THE EVOLUTION AND STATUS OF THE DAΦNE CONTROL SYSTEM

G. Di Pirro, G. Mazzitelli, C. Milardi, F. Sannibale, A. Stecchi, A. Stella, INFN-LNF, Frascati

Abstract

During the DAΦNE commissioning and run operations the Control System has been continuously evolving in order to fulfill the user requirements and the needs of a complete accelerator management. The original structure of distributed CPUs relaying on a central shared memory proved to be scalable and suitable for adding functionality "on the fly". Also the choice of a commercial software environment for all the control tasks demonstrated to be valid and allowed redesigning the user level with no worries for porting all the developed software. Console applications have been moved from personal computers to Force® VME embedded processors and user interfaces now runs on a Sun® multiprocessor server connected to many lightweight SunRay® terminals. A comprehensive Control System evolution history is reported.

1 DESIGN STATEMENTS

Which is the major worry for people facing the design of an accelerator Control System?

No doubt: it is that the system works.

But what does it mean "to work"? Excluding disasters such as wrong topologies, poor bandwidth or CPU power, bugs in Operating Systems and so on, we can affirm that a Control System works when it allows to drive easily and reliably the accelerator.

The matter is how to gain this target: most likely there is not a golden way but certainly there is a coherent approach based on well-defined choices. When we started the DAΦNE [1] Control System [2] design, we decided to deal with commercial technologies as much as possible and this was for good reasons. A commercial object is characterized by a broad distribution, which means a lot of feedback from the users and consequently deep debugging. Furthermore the wider is the distribution of a product the more reliable is its support from the producer.

Another criterion was to privilege "easy development and maintenance".

We decided to use:

- personal computers with standard Operating Systems everywhere;
- LabVIEW[3] as development environment for all the software;
- industrial VME bus for front-end interfacement.

2 SYSTEM GENERAL DESCRIPTION

We distributed VME crates all around the machine in order to have the front-end hardware close to the devices to be controlled. We adopted as VME processors customized Macintosh® LCIII able to run LabVIEW code.

The distributed CPUs make up the system 3rd level where the applications dedicated to the device handling and control reside. All the CPUs run asynchronously and write into their own VME memory the result of the control tasks for all the devices of which they are in charge. The refresh time depends on the number of connected devices, on the complexity of the preprocessing implemented on the CPU and on the type of front-end connections. The CPU load (in terms of number of devices) is tuned in order to keep the control refresh rate within the desired value. Depending on the needs, this rate ranges from few Hz to 50 Hz even though dedicated applications could allow significantly higher rates.

From a data point of view the system 3rd level consists of several local memory pages where all the machine objects are represented as descriptive records continuously updated at their own rate. All 3rd level VME crates are connected to a central cluster of VMEs through point-to-point optical links to constitute a central common addressing space called 2nd level.

The 2nd level can be logically thought as a single bus thanks to a *crate interconnect* VMV [4] system.

The end result is a central virtual memory where the machine RTDB (Real Time Database) resides. Once the system has been properly initialized, the transaction from the 2nd up to the 3rd level is transparent to the user that can fetch any descriptive record from a remote memory page with a simple VME read cycle. These read actions are performed from the system 1st level, where the consoles and the user applications reside.

3 SYSTEM 1 IMPLEMENTATION

For the first system version we used Macintoshes 68040 as consoles.

The connection between the 1st and the 2nd level was done by plugging into each Macintosh NuBus a dedicated VMV interface and connecting in multidrop all the consoles up to the first 2nd level VME through a VMV branch.

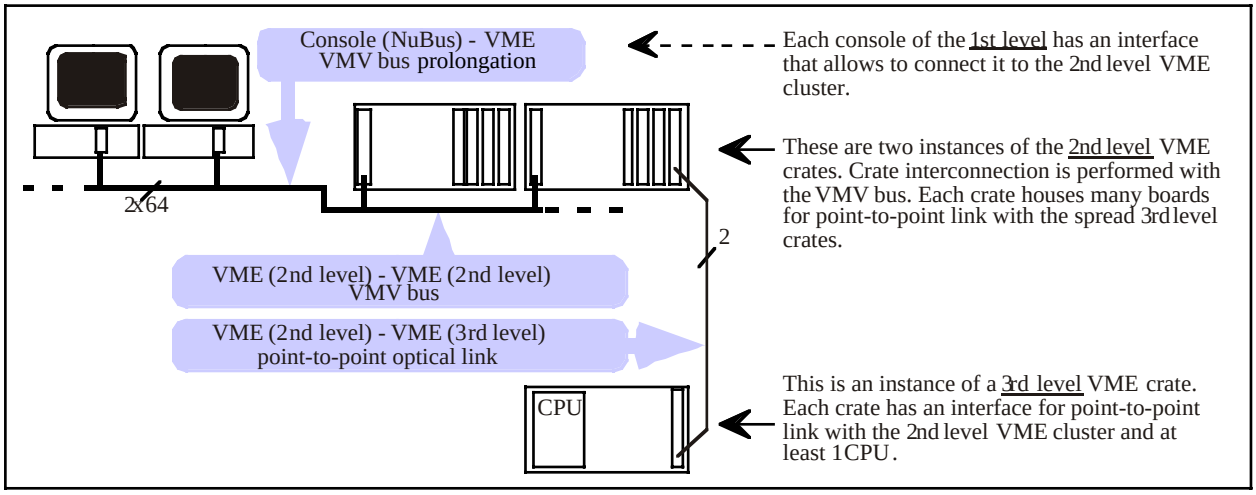

Figure 1: First implementation of the DAΦNE Control System. The consoles at 1st level were personal computers (Macintosh 68040) capable to access the central shared memory through dedicated internal interfaces.

The system version 1 (Fig. 1) main peculiarities were:

- "true" memory mapping of the central virtual memory into the consoles internal address space;
- uniform hardware and OS (Macintosh at any level);
- full accomplishment of the " Design statements".

The DAΦNE commissioning started and continued until December 1999 with this setup. The overall system behavior was good enough even though since the beginning we had to deal with a large number of bus errors due to two distinct causes.

First: the Macintosh NuBus timeout for a read/write cycle is 20.5 μs and when two or more consoles attempted to fetch simultaneously data from the shared memory, some of them could run in timeout.

Second: the Macintosh interfaces dedicated to VMV bus connection demonstrated to be not reliable in a such populated setup (we had 7 consoles and 7 VME crates on a single VMV branch) both for the arbitration and the bus current load. The workaround was to introduce software arbitration at the 2nd level and to split the VMV bus into two branches: one for the consoles and one for the 2nd level VMEs.

After this tune-up the system became more stable and grew smoothly following the introduction of new devices and the requests coming from the commissioning staff.

4 SYSTEM UPGRADE

Why to upgrade something which is working? In our case the general answer was that after 5 years of operation the system had reached some intrinsic limits:

- the Macintosh NuBus platform was dismissed;
- the 68040 μP was no longer able to stand the load of always-heavier requirements;
- the VMV consoles branch limited the number of installable consoles to about 10.

Moreover, the affirmation of new services based on Internet and the trend of "big" computers to be much easier and smarter pointed out that the choice of stand alone personal computers was no longer the most straightforward.

We focussed on the following upgrade targets:

- improve the overall system reliability;
- remove the consoles connection bus and therefore the limit to the number of consoles;
- enhance the consoles performances;
- gain remote access on the consoles;
- have Internet media and services fully available.

5 SYSTEM 2 IMPLEMENTATION

The system general structure based on distributed CPUs and a central shared memory demonstrated to be valid with no limitations from the hardware. It allowed easy and fast data gathering and correlation with no bandwidth worries.

Also the distributed processors showed to be suitable for the front-end tasks hence we focussed on the 1st level for the upgrade project. An obvious issue was to reuse as much as possible the software already developed and this somehow imposed to adopt computers able to run LabVIEW. Fortunately during last years LabVIEW had a spectacular affirmation in the industrial automation field as well as in the scientific community so that it was available on almost all the major platforms.

We adopted Spark 50T[5] VME embedded computers by FORCE instead of the Macintosh consoles. The 50T runs Solaris and is 100% compatible with the Sun UltraSpark Iii architecture. The first benefit of using VME embedded processors was to get rid of all bus-to-bus interfaces and cables and to gain RTDB data access without software arbitration.

First of all we developed the VME read/write basic routines and then we encapsulated them into conventional LabVIEW graphic nodes. After this, the porting of all the user applications in LabVIEW for Solaris went on smoothly and required just a little cosmetic make-up and a few minor adjustments.

The Spark 50 T is a multi-user machine and we estimated to load up to 5 sessions on each of them.

We installed four diskless 50T and a Sun Enterprise 250 as server over an Ethernet 100 Mbps switched network. The access to the user applications, which run on the VME processors, is done by mean of *SunRay*[6] lightweight terminals. These terminals are centrally managed by, and draw their computing resources from the *SunRay* server software that runs on the Enterprise 250.

The system performance, concerning the graphic presentations and the window management, greatly improved and also console hangs due to low memory disappeared. This architecture (Fig. 2) is fully scalable: it is possible to increase the number of VME embedded processors and hence the CPU power dedicated to user applications and to add "on the fly" *SunRay* terminals in order to have more working points.

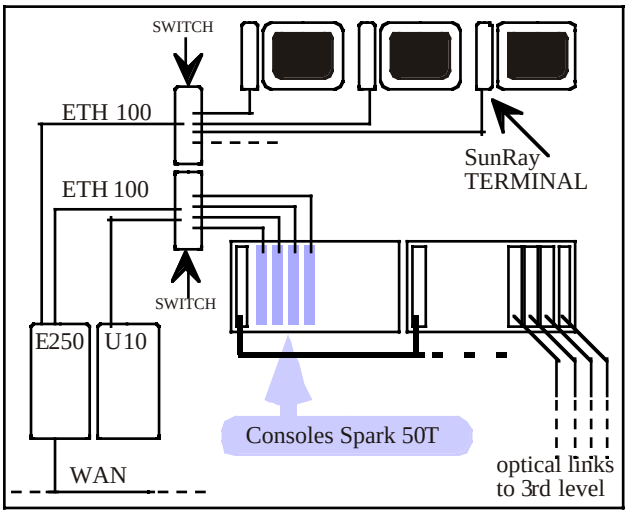

Figure 2: Second implementation of the Control System. The VME consoles and the SunRay terminals are on two separate ETH 100 Mbps switched networks.

Now the Control System integrates acquisition, disk storage and WWW online publishing [7] of machine data as well as dynamic data interchange with the experiment computing center.

CONCLUSIONS

After 6 months of development, tests and debugging and 3 months for the installation, the new system meets all the upgrade targets and is driving the accelerator since March 2000 (Table 1 summarizes the number and type of processors employed in the system in its present status).

Table 1: System main components

3rd level CPUs	Custom Mac LCIII MC68030	43
2nd level CPUs	FORCE 50T UltraSpark II	4
1nd level term.	SunRay MicroSpark	15
system server	Sun Enterprise 250 2 x UltraSPARC II	1
WWW server	Sun Ultra 10 UltraSpark II	1

ACKNOWLEDGEMENTS

We want to thank G. Baldini and M. Masciarelli for their commitment and essential contribution in the system implementation, O. Coiro and D. Pellegrini for their support in the hardware installation and setup, I. Sfiligoi for its contribution in the UNIX configuration and management.

We are also grateful to all the Accelerator Division staff for their continuous suggestions and stimulus for making a good and useful job.

REFERENCES

[1] G. Vignola and DAΦNE Project Team, DAΦNE: The First Φ-Factory, EPAC'96, Sitges, June 1996, p. 22.

[2] G. Di Pirro et al. "DANTE: Control System for DAΦNE based on Macintosh and LabView", Nuclear Instrument an Methods in Physics Research A 352 (1994) 455-475.

[3] LabVIEW, National Instruments Corporation, 11500 N Mopac Expwy, Austin, TX 78759-3504 USA (http://www.ni.com).

[4] Creative Electronic System S.A., Route du Pont-Butin 70 CH-1213 Petit-Lancy 1 Geneva Switzerland (http://www.ces.ch).

[5] Force Computers GmbH, Prof.-Messerschmitt-Str. 1, D-85579 Neubiberg/München., (http://www.forcecomputers.com).

[6] Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA (http://www.sun.com).

[7] G. Di Pirro, G. Mazzitelli, A. Stecchi, "Data handling tools at DAΦNE", TUP1B04 this Conference.