The Event Filter Farm of the CMS detector

Nick J. Sinanis
CERN
ECP/CMC
CH-1211 Genéve 23
Switzerland

The next generation of physics experiments at LHC, with the challenging physics problems they are called to solve, pose very high requirements to the underlying trigger and data acquisition infrastructure. This paper describes the concept and the architecture of the event filter farm of CMS. Special weight is given to the performance issues of computer systems, reviewing the technological trends that may affect the farm design and its implementation phase.

I. Introduction

The Compact Muon Solenoid (CMS) detector [1, 2] is one of the two largest experiments planned to operate at the Large Hadron Collider (LHC) at CERN. CMS is designed to search for new physics, like the standard model Higgs particle, in the collision products of the high-luminosity proton-proton beams of LHC. Among the CMS physics design parameters are the excellent identification and high precision measurements of muons, electrons and photons over a wide energy spectrum and at high luminosity.

The operating conditions and the physics requirements, impose severe constrains on the read-out, trigger and Data Acquisition (DAQ) subsystems. Advanced techniques of detector read-out, first-level triggering, event building and filtering will have to be implemented to cope with the unprecedented amount of data, produced during the detector operation. A powerful computer farm will be assigned the task to extract only the interesting physics events.

The schedule of the CMS project foresees a design and implementation phase, lasting almost 10 years and an operation phase of 15 years approximately. For sub-systems like the Event Filter Farm (EFF), this schedule leaves enough time for the different requirements to be satisfied better, but also raises the need of studies on its feasibility. During such long runtime period, the computer technology, will evolve by several generations and the changes that the EFF will undergo have to be taken into account.

The aim of this paper is to present the concepts of the CMS event filter farm and its building blocks, the Event Filter Units (EFU). After an introduction to the CMS DAQ system and a description of its operation, the EFF and its building blocks are presented. The concept of farm is introduced and other alternatives are discussed. Multiprocessor computer systems are considered as EFUs. Their different architectures are described, together with a discussion on their performance issues. Finally, computer industry trends that may influence the design of the EFF are discussed.

II. CMS DAQ system

The CMS DAQ system as depicted in Fig. 1, is designed to operate at the LHC bunch-crossing frequency of 40 MHz. At the nominal LHC design luminosity of 10^{34} $cm^{-2}s^{-1}$ an average of 20 events will occur in every bunch-crossing. This gives up to 10^9 interactions per second that have to be reduced to ~ 100 Hz; the maximum assumed rate of events accepted for recording.

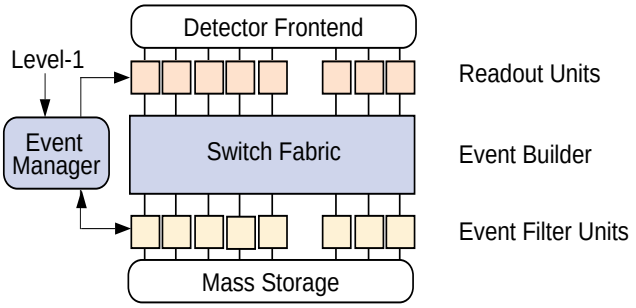

Fig. 1. The CMS Data Acquisition architecture.

A. Read-out and level 1 trigger

The task of first level triggers (LV1) is to reduce the data rate of 40 MHz produced by the front-end circuits of the detector to a maximum rate of 100 KHz. This is achieved by utilising different physics trigger-channels of the calorimeters and muon detectors, all of them running in parallel. Each one operates on a subset of the detector data, in order to minimise the decision time.

The Global Trigger System (GTS) combines together all LV1 conditions and issues the final decision of acceptance or rejection of an event. On acceptance it assigns an event number to the event fragments, over the front-end buffers. The GTS is the main device where the data rate at this trigger level will be controlled.

A physics event at this stage of the DAQ-chain is a set of fragments spread over the Read-out Units (RU). The detector granularity and the design luminosity result in a total event size of approximately 1 MB (with zero suppression taken into account). The contributions of the different subdetectors to this figure is shown in Table I.

The high data rates even after the LV1 trigger, create a data stream still difficult to handle. Therefore, two additional levels of event filtering level 2 (LV2) and level 3 (LV3)are introduced. Both LV2 and LV3 will utilise different event-selection criteria to filter the interesting events and reduce the final data rate.

TABLE I

Sub-detector	Channels	Occupancy %	Event size (KB)
Pixel	80,000,000	0.01	100
Inner Tracker	16,000,000	3	700
Preshower	512,000	10	100
Calorimeter	250,000	10	50
Muon	1,000,000	0.1	10
Trigger	10,000	100	10

B. Event building and high-level triggers

The functional elements of the event builder are 1000[1] Read-out Dual Port Memories (RDPM) as data gathering, storage and transmitting devices and 1000 Switch Farm Interfaces (SFI) at the receiving end, connected together with a 1000×1000 switching network.

With 1 MB event size and an LV1 rate of 100KHz, a total bandwidth of 100 GB/s has to be sustained at the inputs of the event builder.

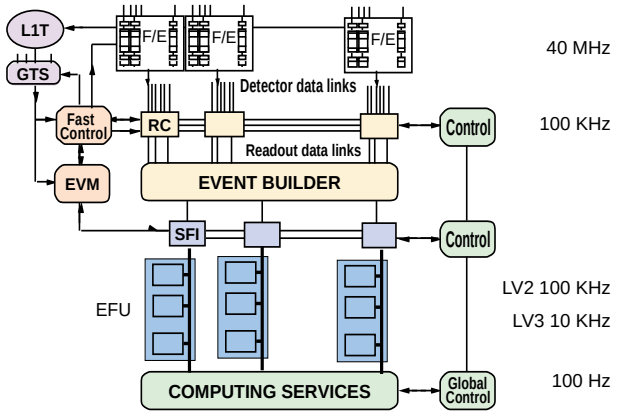

Fig. 2. CMS DAQ structure and parameters.

The event fragments of events assigned an event number (i.e accepted) are stored in the read-out buffers. The task of the RDPM is to read the read-out buffers upon an LV1 signal, store the data in their internal memory and transmit requested data to a target processor. The switch fabric will forward data fragments from the RDPM to the SFI destinations.

The task of the Event Manager (EVM) is to coordinate the assignment of events from the RUs to the EFUs. When an event is accepted by LV1, the EVM will assign to it a destination address in the EFF address space. This information will be broadcasted to all RDPMs, together with a request all data fragments relevant to LV2, to be forwarded to the destination address. After all event fragments have reached their SFI destination through the switch, the LV2 trigger algorithm will start executing in the respective EFU. At this point, the partial event will either be accepted for further processing by LV3 or rejected. In both cases the EVM will get knowledge of the LV2 decision. While in the former case EVM will broadcast a signal to the RDPMs to clear their buffers, in the later case will request from the RDPMs to send the rest of the event to the same destination were LV2 was executed, so the LV3 algorithm can start.

This procedure of processing partial events at LV2, defines the *virtual LV2* trigger concept. At the current design phase, the virtual LV2 trigger will base its decisions of acceptance or rejection of an event only on data from the muon detectors and the calorimeters. The LV3 trigger only will process the full data of an event.

The biggest advantage of virtual LV2 is the efficient bandwidth utilisation of the switch fabric (the largest part of an event is forwarded through the switch only when needed). Also, less bandwidth needs to be sustained at the SFI inputs and outputs. The virtual LV2 principle leaves also enough configuration freedom on which event parts will be used for LV2 and LV3 processing, too.

The switch fabric is the transport medium for all event fragments (LV2 and LV3) flowing from the RDPM to the SFI. With an approximate size of 1000×1000 and operating in the above described environment, the switch intersection bandwidth has to be very close to 500 GB/s. Switching networks utilising standard communication protocols based on ATM, Fibre Channel and SCI are considered as possible candidates for the event builder's switch fabric.

The utilisation of standardised communication protocols between the RDPM and SFI makes it possible to use a switch fabric from the communications industry. The rapid evolution of switch technologies, as well as the continuously increasing demand in that area, are very encouraging facts that an affordable commercial switch fabric can be purchased on time for CMS.

C. Event filter farm and computing services

As was explained before, the SFIs are at the receiving end of the event building stage. With a functionality resembling that of the RDPM, they collect all the event fragments of a LV2 or LV3 data stream transmitted from the respective RDPM, assemble them and finally send them to their peer Event Filter Unit (EFU). The design of the SFI is actually the

1. The exact figure is implementation dependant and has yet to be defined. For the purposes of this work the number 1000 is assumed.

same as that of RDPM plus some additional logic.

Each EFU receives a continuous stream of LV2 and LV3 event parts from its associated SFI. Its task is to execute the respective algorithms and return to the EVM (through the SFI) the result of the LV2 trigger. If an event passes the LV2 trigger, the execution of the LV3 algorithm will start on the same event and at the same EFU, once the reception of the rest of the event is completed.

The virtual LV2 technique requires fast communication channels between each EFU and the EVM. Moreover, the LV2 task has to have very predictable execution time in order to avoid running out of memory in the RDPM holding the LV3 event fragments. Indicative figures of the LV2 and LV3 timing are shown in Table II. Some of those figures (such as the LV2 decision time and the rejection rate) are strongly coupled to the RDPM design parameters.

TABLE II

	Virtual LV2	LV3
Event size	200 KB	1MB
Rejection factor	10	100
Execution time	10ms	1s

Once an event has passed the LV3 filtering phase, it is forwarded to the Computing Services (CS) stage for permanent storage and later analysis. An integral rate of 100Hz and an event size of 1 MB result in a 100 MB/s required bandwidth of the storage device. A communications network connected to all EFU and the CS may be used to drain all events for recording.

III. Event Filter Farm

The CMS EFF comprises the SFIs and the EFUs attached to them, as well as part of the EVM. It has to provide sufficient computing power to accomplish the LV2 and LV3 filtering tasks. Estimate figures of the total needed computer power are in the range of few 10^6 MIPS.

A computer farm model is defined by its farming process and the workers. The farming process performs the scheduling and the work distribution and the workers service the requests of the farming process. Following this paradigm, the CMS EFF has as farming process the EVM and as workers approximately a thousand of EFUs.

The ratio of worker performance to the number of workers is a very important design factor of the farm. It is expected to increase dramatically in the next years before the construction of CMS, following the advancements of modern computer industry. Several trade-offs have to be made to optimise this ratio. Among them are:

1) The performance of each EFU and the number of SFI outputs connected to them. Higher performing EFUs (in terms of CPU power and I/O throughput) might be capable of sustaining the flow of more than one SFI output.

2) Synchronisation complexity, maintenance and failure rates are proportional to the number of EFUs.

Behavioural simulations of different farm configurations and measurements on small-scale prototype systems, help to better understand and optimise the different EFF parameters.

Another interesting aspect of the CMS EFF is the possibility of using its resources during the shut-down periods of the detector for tasks like the off-line analysis of the stored events and Monte-Carlo simulations.

A key design requirement of the CMS EFF is to utilise commercial state-of-the-art computer systems in place of the EFU. In combination with the event builder choice, the deployment of standards at the farm interfacing systems is a very important requirement too.

If we take into account the life-time of the experiment and the computer industry evolution rate, it may very well happen that the used equipment is not supported any longer. After some years of operation, the EFF may gradually turn to farm with a very different structure from the original one. Thus configuration flexibility is a very important issue. The EFF should be able to operate even if heterogeneous EFUs are used.

A. Farm Scheduling

The event builder in the process of assembling event fragments, creates some 1000 independent event streams. Already at the start of an event building phase, a destination node in the EFF is assigned to the event fragments by the EVM. Hence, the EFF or more precisely the EVM scheduling is a task of major importance, determining the behaviour not only of the farm, but also that of the event builder.

A dynamic, real-time scheduling policy of the farm and particularly of each EFU is dictated by the farm size and the time-critical data flow requirements.

The virtual LV2 technique, while minimising the high bandwidth requirements of the event builder, increases the necessary signalling between the EVM, RDPM, SFI and EFU. Particularly, the EVM has to collect status information from all EFUs at a faster rate than the LV1 trigger rate (100 KHz), so that it can issue an optimal scheduling decision. Several possible farm scheduling methods are considered like request driven by the SFI and its EFU, progress monitoring of the SFI input queues or round-robin. The various scheduling options have different SFI to EVM communications requirements, which may lead to different implementations.

B. Monitoring and control

Of equal importance to the scheduling problem are the monitoring, control, fault detection and recovery, as well as the maintenance and support of the EFF.

Several performance parameters have to be collected periodically during the operation of the EFF. They will be used

also to dynamically allocate resources of the EFF depending on the luminosity and the LV1 trigger rate.

Of particular importance is the failure identification and isolation, so that the operation of the farm is not disrupted. Recovery mechanisms like rebooting or replacing an EFU without interfering to the normal operation of the rest of the system, should also exist.

The maintenance of the EFF is an issue of equal concern, too. High availability needs to be guaranteed especially during the data taking periods.

IV. Event Filter Unit

The EFUs as discussed above, are intended to be state-of-the-art and off-the-shelf commercial computer systems, with enough processing power to execute the LV2 and LV3 algorithms in time.

Important advantages of commercial over custom choice, are the better maintenance, support and the wider choice. In addition, as the desktop market is developing so rapidly, it is very likely that high-end desktop systems may easily perform well enough to make them very good candidates for the EFUs. In such case development phase of the filtering software will also benefit as the target architecture will be if not exactly the same, very similar to that used for development.

Additional requirements of the EFUs are compact packaging (rack mounted boards, etc.) to ease the installation, cooling and maintenance.

We see the high-end desktop evolving towards the multiprocessor (MP) architectures rather than merely increasing CPU clock speed of uniprocessor (UP) systems. Already today, more and more MP desktop systems are becoming available. That is why this review will focus only on MP systems and how they compare to UP systems.

A. Multiprocessor architectures

In the continuous quest for higher computer power, apart from developing faster processors, performance could be increased further if more processors were added to a system.

The first MP computer systems originally appeared in the middle of the seventies as large and expensive mainframe systems. Since then, the tremendous advances of microelectronics and microprocessor technology, brought MP systems to the user desktop with performance outrunning by orders of magnitude that of the original systems. Additionally, the developments of computer science and particularly in parallel computing and computer architectures, led to cost-effective and high performing MP systems.

Several classifications of MP systems can be made focusing on different characteristics. We mention some of them with a key role in performance.

1) Symmetric vs. asymmetric MP systems. It reflects the ability of an MP system to treat its processors equally. In a symmetric MP (SMP) system there is no privileged processor for certain operations like interrupt servicing, I/O handling etc. In asymmetric (ASMP) systems certain operations can be executed by only one processor. Very often the more general term tightly coupled processors is attributed to SMP systems.

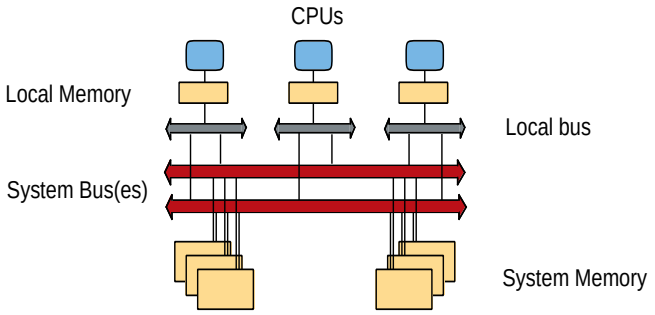

Fig. 3. Traditional multi-processor bus based architecture

2) From the point of view of the memory to CPU interconnection architecture, MP systems are classified in two main categories. Those with a single or multiple bus interconnecting processors, memory and I/O devices (Fig. 3) and those using a crossbar switch instead (Fig. 4).

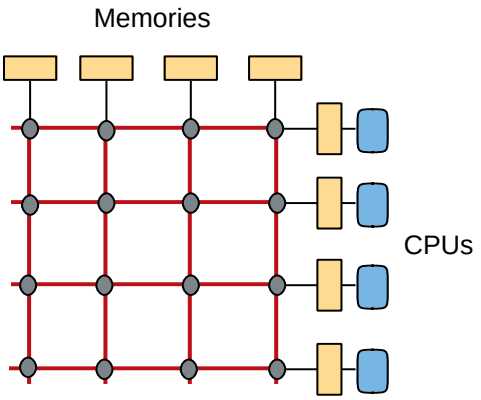

Fig. 4. Multi-processor architecture based on a crossbar interconnect

3) Depending on the structure of the memory hierarchy, MP systems can be classified further in those with shared external cache memory and those with private (non-shared) cache memory.

B. Performance Issues of Multiprocessors

In the environment of a UP system, very few operations and usually related only to I/O, may happen in parallel. Concurrency is achieved by making system calls to the operating system (OS), which in turn using a time-sharing process scheduling policy will guarantee fair access to the CPU resources. Systems with more than one memory level (i.e internal or external cache attached to a CPU) and depending on the memory model, adopt various methods to ensure data consistency. Performance problems, such as bus congestion

are very likely to appear as the number of memory and I/O service requests increases. That is, the system bus can forward one request at a time.

In an MP system, concurrency will increase simply by scheduling additional processes to the free CPU. However, the system bus of an MP system has to service not only the usual memory and I/O requests, but also CPU to CPU messages necessary to keep data located in the cache memories consistent. The interprocessor handshaking is usually handled by the hardware.

Several techniques exist to handle the cache consistency. In one of them, called *snoopy bus*, during the CPU to memory requests, other processors caches are listening to the bus traffic and they reply if they hold a modified copy of the requested memory location.

The cache coherency related bus traffic has a very high impact on the overall performance of MP systems. As the number of processors increases, this traffic increases too. Weak architectural designs of SMP systems, in conjunction with inefficient scheduling policies, may even result in negative performance scaling.

From the scaling behaviour of bus based SMP systems, as depicted in Fig. 5, is not always obvious that a better performing MP system will result if we add more processors to it. In fact, any SMP bus-based system has always a limit of positive scaling behaviour.

In order to achieve better scaling of SMP systems, crossbar switching networks were deployed as the CPU, memory and I/O interconnect (Fig. 4). These systems scale better than their bus-based counterparts from the simple fact that several memory accesses may go on simultaneously, utilising the different paths of the crossbar.

The utilisation of an MP system may increase further if a classical process is split into smaller pieces (often called threads of control) which are scheduled separately to different processors. This inherent parallelism is strongly dependant on the nature of the application. Even more, the OS itself using kernel threads of control, may run on more than one CPU at a given time.

The memory model of an MP system defines how memory is accessed for operations like load an store. The simplest and most commonly used memory model is strong ordering, while others like partial and total store ordering are also common. Because memory accesses are not always deterministic, atomicity has to be ensured (e.g an atomic read-modify-write), otherwise the system will stop operating correctly or data may be lost. In addition, special provisions have to be made by the OS, to ensure the integrity of its internal data structures when interrupts are occurring. More than one interrupt of the same kind may be serviced simultaneously on different processors.

Modern OS are capable to ensure the data integrity of their data structures and those of the applications running on SMP systems, implementing a variety of locking mechanisms, although not always for free. The different synchronisation schemes used in modern OS, if not used properly may deteriorate the overall performance of a system. A classical example is the use of spin-locking mechanisms for rather long operations were CPU power will be consumed for non useful work.

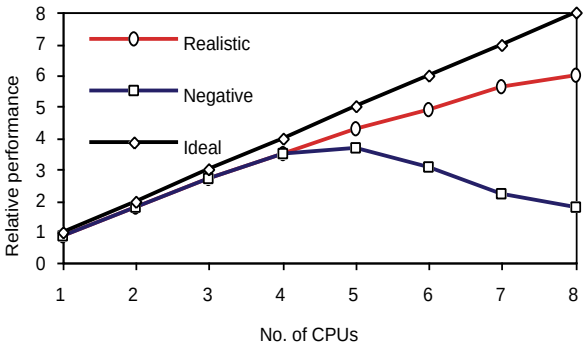

Fig. 5. Multiprocessor performance scaling

V. Trends of Computer Systems

The design parameters of EFF and particularly those of the EFU cannot easily be satisfied by today's computer systems. Thus, the study of the trends of computer systems is an important aspect of the current design phase. Among the many directions of computer industry evolution, two of them will play a key role at the EFF implementation.

A. CPU performance

Without pretending any detailed forecast analysis, we follow the evolution of some of the CPU performance indices like SPECfp92, SPECint92 and CPU clock speed (Fig. 6, 7 and 8 respectively) over the past five years.

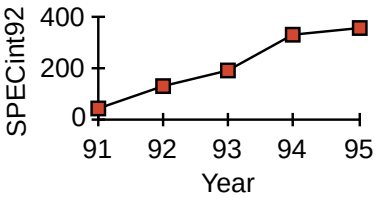

Fig. 6. SPECint 92 evolution

Precise extrapolations are not easy to make, although the evolution of these three indices outline two main directions of CPU performance improvement.

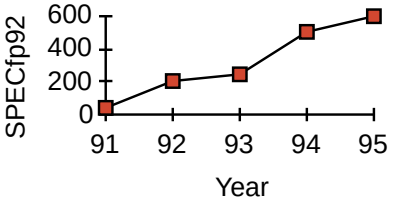

Fig. 7. SPECfp 92 evolution

These are the increase of the CPU clock speed which obviously leads to faster execution of code and the efficiency improvements of CPU, regarding the integer and floating point calculations. As can be seen from Fig. 3. the clock speed has a slower increase rate, than that of integer and float calculation indices. This reflects the tendency to make more computational powerful processors than simply faster ones.

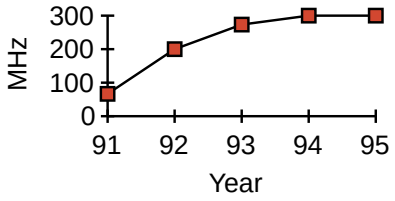

Fig. 8. CPU clock frequency evolution

The more powerful processors become available, higher the demands for faster peripherals are. Faster CPU will not perform well waiting for memory or bus resources to become available. Indeed, bus throughput and memory bandwidth are very important marks of a computer system and their value has great effects in the overall performance image.

Research areas like multi-threaded CPU architectures [5] promise to increase the overlap of computations, putting a lot of what is done today in software, into hardware.

B. Architecture

Multiprocessor systems offer the appeal of more tasks performed per unit time using more economical CPU technology than if a very fast CPU is built to handle the same load.

SMP systems have already made their appearance and they form a direction from which still a lot will come out. They will benefit from the increase of CPU power and the different architecture developments will scale them closer to the theoretical maximum.

The software overhead of maintaining cache consistency will be reduced further as the processors support different cache consistency protocols. The OS will have better chances to adapt itself to the workload's memory-access patterns, hence reducing the bus communication traffic and avoiding contention.

Crossbar switch based architectures come to contribute to higher scalability and efficiency. An SMP system could be sized (more CPU added) to handle even larger applications and still have enough potential for future upgrades.

VI. Summary and Conclusions

The CMS data acquisition system is a highly sophisticated and flexible system, able to cope with the high interaction rate and tremendous amounts data, typical for the LHC-generation of physics experiments.

In particular, the concept of event filter farm, together with the event builder proves to be flexible enough to operate and recover under the diverse design requirements and also represents a solution which adopts at many of its functional stages, off-the-shelf commercial solutions.

Multiprocessor computer systems are good candidates for event filter units. Their major performance issues were examined, as well as the general industrial trends. It is believed that multiprocessor implementations based on high performing CPUs, will always exist as an option for a higher end solution, from the fact that they can offer more economical increase of the computer power. The architectural enhancements together with the developments on operating systems, make multiprocessor systems even more attractive.

Acknowledgments

Special thanks to S. Cittolin, W. Jank and J-P. Porte and the CMS collaboration for their support and encouragement, as well as to D. Denegri for his invitation to this conference.

References

[1] CMS Letter of Intent, CERN/LHCC 92-3, (1992)
[2] CMS Technical Proposal, CERN/LHCC 94-38, (1994)
[3] S. Cittolin, A. Fucci, P. Sphicas, K. Sumorok, Dual Port Memories in LHC Experiments, CERN, CMS-RD12 TN/95-04, (1995)
[4] C. Schimmel, UNIX systems for modern architectures, Addison-Wesley, (1994)
[5] R. Alverson, et al., Tera Computer System, TERA computer company, Seattle WA, USA