internet con la versión 6 del protocolo IP; de esta manera, se espera contribuir de una manera práctica en el desarrollo de esta importante línea que está en pleno desarrollo. Los elementos fundamentales que cimientan el documento son:

* Herramientas utilizadas.
* Inclusión de archivos necesarios y diseño para el soporte de IPv6 en el microcontrolador. Algoritmo de procesamiento de paquetes.
* Implementación en red. Análisis y resultados.

IP resulta ser una opción para la intercomunicación de sistemas embebidos y el control de estos debido a varias características, entre las que destacan:

* Interoperabilidad.
* Arquitectura evolutiva y versátil.
* Estabilidad y universalidad de la arquitectura.
* Escalabilidad.
* Configuración y manejo.

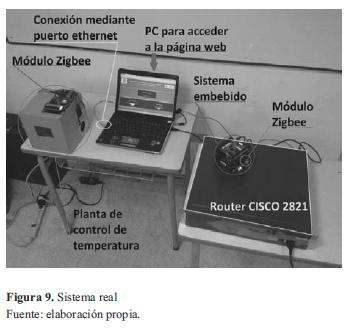
**Ejemplo del uso del protocolo en un sistema embebido**

**Implementación**

Para comprobar el funcionamiento de IPv6 sobre el sistema embebido de acuerdo a lo planteado anteriormente, se ha realizado un montaje que permite hacer un control de temperatura de tipo ON/OFF de manera remota, con el uso de módulos Zigbee como lo muestra el diagrama de la [figura 8](http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0123-921X2013000300015#f8).



Se ha utilizado un router cisco 282I, este dispone de dos interfaces Gigabit Ethernet. En una de ellas es conectado el sistema embebido (Tower System) por medio de un cable Ethernet, en la otra un PC. Este último tiene una tarjeta de red con puerto Ethernet y un navegador de Internet, el montaje real se puede apreciar en la [figura 9](http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0123-921X2013000300015#f9).



Desde el PC que se encuentra conectado al router se accede a la página web almacenada en el sistema embebido, digitando en la barra de direcciones su IP y adicionándole /index.shtml, para que quede de la siguiente manera [2001:3::2]/index.shtml. Después de que la página ha sido cargada en el navegador del PC, este actualiza el campo de "temperatura actual" cada segundo.

**ARDUINO**

Una de las principales ventajas de Arduino es que podemos dotarlo de comunicación de una forma sencilla añadiendo un shield o una breakout board y dispondremos de casi cualquier tipo de comunicación tanto de acceso a Internet como de para comunicar arduinos entre sí o con otros dispositivos de una red privada.

**Ethernet**: la forma más clásica de comunicar arduino mediante el shield de Ethernet

**Ethernet**: la forma más clásica de comunicar arduino mediante el shield de ethernet. <https://www.arduino.cc/en/Main/ArduinoEthernetShield>

**Wifi**: Hay múltiples formas de conectar Arduino a internet mediante wifi:

* Shield: <https://www.arduino.cc/en/Main/ArduinoWiFiShield101>
* Wifi Integrado en MCU: <https://www.arduino.cc/en/Main/ArduinoMKR1000>
* ESP8266: <https://espressif.com/en/products/hardware/esp8266ex/overview>
* HW compatible Arduino con Wifi Intergrado: [ESP8266](https://espressif.com/en/products/hardware/esp8266ex/overview)o [ESP32](https://espressif.com/en/products/hardware/esp32/overview)
* Placas con Wifi integrado: <http://www.arduino.org/products/boards/arduino-uno-wifi>
* Arduino Yun: <https://www.arduino.cc/en/Main/ArduinoBoardYun>

