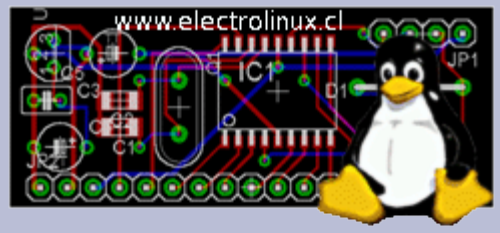


Control Distribuido con Linux desde Cero

6to Encuentro Nacional de Linux 2005

Organizador del evento

Universidad Arturo Prat de Iquique

20 de Octubre 2005

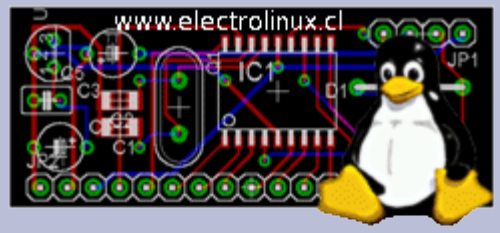
Diseño y desarrollo:

Sr. Ricardo Albarracin B. rab@electrolinux.cl

Programación y server WEB minimalista:

Sr. Eduardo Silva P. Ing. en Informática

Electrolinux http://www.electrolinux.cl



Control Distribuido con Linux desde Cero

Apagar los celulares por favor

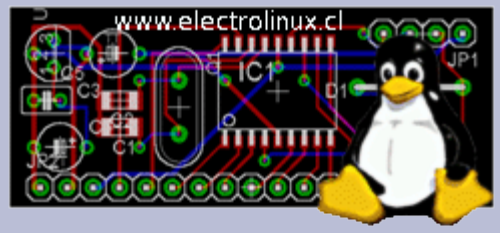
Diseño y desarrollo:

Sr. Ricardo Albarracin B. rab@electrolinux.cl

Programación y server WEB minimalista:

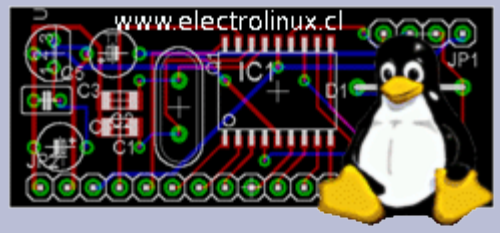
Sr. Eduardo Silva P. Ing. en Informática

Electrolinux http://www.electrolinux.cl



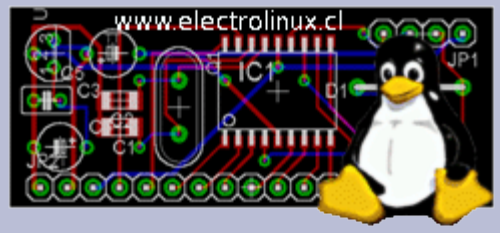
Objetivos de la Presentación

- Esta tiene por objeto demostrar en forma práctica un desarrollo completo de un sistema de *control distribuido con Linux desde cero*, con administración remota vía WEB (TCP/IP) lo que permite realizar la administración y gestión en sistemas multiplataforma y con todas las ventajas de uso bajo un sistema operativo seguro, robusto, escalable, confiable como Unix para arquitecturas i386, usando kernel Linux o BSD.
- Veremos la situación actual del desarrollo tecnológico en Chile, algunos comentarios generales de la educación en tecnologías de información y comunicaciones, lo definido como TIC.
- Algunos comentarios menores de BD relacional para aplicaciones en la gestión del control a distancia.
- Algunas aplicaciones de ejemplo de código 'C' y finalmente una implementación práctica de lo planteado.



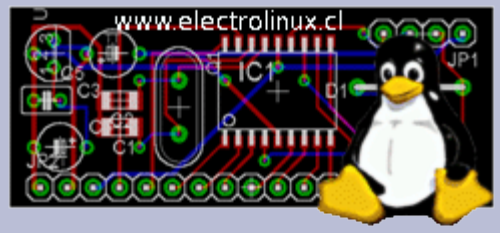
Puntos de una Reflexión Tecnológica

- Nuestro país ha crecido sostenidamente en los últimos años, de acuerdo a informes internacionales, los que nos han evaluado como el mejor país latinoamericano, sobre Brasil, Argentina, Méjico, pero con graves falencias en el desarrollo tecnológico y en la educación.
- IMHO seguimos siendo un país *Colonial* tecnológicamente hablando, aún nos pesa la historia, no nos hemos dado cuenta de nuestras capacidades y potencialidades.
- Los países que hoy llamamos desarrollados y que se encuentran en la cresta de la ola tecnológica, pasaron por problemas de desarrollo, cometieron errores y los superaron, un gran motor de ello fueron las crisis de guerras mundiales, ya que no tenían tecnología y estas crisis los obligó a desarrollarla.
- Nuestro país ha perdido muchos recursos económicos por la dependencia tecnológica, ya sea por la obsolescencia, por el cambio de tecnología de productos o por no tener peso frente a multinacionales, somos un mercado muy pequeño comparado con países vecinos, Brasil, Argentina.
- En Chile, somos capaces de realizar desarrollos profesionales sustentables, pero nos falta el compromiso entre nosotros, empresarios y tecnócratas, lo que permite implementar soluciones locales. Esto nos permitiría crecer, ser independientes en el conocimiento tecnológico y sobre todo tener soluciones propias, con nuestra idiosincrasia y metodologías.



Puntos de una Reflexión Tecnológica II

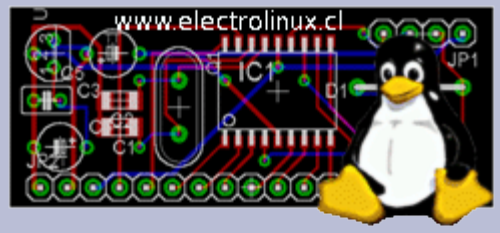
- Nuestro país ha crecido en exportaciones, de ser un país mono exportador de cobre, hemos pasado a exportar además materias primas y alimentos, pero nada con valor agregado mayor.
- Nuestro país por una parte vende cobre, pero por otra importa cables y conductores eléctricos, por nombrar un ejemplo, es decir recompramos nuestros propios productos procesados en países de mayor desarrollo tecnológico.
- El *"valor agregado"* de los productos de exportación esta relacionado con el desarrollo tecnológico, este con la industria y esta con la calidad de la educación.
- Nos falta mano de obra calificada para poder enfrentar un desarrollo tecnológico masivo en las TIC, la electrónica, desarrollo de software orientado a procesos productivos, desarrollo de aplicaciones sobre BD abiertas (postgreSQL), sin embargo necesitamos como país, disponer de este desarrollo.
- Son muchas las pymes que administran hoy en día con planillas electrónicas y carecen de sistemas de gestión de información orientadas a los servicios que proveen.
- En países como Rusia, Japón y sobre todo China, desarrollan sus propias soluciones tecnológicas en base a software libre, donde usan fuertemente Unix/BSD como Netbsd, Freebsd, Openbsd, Linux y han colaborado en muchos proyectos open-source.



Generalidades de PostgreSQL

PostgreSQL un RDBMS con poder

- PostgreSQL tiene sus orígenes en la Universidad de Berkeley en 1977, con el nombre de POSTGRES. En 1986, Michael Stonebraker de Berkeley, empezó a trabajar sobre ese código abandonado de esta RDBMS y lo renombró a Postgres95, el objetivo era desarrollar un RDBMS libre el proyecto fue finalmente bautizado con PostgreSQL.
- Hasta el día de hoy, es un proyecto con una gran actividad y muchas grandes empresas como RedHat, Fujitsu, Afilias, Pervasive, EnterpriseDB, Green Plum, Command Prompt, 2nd Quadrant entre otras lo apoyan.
- Las características más particulares de PostgreSQL:
- * Extensibilidad, permite escribir procedimientos en C,C++,C#,Perl,PHP,Pyton,Java,Tcl y más.
- * Permite definir su propio lenguaje y tipos de datos si se desea y nuevas funciones de agregación.
- * Objetos relacionales
- * Datos peculiares como son:
- 1. Direcciones MAC, Direcciones IP (IPV4 e IPV6).
- 2. Figuras geométricas.
- 3. Coordenadas geográficas.
- 4. Textos y datos binarios de hasta 1GB, permite almacenarlos con compresión.
- Se distribuye bajo la licencia BSD, lo que implica que no tiene costo alguno, sobre cualquier plataforma.



Generalidades de PostgreSQL II

- ## PostgreSQL un RDBMS con poder

- El soporte técnico esta respaldado por una gran comunidad de usuarios que en foros, IRC y listas de correos se puede encontrar ayuda para resolver los problemas y/o dudas que nos surjan. Además se puede contratar soporte con algunas de las empresas mencionadas.

- Dentro de sus principales características técnicas se encuentran:

- * Transacciones

- * Subselects

- * Triggers

- * Vistas

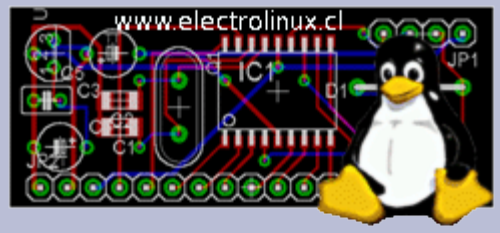
- * Integridad Referencial de claves externas

- * Sistemas de bloqueo

- * Control de Concurrencia Multiversion MVCC

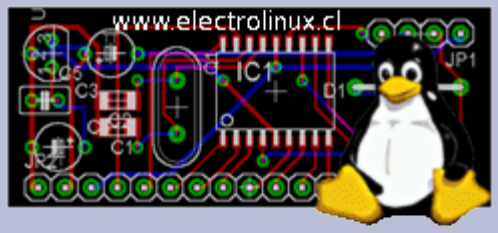
- * Sistema de reglas que permite definir vistas actualizables

- También cuenta con otras funcionalidades las cuales muchas RBDMS comerciales no tienen como son, definir propios tipos de datos, disponer de herencia, reglas y control de concurrencia multi-versión, que permite reducir el bloqueo de conexión, tener un sistema de replicación Slony-I y contar con soporte SSL, sistema de autentificación nativa mediante kerberos (v.4 y v.5), procedimientos almacenados (stored procedure), optimización de consultas “data-warehousing” y OLAP sean más rápidas, ahora se encuentra optimizado para AMD opteron gracias a la participación de la Universidad de Massachusetts en Amherst. Tambien tiene soporte para el estándar de comunicación TCP/IP IPv6 y el protocolo Rendezvous de Apple para funcionamiento sin configuración.



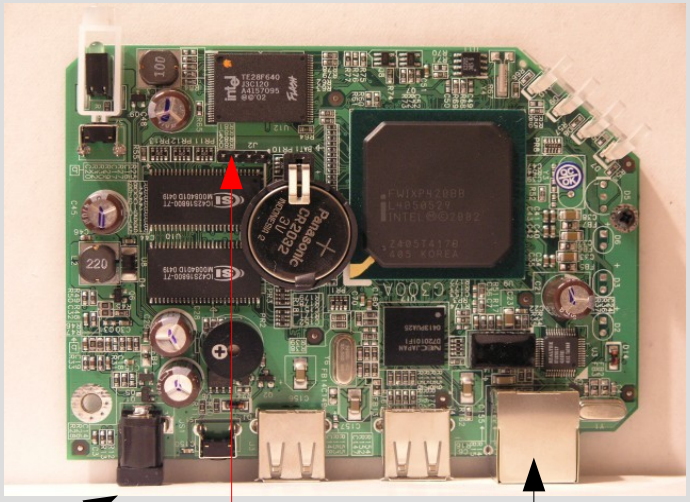
Introducción

- En nuestra pagina WEB disponemos de varios documentos que explican como usar herramientas libres para hacer desarrollo de electrónica digital en base a Microcontroladores, los invito a revisarlos.
- Con esta base hemos desarrollado un sistema de control distribuido partiendo desde la base, fabricando un hardware mínimo para ello, como ejemplo hemos basado esta presentación sobre nuestro kit más pequeño el HE251, sobre este hardware, hemos desarrollado un protocolo multipunto y una aplicación de alarmas, la que explicaremos su arquitectura, potencialidades y aplicaciones.
- La gracia de ello, es que se ha hecho sobre un hardware mínimo con muy pocos recursos, una memoria RAM mínima (solo 128 Bytes), baja velocidad de proceso comparado con procesadores de 16/32bits, pero lo suficiente como para este desarrollo.
- La verdadera ventaja es permitir bajar costos en procesos de control distribuido e implementarlos sobre un hardware de bajo costo por punto.
- Permitir que la información sea gestionada por aplicaciones sobre una RDBMS de verdad y pueda ser integrada a los sistemas de gestión de las empresas.



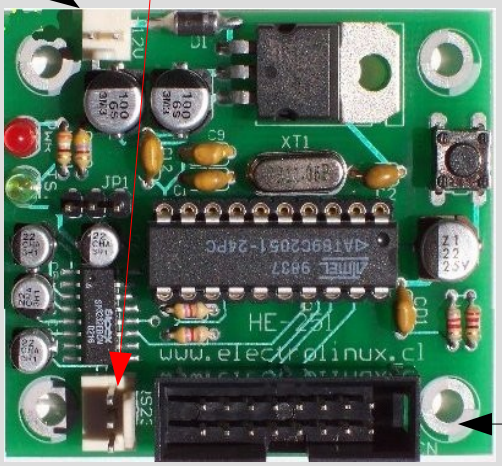
Arquitectura del sistema Distribuido

Server



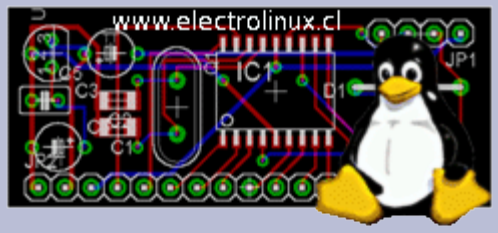
Power COMM 10/100 BT

Nodo



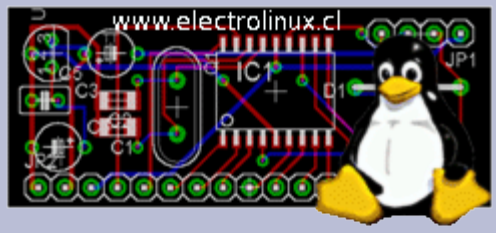
12 bits I/O

- Esta arquitectura se basa en un server con kernel Linux embebido (32 bits), que atiende requerimientos e informa de estados sobre una WEB, la que es vista sobre una red (intranet o extranet o ambas), cuidando los aspectos de seguridad.
- Este server se conecta con los uC con un bus común (RS422/RF), atendiendo a los nodos (uC) de la red interna, además se conecta a una red TCP/IP, actuando como un verdadero de gateway.
- Los nodos (uC) están compuestos por el kit HE251, el cual tiene un microcontrolador MCS51, este puede controlar hasta 12 I/O, por controlador.

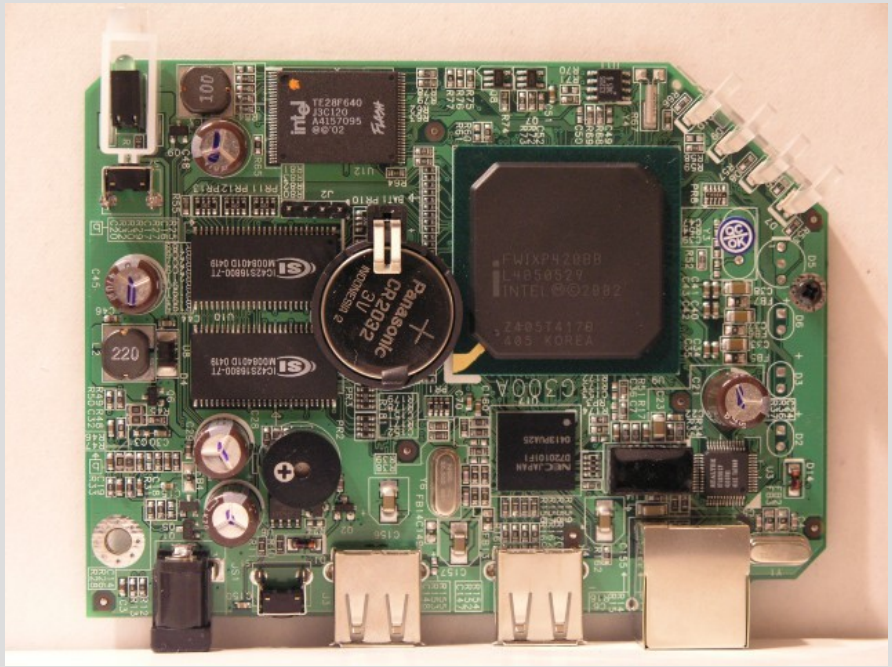


Aplicación de Administración WEB

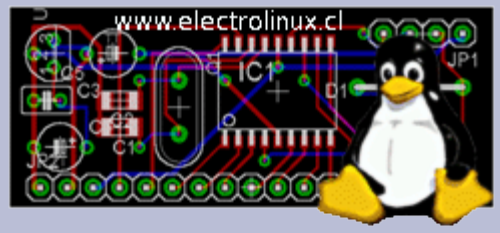
- Tradicionalmente los sistemas de Control se basan sobre aplicaciones propietarias y cerradas, al mejor y más puro estilo Hasefroch.
- Otras aplicaciones trabajan sobre BD propietarias no relacionales y con pocas herramientas de exportación de datos hacia los SIA.
- Todo esto lleva al cliente a estar cautivo permanentemente por el proveedor, con contratos de exclusividad o mantenimiento, esto aumenta costos y disminuye capacidad de desarrollo local.
- Nuestra propuesta se basa en administración vía WEB, sobre un servidor minimalista (Monkey) corriendo sobre una maquina Linux embebido.



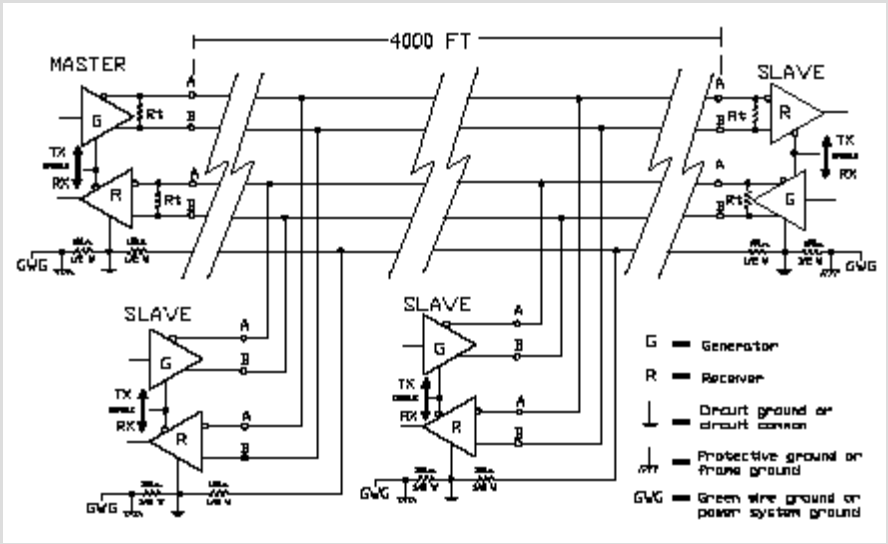
Servidor de esta Arquitectura



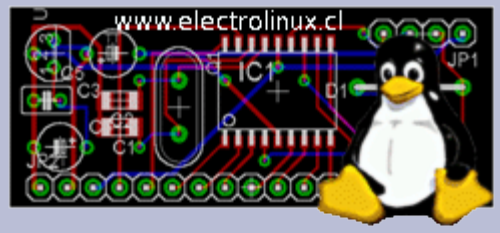
- El server se basa en un hardware de 32bits, con 64/32MB de Memoria Flash, 32MB de SDRAM.
- En esta placa se instala Linux embebido, GCC y aplicaciones desarrolladas para la tarea final.
- Se incorporan servicios de red y funcionalidades de seguridad como lo es la autentificación segura.
- Dependiendo de la capacidad, se incorporan detección de intrusos (snort) u otros similares.
- Son implementados además sistemas de chequeo de firma electrónica.
- Al usar un S.O. escalable, se puede aumentar los servicios y capacidades del hardware sin inconvenientes.



Bus de Comunicaciones Interno



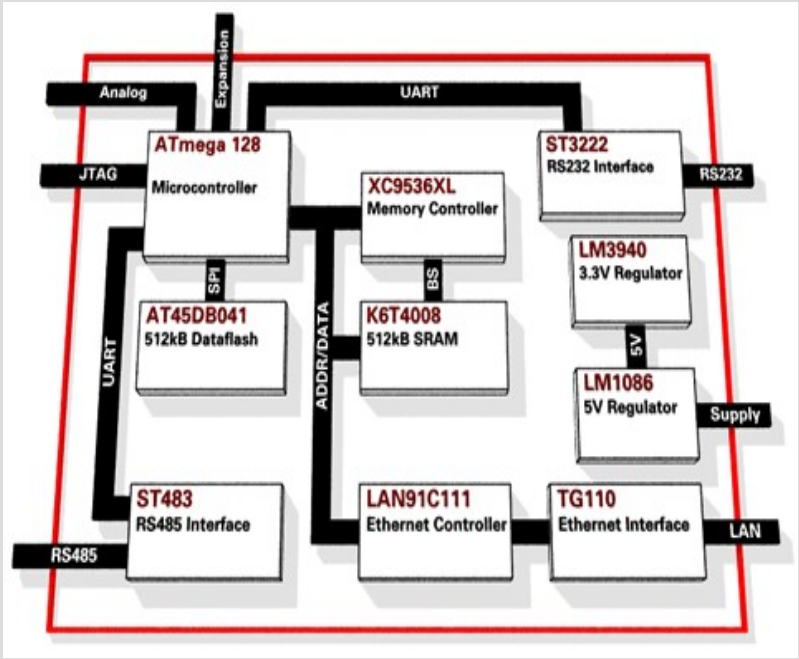
- Las comunicaciones entre el Server y los nodos (uC) se realiza sobre un BUS multipunto, normalmente RS422 para cobre, pero el medio puede ser también inalámbrico.
- Dependiendo del server se pueden integrar más buses con nodos sobre la misma máquina servidora, incluso pueden ser buses híbridos alámbrico e inalámbrico.
- Las comunicaciones soportan un ancho de banda de hasta 4Mbps, en el ejemplo disponemos de 19200bps esto nos da un tiempo de 1mS por trama, lo que es mas que suficiente para la aplicación pensada.
- Para requerimientos más exigentes se puede llegar a tiempos de ~100uS o 35uS por trama de datos.



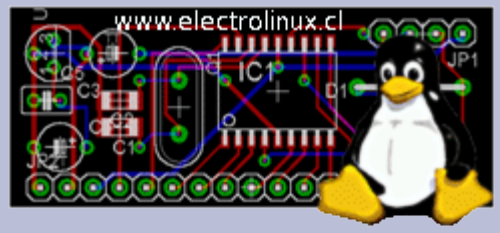
Protocolo de Comunicaciones Interno

Estructura General del Protocolo

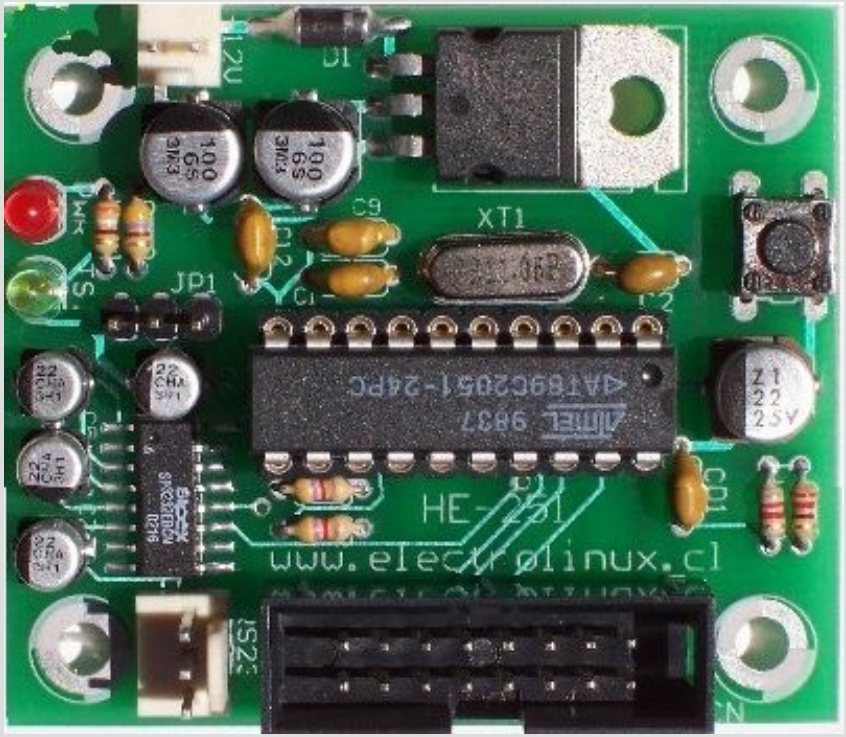
| stx | he2 | he1 | flags | data | crc | eot |



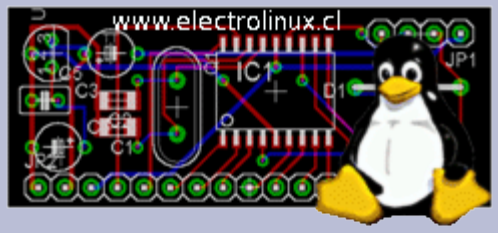
- El protocolo de comunicaciones en el que esta basado este desarrollo, es un protocolo desarrollado por Electrolinux.
- Este protocolo tiene conceptos TCP/IP, como id, direcciones de origen/destino, frame de control y data, ping, comandos de estados, basado en IRQ de uC y varias funcionalidades más.
- Una de las características importantes es que es de muy pequeño tamaño, 10 a 20 bytes típicos, pero es escalable hasta 512 byte de datos, hasta 3 byte de flags, 2 headers, orientado al bits y actúa por eventos NO es POLL-SELECT, lo que trae importantes ventajas en el área de control distribuido.
- Puede ser usado por controladores de muy escasos recursos hasta máquinas i686, con Linux o BSD.
- Permite la comunicación entre nodos y en forma jerárquica, ideal para máquinas de distintas y variadas arquitecturas, desde los 8 bits hasta los 32 bits.
- Provee identificación del tipo de máquina, modelo, familia, versión de código, fecha de fabricación, cliente y varias características más.
- Soporta uC de 8/16/32/64 bits, con arquitecturas CISC o RISC y este protocolo esta desarrollado completamente en lenguaje 'C'. Los fuentes han sido compilados desde los 8 bits a los 32 bits.



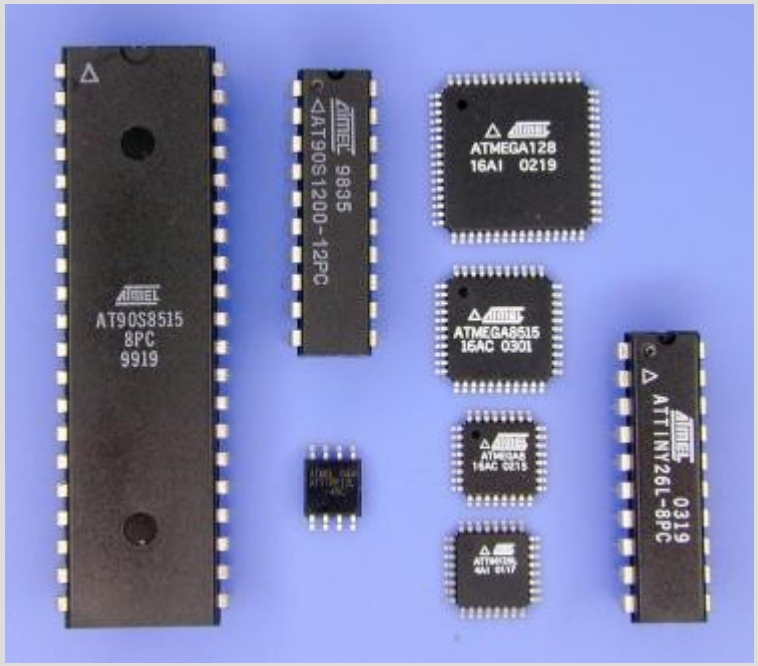
Nodos o Microcontroladores

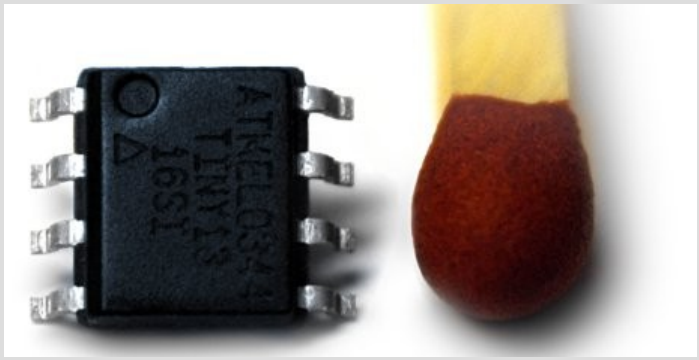


- En este ejemplo tal como se indicara anteriormente se ha basado esta demostración sobre nuestro kit más pequeño, lo que permite demostrar que es posible realizar desarrollos sobre plataformas mínimas, con las siguientes características:
  - Microcontrolador de 8 bits, familia MCS51
  - Memoria RAM interna de 128 bytes.
  - 5 Niveles de Interrupciones, serial, externas e internas.
  - Comunicaciones seriales, en este caso RS232 a 3 hilos (TX/RX/GND).
- Los sistemas de control pasaron de ser grandes microcontroladores (16 o 32 bits) a muchos uC que realizan tareas especificas pero trabajando en redes de datos, lo que facilita su administración y control.
- Un ejemplo de esto es que actualmente los automóviles pasaron de tener un solo controlador central a tener alrededor de los 200 uC orientados a distintas y especificas funciones, como la medición consumo de combustible, carburación, aceleración, control del sistema de frenos ABS, Aire/Acondicionado, temperatura, climatización, alza vidrios, señalización, luces, alarmas, etc...

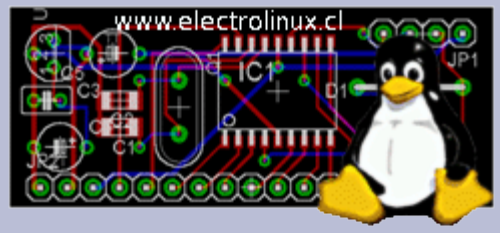


Nodos o Microcontroladores II





- Los uC actuales están orientados al control industrial, esto permite disponer de herramientas de hardware que antes debían ser implementadas externamente en el diseño, tales como:
  - Watchdog/Timer, Power-fail, esto permite evitar que los uC se 'cuelguen', esto evita fallas en el sistema de control.
  - Mayor capacidad de proceso y velocidad que llegan a 24MHz/16MIPS para máquinas de 8 bits, impensado hace 8 o 10 años.
- Actualmente se disponer de mejores herramientas open-source que hace algunos años, tal como mejores compiladores de assembler y lenguajes de alto nivel como 'C'.
- Esta capacidad nos permite embeber en estos pequeños uC, criterios de control y condiciones de borde las que permiten que estas unidades puedan tomar decisiones definidas por este criterio utilizado y/o el tipo de proceso productivo.
- Además permiten implementar conceptos de inteligencia artificial distribuida como “granjas de uC” debido a las actuales capacidades de proceso de estas pequeñas unidades.



Tendencias en el Control Distribuido

- UNIX es un S.O. que tiene como concepto de diseño en su desarrollo, que se define como un conjunto de programas que hacen tareas especificas, en forma robusta y eficiente.

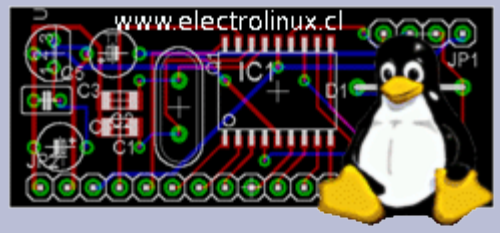
- Este concepto lo hemos adoptado e implementado en un sistema de control distribuido, el cual lo hemos complementado con la administración remota vía WEB, esto permite orientarlo a una administración abierta, sin tener que obligar al usuario a usar un S.O. específico.

- Actualmente los sistemas de control se basan en sistemas operativos confiables y escalables, nadie que implemente soluciones de control piensa en M$, solo se usan sistemas de Tiempo Real (**R**eal **T**ime **L**inux) en los casos que se requieran o sistemas UNIX en general. El tiempo ha demostrados que los sistemas de control basados en M$ han sido un completo fracaso por la mala calidad de este.

- La “Inteligencia distribuida”, ofrece ventajas de procesamiento en el lugar del evento, es decir los datos y/o conversiones de variables físicas se hacen en el lugar de captura del dato, esto permite realizar al vuelo en el lugar un proceso o tomar una decisión inmediata, evitando el envío de datos hacia el Computador Central para que este tome la decisión y responda. Esta capacidad permite implementar distintos criterios de control dependiendo del proceso en cuestión y actuar por evento producido.

- Este concepto tiene ventajas sobre sistemas con muchas variables o muchos estados simultáneos, permitiendo soportar fuertes cargas de proceso sin problemas.

- El sistema es llevado finalmente a una BD relacional para gestionar la completa administración del sistema, pudiendo interactuar con información de gestión administrativa (SIA).

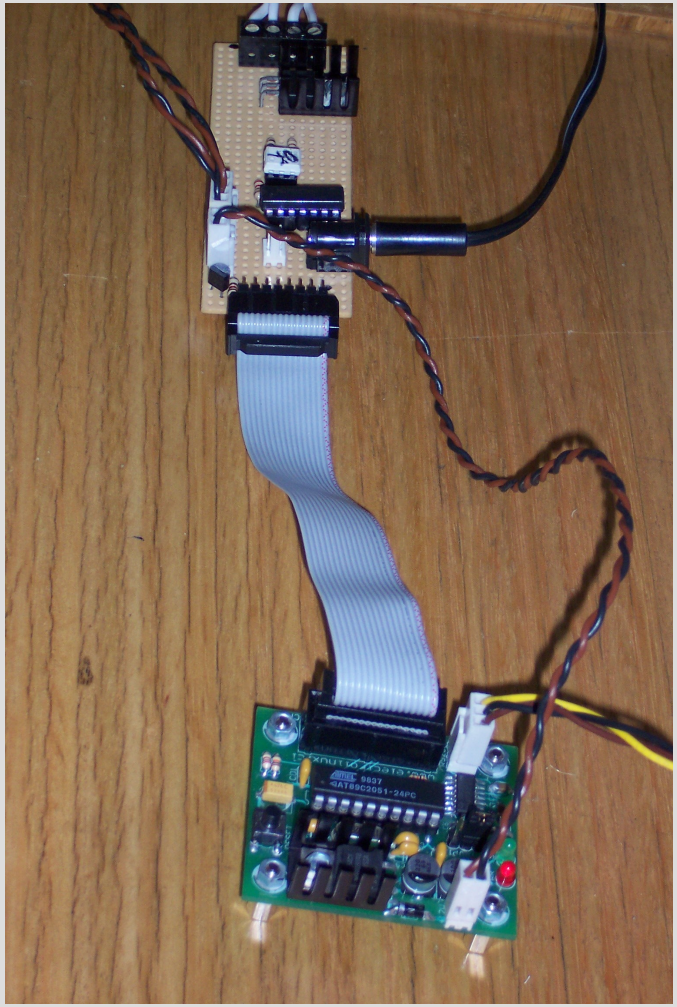


Aplicaciones de Control Distribuido

Ejemplos en Alarmas domésticas:

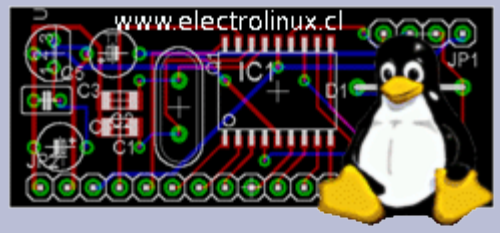
- Los Sistemas de Control Distribuido, normalmente son aplicados a la industria, esto se debe casi exclusivamente a su alto costo en su implementación.
- Nosotros lo hemos llevado a lo cotidiano lo hemos querido usar en aplicaciones domésticas y en este caso particular lo hemos orientado a un sistema de alarmas, lo que escapa del concepto tradicional.
  - Actualmente las alarmas domiciliarias tienen un solo flujo de datos, desde el sensor hasta la unidad central y de allí al proveedor del servicio. Esto hasta ahora es lo normal.
  - Una de las falencias más importantes de las alarmas son el actuar frente a una causa en forma fija, es decir actuar siempre igual frente al evento el cual puede ser la activación de un sensor, por lo que no hay ningún análisis de este, esto lleva a activaciones “falsas”, lo que hace que nadie le tome atención a la alarma y sea simplemente un contaminante acústico, perdiendo efectividad y sentido.
  - Los sistemas de imagen remota (cámaras web) solo 'muestran' el extremo remoto, pero nadie actúa sobre este, lo que frente a un intruso solo es factible el envío de personal al lugar remoto, cosa que en muchos casos es tarde y caro.
- Frente a estos problemas, los hemos resuelto por la incorporación de un mayor desarrollo tecnológico, actualmente con este sistema es posible actuar sobre el extremo remoto, pudiendo manipular artefactos remotamente sin tener que enviar personas para ello.

Aplicaciones de Control Distribuido II



Ejemplos en Alarmas domésticas:

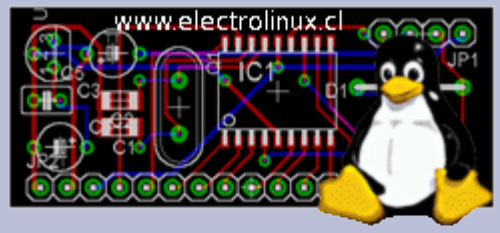
- En este caso hemos querido poner un ejemplo didáctico para que sea comprendido fácilmente.
- El hardware mostrado tal como se indicara se ha hecho sobre nuestro kit más pequeño el HE251, con unos sensores y actuadores simples.
  - Con esto se quiere demostrar que se pueden desarrollar proyectos complejos con simples componentes al alcance de cualquier persona.
  - La aplicación es embebida en el uC y desarrollada en lenguaje 'C'.
  - Sistema con muy pocos recursos de hardware disponibles (8bits/128B SRAM).
  - Comunicaciones punto a punto (RS232) y multipunto (RS422), en este ejemplo hemos usado RS232, para efectos de la demostración.



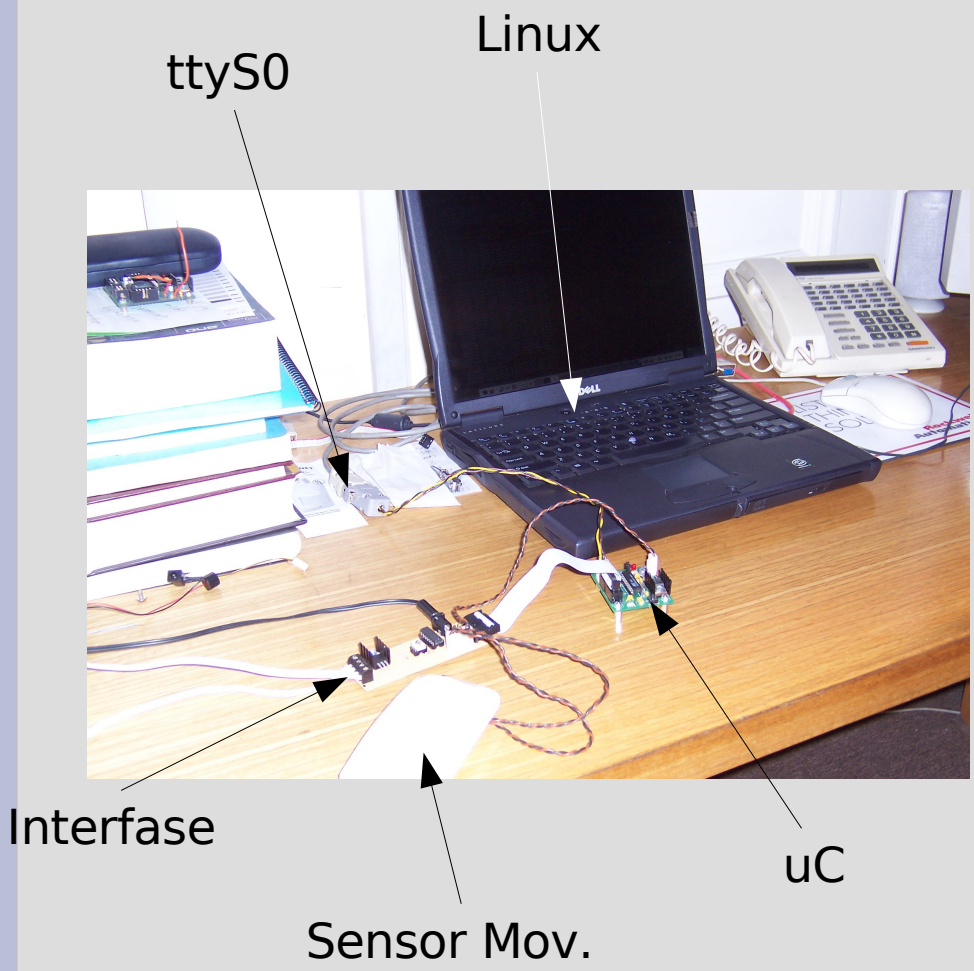
Aplicaciones de Control Distribuido III

Otras posibles aplicaciones:

- Monitoreo de estaciones remotas, como antenas repetidoras de sistemas de comunicaciones, permitiendo detectar variables físicas (banco de baterías, temperatura, detección de intrusos, niveles de combustible, estados de circuitos de tableros eléctricos).
- Sistemas de vigilancia remotas, con activación de sistemas electromecánicos a distancia, como grupos generadores, sistemas de riego en la industria agrícola, sistemas de iluminación, motores eléctricos, sistemas de detección de incendios y muchas otras posibilidades de control a distancia.
- Sistemas de control de acceso en base a sistemas de tarjetas de identificación en linea basados en BD relacionales.
- Captura de información en procesos industriales en linea, para gestión de la información sobre SIA's.
- Sistemas de gestión de seguridad comunitarias, como edificios, condominios y grandes superficies que se requieran vigilar con baja dotación de personal.
- Y en general lo que la imaginación pueda sobre esta base tecnológica.

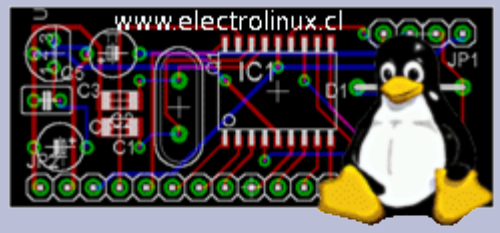


Alarma con el Control Distribuido



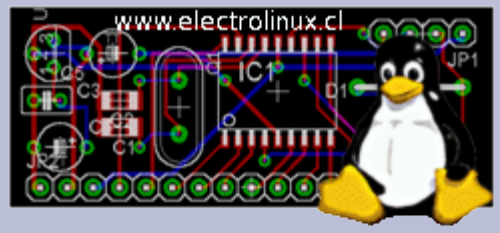
Partes de este ejemplo:

- En este ejemplo se tiene un pequeño microcontrolador con un detector de movimientos convencional conectado al uC (HE251), un sistema de alarma de ruido usando un beep suficientemente molesto, un sistema de activación de luz indicado por un led verde.
- Aplicaciones embebidas desarrolladas en 'C' corriendo sobre este uC. Emite un sonido al detectar movimiento y el server recibe el mensaje de violación de la seguridad.
- Como server se tiene una maquina con Linux, pudiendo ser implementada perfectamente sobre un hardware embebido tal como se mostró en las diapositivas anteriores.
- Se muestra la aplicación en texto a fin de que se entienda mejor y ver el ejemplo.
- Programas en 'C' corriendo por el lado del server.



Demostración del ejemplo

Demostración práctica de la charla



Agradecimiento a los Organizadores

Ahora pasamos a sus consultas y comentarios.

Espero que esta charla les haya agradado y sea el punto de partida para que desarrollen sus propios proyectos.

Agradecemos a la UNAP como organizadores de este Encuentro Nacional de Linux 2005

Atentamente

Ricardo Albarracín B. rab@electrolinux.cl

Electrolinux