μEnable: A FPGA-based Coprocessor proposed for Pre-processing in ATLAS [1]

Y. Gal [a], H. Högl [b], A. Kugel [b], L. J. Levinson [a] J. Ludvig [b], R. Männer [b], K.-H. Noffz [b], S. Rühl [b], M. Sessler [b], M. Sidi [a], H. Singpiel [b], R. Zoz [b]

[a] *Faculty of Physics, Weizmann Institute of Science, Rehovot, Israel*

[b] *Lehrstuhl für Informatik V Universität Mannheim, Mannheim, Germany*

In order to limit the demands on computing power and bandwidth for the ATLAS Level-2 trigger, events are pre-processed in the read out buffers (ROB) before passing them to the Level-2 trigger processors. Pre-processing consists of local tasks like event reformatting, zero compression, or coordinate transformations. Most of these tasks involve bit shuffling and allow pipelined and parallel execution, making them ideally suited for execution in FPGA [a] reconfigurable accelerators.
This paper discusses the FPGA-based coprocessor, μEnable, proposed for ATLAS. The board is a PCI board closely coupled to its ROB host CPU via the PCI bus. Preliminary results indicate a very effective implementation of pre-processing algorithms in the FPGA of the system. Although it is a small and inexpensive solution, a single μEnable coprocessor will be able to execute the pre-processing algorithms for a whole ROB crate rather than for a single ROB. This provides a very compact and cost effective pre-processing system for ATLAS.

[a] Field Programmable Gate Array

Key words: Reconfigurable processing; FPGA processor; ATLAS; Level 2 trigger; pre-processing

[1] This research was supported by a grant from the G.I.F., the German-Israeli Foundation for Scientific Research and Development.

1 Introduction: the ATLAS Level-2 trigger

In the concept of the ATLAS Level-2 trigger the Level-1 event data stream is temporarily stored during Level-2 processing in a huge distributed memory: the Read out Buffer (ROB) system. Accepted events are then passed to the Level-3 trigger. In order to cope with the enormous input bandwidth of 130 GByte/s, the ATLAS ROB consists of about 1000 individual units. The data required for the Level-2 processing of an event is distributed over these ROBs.

The Level-2 processing is guided by Regions-of-Interest (RoI's) in a detector identified by the Level-1 trigger. RoIs limit the fraction of the detector read out to only a few percent of the whole event data stream. This allows, unlike Level-1, computing with the full granularity of the detectors and including the tracking detectors in the trigger.

For all farm based solutions for the Level-2 trigger processors it is vital to further limit the bandwidth and computing power requirements. A promising possibility is to pre-process the event data inside the ROBs before passing it to the trigger processor.

2 Pre-processing

Typically the data for a full Region-of-Interest is spread over several ROB units which forbids the execution of partial trigger algorithms in the ROB. Instead, pre-processing consists of tasks to prepare the event for the actual algorithm or to compress its data volume.

Typical examples of pre-processing algorithms are:

(i) *Calibration* of event data.

(ii) *Event reformatting*: The data of an event is converted into a format optimally suited to the requirements of the trigger processor. An example is the alignment of the data bits on byte or word boundaries to reduce the bit shuffling inside the Level-2 trigger processors.

(iii) *Zero suppression*: The suppression of all zero data can limit the bandwidth requirements substantially. An example is the Transition Radiation Tracker (TRT) which outputs events as a pixel image. The pre-processing task consists of the conversion into a list of coordinates of all set pixels.

(iv) *Coordinate transformation and data correction*: Usually event data contain the local coordinate system of the detector – the logical number of a calorimeter cell or a silicon wafer. For effective processing these coordinates have to be transformed into a global $\eta - \phi$ coordinate system. Also corrections based on local parameters can be applied to the measured data.

(v) *Partial algorithms*: In some cases part of the actual trigger algorithm can be executed at the pre-processing level. A good example is the summing of cells in the calorimeter. Another is redoing the Level 1 logic to verify its proper functioning.

3 Reconfigurable co-processors as a solution

Typically pre-processing consists of several simple tasks to be performed in sequence. Furthermore, as the data passes through the pre-processor, only a small window of the data stream is needed at any given moment by the algorithm. There is inherent parallelism in the sequence of tasks and their repetitive application to the data stream. Programmability is essential since the pre-processing algorithms will evolve along with the Level-2 algorithms. These features make pre-processing ideal for FPGA-based reconfigurable processors. They can almost attain the speed of custom hard-wired processors, but have the ability to be re-programmed as needed. Modern hardware languages enable programming of algorithms at a high, user understandable, level. In a Level-2 architecture based on a farm of conventional RISC CPUs, an FPGA co-processor tightly coupled to a RISC CPU host offers an excellent combination of speed and flexibility for operations which are poorly handled by a RISC CPU.

4 The μEnable implementation

A small, FPGA-based, PCI-bus co-processor, named μEnable, has been developed at the University of Mannheim and the Weizmann Institute. Targeted at pre-processing, the co-processor can be used by its host (i.e. ROB) microprocessor as a programmable hardware accelerator due to the high bandwidth provided by PCI (around 100 Mbyte/s). It has the standard PCI card format and will soon be available also as a PCI Mezzanine card (PMC). The board supports the evolving ATLAS standard S-Link interface as an external link. It consists of a fast FPGA (Xilinx XC4013 or XC4028) and 0.5 to 2 Mbytes fast SRAM. Two scatter/gather DMA channels can move data in and out of the co-processor. The FPGA can interrupt its host. Custom logic can be added on a daughter board.

5 Studies of algorithms

Investigation of algorithm execution times has started with one of the most computationally intensive algorithms, the conversion of the TRT pixel image into an ordered list of coordinates. This conversion reduces the bandwidth requirements and significantly simplifies subsequent processing in the Level-2 farm processors.

The first version benchmarked on μEnable uses a pixel image in a rather simple format. The execution time of the implemented algorithm is proportional to the number of set pixels. It runs on μEnable with a frequency of 25 MHz occupying about 15% of the FPGA logical resources. Assuming a worst case occupancy of 30% the execution time is 17 μs.

The second version additionally includes the reformatting of the TRT image and the compression of the bunch crossing information. The reformatting task is required by the fact that the pixel information is not aligned to 32 bit boundaries. Preliminary results indicate an execution time on μEnable of 35 μs requiring 25% of the FPGA resources. The same algorithms implemented on a HP700-series RISC CPU takes over 0.5 ms.

In the current model of the ATLAS Level-2 trigger, an individual ROB is requested to send data to the Level-2 processors with a mean frequency of 3 kHz. Assuming the execution times mentioned above as typical values, a single μEnable co-processor is able to handle the preprocessing for 10 ROBs. This allows the ATLAS pre-processing system to be made more compact and less expensive.

6 Conclusions

The high performance of the FPGA-based processing allows the implementation of μEnable in a small and inexpensive system. Nevertheless the system is able to cope with the pre-processing algorithms for ATLAS for a complete crate of ROBs rather than for a single ROB. The μEnable FPGA processor can be a very compact and cost effective preprocessing system for ATLAS.