Sistemas Embebidos y Sistemas Operativos Embebidos

David A. Pérez A.[1]

[1] Laboratorio de Redes Móviles e Inalámbricas (ICARO) – david.perez@ciens.ucv.ve

1. Introducción

Los sistemas de computación se encuentran en cualquier parte. Es probable que nadie se sorprenda al saber que millones de sistemas de computación son producidos y vendidos cada año, como por ejemplo computadores de escritorios o computadores personales. Sin embargo, sorprende saber que billones de sistemas de computación de propósito específico son construidos y vendidos cada año; los sistemas embebidos se encuentran alrededor de nuestras vidas, en forma de teléfonos móviles, equipos médicos, sistemas de navegación aérea, reproductores MP3, impresoras, automóviles, etc. Cada vez que se mira alrededor es posible identificar un dispositivo que contiene un microprocesador, y probablemente se ha encontrado un sistema embebido. Por estas razones resulta conveniente estudiarlos.

En este documento se abordarán conceptos pertinentes a sistemas embebidos y parte del software que se ejecuta en ellos. En la Sección 2 se definen los sistemas embebidos; en la Sección 3 se describen las características de los sistemas embebidos; en la Sección 4 se discute acerca del hardware encontrado en los sistemas embebidos; la Sección 5 expone algunos principios de diseño de los sistemas embebidos, en la Sección 6 se presentan aplicaciones comunes para los sistemas embebidos, la Sección 7 muestra algunos sistemas operativos embebidos y explica algunas funcionalidades; finalmente la Sección 8 concluye este documento.

2. ¿Qué es un sistema embebido?

Existen numerosas definiciones de sistemas embebidos, algunas son:

- "Un sistema embebido es cualquier dispositivo que incluye un computador programable, pero en sí mismo no es un computador de propósito general" [15].
- "Un sistema embebido es un sistema electrónico que contiene un microprocesador o microcontrolador; sin embargo, no pensamos en ellos como un computador" [6].
- "Las personas usan el término sistema embebido para referirse a cualquier sistema de cómputo escondido en algún producto o dispositivo" [8].
- "Un sistema embebido es un sistema cuya función principal no es computacional, pero es controlado por un computador integrado. Este computador puede ser un microcontrolador o un microprocesador. La palabra embebido implica que se encuentra dentro del sistema general, oculto a la vista, y forma parte de un todo de mayores dimensiones" [14].

Un sistema embebido posee hardware de computador junto con software embebido como uno de sus componentes más importantes. Es un sistema computacional dedicado para aplicaciones o productos. Puede ser un sistema independiente o parte de un sistema mayor, y dado que usualmente su software está embebido en ROM (Read Only Memory) no necesita memoria secundaria como un computador. Un sistema embebido tiene tres componentes principales:

1. Hardware.
2. Un software primario o aplicación principal. Este software o aplicación lleva a cabo una tarea en particular, o en algunas ocasiones una serie de tareas.
3. Un sistema operativo que permite supervisar la(s) aplicación(es), además de proveer los mecanismos para la ejecución de procesos. En muchos sistemas embebidos es requerido que el sistema operativo posea características de tiempo real.

Es importante resaltar que el software que se ejecuta en un sistema embebido es diseñado bajo algunas restricciones importantes: (i) cantidades pequeñas de memoria, generalmente en el orden de los KB, (ii) capacidades limitadas de procesamiento, generalmente los procesadores poseen velocidades que no superan los Mhz, (iii) la necesidad de limitar el consumo de energía en cualquier instante, bien sea en estado de ejecución o no.
En la Figura 1 se muestra la anatomía de un sistema embebido típico [5]. Este diagrama muestra la arquitectura de hardware a alto nivel de un punto de acceso inalámbrico. Algunos de los componentes de hardware vistos en el diagrama serán explicados en las próximas secciones; pero básicamente el sistema se centra en un procesador RISC (Reduced Instruction Set Computer) de 32 bits. La memoria FLASH es utilizada para almacenamiento de datos y programas de forma persistente. La memoria principal típicamente cuenta con algunos megabytes o cientos de megabytes, en los cuales se almacenan valores temporales para la ejecución de programas. En este ejemplo también se observa una interfaz Ethernet y una USB, un puerto serial y un chip que contiene la implementación del estándar IEEE 802.11.

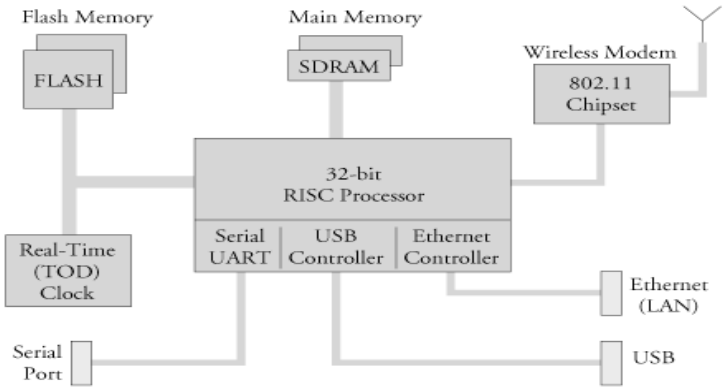

Figura 1. Ejemplo de un sistema embebido

3. Características de un sistema embebido

Los sistemas embebidos poseen ciertas características que los distinguen de otros sistemas de cómputo, a continuación estudiaremos las más importantes [12]:

1. Funcionamiento específico. Un sistema embebido usualmente ejecuta un programa específico de forma repetitiva. Por ejemplo un **pager**, siempre en un pager. En contraste, un sistema de escritorio ejecuta una amplia variedad de programas, como hojas de cálculo, juegos, etc.; además nuevos programas son añadidos frecuentemente. Por supuesto pueden haber excepciones, podría ocurrir que el programa del sistema embebido fuese actualizado a una nueva versión. Por ejemplo, un teléfono celular podría actualizarse de alguna manera.
2. Fuertes limitaciones. Todos los sistemas de computación poseen limitaciones en sus métricas de diseño, pero en los sistemas embebidos son muy fuertes. Una métrica de diseño es una medida de algunas características de implementación, como: costo, tamaño, desempeño, y consumo de energía. Los sistemas embebidos generalmente deben ser poco costosos, poseer un tamaño reducido, tener un buen desempeño para procesar datos en tiempo real, y además consumir un mínimo de energía para extender el tiempo de vida de las baterías o prevenir la necesidad de elementos adicionales de enfriamiento.
3. Reactivos y tiempo real. Muchos sistemas embebidos deben ser reactivos o reaccionar ante cambios en el ambiente, además de realizar algunos cálculos en tiempo real sin ningún retraso, es decir, se deben tener resultados en tiempos fijos ante cualquier eventualidad. Por ejemplo, el módulo de control de viaje de un automóvil continuamente monitorea la velocidad y los sensores de frenos, reaccionando ante cualquier eventualidad. Ante un estímulo anormal, el módulo de control debe realizar los cálculos de forma precisa y acelerada para garantizar la entrega de los resultados dentro de un tiempo límite, una violación en este tiempo podría ocasionar la pérdida del control del automóvil. En contraste, un sistema de escritorio se enfoca en realizar cálculos con una frecuencia no determinada y la demora de los mismos no producen fallas en el sistema.

4. Hardware de sistemas embebidos

El término hardware en cualquier sistema se refiere a los componentes físicos que lo forman o constituyen; estos componentes permiten realizar un conjunto de tareas al ejecutar programas o software.

Los componentes físicos de un sistema embebido por lo general difieren en algunos aspectos de los que conforman un sistema de propósito general, como un computador de escritorio en: tamaño, capacidad de cómputo, requerimientos de energía, etc. Por esta razón es de gran importancia conocer el funcionamiento del hardware para poder desarrollar sistemas embebidos y las aplicaciones que se ejecutarán en él. A continuación se estudiarán los aspectos más relevantes del hardware comúnmente utilizado en sistemas embebidos.

4.1 Microprocesadores

La tecnología de procesadores trata de la arquitectura del núcleo computacional usado para implementar las funcionalidades deseadas de un sistema [12]. En particular, un microprocesador "es un componente LSI que realiza una gran cantidad de funciones o tareas en una sola pieza de circuito integrado" [16].

El término LSI (Large Scale Integration) se refiere a la tecnología que permite integrar varios miles de transistores en un solo circuito integrado. Otra definición de microprocesador podría ser "componente LSI que asocia en una sola pieza de circuito integrado las funciones de una unidad aritmético-lógica y la unidad de control asociada" [16].

Las métricas de diseño pueden variar de un procesador a otro de acuerdo a su funcionalidad, por ejemplo la compresión de imágenes. En la Figura 2 se pueden observar estas variaciones. La cruz muestra la funcionalidad deseada y en la parte inferior se observa como diferentes procesadores se ajustan a ella.

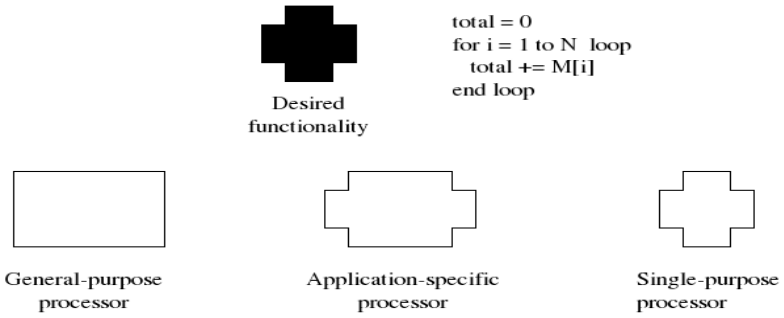

Figura 2. Diferencias entre las funcionalidades de un procesador

4.1.1 Procesadores de propósito general

Un procesador de propósito general o microprocesador es un dispositivo programable adaptable a una gran variedad de aplicaciones. Una característica de ellos es que poseen memoria para la ejecución de programas, este componente es necesario ya que no se conoce a priori cuál programa será ejecutado. Otra característica es que poseen un camino de datos genérico, conformado por varios registros y una o varias ULAs (Unidad Lógica Aritmética). Todo esto hace posible la ejecución de aplicaciones de diversos propósitos [5].

4.1.2 Procesadores de propósito específico

Un procesador de propósito específico es un circuito digital diseñado para ejecutar exactamente un programa. Por ejemplo un codificador/decodificador JPEG, el cual ejecuta un programa que comprime y descomprime marcos de video. Un sistema embebido frecuentemente utiliza este tipo de procesadores, ya que se ajusta completamente a las funcionalidades requeridas. A menudo este

tipo de procesadores es denominado coprocesador, acelerador y/o periférico [2].

4.1.3 Procesadores específicos de aplicación

Un procesador específico de aplicación (ASIP – Application Specific Instruction set Processor) es un compromiso entre las opciones anteriores. Un ASIP es un procesador programable optimizado para una clase particular de aplicaciones que comparten características comunes, tales como procesamiento de señales digitales o telecomunicaciones. De este modo es posible optimizar el camino de datos para una clase de aplicaciones en particular, al agregar unidades funcionales especiales para aplicaciones comunes y eliminar otras de poco uso [2].

4.2 Microcontrolador

"Un microcontrolador es un circuito integrado que consiste en muchas de las mismas cualidades que una computadora de escritorio, tales como CPU (Central Process Unit), la memoria, etc., pero no incluye ningún dispositivo de comunicación con humanos (monitor, teclado, etc.)" [4].

Un microcontrolador es un tipo de procesador ASIP, usado desde hace varias décadas [12]. Un microcontrolador es un microprocesador que ha sido optimizado para aplicaciones de control embebidas. Estas aplicaciones típicamente monitorean y fijan numerosas señales de control de un único bit, pero no realizan cálculos exhaustivos.

Así, los microcontroladores tienden a tener caminos de datos sencillos que se destacan en operaciones a nivel de bit, para su lectura y escritura a nivel externo. Además tienden a incorporar en el chip del microprocesador varios componentes periféricos comunes en el control de aplicaciones, tales como: periféricos de comunicación serial, temporizadores, contadores, modulador de amplitud de pulso, etc. Esta incorporación de periféricos permite la implementación de un único chip, lo que genera productos de menor costo. Algunos ejemplos de microcontroladores son los MCS-48 y MCS-51 de Intel.

4.3 Sensores

Un sensor es un dispositivo eléctrico y/o mecánico que convierte magnitudes físicas en valores medibles de dicha magnitud. Los sensores van a aportar información tanto del entorno como del estado interno del componente que mide [2].

La señal medida usualmente debe transformarse para poder ser interpretada. Este proceso se realiza en tres fases:

- Un fenómeno físico es captado por un sensor, como consecuencia, muestra en su salida una señal eléctrica equivalente al fenómeno captado.
- La señal eléctrica es modificada por un sistema de acondicionamiento de señal, cuya salida es un voltaje que será convertido usando un convertidor analógico/digital (A/D) para ser tratado.
- El convertidor A/D es sensible sólo a rangos limitados de tensiones, frecuentemente 0 a 5V. El convertidor hace que la salida continua se convierta en una salida discreta.

En sistemas embebidos suelen utilizarse diversos tipos de sensores como por ejemplo, sensores de luz, sensores de contacto, sensores de temperatura, etc.

4.4 Memoria

Cualquiera de las funcionalidades de los sistemas embebidos están compuestas de tres aspectos: procesamiento, almacenamiento y comunicación. El procesamiento es la transformación de los datos, el almacenamiento es la retención de los datos para su posterior uso, y la comunicación es la transferencia de los datos. Cada uno de estos aspectos debe ser implementado. Se usan procesadores para el procesamiento, memoria para el almacenamiento, y buses para la comunicación [12]. En esta sección se describe el elemento memoria.

4.4.1 Organización de memoria

El elemento básico de una memoria es la celda. Aunque se utilizan diversas tecnologías electrónicas, todas las celdas de memoria comparten ciertas propiedades [9]:

- Presentan dos estados estables, que pueden emplearse para representar el 1 y 0 binarios.
- Puede escribirse en ellas (al menos una vez) para fijar su estado.
- Pueden leerse para detectar su estado.

La Figura 3 describe el funcionamiento de una celda de memoria [9]. Lo más común es que la celda tenga tres terminales para transportar señales eléctricas. El terminal de selección selecciona la celda, para que pueda realizarse una operación de escritura o de lectura. El terminal de control indica si la operación se trata de una lectura o una escritura. Para la escritura el tercer terminal proporciona la señal que fija el estado de la celda a 1 ó 0. En la lectura el tercer terminal se utiliza como salida del estado de la celda.

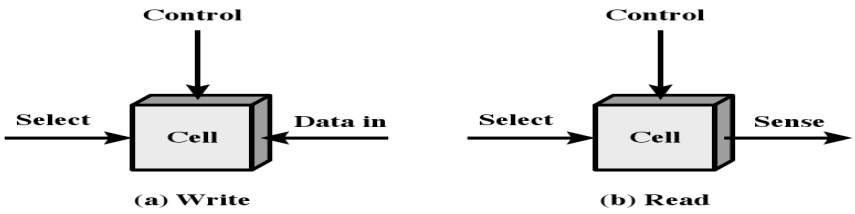

Figura 3. Funcionamiento de una celda de memoria

4.4.2 Tipos de memoria

A continuación se describen los dos tipos de memoria (RAM y ROM), junto con algunas de sus variantes.

4.4.2.1 Memoria RAM

Una de las características distintivas de las RAM (Random Access Memory) es la posibilidad de leer datos, como escribirlos rápidamente. La otra característica distintiva es que una RAM es volátil. Una RAM debe estar siempre alimentada por corriente eléctrica, si se interrumpe la alimentación se pierden los datos [9].

Las dos formas básicas de memoria de acceso aleatorio son la RAM dinámica (DRAM – Dynamic RAM) y la RAM estática (SRAM – Static RAM). Una DRAM está hecha con celdas que almacenan los datos como cargas eléctricas en condensadores. La presencia o ausencia de carga en

un condensador se interpreta como el 1 ó 0 binarios. Ya que los condensadores tienen una tendencia natural a descargarse, las DRAM requieren refrescos periódicos para mantener memorizados los datos. Una SRAM es un dispositivo digital, basado en los mismos elementos que se usan en el procesador. En una SRAM los valores binarios se almacena utilizando **flip-flops**, los cuales mantienen sus datos en tanto se mantengan alimentados.

Otras formas de RAM son la RAM pseudo estática (PSRAM – Pseudo Static RAM) y la RAM no volátil (NVRAM – NonVolatile RAM) [12]. La PSRAM es una DRAM con un controlador refrescador de memoria embebido. La NVRAM es una variación especial de la RAM que es capaz de mantener datos, incluso luego de que se remueve la alimentación de poder. Esto se logra almacenando los valores en una memoria EEPROM (explicada en la siguiente sección), justo antes de perder la alimentación.

4.4.2.2 Memoria ROM

Una memoria de sólo lectura (ROM – Read Only Memory) contiene un patrón permanente de datos que no puede alterarse. Una ROM es no volátil. Además suele ser utilizada para almacenar subrutinas de bibliotecas para funciones de uso frecuente, programas del sistema, etc.

Existen varios tipos de memoria ROM, entre los cuales se encuentran [12]:

- ROM programada por máscara. Las conexiones son programadas cuando el chip es fabricado, creando un conjunto apropiado de máscaras.
- OTP ROM (One Time Programmable ROM). Esta puede ser programada por un diseñador en un laboratorio, después de la manufactura del chip. La OTP ROM más básica usa un fusible para conexiones programables, dichos fusibles son fundidos para la representación del valor 0 en binario.
- EPROM (Erasable Programmable ROM). Este dispositivo usa un transistor como componente programable. Esta se lee y escribe electrónicamente, pero para el borrado de la misma es necesario exponerla a radiación ultravioleta para reasignar carga negativa a todas las celdas.
- EEPROM (Electrically Erasable Programmable ROM). En esta memoria se puede escribir en cualquier momento sin borrar todo el contenido anterior, ya que permite actualizaciones a nivel de byte. El borrado de los datos se realiza mediante un voltaje específico a los bytes de memoria direccionados.
- Flash. Esta memoria se introdujo a finales de los 80s como una mejora de las memorias EEPROM. Su funcionamiento es similar a la de su antecesora, con una mejora significativa en la velocidad de lectura y escritura.

5. Principios de diseño de un sistema embebido

El diseñador de un sistema embebido debe construir una implementación que satisfaga la necesidad deseada. Pero una dificultad adicional es construir una implementación que simultáneamente optimice diversas métricas de diseño. Una implementación consiste en un microprocesador con un programa que lo acompañe, una conexión de compuertas digitales, o una combinación de ambos. Una métrica de diseño es una característica medible de la implementación del sistema. Métricas comunes incluyen [12]:

- Costo NRE (Nonrecurring Enginnering). El costo monetario de diseñar el sistema por primera vez. Una vez que el sistema está diseñado, cualquier número de unidades puede ser manufacturado sin incurrir en costo adicionales de diseño.

- Costo unitario. El costo monetario de manufacturar cada copia del sistema, excluyendo el costo NRE.
- Tamaño. El espacio físico requerido para el sistema, usualmente medido en bytes para software, y compuertas o transistores para hardware.
- Desempeño. El tiempo de ejecución del sistema.
- Energía. La cantidad de energía consumida por el sistema, que puede determinar el tiempo de vida de la batería.
- Flexibilidad. La habilidad de cambiar la funcionalidad del sistema sin incurrir en un gran costo NRE.
- Tiempo para crear un prototipo. El tiempo necesario para construir una versión funcional del sistema.
- Tiempo de mercadeo. El tiempo requerido para desarrollar un sistema al punto en que pueda ser lanzado al mercado.
- Mantenimiento. La habilidad de modificar el sistema luego de su lanzamiento inicial, especialmente por diseñadores que no participaron en el diseño original.
- Correctitud. Confianza en que la funcionalidad del sistema es la correcta.
- Seguridad. La probabilidad de que el sistema no cause daño.

Al diseñar un sistema embebido deben tenerse en cuenta las métricas mencionadas; sin embargo, las métricas típicamente compiten entre sí. La mejora de una métrica generalmente conlleva al degrado de otra. Por ejemplo, si se reduce el tamaño de una implementación su desempeño puede decaer, tal como se muestra en la Figura 4.

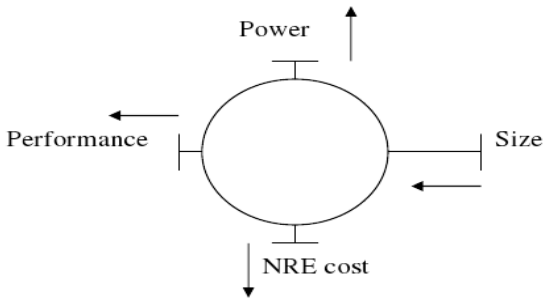

Figura 4. Competencia de métricas de diseño

6. Aplicaciones de un sistema embebido

Los sistemas embebidos se encuentran en una variedad de dispositivos electrónicos comunes, tales como consumibles electrónicos (teléfonos celulares, pagers, cámaras digitales, video juegos portátiles, calculadoras, PDAs, etc.), electrodomésticos (hornos microondas, máquinas contestadoras, termostatos, lavadoras, etc.), equipos de oficina (fax, copiadoras, impresoras, scanners), equipos de negocios (caja registradora, sistemas de alarma, lectores de tarjeta y cajeros automáticos), y automóviles (control de transmisión, control de viaje, inyección de combustible, ABS, etc.). Podría decirse que prácticamente cualquier dispositivo que se ejecute con electricidad o ya tiene un sistema computacional embebido o próximamente lo tendrá [12].

7. Sistemas operativos embebidos

Un sistema operativo es un programa que se ejecuta continuamente en un dispositivo, brindando una capa de abstracción para los usuarios facilitándole el uso del dispositivo; además de ocultar el hardware del sistema y encargarse de la administración de sus recursos [7][10].

Un sistema operativo embebido es un sistema operativo que se ejecuta sobre un sistema embebido, los cuales han sido descritos previamente. Los sistemas operativos embebidos generalmente se ejecutan sobre dispositivos que difieren de un computador común, como televisores, hornos microondas, y teléfonos móviles. Usualmente tienen algunas características de sistemas de tiempo real, pero a la vez tienen restricciones de tamaño, memoria y energía que los hacen especiales [10].

Algunas definiciones alternativas de sistemas operativos embebidos se listan a continuación:

- "Algunos sistemas embebidos incluyen un sistema operativo, que se conoce como sistema operativo embebido. Este puede ser un sistema de software muy pequeño desarrollado específicamente para ser usado con un algún sistema embebido en particular, o en ocasiones puede ser una versión reducida de algún sistema operativo que se utiliza en una computadora de propósito general" [11].
- "Un sistema operativo es definido como una capa de software que permite multiplexar abstracciones de hardware como: memoria volátil, ciclos de procesador, dispositivos de entrada salida, etc., para los programas de aplicación. Un sistema operativo embebido debe realizar las operaciones expuestas anteriormente, pero en un ambiente donde los programas de aplicaciones poseen numerosas restricciones, particularmente en cuanto a consideraciones de tiempo y energía" [1].
- "Un sistema operativo para un sistema embebido usualmente es diseñado para una aplicación específica, y por lo tanto es más estático que un sistema operativo de propósito general" [3].

7.1 Casos de estudio

En la Tabla 1 se muestra un resumen de algunos sistemas operativos embebidos vigentes en la actualidad, resaltando características de interés.

Nombre	Gratuito	Código abierto	Última actualización	Uso Principal
VxWorks	No	No	Feb, 2009	Modems, Firewalls, Wireless routers
QNX	No	No	May, 2009	Dispositivos de tiempo real
Windows CE	No	No	Nov, 2007	PDAs, Smartphones
Windows Mobile	No	No	May, 2009	PDAs, Smartphones, Dispositivos multimedia
eCos	Sí	Sí	Mar, 2009	Dispositivos de tiempo real
LynxOS	No	No	Sep, 2006	Aeronaves, automóviles
TinyOS	No	Sí	Ago, 2008	Redes de sensores, motes
Palm OS	No	No	Sep, 2006	PDAs, Smartphones
Garnet OS	No	No	May, 2009	PDAs, Smartphones
Linux Embebido	Sí	Sí	Ago, 2006	PDAs, Smartphones, Wireless routers
Symbian	No	No	Oct, 2008	Dispositivos Móviles (Smartphones)
Inferno	No	Sí	Feb, 2007	Sistemas distribuidos de tiempo real

Tabla 1. Ejemplos de Sistemas Operativos Embebidos vigentes en la actualidad

Para entender de forma clara el funcionamiento de un sistema operativo embebido, se estudiará como algunas implementaciones realizan el manejo de abstracciones básicas del sistema. En particular, se estudiarán los sistemas QNX, Linux Embebido y Windows CE [13].

QNX es un sistema operativo embebido desarrollado por QNX Software Systems Ltd, para aplicaciones de dispositivos electrónicos, telecomunicaciones, sistemas automotrices, etc.; que necesitan una gran confiabilidad, desempeño, funcionalidades específicas, y una escalabilidad masiva.

Linux embebido es un sistema Linux típico del cual se han removido programas de utilidad, herramientas, y otros servicios del sistema que no son necesarios en un ambiente embebido.

Windows CE fue introducido en un conjunto de productos de PC de mano en el año 1996, pero posteriormente se transformo en un sistema operativo embebido altamente configurable. Entre sus ventajas se encuentran que incluye un subconjunto del API Win32 enfocado a los servicios que comúnmente se necesitan, así como una optimización en el consumo de energía.

7.1.1 Arquitectura

Los sistemas operativos embebidos usan una arquitectura de microkernel o una arquitectura modular, esto los hace fácilmente adaptables para ajustarse a diferentes requerimientos de aplicaciones.

QNX contiene un microkernel muy pequeño rodeado por un conjunto de procesos cooperantes, que proveen servicios de alto nivel. El microkernel de QNX implementa cuatro servicios: 1) Comunicación entre procesos. 2) Comunicación de red a bajo nivel. 3) Planificación de procesos. 4) Manejo de interrupciones. Los procesos de servicio de sistema operativo son opcionales, y el usuario puede elegir cuales son necesarios para sus aplicaciones.

La arquitectura de Windows CE es jerárquica. En el fondo se encuentran los drivers del dispositivo. Sobre ellos está el subsistema de gráficos, ventanas y eventos, el kernel y la pila de comunicaciones. Sobre el kernel se ubica el sistema de archivos. Las aplicaciones se ejecutan en su propio espacio de direcciones e interactúan con el resto de Windows CE vía llamadas al API de Win32.

Linux embebido posee una estructura en capas complementada con módulos. Estas capas generalmente son el kernel de Linux, el sistema de archivos, los drivers de dispositivos y los protocolos de red. Linux embebido toma el kernel de Linux y extrae los módulos que no son necesarios. Dentro de la capa del kernel, Linux embebido está compuesto de cinco grandes subsistemas: el planificador de procesos, el manejador de memoria, el sistema de archivos virtual, la interfaz de red, y la comunicación entre procesos.

7.1.2 Manejo de procesos

En QNX el manejador de procesos no reside en el microkernel QNX. Aunque comparte el mismo espacio de direcciones que el microkernel, el manejador de procesos se ejecuta como otro proceso planificado por el microkernel, utilizando las primitivas de pase de mensajes para comunicarse con otros procesos en el sistema. El manejador de procesos es responsable de la creación de procesos y del manejo de sus recursos, además de tomar las siguientes decisiones:

- Atender a un proceso que sale del estado bloqueado.
- Gestionar la expiración de la ranura de tiempo de ejecución asignada a un proceso.
- Desalojar a un proceso del procesador.

En QNX cada proceso posee una prioridad y el planificador los ordena por esta.

Windows CE soporta tanto procesos como hilos. La planificación de hilos es llevada a cabo de forma apropiativa usando ranuras de tiempo, además de asignar ocho diferentes niveles de prioridad a los hilos. Esto implica que Windows CE es un sistema operativo multitarea apropiativo, permitiendo a múltiples aplicaciones o procesos ejecutarse a la vez (máximo 32 procesos).

En Linux embebido el hilo es la entidad de planificación, distinguiendo tres clases: 1) Los hilos FIFO (First In First Out) de tiempo real tienen la mayor prioridad y no son apropiativos. 2) Hilos round robin de tiempo real, los cuales son apropiativos; 3) Hilos de tiempo compartido con menor prioridad. Cada hilo tiene una prioridad de planificación; y además los hilos 2 y 3 tienen un quantum asociado. Linux planifica los hilos con un algoritmo que decide ejecutar el hilo con mayor prioridad, decrementando el quantum del hilo una unidad en cada ejecución.

7.1.3 Comunicación entre procesos

QNX utiliza el pase de mensajes para la comunicación entre procesos. Las primitivas soportadas por QNX copian directamente los datos de proceso a proceso sin la necesidad de colas o mecanismo intermedios, esto permite que el desempeño en el pase de mensajes se aproxime a las capacidades de ancho de banda de la memoria. En los sistemas de pase de mensajes tradicionales, usualmente los datos ocupan un área contigua en memoria, las primitivas de **mensajes multipartes** de QNX permiten que el mensaje resida en porciones no contiguas aprovechando así la memoria.

Windows CE posee soporte para pase de mensajes entre procesos, además de permitir mapeo o compartición de memoria entre procesos. Esta característica hace que la comunicación entre procesos cooperantes sea muy rápida, lo que resulta deseable para ciertas aplicaciones con límites de tiempo.

Linux embebido puede incorporar todos los mecanismos típicos de pase de mensajes de un sistema Linux convencional. Algunos de estos mecanismos son:

- Señales. Mecanismo para la emisión de mensajes asíncronos entre procesos.
- Pipes (Tuberías). Primitiva que permite una comunicación orientada a la conexión bidireccional, entre dos procesos.
- System V IPC (Semáforos). Mecanismo de comunicación entre procesos utilizado para controlar accesos a regiones compartidas.

En Linux embebido se puede escoger entre diversos mecanismo de comunicación entre procesos (algunos nombrados anteriormente); claro está si en la construcción del sistema operativo se incluyeron dichas funcionalidades, como parte del kernel o como módulo.

7.1.4 Manejo de memoria

La mayoría de los sistemas operativos modernos convencionales usan memoria virtual paginada, donde la página es la unidad de protección y asignación de la memoria. El uso de procesos y protección de memoria en sistemas embebidos es muy importante; si se utiliza un único espacio de direcciones para todas las aplicaciones, una falla de software de una aplicación puede resultar en la corrupción de la memoria, ocasionando una falla de sistema. La desventaja, sin embargo, es que la

protección de memoria requiere que el CPU soporte MMU (Memory Management Unit), lo cual resulta en un CPU más complejo.

A diferencia de los sistemas operativos convencionales, la mayoría de los sistemas operativos embebidos están enfocados a un CPU simple, que usualmente no tiene MMU. Además el resto del sistema tiene memoria limitada, poco o ningún espacio en disco; así que usualmente no usan memoria virtual.
Por ejemplo, los sistemas Linux embebido y QNX no soportan paginación, esto quiere decir que los datos, texto y pila comparten un espacio contiguo de memoria. Esto implica que no existe protección a nivel de memoria, por ejemplo, la pila podría crecer hasta ocupar el espacio de texto o datos; o un proceso podría leer o escribir datos de otro proceso.

Por otro lado, Windows CE tiene un manejo de memoria más elaborado. Soporta parcialmente memoria virtual paginada (esto implica soporte de TLB (Translation Lookaside Buffer) en el CPU; y brinda protección de memoria a nivel de procesos e hilos.

7.1.5 Soporte de red

El soporte de red es importante en sistemas embebidos ya que les facilita comunicarse con el mundo exterior, así como su actualización. QNX contiene comunicación de red de bajo nivel en su microkernel; Windows CE posee varias pilas de comunicación a nivel de kernel (IP, PPP, IrDA, etc.).

Linux embebido por su parte posee un soporte de red heredado de los ambientes cliente/servidor soportados por Linux, ofreciendo pilas de red y protocolos de Internet.

8. Conclusiones

El crecimiento de los sistemas embebidos en los últimos años ha ocasionado que estos sean más populares y más complejos. Esta complejidad requiere que la comunidad que trabaja sobre sistemas embebidos posea sólidos conocimiento de los componentes de hardware y software usados en los mismos; ya que de esta manera el diseño de cualquier solución será concebido considerando las métricas y restricciones particulares de dichos sistemas.

Durante este documento se han estudiado aspectos claves de los sistemas embebidos, en particular del software que se ejecuta en ellos, tal como los sistemas operativos embebidos. Resulta interesante el hecho de conocer los cambios incorporados en las funciones básicas de un sistema operativo (procesos, administración de memoria, etc.) cuando se ejecutan sobre un sistema embebido, lo cual es consecuencia directa de las fuertes limitaciones de hardware o métricas que poseen estos sistemas. El estudio de esta nueva tendencia en los sistemas computacionales abrirá paso a un nuevo campo de investigación donde las limitaciones de cómputo, espacio y energía serán tópicos de gran importancia.

Referencias

[1] Chen-Mou, C. “An Operating System Architecture for Embedded Systems – Design and Implementation”. Department of Electrical Engineering. National Taiwan University. 1998.

[2] Díaz, E. et al. “Introducción al Diseño de Microrobots Móviles”. Universidad de Alcala. 2006

[3] Friedrich, L. “A Survey on Operating System Support for Embedded Systems Properties”. Departamento de Informática e Estatística.

[4] Gilliland, M. “What is a Microcontroller”. Parallax. 1999.

[5] Hallinan, C. “Embedded Linux”. Prentice Hall. 2006.

[6] Morton, T. “Embedded Microcontrollers”. Prentice Hall. 2000.

[7] Silberschatz, et al. “Operating System Concepts”. Seventh Edition. John Wiley & Sons. 2005.

[8] Simon, D. “And Embedded Software Primer”. Addison-Wesley Professional. 1999.

[9] Stallings, W. “Organización y arquitectura de computadores”. 7a Edición. Prentice Hall. 2006.

[10] Tanenbaum, A. “Sistemas Operativos Modernos”. 2da Edición. Prentice Hall. 2001.

[11] The Linux Information Project (LINFO). “Embedded System Definition”. 2006. http://www.linfo.org/embedded_system.html

[12] Vahid, F. & Givargis, T. “Embedded System Design”. John Wiley & Sons, Inc. 2002.

[13] Wang, C. “A Survey of Embedded Operating System”. 2005.

[14] Wilmshurst, T. “An Introduction to the Design of Small Scale Embedded Systems with examples from PIC, 80C51 and 68HC05/08 Microcontrollers”. Palgrave Foundations. 2003.

[15] Wolf, W. “Computers as Components: Principles or Embedded Computing System Design”. Second Edition. Morgan Kaufmann. 2008.

[16] Zazks, R. “Microprocessors: From Chips to Systems”. Sybex Inc. 1981.