High Availability Linux JUST KEEP ON RUNNING

LARRY G. CRUZ

In the telecommunications industry computer systems are required to exceed 99.999% (five nines) availability. The Motorola Computer Group is a major manufacturer of embedded system platforms for this market and now offers Linux as one of the operating systems that may be used. Larry G. Cruz describes the architecture of this hardware and the changes and additions that were made to Linux so as to make it into a high availability operating system.

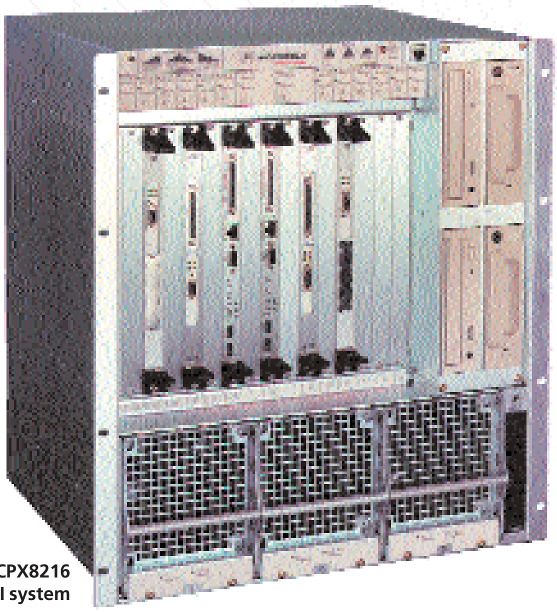

Fig. 1: A Motorola CPX8216 CompactPCI system

The Motorola Computer Group (MCG) manufactures a variety of embedded system platforms for the telecommunications industry. The CPX8000 family of CompactPCI systems are application ready platforms that are designed to meet five nines availability. CompactPCI provides the primary bus architecture for these platforms, but additional hardware and software is required to provide a system that can be deployed in critical telecommunication infrastructure applications. One of the operating systems offered for these systems is the Linux operating system.

The CPX8216 systems are Network Equipment Building System (NEBS) and European Telecom Standard (ETSI) compliant platforms. They have built-in redundancy enabling them to be configured and programmed for Hot Swap and High Availability environments. Standard features include:

- Dual 8 slot CompactPCI(r) backplanes (H.110 bus optional)
- Hot-Swappable CPU and I/O boards
- PowerPC or Pentium II processors
- Up to three 350W Hot-Swappable power supplies
- Three Hot-Swappable fans
- Four Hot-Swappable drive bays
- NEBS compliant, Hot-Swappable alarm status display panel
- Hot Swap Controller providing CompactPCI(r) bus and slot control compliant with PICMG 2.1 rev 1.0 CompactPCI(r) Hot Swap Specification

The CPX8216 has a dual host architecture (Fig. 2). The two independent CompactPCI(r) bus segments form redundant system domains that are cross connected via bridges. In a simple Active Standby I/O configuration the bridges are configured such that CPU-A controls all twelve I/O slots in the system. If CPU-A were to fail the bridges would then be configured such that CPU-B would have control of all twelve I/O slots.

The ability to switch from the active CPU to the standby CPU is obviously useful in the case where the CPU hardware or software fails. An additional benefit is the ability to perform CPU, software and firmware upgrades without taking the system down. To do this the CPU, software or firmware is first upgraded on the standby CPU. When completed the standby CPU takes control of the I/O bus segments and becomes the active CPU. The previously active, and now standby, CPU is then upgraded.

Driven by the appropriate software, the CPX8216 can be configured and maintained at varied degrees of complexity and granularity ranging from simple Hot Swap implementations to Highly Available, "five nines" systems.

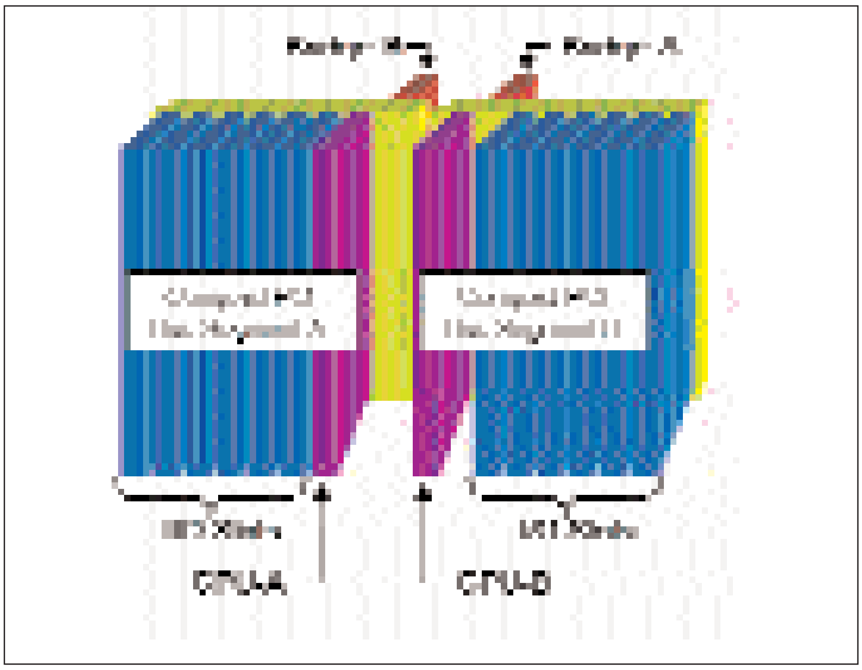

Fig. 2: CPX8216 Dual Host Architecture

Hot Swap and High Availability

Fundamental to the need for Hot Swap and High Availability is the premise that the ability to add and remove components from a live system reduces system downtime. A system without Hot Swap capabilities must first be shut down and powered off prior to the addition, removal or replacement of system boards and components. Systems with Hot Swap capabilities range in their ability to reduce this downtime from hours of system unavailability per year to less than five minutes per year. Predictably, as the availability of a system increases so, too, does the complexity of the hardware and software of that system.

Although definitions abound for the terms "Hot Swap" and "High Availability" PICMG has characterised three different levels of Hot Swap capabilities: *Basic Hot Swap*, *Full Hot Swap* and *High Availability*. Each level builds on the capabilities of the prior level to increase system availability.

Basic Hot Swap provides the fundamental capability to add and remove boards from an active system. The staged (differing length) pins in a CompactPCI(r) connector cause some pins to make connection to the bus before others when inserting a board. The reverse occurs when a board is removed. Special circuitry is provided that allows boards to be inserted and removed from an active bus without causing signal or DC power glitches. CompactPCI(r) boards conforming to the PICMG CompactPCI(r) specification PICMG 2.0 R2.1 are electrically Hot Swappable.

Systems supporting Basic Hot Swap are fairly simple in their implementation and require operator intervention and direction to perform the Hot Swap activity. The operator must first access the system console and direct the system to, as gracefully as possible, stop using or "de-configure" the board to be removed. Once the operator sees that application and operating system software have terminated their use of the board the operator can instruct the board to disconnect from the bus and power off. The board can then be removed and replaced. After this the operator must reverse these steps to complete the replacement of a board into the system.

Full Hot Swap extends the Basic Hot Swap model with additional hardware and software. If the operating system can be notified automatically in advance of a board's removal or insertion then the operating system can automatically de-configure or configure the board without requiring operator direction. This would reduce both the opportunity for mistakes and the amount of time required to perform the Hot Swap activity.

To provide this capability a microswitch is added to the lower injector/ejector handle of the CompactPCI(r) board. To remove a board an operator must first press down on the lower ejector handle. This action activates the microswitch and causes the ENUM interrupt to be generated. The operating system must then field the ENUM interrupt, identify the board that is about to be removed and gracefully de-configure the board. An LED then lights on the face of the board to tell the operator that it is safe to complete the removal of the board. As a board is inserted into the bus the microswitch is activated and the ENUM interrupt is processed. In this case the operating system must determine the type of board being inserted and then configure the board for use. The LED on the face of the board is then turned off to indicate to the operator that the board has been accepted into the system.

High Availability systems significantly expand the scope of the operating system's involvement in handling events occurring in the system chassis. Middleware (user level software outside the O/S kernel) is added to manage the configuration and allocation of system resources. This middleware also provides an interface to user applications through published APIs.

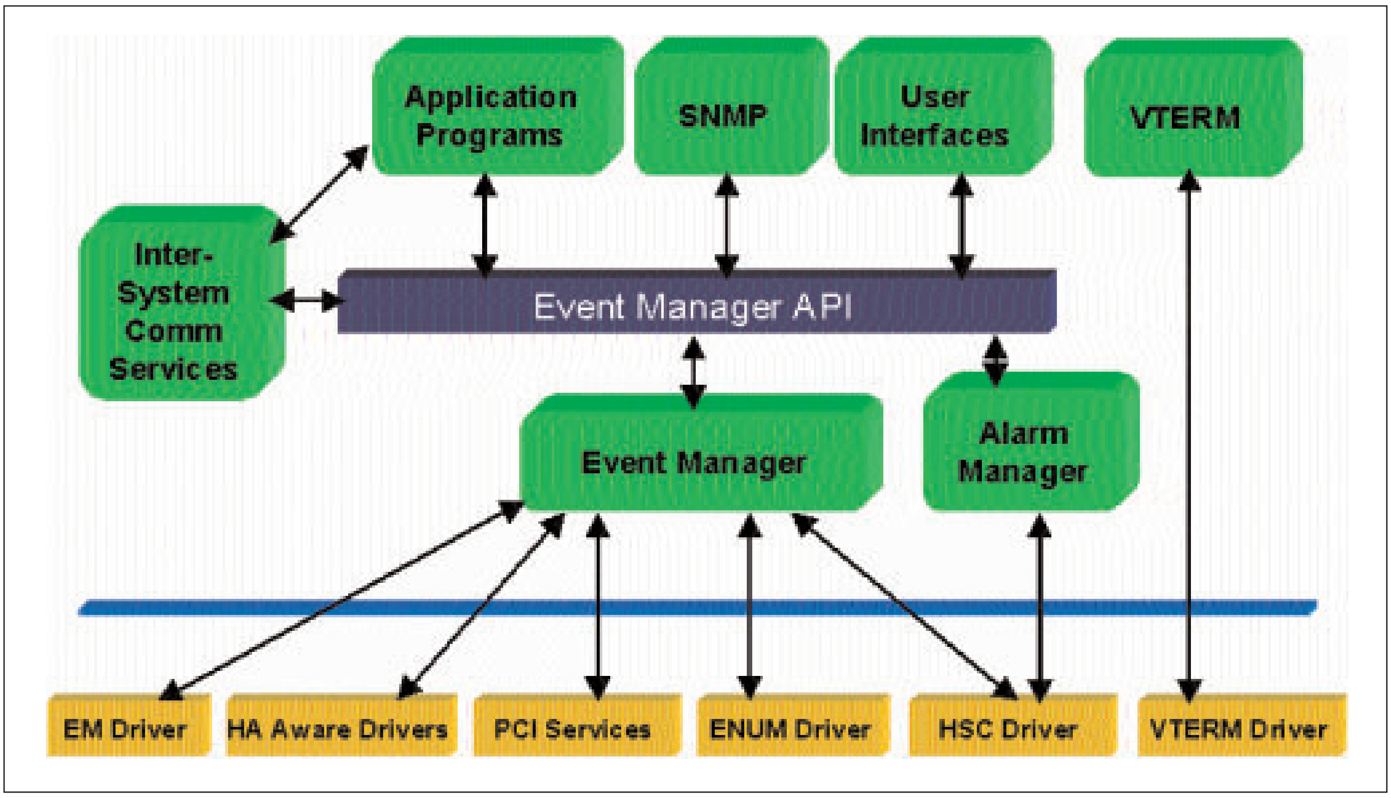

Fig. 3 HA-Linux Block Diagram

In a High Availability (HA) system software can control the state of chassis components via the addition of the Hot Swap Controller (HSC). With this hardware addition automation and fine granularity of status and control are possible. The HSC developed by Motorola provides CompactPCI(r) bus and slot control as well as chassis alarm and status registers and is itself Hot Swappable.

High Availability Linux

In recent months there has been explosive growth in the use of Linux in commercial applications. Motorola's decision to develop High Availability software for the CPX8000 family of platforms using Linux was driven by many factors. Not only does Linux provide a full featured OS with proven reliability, but the open source model enables lower cost, greater control and simplified licensing. Linux also fitted in well with Motorola's existing service and support model for UNIX systems.

"HA-Linux" is the name given to the collection of Motorola's added value extensions and enhancements to Linux that implement High Availability features. Motorola's HA-Linux is not "yet another" new Linux distribution. In fact, one of the design goals of HA-Linux was to create as portable an implementation as possible. The CPX8000 HA platforms are available in configurations using both PowerPC and Intel(r) processors. Today, HA-Linux runs on two Linux distributions: Red Hat 6.1/6.2 for Intel processors and a derivative of LinuxPPC for PowerPC processors.

Fig. 3 presents a high level block diagram of the HA-Linux components. HA-Linux consists of enhancements to the Linux kernel, additional kernel modules, user level programs, utilities and APIs. A CPX8000 system with HA-Linux in place is ready for customer applications to be added to run in a High Availability environment.

HA-Linux Components

The following is a brief description of the HA-Linux components. Motorola is open sourcing all components covered by the GNU General Public License (GPL). Additionally, Motorola intends to publish any component integrated into the kernel including drivers that are not covered by the GPL.

PCI Kernel Services

Motorola has developed a revision of PCI kernel services for 2.2.x versions of the Linux kernel. Since PCI devices are removed from or inserted into an HA-Linux system the operating system must be able to dynamically maintain data structures that represent the current state of the bus. The 2.2 version of the Linux kernel does not provide a model for the dynamic allocation and de-allocation of PCI bus resources. Motorola's revision of PCI kernel services provides this capability through enhancement of the existing GPL code.

ENUM Driver

The ENUM driver is implemented as a kernel module and is responsible for the asynchronous event notification of Hot Swap activity. The ENUM# interrupt is generated when the lower ejector handle of a CompactPCI(r) board is pressed down. The interrupt is also generated when a board is inserted into the bus. The ENUM driver processes this interrupt and determines the source of the interrupt.

Hot Swap Controller Driver

The Hot Swap Controller driver is implemented as a kernel module and supplies an interface to the HSC. The HSC controls all of the system infrastructure and provides asynchronous event notification of chassis component state changes. Access to user level applications is through a published API that provides these facilities:

- Visual and audible alarms on the chassis alarm panel can be set or queried for status.
- Individual board slots can be reset, powered off or on or queried for status.
- Disk drives can be powered off or on.
- Cooling fan speed can be adjusted, powered off or on.
- Power supplies can be monitored, powered off or on.
- Bridges can be configured for domain control.
- Chassis environmental conditions can be monitored.

Although the API for the HSC is published, most user applications will not drive the HSC directly. Instead, higher level HA-Linux software will interact with the HSC on behalf of the user application.

HA Aware Drivers and the EM Driver

HA-aware drivers are either existing drivers that have been enhanced or new drivers that have been written to conform to the HA-Linux "HA Driver Specification." An HA-aware driver adheres to the PCMCIA specification enabling drivers to cope with the appearance and disappearance of individual devices. Drivers may also be enhanced with fault detection and diagnostic capabilities. If applicable, these drivers might also be configured to perform low level device failover. For example, the ethernet driver can be configured such that it views two physical connections as one active and one standby connection. If the driver detects that the active connection has failed it can switch activity to the standby connection prior to notifying the system of the failure and thereby reducing the possibility of data loss.

The Event Manager driver is implemented as a kernel module and provides an interface to the Event Manager for HA aware drivers.

VTERM Driver and Utilities

The VTERM driver and utilities implement a very useful and "nifty" feature in an HA-Linux system. Through VTERM a virtual terminal interface can be established between the CPU board and any of the I/O boards to interact with the I/O board's firmware (BIOS) via the PCI bus. User applications can interact with, configure and control the firmware of an I/O processor board. Diagnostics can be run, MAC addresses can be read, boot options can be configured and network booting can be initiated. In short, almost anything that can be done via firmware at the I/O board's console port can be done via VTERM.

Event and Alarm Managers

The Event Manager is a user level process that serves as the brains of an HA-Linux system. The Event Manager is responsible for the system configuration and event management of the system. The Alarm Manager in conjunction with the Event Manager controls the system LEDs and alarms. The Event and Alarm Managers are easily configured and extended by users to meet their specific application needs.

SNMP

The Simple Network Management Protocol (SNMP) agent included in HA-Linux is an implementation of the open source UCD SNMP v3 agent. The agent supports MIB-2 and UCD agent extensions for processors, disks, memory, load average, shell commands and error handling. HA-Linux extends the SNMP agent with a MIB for the Event Manager.

ISCS

The Inter-System Communication Services (ISCS) process provides a method for data communication between the two domains in the CPX8000 chassis. ISCS provides a robust interface between the two domains with redundant serial connections and a network interface. ISCS is used by both the Event Manager and user applications to communicate between domains. Through published APIs, application to application messaging and data transfer capability is available to all applications. Utility programs for file transfer, remote program execution and logging are included in HA-Linux.

System Configuration and Event Management

The primary purpose and function of HA-Linux is to provide system configuration and event management capabilities within the chassis. The software components described above work together to achieve this. There are four major functions of System Configuration and Event Management:

System Setup

- Configure system and application objects at boot time or on a running system.
- Possibly combine physical devices into logical devices comprised of redundant sets of physical devices.

Respond to Change

- Automatically reconfigure the system after a failure.
- Take actions specified by subsystems when the system state changes.
- Notify a system administrator or service center of a failure.
- Execute operator requests to add or remove objects and/or devices.
- Execute operator requests to re-integrate objects and/or devices following repair or replacement.

Display System State

- By illuminating lights or activating alarms on the system alarm panel.
- By illuminating lights or indicators on the object or device.
- By updating the display of a graphical user interface.

Maintain System History

- Keep a record of system configuration and events in system or remote logs.
- Update system configuration maintained by the standby CPU.

HA-Linux implements all of these functions, and more, relieving the application developer from having to manage low level devices within the chassis and allowing him to focus on the specifics of the application.

Achieving High Availability

Many aspects of the system come into play in order to achieve high availability within a chassis. To maximise the effectiveness in reducing system downtime, applications must be aware of and interact with HA-Linux. The amount of interaction required is of course dependent on the complexity of the application and failover models desired.

HA-Linux is built upon a collection of simple text files. Editable text files exist that:

- describe the configuration of the system;
- define objects to be managed;
- establish rules and policies to be followed when the state of an object changes;
- specify the actions to be taken based on state changes.

Each of these files is implemented using a simple scripting language. HA-Linux comes complete with configuration, definition, rules and action files for the objects that exist in a CPX8000 chassis. Users can easily modify, extend or add definitions, rules and actions for any software or hardware objects of interest to their application of the system. There is also an API for the Event Manager that when linked with applications allows them to interact with the Event Manager so as to manage configuration, control, status and event notification.

In addition to tailoring system configuration and event management, users can easily integrate the network management of the system via SNMP. All the functions that are available to programs via the Event Manager API are also available via the Event Manager MIB.

The Inter-System Communication Services can be used to send messages between applications running on the active and standby CPUs. Again, this facility is provided via simple utility programs or published APIs. The frequency, volume and type of data transmitted is entirely up to the application program. The ISCS interface can be used to implement software upgrade scripts allowing the active CPU to drive the upgrade process on the standby CPU.

Through a combination of user applications, HA-Linux, the Linux operating system and the CPX8000 platform high availability is achieved. This is essential to enable network operators to achieve the reliability, performance and scalability they require to compete in the telecommunications market.

The Future

Future releases of HA-Linux will focus on advancing the scalability and availability of CPX8000 systems. Clustering, availability management, backplane messaging and network management will be added. Although HA-Linux systems support ethernet and ATM, additional communication protocols will be supported with HA-aware drivers. As the CPX8000 family of systems grows and the underlying architecture and hardware is enhanced, HA-Linux will also grow to take advantage of these changes. The direction will always be toward achieving higher and higher availability while providing new features and functions that further enable the “application readiness” of the platform for the telecoms industry.

Summary

With HA-Linux it is possible to remove and replace system components without interrupting the operation of the system. In fact, with HA-Linux an operator can pull the active CPU from a CPX8000 system and observe the system switch to the standby CPU with minimal or no interruption of system processing. Operating System software, application software as well as CPU board firmware can be upgraded without experiencing any system downtime. Even the firmware on the active CPU can be upgraded while Linux is running. HA-Linux provides the full system configuration, event management and interfaces required to achieve in excess of 99.999% availability.

Motorola’s implementation of HA-Linux on the CPX8000 family of CompactPCI systems provides an application ready five nines platform for the telecoms industry. Motorola has developed an open system solution that makes application integration simple and easy. This high availability solution is proof that Linux is ready for deployment in these demanding environments. ■

Info

Motorola Linux website
http://www.motorola.com/computer/Linux

Motorola Computer Group
http://www.mcg.mot.com/