

Open architecture in the core of AXE

Mats Jonback and Stefan Schultz

The APZ 212 40 is the first central processor of a new generation based on industry-standard microprocessors. It introduces a new *warm-standby/hot on-demand* system principle: the CP has two independent sides, each of which contains two processors—one that serves as an instruction processing unit and one that serves as a signal processing unit—which run as a two-way symmetrical multiprocessing computer. The APZ virtual machine handles ASA execution and is the middleware that guarantees telecommunications-grade availability.

The authors describe the APZ 212 40 hardware and software, the *warm-standby* concept for high availability, and differences and similarities between this and previous APZ processors.

Introduction

Being based on industry-standard microprocessors, the APZ 212 40 central processor (CP) is a milestone in Ericsson hardware implementation. The latest in a line of central processors, the APZ 212 40 is the first central processor of a new generation, and the platform for future multiprocessor solutions (Figure 1).

The APZ 212 40 introduces a new system principle. In place of the parallel synchronous (lock-step) machine, we are introducing a *warm-standby/hot on-demand* principle. The CP has two independent sides, A and B, that are loosely coupled via a high-speed interconnect and a maintenance channel. Each side contains two processors: one that serves as an instruction processor unit (IPU), and one that serves as a signal processor unit (SPU), running as a two-way symmetrical multiprocessing (SMP) computer. The hardware and the operating system define a two-node cluster of two-way SMP computers.

The APZ virtual machine (VM), a new part of the system, handles the ASA execution and is the middleware required to achieve telecommunications-grade availability. It provides recovery and repair for hardware faults, software upgrades and the like while minimizing traffic disturbance. At the same time, Ericsson is introducing an industry-standard hardware equipment practice, cPCI, and a standard operating system. Parts of the CP core have been rewritten in standard C++ code using commercial development tools.

The APZ 212 40 will be deployed as a high-capacity central processor for all kinds of application. It offers processing capacity three times as great as the fastest APZ 212 30, and is fully backward-compatible—that is, it can run all existing AXE application software.

APZ 212 40 hardware

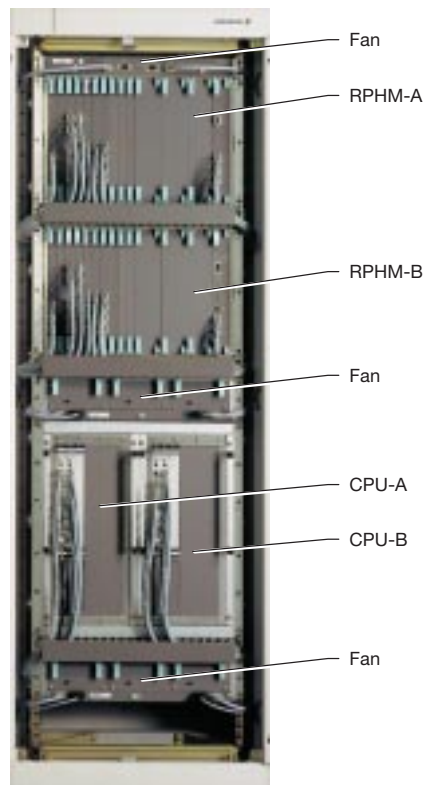
Hardware compatibility and structure

Processor technology is advancing rapidly, and Ericsson has decided to make the most of research and development in the mainstream of the computer industry by using industry-standard microprocessors. This reduces Ericsson's time-to-market, especially for upgrades, which can be made available as improvements are achieved in third-party development. At the same time, this makes it possible to concentrate in-house design efforts on developing services, improving robustness, and integrating solutions.

Each APZ 212 40 central processor magazine (CPUM, see Figure 3) contains five boards and a power module (Box A). The CPU board contains the industry-standard microprocessors. Connections are provided by

- the regional processor handler magazine

Figure 1
APZ 212 40 central processor cabinet.
Note: CDU panel not shown.



interface (RPHMI)—through cable connections;

- the base input-output (IO) board—through access ports and some general support functions; and
- the update bus board (UPBB)—through interfaces such as the new Ethernet bus and cable connections.

The new cPCI backplane is connected to the CPU backplane via a PCI interface module (PIM).

Ethernet and the interplatform network

To speed up central processor-to-APG40 communication, and thus reduce reload time, boot time, backup time, and the like, the APZ 212 40 uses direct communication

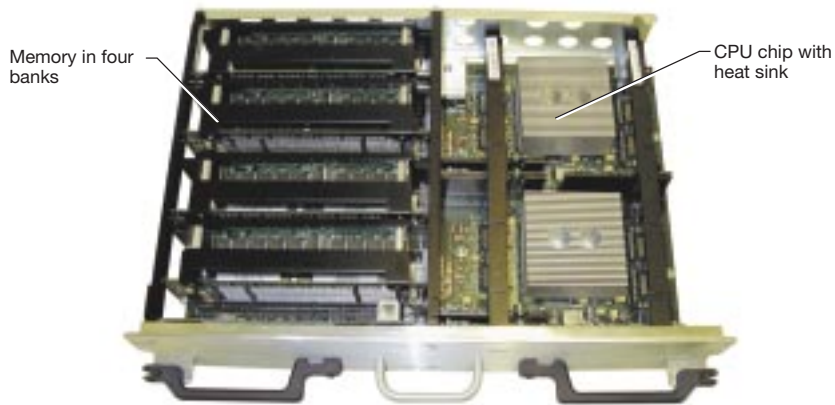


Figure 2
CPU magazine.

BOX A, APZ 212 40 BOARDS

CPU board

- 2 high-performance (GHz) microprocessors, each with 8 MB level 2 cache
- 8 GW (16 GB) SDRAM memory

RPHMI board

- Cable connections to the magazines (RPHB) for the A- and B-sides RPH-A and RPH-B
- Cable connection to the RPHMI on the other CP side for error information and for control of system states (WSB).

Base IO board

- Access port (connected to UPBB) for obtaining detailed low-level system-error information

from the microprocessor system (CPU board)

- Some general support functions for the CPU board and power module

UPB board

- One 1 Gbit/s Ethernet optical fiber connection to the other CP side for updating
- Two 100 Mbit/s Ethernet cable connections to the adjunct processor over the interplatform network (IPN), connected via the IPNX Ethernet switch in RPHM
- One 10 Mbit/s Ethernet cable connection to the adjunct processor, a processor test bus (PTB) for access with the central processor test (CPT) command interface, connected via

the IPNX Ethernet switch in RPHM

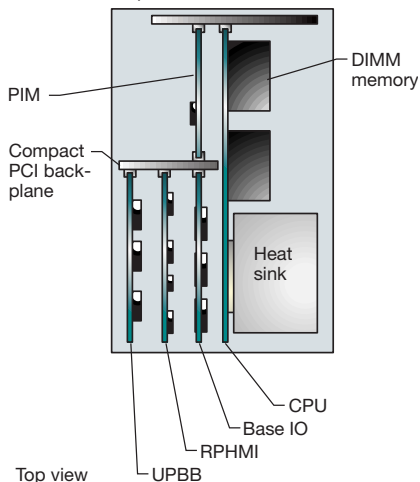
- Cable connection to the central display unit (CDU) panel at the top of the cabinet
- Cable connections to the CPU magazine fan units for monitoring and controlling the fans

The PCI interface module (PIM) card (not visible from the front) includes functionality for connecting the CPU backplane with the cPCI backplane.

Power module (in bottom left part of the CP sub-rack)

- Duplicated -48V external connection
- DC/DC converters for internal system voltages

Processor backplane



Front view

Figure 3
CPU subrack.

via Ethernet. Figure 4 shows the Ethernet switch used to bridge communication between the APG40, CP, AXD 301, and other equipment connected to the interplatform network (IPN). Communication between the central processor and APG40 uses cross-connected 100 Mbit/s Ethernet links. The two CP sides are connected by end-to-end 1 Gbit/s Ethernet link optical fiber.

The central processor test (CPT), which is used for maintenance and repair activities, is also connected to the Ethernet switch. All Ethernet connections to and from the central processor are connected to the UPBB board via the Ethernet switch. The use of Ethernet for CP-to-IO (APG40) communication also decreases the load on the regional processor handler bus (RPHB).

Cross-connected regional processor handler bus

As in previous models of the APZ, an RPHB runs between the signal processor unit and the RPHM. However, unlike previous APZs, the RPHB is cross-connected in the APZ 212 40 for redundancy. This connection is between the RPHMI board in the

central processor unit magazine and the regional processor input-output (RPIO) board in the RPHM. With the exception of the RPIO board, the two RPHMs are identical to those developed for the APZ 212 30. The RPIO board has been updated to support the cross-connection.

Of the two CP sides, one is always the executive side; the other is on standby. The executive CP side has control over one or both of the RPHMs. In a normal traffic situation, the executive side controls both RPHMs. Because one of the two RPHMs is always active (used in traffic), if an error occurs, the standby RPHM can take over immediately. The RPHMs can also be separated, and each can be assigned to its own CP side.

To avoid accidents, such as the inadvertent disconnection of the active RPHB, the state of the RPHMs is shown on the CP control and display unit (CDU) panel at the top of the cabinet.

APZ 212 40 software

Layered software structure

The structure of the APZ 212 40 can best be described as layered (Figure 5). From the bottom, we have the CPU hardware platform, with microprocessor, memory, firmware console, privileged architecture library (PAL) code, and other functions. On top of that, we find a suitable operating system that interfaces with the hardware and the application—in this case, software developed by Ericsson.

The APZ virtual machine is a PlexEngine that executes traffic. It is the interface between the existing application software and the hardware platform, similar to a microprogram. The virtual machine communicates with the CPU platform via two additional thin layers: an operating system application interface (OS API) and the hardware abstraction layer (HAL). These two layers are intended to make the APZ less dependent on a particular microprocessor architecture or operating system. The idea is that if the operating system or hardware platform is replaced, the virtual machine and the layers above it can remain more or less intact.

The APZ virtual machine accesses the resources and services provided by the underlying CPU hardware and operating system via HAL and OS API. In Figure 5, PLEX/ASA indicates where in this layered structure the APZ software units and the

BOX B, TERMS AND ABBREVIATIONS

AMU	Automatic maintenance unit	IP	Instruction processor
AOT	Ahead of time	IPN	Interplatform network
AP	Adjunct processor (external IO system used for man-machine communication, external boot media and charging output)	IPNX	IPN switch
APG40	The latest generation of adjunct processor most suitable for the APZ 212 40	JAM	Jump address memory
ASA	Programming language assembler, also machine language in earlier APZ CPs	JIT	Just in time
BOOTP	BOOT protocol	MAU	Maintenance unit
BUMS	Backup in the main store	MW 16	Megaword (1,048,576 words) 16-bit word length
CDU	CP control and display unit	OS	Operating system (commercial)
CP	Central processor (includes RPHM, CPUM and cabling)	OS API	OS application program interface
cPCI	Compact peripheral component interconnect	PAL	Privileged architecture library
CPIO	Central processor input-output	PCI	Peripheral component interconnect
CPT	Central processor test	PIM	PCI interface module
CPU	Central processor unit. In this machine, same as CPUM	PLEX	Proprietary programming language for exchanges
CPUM	CPU magazine (subrack), one CPU side	PS	Program store
DIMM	Dual in-line memory module	PTB	Processor test bus
DS	Data store	RP	Regional processor
FTP	File transfer protocol	RPHB	RP handler bus
GW 16	Gigaword (1,073,741,824 words) 16-bit word length	RPHM	RP handler magazine (subrack)
HAL	Hardware abstraction layer	RPHMI	RPHM interface board
IO	Input-output	RPIO	Regional processor input-output
		RS	Reference store
		SDRAM	Synchronous dynamic random access memory
		SMP	Symmetrical multiprocessing
		SP	Signal processor
		UPBB	Update bus board
		VM	Virtual machine
		WSB	Working state bus

rest of the application can be found. In the actual target machine, only ASA code is executed. PLEX-to-ASA compilation is done off target.

The APZ virtual machine employs the operating system application interface to communicate with the operating system and to use its functions when required. The OS API layer is intended to make the system less dependent on particular operating system platforms. If, for some reason, it becomes beneficial to change the platform, the main changes will be made in the OS API. The central processor design is based on a commercial CPU, making it feasible to use a commercial operating system that

- is available for the chosen microprocessor architecture;
- supports multiple processors;
- uses 64-bit processor architecture;
- supports dynamic-link libraries; and
- has memory protection between processes and threads.

Two ASA compilers

The APZ 212 40 CP compiles ASA binary code using two methods: just-in-time (JIT) and ahead-of-time (AOT). Thus, the APZ includes a JIT compiler and an AOT compiler to compile the ASA binary code into native CPU code. This is transparent to the user.

Just-in-time compiler

The APZ virtual machine calls the JIT compiler when there is no compiled code—that is, when neither AOT-compiled nor JIT-compiled code is available. The JIT program compiles and stores a “compilation unit,” which is a small part of the ASA code—a sequence of instructions out of which there are no jumps (short compilation time is the main focus of the JIT compiler). The next time that piece of code is called, the compiled code will be executed directly. If anything happens that affects the ASA binary code, such as a correction being inserted or a trace on an instruction address using the test system, the generated code is invalidated and must be recompiled the next time it is called.

Compared to earlier versions of APZ, the JIT-compiled code has full register coherency. Consequently, the JIT compiler is the one to use during troubleshooting. It has full support for the test system and the current correction handling—the code is JIT-compiled until the AOT compilation is finished.

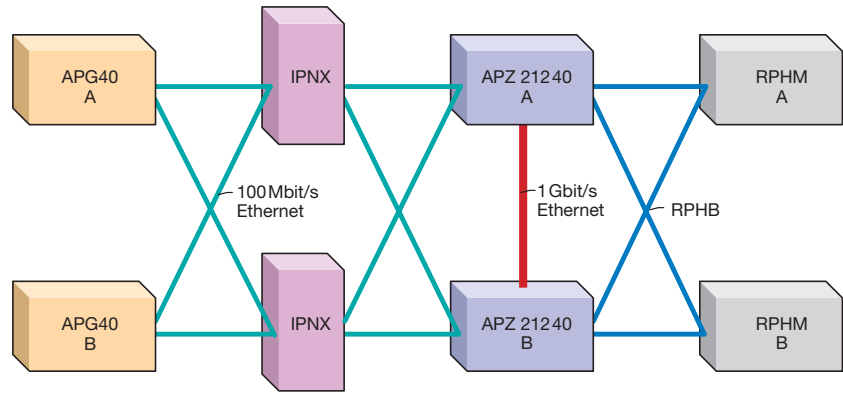


Figure 4
Ethernet links and cross-connected RPHM.

Ahead-of-time compiler

The AOT program compiles entire software units of ASA binary code in background execution, including all currently loaded ASA corrections. The AOT compilation is performed in a separate process on the SPU processor. Its purpose is to generate code that is as efficient as possible. The AOT compiler is allowed longer execution time to produce code. Thus, unlike the JIT compiler, it can optimize over an entire ASA

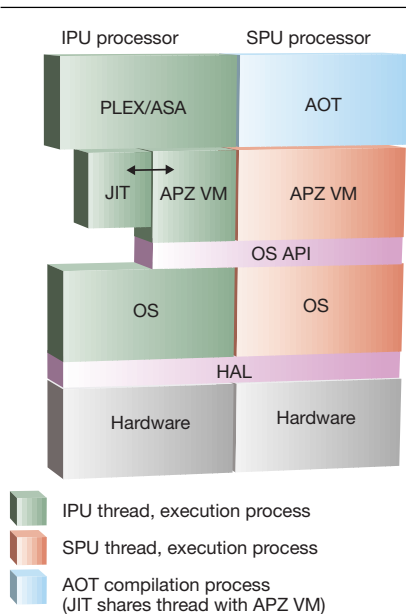


Figure 5
Central processor structure of the APZ 212 40.

software unit. Capacity is the main focus of the AOT.

The AOT program compiles only the most frequently executed software units. These are either selected by the system or decided manually—that is, they are preset. The number of blocks is estimated to be 10-15% of the total number, or around 150 blocks. This should be sufficient to cover at least 90% of the most frequently executed code, the rest is compiled using the JIT compiler.

The AOT compiler supports signal trace and some forlopp traces, but not trace on out-signal. If any other traces are requested by the test system, the JIT compiler will take over program execution. The *trace on every jump* option using the test system is no longer available.

The switch between AOT-compiled and JIT-compiled code is transparent to the user. For instance, when a trace is activated, the system switches from AOT-compiled to JIT-compiled code. When the trace is deactivated, the system automatically switches back to the AOT-compiled code.

Record-oriented data-store architecture

The APZ 212 40 introduces a record-oriented data store (DS) architecture. Record-oriented data allocation is based on the observation that after one piece of data has been read into memory, another variable in the same record (with the same pointer value) will likely be read soon after.

In contrast to existing central processors, modern microprocessors rely heavily on caching, so they spend much less time reading data if they have just read the preceding data. Thus, when the APZ 212 40 reads data, some data that follows immediately after is also read and put in the cache memory. The introduction of the record-oriented data-store architecture only affects a few PLEX or ASA software units in the APZ operating system and not the application.

APZ virtual machine

The APZ virtual machine is the compiled and linked module of C++ code that executes the application. The APZ virtual machine, which is compiled into native code off-line, is the first module loaded after the operating system when the system is powered on. This microprogram substitute takes care of the incoming signals from the regional processors (RP) and from the IPN network—it schedules, dispatches, and distributes the signals that result from the jobs. The APZ virtual machine also offers special services or support functions to the software layers above it (Figure 5).

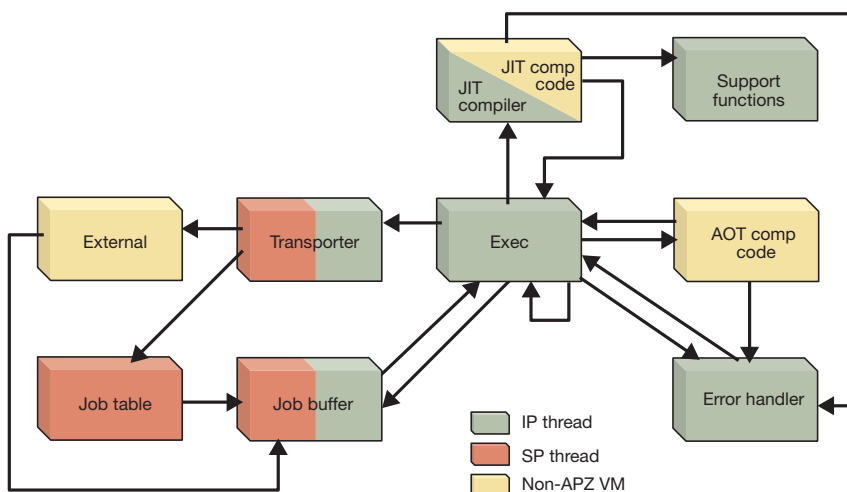
The APZ virtual machine executes in a process with two threads, referred to as the instruction processor (IP) thread and the signal processor (SP) thread. The instruction processor thread handles all job scheduling (Figure 6). The main loop of the instruction processor thread starts by polling the job buffers for incoming jobs. The jobs are then dispatched and executed. The main parts of the instruction processor thread are

- Exec—which is the virtual machine kernel, dispatcher, scheduler, and signal distributor;
- error-handling and recovery;
- the run-time log;
- the JIT compiler; and
- an interface to the AOT-compiled code.

The signal processor thread handles external communication and controls the job table. Basically, it runs in an infinite loop checking the external and IPU interfaces for new signals or messages to store in job buffers—for subsequent execution in the instruction processor thread. The main parts of the signal processor thread are

- job-table scanning;
- error-handling and recovery;
- communication support (transports and external signaling);
- RP signaling; and
- IPN signaling.

Figure 6
APZ virtual machine.



Warm-standby concept for high availability

The CPU sides do not run in parallel synchronous (lock-step) mode. Instead, a *warm-standby/hot on-demand* principle is applied. In normal traffic situations, one CP side executes and the other side is on standby (normal). The same system that is executing on the executive side is preloaded into memory on the standby side. The standby side is also regularly updated with transient data.

Likewise, the standby side is updated through the automatic backup function. When a side is scheduled to be switched or a function is scheduled to be replaced—for example, when testing, performing maintenance, or adding hardware—the standby side is “heated up”; that is, the entire memory is copied from the executive side to the standby side, so that the standby side becomes a mirror image of the executive side. A 1 Gbit/s Ethernet link (the updating bus) between the two CP sides provides the capability to heat up the standby side. The copy procedure can be divided into two phases (Figure 7): background copy (raw copy) and traffic-oriented copy (high-level copy). Memory is copied page by page using a fault-on-write mechanism. This means that a copied page is set to write protect. Then, when a write order to that page is received, the page is logged before it is written. The page is copied again later (Figure 8).

Measurements and estimates have been made for copying data between CP sides. With one 1 Gbit/s link, the capacity will drop marginally for a period less than 15 seconds. At the end of the copy interval, the system is frozen for around 0.5 seconds, allowing the CP states and the APZ virtual machine to be copied. During this interval, traffic from the RPs is buffered, so no data is lost.

If the standby side is put into operation because of a debilitating hardware fault in the executive side, a restart with reload is ordered. In the normal state, a system will already have been loaded, so what actually occurs is a nearly instantaneous reload followed by a large restart. The command log is later read from the adjunct processor without affecting downtime.

The advantage of the *warm-standby/hot on-demand* concept is in-service performance during software recovery, which fully compensates for any negative in-service performance from CPU hardware recovery. As

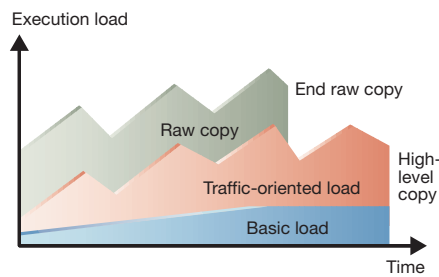


Figure 7
“Heating up.”

with previous APZs, planned events do not result in any traffic disturbance.

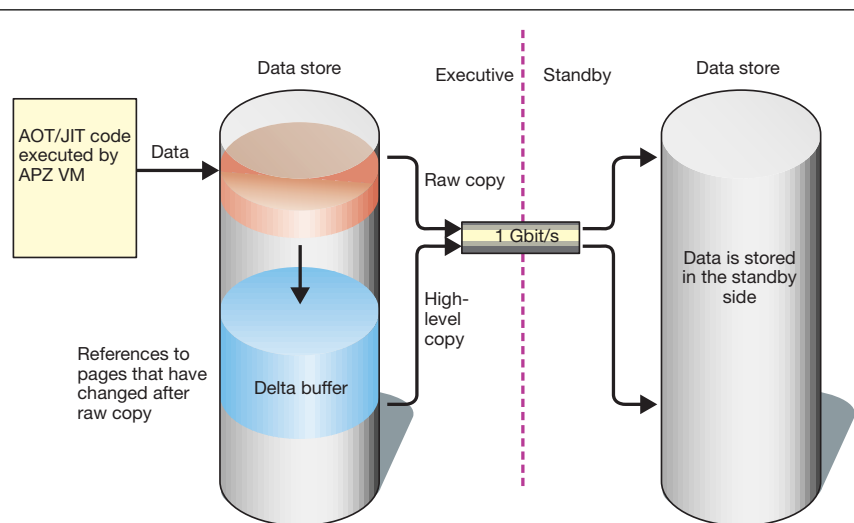
Previous APZ processors—similarities and differences

New concepts in APZ 212 40

The *warm-standby* concept, which is new to APZ design, obviates the need for maintenance hardware to detect matching errors between CP sides. Consequently, the APZ 212 40 has no separate maintenance unit (MAU).

As explained above, the APZ central processor is based on a commercial micro-processor that requires a substitute for the old

Figure 8
Data transfer to standby side.



microprogram. The substitute is found in the APZ virtual machine and the new compilers. Hence, the information that the system provides in fault situations and other specific states differs from that of earlier CP versions. For example, the jump address memory (JAM) contains different information.

An Ethernet switch (Figure 4) has been introduced to connect ports in the central processor to the adjunct processor and other systems, such as the AXD 301.

Hardware dissimilarities

The smallest replaceable hardware unit in the new central processor is the entire

CPUM—that is, a complete CP side. For the RPHM, the smallest replaceable hardware unit is the same as in previous versions: individual boards. The CP working state logic is implemented on the RPHMI board; a special working state bus (WSB) cable connects the RPHMI boards in the two CPU sides.

The memory can be configured in steps of 1 GB. The normal way of upgrading or increasing physical memory is to update a configuration file on the adjunct processor, replace the CPUM, add physical memory, and then use regular size-alteration commands to inform the applications that more memory is available. As usual, this process is ended with a backup.

A new CP control-and-display unit panel indicates which side is executing and which side is on standby. It also indicates which RPHM is handling traffic at any moment. The CDU panel indicates normal system state when the executing side is switching traffic and the standby side is ready to take over control. The standby side is ready to take over when

- it has the same system generation preloaded in system memory as the executive side; and
- when it is receiving transient data.

A new state (normal, NRM on the CDU panel) has been defined for the standby side to indicate when these criteria are satisfied.

Software dissimilarities

The C and C++ parts of the system are treated separately from the rest of the system. These separate files on the adjunct processor are loaded using BOOTP and FTP. They are not part of the system backup.

To ensure short reload time, backup in main store (BUMS) must be active. When

BOX C, SYSTEM CHARACTERISTICS AND TECHNICAL SPECIFICATIONS

Equipment practice	BYB 501		
Footprint	600 x 400 mm (23.62 x 15.75 inches)		
Height	1800 mm (70.87 inches)		
Cabinet subracks	2 CPU subracks (one each for CPU-A and CPU-B)		
	2 RPH subracks (RPHM) (one each for sides A and B)		
Electromagnetic compatibility (EMC) class:	Fulfills EMC Class B		
Cooling type	Fans		
CPU subrack	2 x 3 packages with 2 fans each		
RPH subrack	1 package with 3 fans each		
CPU subrack (one of the CP sides)	1 processor board including memory 1 interface board to RPHM 1 interface board to other CP side 1 base IO board		
RPH subrack	1 interface board to CPUM Max. 16 RPH boards (parallel type) or Max. 8 RPH boards (serial type) 1 IPNX Ethernet switch board		
Memory size	8 GW SDRAM		
Power supply	-48 V		
Power consumption	~1400 W		
System limit	APZ 212 20	APZ 212 30	APZ 212 40
Data store (DS)	1.5 GW	4 GW	7.7 GW
Program store (PS)	64 MW	96 MW	256 MW
Reference store (RS)	2 MW	16 MW	16 MW
	32 bits	32 bits	32 bits
Number of RPs	1,024	1,024	1,024
Number of EMGs	1,024	1,024	1,024
Capacity increase from the APZ 212 20	1.0	3.5	10

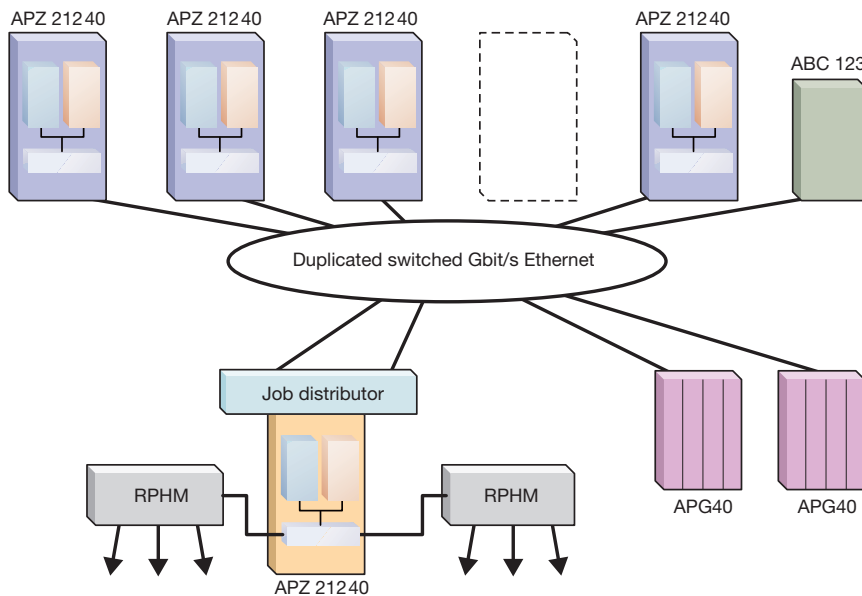


Figure 9
Example of a node in the future.

backup of the system data is first made, it is stored in the backup area of primary memory. As part of the backup function, the system is then transferred to external media—the APG40. The automatic backup function is routed via the standby side to the adjunct processor. In the standby side, an automatic reload is initiated to keep the standby side “warm.”

When the APZ VM, JIT, HAL, OS API, AOT, or OS needs to be updated, the new version is loaded as a normal file onto the adjunct processor. A boot is then initiated in the CP standby side, and the new versions are loaded. The CP sides (roles) are then switched and the sequence is repeated.

Conclusion

The new APZ hardware architecture makes the most of increased processing speed through the use of industry-standard microprocessors and innovations such as internal Ethernet buses. The mainstream sourcing of certain processors allows AXE users to benefit from general advances in microprocessor technology.

The latest version of Ericsson’s APZ central processor introduces several innovative concepts, but backward compatibility with previous versions of AXE hardware and software has been ensured, thereby safeguarding operator investments.

BOX D, KEEPING PACE WITH FUTURE DEVELOPMENTS

With its architecture and use of commercial microprocessors, the APZ 212 40 allows us to take advantage of rapid evolution in the computer industry. Development will keep pace with the mainstream computer industry and enable improvements in characteristics, such as capacity and robustness, with limited changes in software. The APZ central processor design is also open enough to simplify moves between different microprocessor suppliers and the operating systems they support.

The introduction of the interplatform network (IPN) via directly connected Ethernet makes the APZ212 40 the obvious platform for future multi-

processor systems. In a distributed multiprocessor system, replicated processors can improve in-service performance and simplify handling when adding capacity to the node. Figure 9 shows an example of a multiprocessor system.

A powerful duplicated processor handles job distribution between a number of replicated processors (call handlers). The APG40, for IO communication, is connected to the Gigabit Ethernet used for communication. Other equipment can be attached to the network based on the needs of applications.