MCC: the Module Controller Chip for the ATLAS Pixel Detector

R. Beccherle, [a] G. Darbo, [a,1] G. Gagliardi, [a] C. Gemme, [a]
P. Morettini, [a] P. Musico, [a] B. Osculati, [a] P. Oppizzi, [a]
F. Pratolongo, [a] E. Ruscino, [a] C. Schiavi, [a] F. Vernocchi [a]

[a] *INFN e Dipartimento di Fisica dell'Università di Genova, Genova, Italy*

L. Blanquart, [b] K. Einsweiler, [b] G. Meddeler, [b] J. Richardson [b]

[b] *Lawrence Berkeley National Laboratory and University of California, Berkeley, CA, USA*

G. Comes, [c] P. Fischer [c]

[c] *Physikalisches Institut der Universität Bonn, Bonn, Germany*

D. Calvet [d]

[d] *Laboratoire de Physique Corpusculaire, Clermont-Ferrand, France*

R. Boyd [e]

[e] *Department of Physics and Astronomy, University of Oklahoma, Norman, OK, USA*

P. Šícho [f]

[f] *Institute of Physics, Czech Academy of Sciences, Prague, Czech Republic*

ATL-INDET-2002-002
05/04/2002

Abstract

In this article we describe the architecture of the Module Controller Chip for the ATLAS Pixel Detector. The project started in 1997 with the definition of the system specifications. A first fully-working rad-soft prototype was designed in 1998, while a radiation hard version was submitted in 2000. The 1998 version was used to build pixel detector modules. Results from those modules and from the simulated performance in ATLAS are reported. In the article we also describe the hardware/software tools developed to test the MCC performance at the LHC event rate.

[1] Corresponding author. E-mail: giovanni.darbo@ge.infn.it

1 Introduction

The ATLAS experiment at the future LHC collider will use silicon pixel detectors [1–5] for the three innermost layers of the semiconductor tracker. The elementary building blocks are pixel modules with an active area of 16.4×60.8 mm^2. They are arranged on 3 layers of cooling staves in the barrel region and on 2×3 ring-shaped 'disks' in the forward and backward regions. Every module consists of a n^+ on n^--type silicon sensor [6,7] segmented into 46,080 pixels with a size of 50×400 μm^2 (61,440 pixels of 50×300 μm^2 in the innermost layer). In the whole detector there will be 1744 modules, corresponding to almost 10^8 read-out channels. All components on the module are exposed to a high level of radiation: more than 10^{15} 1-MeV equivalent neutrons per cm^2 and 500 kGy during the operating life of the experiment, so that radiation-tolerant components must be used.

Fig.1 shows the different elements that compose a detector module. The pixel module is a sandwich of several layers. From bottom to top we first find the front-end (FE) read-out chips, then the sensor in the middle and a high-density two-routing-layer kapton film ("flex hybrid") on top [8]. Finally, on the top of the kapton we have SMD components (like decoupling capacitors and a temperature sensor) and the Module Controller Chip (MCC). The sensor diodes are connected to the input of the preamplifier with the high-density In or SnPb bump-bonding technique. The interconnectivity is completed by wire bonds from the FE chips to the kapton, on the two long module sides, and from the MCC to the kapton. Last, a pig-tail kapton circuit is connected to the kapton flex hybrid to bring signals in and out, to bring power in for the chips and to bias the sensor. At both ends of the detector barrel and at the outer radii of the disks all signal lines are transformed into opto-signals for the optical fibres.

2 Pixel Detector System Architecture

Electron charges, released by ionising particles, in the cells of the sensor array are collected by 16 FE chips per module, arranged in two rows of 8 chips. Each chip has 2880 pixel cells, organized in 160 rows and 18 columns. Each pixel cell in the FE chip has the same footprint as the array of sensor diodes. In each pixel cell, the preamplifier/shaper input is connected to a sensor diode by a high-density bump-bonding technique. The preamplifier output goes to a discriminator. When the charge deposited by a particle is above a programmable threshold, a time stamp to associate the hit to the correct bunch crossing is temporarily stored in the pixel.

The read-out architecture in the FE chips is 'column based'. With this kind of architecture all control logic, hit buffers and I/O pads can be easily located in the 'bottom' of the chip. Most of this area is used for the End-of-Column (EoC) buffers

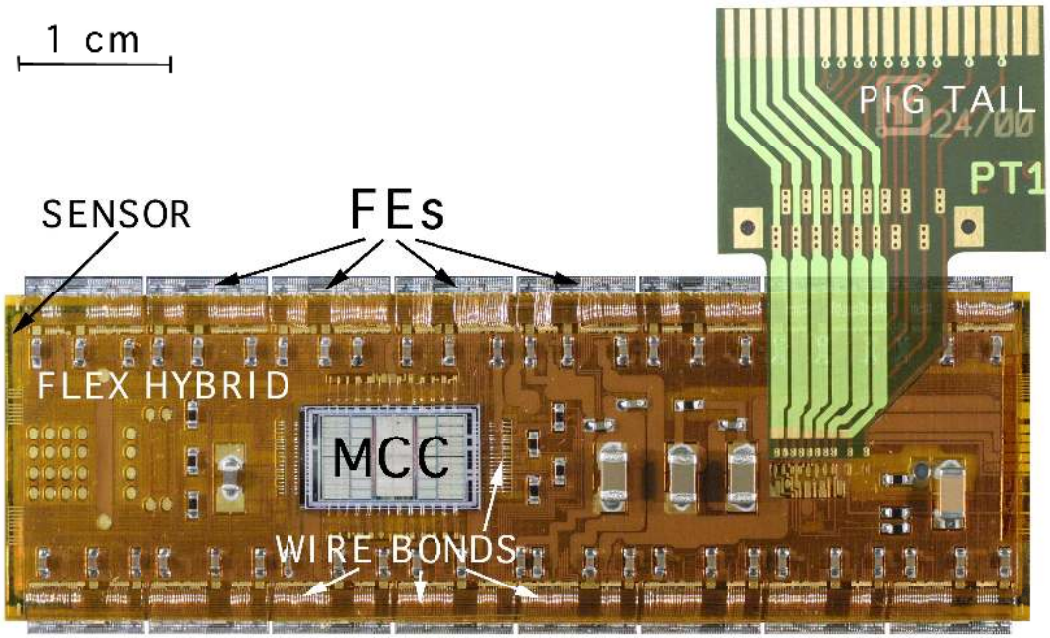

Fig. 1. The ATLAS Pixel Detector module.

necessary to store address/time information for the hits extracted from the pixel cells. This buffering is required because hits are transferred to the EoC as soon as they are detected by the discriminator in a pixel. Once in the EoC buffer, the hit information waits for the Level 1 (L1) trigger decision (2.5 μs): only hits belonging to events accepted by L1 are extracted from the FE chip, all other ones are cleared from the EoC buffers.

Several flavours of FE chip architectures were proposed and implemented, from 1997 to today. All of them have the same sensor footprint and the same electrical and functional specifications to operate with the MCC.

A simplified block diagram of the module is shown in Fig.2. All 16 FE chips are connected to the MCC by serial links. The MCC is, in turn, connected to the PIN Diode / DORIC chip, which converts the opto-signal from the down-link fibre into separate clock and data lines, and to the VDC chip / VCSEL, which converts the MCC output data to an optical signal for the up-link fibre. The optical fibres go to the off-detector Read-out Driver (ROD) modules, which have the functions to transmit trigger and command signals to the MCC, and to read events back and convert them to the ATLAS event format. The ROD also performs calibration and initial configuration of the Pixel detector module. Finally the ROD has dedicated DSP's to perform monitoring on sampled events [9–11].

The interconnections were kept very simple, and all connection which are active during data-taking use the low-voltage differential signalling (LVDS) standard to reduce EMI and balance the current flow. In order to further reduce possible EMI coupling with the sensitive FE inputs, the data out (FE-DO) signals from the FE's

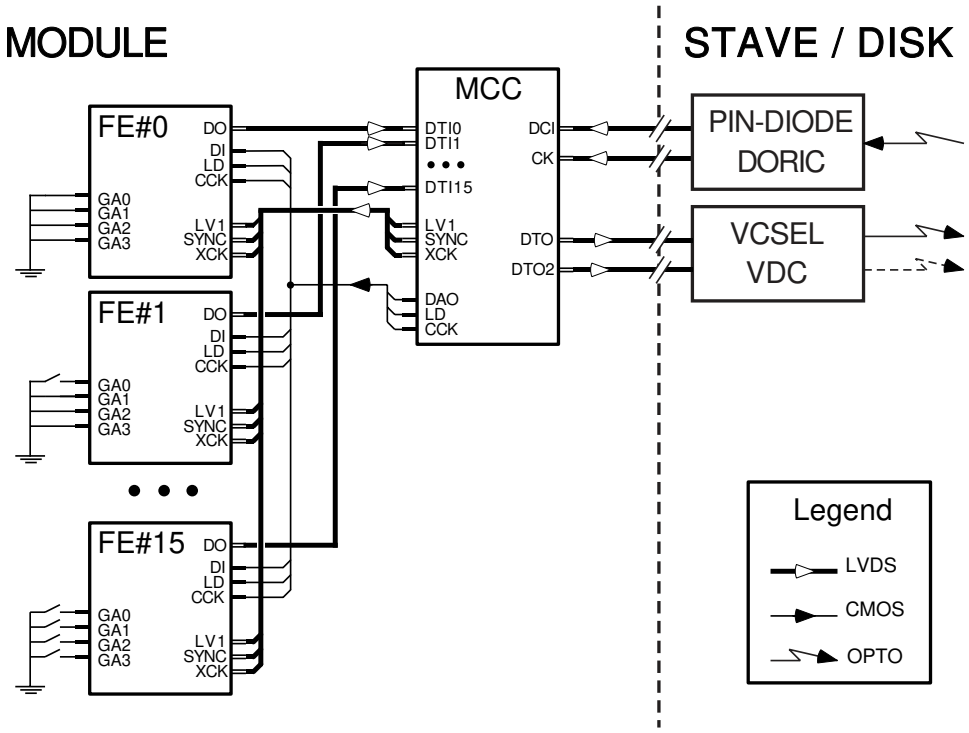

Fig. 2. Simplified block diagram of the interconnections between components of an ATLAS module.

use 500 μA current drivers instead of 3 mA of the LVDS standard. The FE-DO signals are terminated inside the MCC by 500-Ω integrated resistors. This high value of the termination resistance is acceptable because the interconnection length never exceeds 3 cm and, consequently, line reflections are negligible. Other signals are full-swing single-ended CMOS in order to reduce pin counts and to simplify the two routing layers used in the kapton.

The interconnection topology between the MCC and the 16 FE chips in a module is a star topology using unidirectional serial links. This topology was chosen to improve the tolerance of the system to failure of some components, as well as to improve the bandwidth by operating the serial links in parallel.

Two protocols are defined for data transfer between the FE chips and the MCC:

- *Configuration data.* In order to load data into the FE's, the MCC provides data (DAO) together with a clock to latch them (CCK) on the positive edge. The LD signal is used to distinguish, in the bit stream from the DAO pin, the address plus control words from the subsequent data words. The CCK is generated by the MCC only when serial data present at the DAO output must be latched in the FE. This clock has a lower frequency (5 MHz) than the XCK to reduce the timing requirements on the internal registers which can be loaded or read back using this protocol. FE chips are selected for parameter read/write by geographical addressing (FE-GA0 $\div$ FE-GA3). Before sending configuration data to a specific

FE chip, an address/control bit stream in the downloaded data packet selects both the FE chip and the secondary address or command to execute.
- *Event readout.* Data are generated by the FE's in response to MCC-LV1 trigger signals and transmitted serially from the FE-DO pin. Data at the output of the FE are sampled by the MCC at the positive edge of the 40 MHz clock (XCK). The XCK is regenerated by the MCC from the CK input clock and is fanned out to all the FE chips. XCK is the system clock for the pixel module, while CK is received from the off-detector electronics and is used at the MCC input to synchronize off-module signals.

The MCC protocol to communicate off-module uses three data lines: the DTO/DTO2 for the data output and the DCI for both data and command input. The three lines transmit or receive information using the CK timing signal as a reference clock. We foresee to use only the DTO line in the two outer pixel layers, while the additional DTO2 will be used in the B-layer, to increase the available bandwidth, to cope with the higher mean occupancy of the detector.

3 MCC Architecture

The MCC architecture went through a progressive evolution from its first definition in 1997 for the ATLAS pixel demonstrator [12], implemented in 1998 in the MCC-AMS (0.8 μm CMOS technology from Austria Micro Systems [13]), to the latest MCC-D2 (0.8 μm DMILL - "Durci Mixte sur Isolant Logico-Lineaire" technology from ATMEL [14]) designed in 2000 [15]. We are today developing a new rad-hard design using a deep sub micron (DSM) technology. The architecture explained in this section is the one that was accepted for ATLAS and implemented in the MCC-D2, unless stated. Continuously updated information on the MCC project can be found in a dedicated WWW site [16].

3.1 Functional Blocks

A simplified block diagram of the MCC internal architecture is shown in Fig.3. The blocks represented are:

- *Front End Port.* This block interfaces the MCC core to the FE chips. In case the chip is used in "transparent mode" (see Section 3.4) the core is bypassed and the FE port is directly connected to the Module Port.
- *Module Port.* This block interfaces the MCC core to the signals going to the ROD.
- *Command Decoder.* This block receives and decodes all commands sent to the MCC. The L1 trigger command, as well as read/write operations to internal MCC

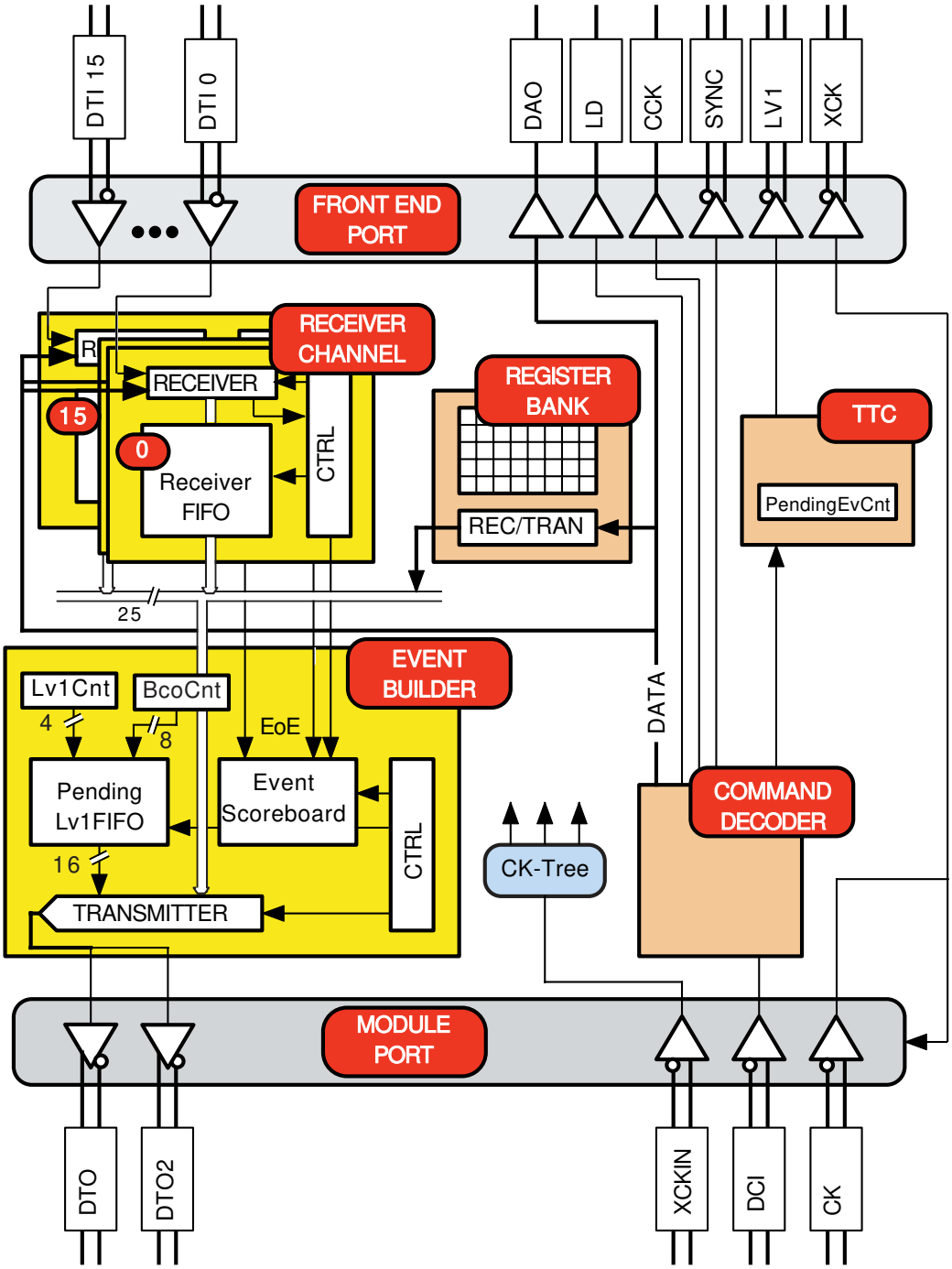

Fig. 3. MCC block diagram.

registers or to FE chips are amongst the commands interpreted by this block.

- *Register Bank.* This block contains the set of configuration and status registers used by the MCC for its operation. The MCC architecture allows the implementation of up to 16 general-purpose 16-bit registers. The registers actually implemented were 11 for the rad-soft (MCC-AMS) and 8 for the rad-hard (MCC-D2) versions of the chip.

- *TTC (Trigger, Timing and Control).* This block generates the L1 trigger (activating the LV1 line for one or more clock cycles) to the FE chips upon reception of the L1 trigger command. It also deals with event memory space overflows in the FE's plus MCC system by stopping the generation of L1 triggers to the FE chips if more than 16 events are pending for transmission to the off-detector electronics (ROD - Read-out Driver).
- *Receiver Channel.* This block is repeated 16 times, one for each FE chip in the module. Each receiver channel has a serial to parallel converter (Receiver), which monitors continuously the input line from the associated FE chip. Upon a complete reception of each hit by the Receiver, the hit is stored into the ReceiverFIFO. The controller (CTRL) has the function of dealing with the ReceiverFIFO pointers, and is capable of detecting end-of-event (EoE) messages sent by the FE chip to signal event completion.
- *Event Builder.* This block keeps two tables necessary for event building: PendingLv1FIFO and EventScoreboard. Each time a L1 trigger is decoded by the MCC Command Decoder, and is not exceeding the storage capacity of the MCC/FE system, the content of the L1 trigger counter (Lv1Cnt) and the 40 MHz bunch-crossing clock (BcoCnt) are stored into the PendingLv1FIFO: when the corresponding event is reconstructed and transmitted the two values are retrieved and inserted in the event block. The EventScoreboard is used to keep track of complete events that are stored in the ReceiverFIFO's: when a given event is fully contained inside the MCC the event can be processed by the Event Builder. The MCC-AMS had only the Lv1Cnt and not the BcoCnt implemented.

3.2 System Tasks

The MCC has three main system tasks: loading parameter and configuration data in the FE's and in the MCC itself, distribution of timing signals like bunch crossing, L1 trigger and resets, and FE chip readout and event building.

3.2.1 System Configuration.

The FE chips and the MCC must be configured after power-up or before starting a data-taking run. It is possible to write and read to all the MCC registers and FIFO's: this is used to configure, to read status information or to test the functionality of the chip. For this last function we provided a special set of commands that allow to write simulated events into the FIFO's, first, and to run the Event Builder with the stored values to check the complete functionality of the chip. This is practical once the MCC is embedded in the Pixel detector module and we want to know if it does correct event building by testing it with known simulated events. Global FE chip registers and parameters in each of the pixel cells have to be written and read back through the MCC.

3.2.2 Trigger, Reset and Timing.

The second task the MCC has to deal with is the distribution of L1 triggers, resets and calibration/timing signals for the FE chips.
Each time a LV1 trigger command is received by the MCC the TTC (Timing, Trigger and Control) logic in the MCC checks if the content of the pending event counter (PendingEvCnt) is less than 16. This means that there are up to 15 events which were triggered in the FE's and are still to be extracted by the MCC/FE system. In this case the L1 trigger is issued to the FE's and a new event will be produced as consequence. If the PendingEvCnt equals 16, the L1 trigger is not generated by the MCC and the event will be lost. In this case the last event, before the lost one, will inform the ROD of how many events will be missing.

In addition to triggers, the TTC logic generates a hierarchy of reset signals: BCR (Bunch Counter Reset), ECR (Event Counter Reset), SyncFE (Resets for FE's). The BCR resets the bunch crossing counter inside the MCC whose least significant 8 bits are recorded in the event headers transmitted to the ROD. The ROD has to issue the BCR such that the value from the MCC matches the value of the ATLAS bunch crossing number. BCR will be issued once per beam turn of the LHC. The ECR resets the whole event data not yet reconstructed and transmitted to the ROD that are in the MCC. This command will be used in case of a system error and is a fast way of resuming normal data acquisition. The SyncFE has no effect in the MCC, but it generates reset signals to the FE's. The type of reset generated by the SyncFE can be programmed as a parameter of the command and has the effect of modulating the width of the SYNC signal to the FE chips. The FE chips interpret the width of the SYNC and generate different levels of internal resets.

The last function of the TTC logic is the ability to issue calibration strobes to the FE's. This is used to calibrate the FE analogue cells on a pixel-by-pixel basis. The delay of the strobe with respect to the clock (XCK) phase can be adjusted with 6-bit resolution (roughly 0.5 ns steps). Also the delay range can be selected by a 4-bit value.

3.2.3 Event Building.

The readout architecture that was chosen for the Pixel Detector is a data-push architecture with two levels of buffering: end-of-column (EoC) buffers in the FE chips and 16 individual FIFO's (ReceiverFIFO) at the MCC inputs. Event readout and building is by far the most complicated task and it is the function that needs most of the chip area to be implemented. Data received from the FE's, in response to a L1 trigger, are deserialized and buffered into 16 FIFO's: one FIFO for each receiving FE line. These FIFO's are used to derandomise the 16 data flows from the FE's and are used by the event builder to extract ordered hits and to prepare them for transmission out of the pixel module. The size of these FIFO's, as well as the num-

ber of EoC buffers in the FE chips, are critical items. In order to keep the read-out inefficiencies at a low level these FIFO's are as large as possible for the technology used given an overall chip area.

Event building is performed by two concurrent processes running in the MCC. The first (Receiver) deals with the filling of the 16 input FIFO's with data received from the corresponding FE chip, while the second (Event Builder) extracts data from the FIFO's and builds up the event.

Each FE sends data as soon as they are available with two constraints: event hits must be ordered by event number and for each event an end-of-event (EoE) word is always generated. EoE is also sent for the case of an empty event to keep event synchronization. The event transmitted to the ROD is organized by the Event Builder process on an event-by-event basis, instead of a hit-by-hit basis as it is received from the Receiver process.

During normal conditions, each Receiver fills data hits from the FE chips into the ReceiverFIFO. The FIFO write pointer (WrPtr) is incremented each time a new hit is stored. When an end-of-event (EoE) is recognized by the Receiver in the data stream the last event pointer (LstPtr) is updated with the content of the WrPtr. In this way the LstPtr always contains the pointer to the end of the last completed event. When an EoE is seen, also the Scoreboard in the Event Builder is updated for the corresponding Receiver. If the FIFO becomes full while storing incoming hits, the partially written event is erased from the FIFO (LstPtr is copied to WrPtr) and only the EoE word is written into the FIFO. In this case, a truncated event flag will be stored in the ReceiverFIFO and then recorded to the MCC output data stream. In case that even the EoE cannot be written in the full FIFO, the Receiver stops and a flag is dispatched to the Event Builder. In this case all the forthcoming events will be reconstructed without data from the blocked Receiver (a flag is inserted into the event data). The Receiver recovers automatically to its normal state when no more events are pending in the MCC (or FE) for reconstruction. This mechanism assures that reconstructed events are not corrupted by FIFO overflows. The Receiver algorithm is illustrated in Fig.4.

As soon as the Event Builder finds that an event is completely received from all of the 16 FE's (Event Builder knows from the Scoreboard about the existence of complete events) it starts building up and transmitting the event. The first information written to the output data stream is the last word in the PendingLv1FIFO. This FIFO is 16-bit wide and 16-word deep and stores the bunch crossing ID (BCID) and the L1 ID (L1ID). The two ID's are the least significant bits of two counters: bunch crossing and L1 counters. The contents of the two counters are transferred to the PendingLv1FIFO each time a L1 is transmitted to the FE chips from the MCC. At this point the Event Builder starts fetching data from the ReceiverFIFO's. After any FIFO read operation, its read pointer (RdPtr) is incremented. RdPtr will never overtake LstPtr and Event Builder shall step to the next FIFO when it finds an EoE

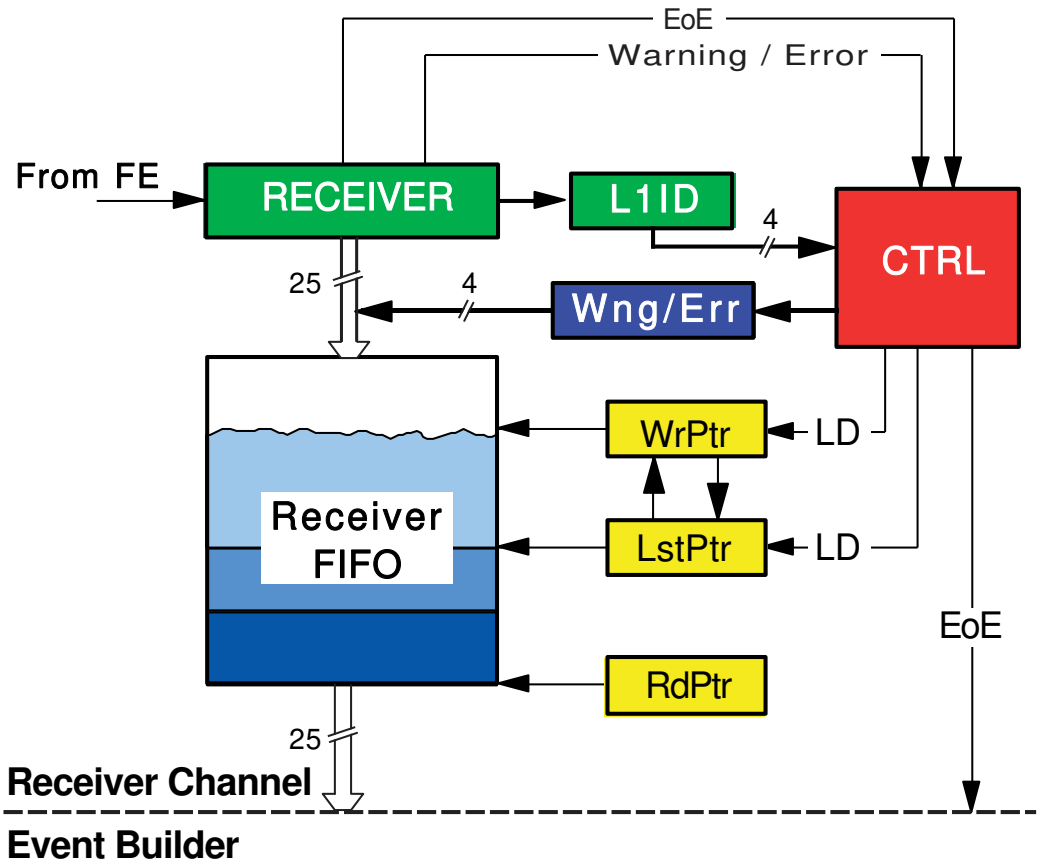

Fig. 4. Logical scheme of the Receiver block.

in the data. The Event Builder erases, for each EoE, the corresponding flag from the scoreboard. The Event Building algorithm is represented in Fig.5.

3.3 Protocols.

Several serial protocols were defined for communication to/from ROD/MCC and MCC/FE. All protocols that are active during data taking use only LVDS (Low Voltage Differential Signalling) type of signals, while signals that are in use only at configuration time use single-ended CMOS to reduce the number of interconnection lines.

3.3.1 MCC to ROD and ROD to MCC protocols.

Communications from the ROD to the MCC use a 40 Mb/s data line (DCI) validated by the rising edge of the 40 MHz clock (CK). There are 3 groups of serial commands:

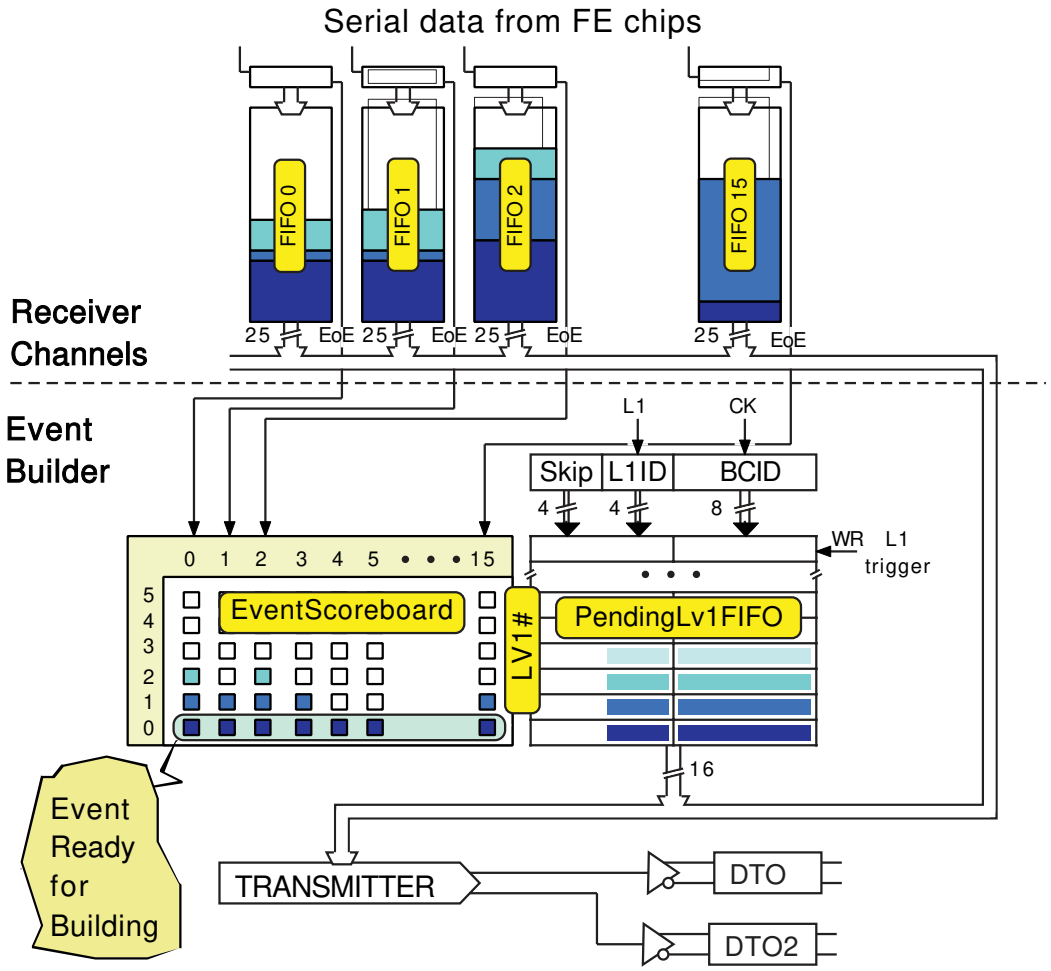

Fig. 5. Logical scheme of the Event Builder block.

- *Trigger.* This is a 5-bit long word (11101) so that a new LV1 can be issued, if needed, at the earliest every 125 ns (5 CK clock cycles, i.e. 5 bunch crossing) as required by ATLAS. This is by far the most frequently issued command while the experiment is running. We increased the length of the trigger command from 3 bits in the MCC-AMS version in order to have the possibility to detect and correct any single bit flip and to generate the trigger signal to the FE's with the correct timing.
- *Fast.* These commands are 9-bit long words, a 5-bit header (10110) and a 4-bit body. They can be issued in the absence of LV1 commands without blocking the data acquisition.
- *Slow.* These commands range from seventeen to thousands of bits. A slow command blocks event data acquisition when issued. The commands are logically subdivided into a 9-bit header (10110.1011), an 8-bit body and a variable-length data field.

The up-link may use 40, 80 or 160 Mb/s data rate: in case of 40 Mb/s a new bit is transmitted at every rising edge of CK, for 80 Mb/s bits are sent at both CK

transitions, and finally for 160 Mb/s both lines and clock edges are used (this can be considered as a 2-bit wide serial link). Only event read-out uses the two higher bit rates, read-out of configuration data is always at the 40 Mb/s rate. The protocol uses a 5-bit header (11101) to wake up the ROD receiver, followed by data bits. When the data transmitted to the ROD are from an event, since the length of the message is not known, the mechanism used is to add a synchronisation bit (Sync = 1). This is done in such a way that, in any event record, there are no more than 21 consecutive zeroes. The end of the event is defined by 22 consecutive zeroes (Trailer). Read-out of configuration data is executed as response of an issued command. In this case the length of the message is known by the ROD and no Sync and Trailer are used. Only the Header is present to signal the start of the message.

3.3.2 MCC to FE and FE to MCC protocols.

Communications from MCC to FE chips use a serial CMOS data bus (DAO), a CMOS control line (LD) and a 5 MHz validation CMOS clock (CCK). All FE chips look at the 3 lines and when the CCK rising edge is detected they strobe DAO and LD. LD (if 0) defines the command part of the message that includes the geographical address of the chip to talk to. If the FE chip recognises its geographical address, it will execute the read or write operation. If it is a write operation, when the LD goes high, every CCK rising edge transmits one bit of data to be written to the selected FE register. In case of a read command, when the LD goes high the addressed FE produces a data bit at every following CCK rising edge. Data from the FE to the MCC are transmitted using 16 individual LVDS serial links (DTI). Those links are also used to receive events hits from the FE's, but in this case the validation clock is the 40 MHz LVDS XCK. No header is used for read/write of configuration data, while a simple 1-bit header is used for each event hit (25-bit length) transmitted from the FE.

3.4 Chip Test Features.

In order to increase the testability of the MCC and of the Pixel module, once built, several design techniques were implemented.
As the observability of internal nodes is rather poor (very few output lines are present in the chip and a large number of logical nodes are very deeply embedded), scan chain flip-flops were used in the whole design. In this way we have both higher controllability and observability of internal nodes and consequently higher fault coverage for the chip.
The MCC can provide additional testing features for the system, giving direct access to the FE chips if set in transparent mode. This behaviour is obtained by a special programming pin (TM - Transparent Mode) and the duplication of some other input pins, which must be connected with their corresponding output pins

going to the FE chips (LDT $\rightarrow$ LD, LV1T $\rightarrow$ LV1 and CCKT $\rightarrow$ CCK). In order to read back signals from the output of individual FE chips, it would have been necessary to add a pin for each of the 16 FE's controlled by the MCC. This would have been too expensive in pin count, especially because the lines are differential. The solution used here is a simple routing of the DTI inputs to the DTO output. This data routing is obtained by a programmable 16-to-1 multiplexer. Transparent Mode operation a was very useful debugging feature when the first modules were built.

4 MCC Implementations

Two full-size MCC prototypes were designed and produced in 0.8 μm CMOS technology. The first (MCC-AMS) was produced by the Austria Mikro Systeme (AMS) using a rad-soft technology; it was then ported (MCC-D2) to the rad-hard DMILL technology from ATMEL. Microphotographs of the MCC dice are shown in Fig.6.

4.1 MCC-AMS

The MCC-AMS design started in 1997 and 15 4-inch wafers were produced in 1998. The chip has a size of 10.6 $\times$ 6.3 mm^2 with 363,000 transistors.

Given the high structural complexity of the MCC design, where many complex finite state machines where needed to control all different chip functionalities, the design methodology chosen is based on behavioural description, logic synthesis and automatic place-and-route. The chip was described and simulated at the behavioural level using Verilog [17]. Correct functionality of the description was validated by a large number of test vectors (one million clock cycles). The Verilog description was then synthesised by Synergy [17] and mapped to the 3.3 V, 0.8 μm CMOS library from AMS. The synthesized netlist has 17,922 standard cell instances. The only full custom blocks used in the design are the 16 FIFO's with 32 25-bit wide words. The design of the FIFO was implemented in full custom layout, as this permits to gain in chip area together with an increased memory depth: the block is repeated 16 times and has a regular structure. As AMS did not provide LVDS I/O pads in its library we designed ourselves the cells. The I/O pads and the FIFO's were placed manually on the floor plan while the standard cells were placed automatically by Cell Ensemble [17] into rows. The design uses type-D flip-flops as storage elements, synchronous with the rising edge of the clock.
In order to distribute the clock we decided to have a clock buffer in each standard cell row, and to connect together all inputs and all outputs of those buffers. The result was a big clock buffer distributed all over the chip surface. This solution is able to keep a low clock skew, and is easier to implement with respect to a balanced

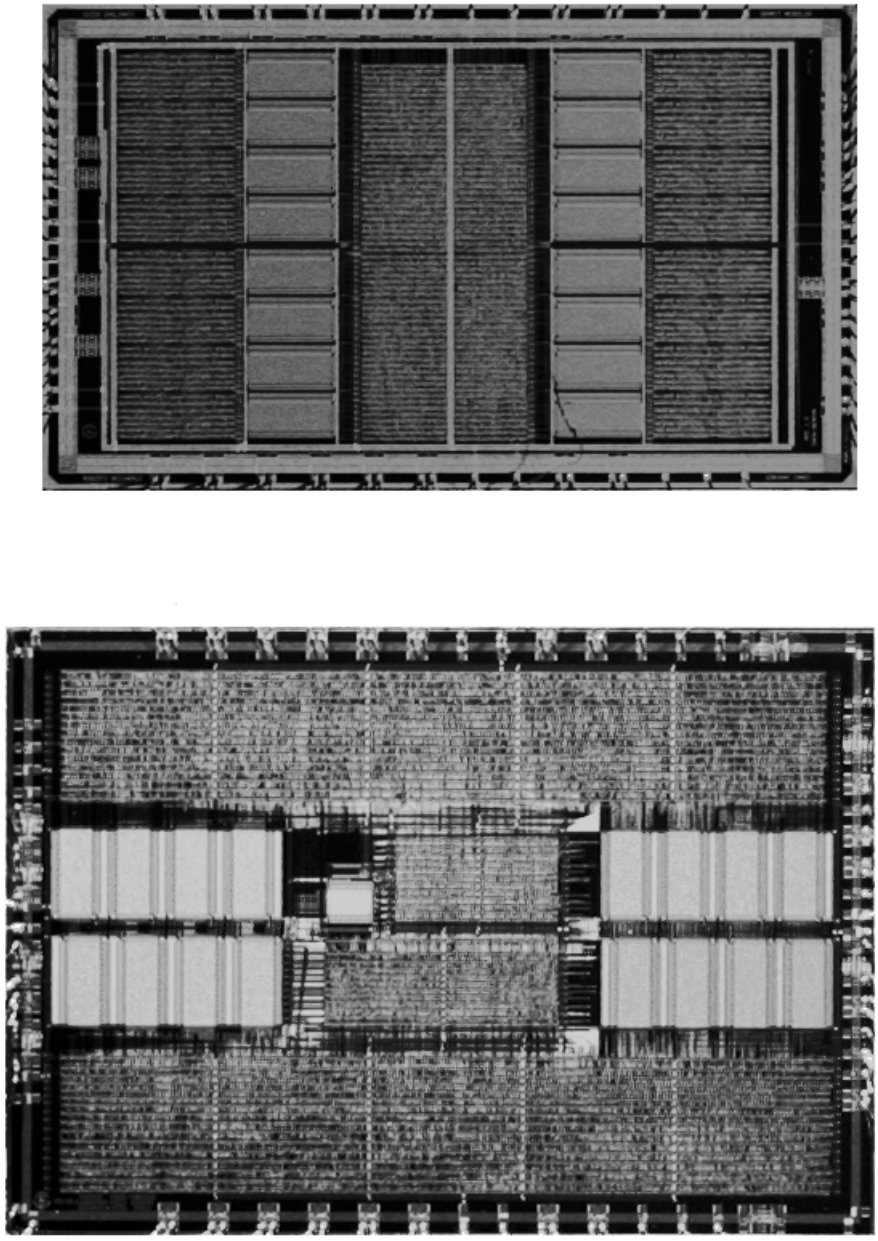

Fig. 6. Photographs of MCC-AMS (10.6×6.3 mm^2) and MCC-D2 (11.9×8.4 mm^2) dice.

clock tree. The final design steps required the design rule check (DRC) and the layout/schematics comparison at transistor level (LVS). All capacitance wire load were extracted producing a back-annotated netlist. We used the Verilog simulation of the back-annotated netlist to validate the timing of the design.
All the wafers produced by AMS were tested by the silicon foundry at 10 MHz

with 700,000 test vectors and an average yield of 64% was found. The power consumption of the chip at 40 MHz with a 3.3 V power supply is 210 mW.

4.2 MCC-D2

The porting of the MCC-AMS to rad-hard technology required 2 years with a small scale prototype (MCC-D0) produced in 1999 and the final version in the year 2000. Many changes were implemented in the MCC-D2: functional specifications, algorithms used in the event building or in the command decoder and the synthesis tool, which was now Synopsys [18]. In addition to the standard cell parts we designed a completely new I/O pad library, two types of full custom FIFO's (ReceiverFIFO and PendingLv1FIFO) and a 6-bit programmable delay line.
The MCC-D2 resulted to have much fewer standard cells (13,446) and transistors (328,000) with respect to the AMS version thanks to the optimisation of the behavioural code and to the better efficiency of Synopsys with respect to Synergy in the synthesis. The final chip size, instead, increased by 50% (11.9×8.4 mm^2) due to the less compact DMILL rad-hard technology we used.
The chips produced were not tested by the silicon foundry and 19 dice were packaged and tested in our laboratory. Only 2 chips out of 19 were found to pass the complete functional test (11% yield). None of the chips were working at the design speed of 40 MHz, the maximum being about 33 MHz. The synthesis tool predicted that the maximum frequency at 3.3 V should have been higher than 67 MHz. The discrepancy might be in the standard cell library characterization or in the interconnect delays that were only statistically estimated and not evaluated on extracted capacitances. The measured power consumption at 40 MHz and 3.3 V power supply is 315 mW.

4.3 MCC-DSM

Due to the unsatisfactory yield conditions of the DMILL technology for a chip of the size of the MCC, and because of the very promising rad-hard results demonstrated by several groups on DSM technology using annular gate n-MOS transistors, we decided to move to this technology. At the moment of writing the paper we have submitted for production a new version of the MCC (MCC-DSM) in 0.25 μm CMOS technology. Apart from refining small aspects of the architecture, the major improvement was the extension of the ReceiverFIFO's to 128 words. Even if the total number of transistor of the MCC-DSM is twice as much as the previous MCC's, the chip area is much smaller (25 mm^2). The MCC-DSM will be reported in a future paper, when the results of its test and the performance inside detector modules will be known.

5 Simulation of System Performance

In parallel to the definition of the system architecture, to the design and test of the chips, we devoted many resources to the simulation of the behaviour of the pixel system to the future LHC environment. This was necessary to understand if there are bottle-necks in the architecture (bandwidth, buffer size) or in the algorithm used to deal with buffer overflows and lost events. The software package is referred here as SimPix.

5.1 SimPix

SimPix is a package written to simulate and analyse the behaviour of the read-out chain of the Pixel detector. The architecture of the package, written in C++, is highly modular. This modularity makes it easy to plug in, for any version of the components in the simulated read-out chain, the model of the physical component that best matches the desired ratio between simulation accuracy and execution time. For example, the MCC model can be any of the following:

- a C++ description,
- a behavioural Verilog model,
- the netlist used to produce the chip,
- even a real chip mounted on the MCCex card (see Section 6.1) or connected to a Logic State Analyser.

SimPix uses a time-oriented simulation engine: each simulated component is interrogated, and its status updated, at every clock cycle. This solution enables to process time-related events (like adjacent triggers, hits, etc...), obtaining the correlations between the position in time of the simulator inputs and the produced results. As an example, a L1 trigger may or may not be accepted, depending on the time at which it was produced; when the system is nearly full, a difference of only one clock cycle may be enough to complete the extraction of a previous event and to have space to accept a new one.
The simulation flow, as shown in Fig.7, is divided into steps. Each step is executed by a different software module: data are produced by an input generator and passed to a FE chip simulator, the output is sent to an MCC module, output results are then collected by an analysis module. More than one MCC module may be active at one time; this feature is useful to compare differences in the results, for instance, between the C++ MCC and the Verilog MCC or the real chip. The input data to the FE chip model (hits and L1 trigger signals) can be generated in two ways: from parametrised random distributions or by extracting events from files produced by the full ATLAS detector simulation, based on the GEANT3 simulation package.
The random generation produces events using parameters like mean L1 trigger rate

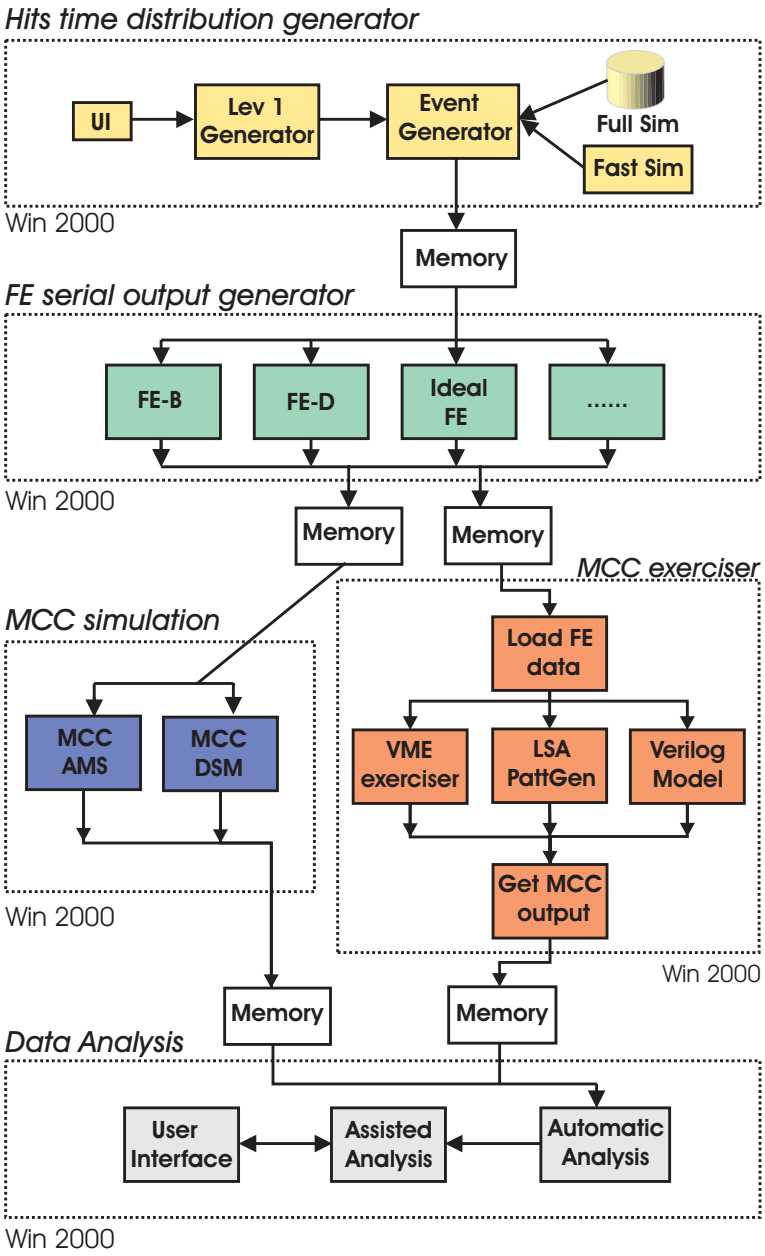

Fig. 7. Block diagram of the SimPix package.

or detector occupancy (i.e. mean number of hits per detector module and bunch crossing). This kind of generator is used to study the read-out performance as a function of input load conditions, when no relevance is given to the physical contents of data. The full simulation is useful when, on the contrary, studies are focused on the impact of electronic inefficiencies on the physics performance.

The C++ models of the FE and of the MCC have internal configurations that are parametrised. For instance ReceiverFIFO's depths in the MCC or EoC buffers sizes in the FE's can be set to different values to study their effects on the physics of the experiment. We wrote also different models for the chips, each one using a different algorithm to describe the event building in the FE's and MCC, in order to test

which option is the best to implement into the chips. This again showed to be a powerful way to optimise the system architecture to the needs of the experiment.
While a software MCC module is running, output data from the FE model can be sent to a real MCC chip or to its Verilog model in parallel. The interface between SimPix and a real MCC is provided by the MCCex (see Section 6.1). SimPix generates the status of the 16 data lines (MCC-DTI's inputs) from the FE chips and the L1 trigger commands (MCC-DCI input) from the ROD for each clock (CK) cycle. All input data are downloaded to the MCCex, the MCCex runs all the input vectors and samples the output data (MCC-DTO) and the L1 trigger (MCC-LV1) sent to the FE's. Finally the sampled outputs are read out from the MCCex by SimPix.
As for the MCCex, the Verilog model runs outside the computer running SimPix. In this case the Verilog simulator runs on a SunOS machine and the link between the two concurrent processes goes over a TCP/IP link. For each clock cycle SimPix sends input data to the Verilog process and receives back the results.
When two MCC models or emulators run in parallel, outputs from both are gathered and bit-wise compared by an analysis module. Aim of the analysis module is to tag any mismatch that may occur and to help in finding out the kind and possible origin. The same module compares also the reconstructed hits with the simulated ones, it monitors error and warning conditions, usage of bandwidth for input and output lines, and occupancy of FE and MCC internal buffers. The output produced by the analysis module (a PAW n-tuple) is also useful to identify the causes leading to data losses in the read-out process and to evaluate the compression factor between the FE and MCC output formats.
Finally, when SimPix runs on GEANT3 input data, two operation modes can be chosen: single pixel module or whole pixel detector. In the second case, SimPix filters the input data file and produces a file with the same format of the input one, containing only the hits that are kept by the read-out chain. The hits in the output files can be used in the ATLAS event reconstruction program and the results can be compared with those from the original hits.
Single-module operation is obviously faster and it is the best choice when studying read-out performance for a particular load condition. Nevertheless, analysing only one module is not sufficient to understand the impact of the inefficiencies of the read-out chain on the performance of the Pixel detector for the reconstruction of the complete events. In fact, due to the spatial hit correlation, induced by the physics processes under study, the efficiency of the event reconstruction may be strongly affected by the read-out chain. Those effects have to be carefully studied in order to tune the system architecture in its early design stage.
SimPix turned out to be a useful tool during the whole MCC development phase: conception, where a software model is used to evaluate efficiency and robustness of the reconstruction algorithm under study; design, where the Verilog description of the chip is debugged and eventually validated by comparison with the software model; functionality test of real chips, where the MCCex is used to operate the device under test and the results are interpreted with the help of the SimPix analysis module.

Out of the many results we obtained using SimPix to study the performance of the MCC-D2 architecture, we report here those we consider most significant: a study of data losses as function of MCC ReceiverFIFO size and output data bandwidth.
In the real read-out chain, both the FE chips and the MCC affect the input data, as buffer overflows may occur in both. In our simulation we used an ideal FE model, i.e. a model with infinite buffer size. This kind of FE always passes on to the MCC any input data it sees, implying that the MCC gets an input data flow higher than in the real case: this is a worst-case limit for the MCC. The track hits, at the FE inputs, come from b-jets generated by the decays of 100-GeV Higgs bosons, mixed with an average of 24 minimum-bias events per LHC bunch crossing, as foreseen at the LHC design luminosity of 10^{34} $cm^{-2}s^{-1}$. The simulated L1 trigger rate, of 100 kHz, is the maximum foreseen for the ATLAS detector.
Fig.8 shows the single pixel occupancy, i.e. the number of hits per bunch crossing and per pixel as function of the pseudorapidity and of the detector layer. The plot shows that the highest hit rate is in the central modules of the B-layer. In the simulations we studied in detail one of those modules, this again to see the worst-case effects. Each simulation result is based on 1000 complete events.

We simulated the MCC-D2 behaviour and looked at the results as function of two parameters: ReceiverFIFO depth (from 32 to 128 words) and maximum output bandwidth (40, 80 and 160 Mb/s). The results are shown in the four plots of Fig.9. In each plot, the values are plotted as a function of the ReceiverFIFO size, while each curve corresponds to one of the three values of output bandwidth. Fig.9.a and Fig.9.b show, respectively, the mean occupancy for the output data link and the mean occupancy of the ReceiverFIFO's. Fig.9.c shows the percentage of L1 triggers that would not be issued to the FE chips, thus loosing a full event. This happens when 16 events are waiting to be extracted from the MCC (coming from the FE buffers), and no more space is available to accept a new trigger. Fig.9.d shows instead the percentage of single hits that are lost due to ReceiverFIFO overflows.

From the plots it appears that, for the B-layer, the 40 Mb/s bandwidth is not enough for an acceptable event and hit efficiency for any ReceiverFIFO size. The increase of the ReceiverFIFO size helps to reduce the hit losses due to overflows, but it also increases the average number of hits kept per event. A larger event size requires more time to extract them, thus increasing the average number of events pending in the system. This fact has the effect of increasing the times during which the PendingLv1FIFO is completely full, with the loss of any incoming trigger. The net result is clear by comparing Fig.9.c and 9.d, where the decrease of lost hits corresponds to an increase of lost events for the 40 Mb/s case. The only way out is to use a larger bandwidth: in this case even a ReceiverFIFO size of 32 words is enough to keep events/hits losses to a low value (only 0.28% and 0.014 % of the hits are lost, respectively for 80 and 160 Mb/s).

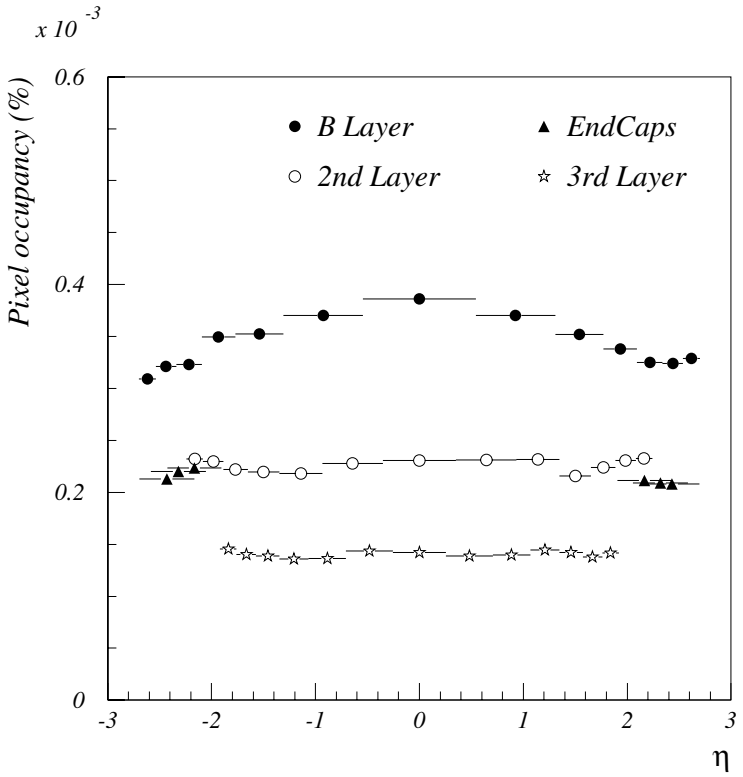

Fig. 8. Mean occupancy of a single pixel as function of the pseudorapidity for the nominal 100 kHz trigger rate. The data sets are for the three barrel layers and for the end-cap disks.

6 Experience with MCC's

Many MCC's were assembled with Pixel Detector modules, and many of those modules were operated in test beam runs. But modules with the MCC are not the right way to validate the MCC at the event rates expected at LHC. In fact, in the test beam, both the event size and the event rate are much lower and saturation of buffers in the chip never happens. This makes it difficult to extrapolate the behaviour and the stability of the MCC's Event Builder at LHC from results at the test beam. In order to overcome all those limitations, we decided to build a test environment where it was possible to load simulated LHC events into the MCC and to read the results coming from the chip. This test environment is called MCC exerciser (MCCex).

6.1 MCC Exerciser

The MCCex test system (see Fig.10) is a 6U VME board that can host a packaged version of the MCC chip. Up to 10 daughter cards can be plugged in on the moth-

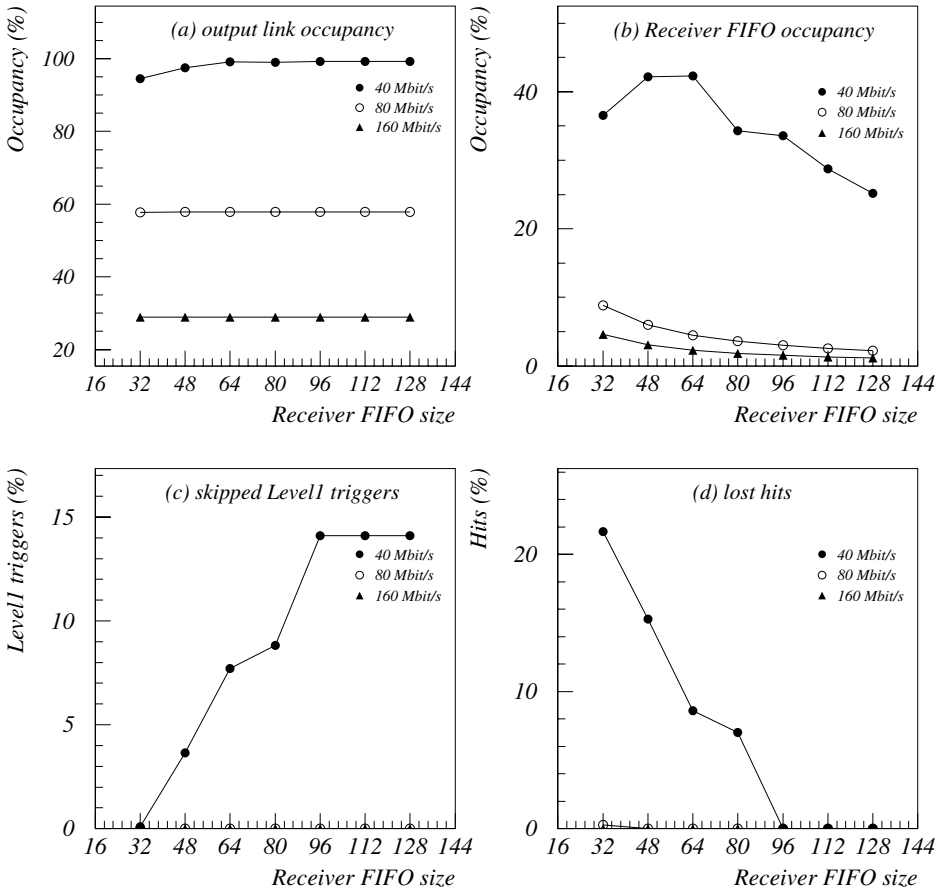

Fig. 9. Effect, for the B-layer, of the ReceiverFIFO size and of the MCC output bandwidth on the occupancy of the output link (a) and of the ReceiverFIFO (b), and on the event (c) and hit (d) lost. For more details see the text.

erboard; each one of them has two memory banks with a capacity of 8 Mb each. Each cannel can be programmed to be either a source of stimuli or a memory to sample and store signals. In a normal test application we use 8 cards to generate simulated events from 16 FE chips (DTI), one channel for command and data (DCI) to the MCC and one for the MCC output (DTO). The two other channels can be programmed to sample other MCC lines.

The whole board is synchronous with a system clock running at 40 MHz. Up to 200 ms of LHC operation can be simulated with this set-up in a single run. Longer runs can be obtained by stopping the clock at the end of each block of data, downloading the sampled outputs and uploading the next block of stimuli. As the MCC is a synchronous machine, stopping the clock freezes the status of the chip, making a virtually infinite simulation time of the MCC possible.

The MCCex needs the support of SimPix to generate the events to load into the board memories and to operate the chip. As explained in Section 5.1, SimPix can

Fig. 10. Photograph of the MCC exerciser.

generate events simulating different LHC luminosities and trigger rates, as well as the expected MCC outputs to be compared with the actual ones.

6.2 Demonstrator Modules

Three generations of Pixel Detector modules were designed and produced over the last few years (see Fig.11) [19]. All the modules produced at the moment of writing use the MCC-AMS. As the MCC-D2 had a too low yield and chips work only below the nominal 40 MHz clock rate, we decided to not build DMILL modules.

The first module we built (see Fig.11.a) had the MCC not on the module itself, as requested by ATLAS, but in a socket on the module support card. Signals from and to the FE chips were wire bonded to the support card and routed to the MCC. This solution allowed easy debugging in the early stage of the design, making the signals among the different components more accessible.

As next step we made a module using a flex hybrid circuit to connect signals between FE's and MCC. In this case a bare MCC is loaded in the centre of the flex (see Fig.11.b). The support card in this second generation of Pixel Module is used only to bring FE chip test pads to easily accessible test points on the support card itself.

In the third generation of Pixel Detector modules, the support card was dropped and we only added a carbon-carbon thermal management tile, glued underneath the module, to extract the heat from the chips (see Fig.11.c).

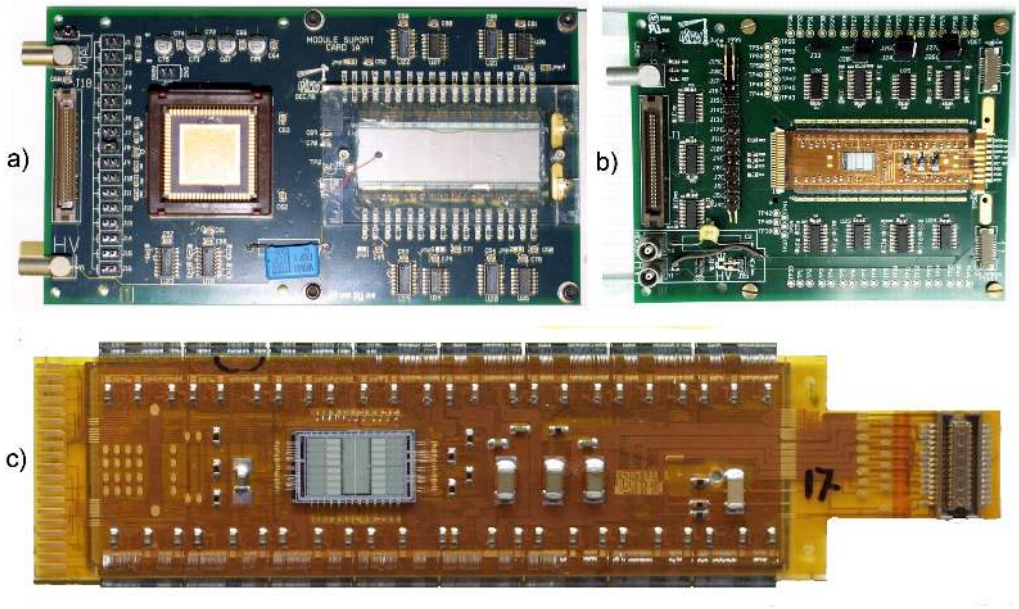

Fig. 11. Photographs of 3 generations of pixel detector modules: a) bare module with support card housing a packaged MCC, b) flex hybrid version 1 with support card and c) flex hybrid version 2 with test pigtail.

Few tens of modules were made overall. Different types of sensors were bump-bonded to the FE chips for each generation. Many of the built modules were tested in the beam or with radioactive sources. All working modules were characterized in the collaboration laboratories. The series of positive results produced by the module tests is an indirect demonstration of the good behaviour of the MCC design and of the system architecture we chose.

6.3 Irradiation in a high-flux proton beam.

We studied the effect of high doses of irradiation on the first prototype we designed in DMILL technology (MCC-D0) [20]. The irradiation was done at the CERN PS/T7 24 GeV/c proton beam. The beam was structured into one or two 200 ms long spills repeated every 20 s, releasing a dose of 50 Gy/spill. The irradiation was done at a temperature of $0 \div 3$ °C, close to the foreseen operational temperature of the ATLAS Pixel detector.
Eight MCC-D0 chips were simultaneously tested by aligning them in the proton beam line. The chips were powered up and remotely operated by the MCCex (Section 6.1). For this application, we developed some additional cards to multiplex the signals from the MCCex to 8 MCC's, to transmit the MCCex signals along 20 m of cables and to support MCC chips in the beam target position. The MCCex system is able to read/write into a single MCC at the time; therefore the test was carried out by continuously reading and writing into one of the MCC's during the spill and by

writing data into FIFO's and registers of the remaining seven before the spill and reading out and comparing the data after the spill. Any one out of the eight MCC's was active in turn, while the remaining ones always received the clock but no command or data. In this way, we monitored continuously the functioning of the chips during irradiation and we performed measurements of single event effects (SEE) due to the beam. The SEE we observed was of 1.5% of MCC's that needed to be reset per spill, and a probability of 1.2% of bit-flip per flip-flop and 2% of bit-flip in the FIFO cells per spill. These numbers must be compared with 500 kGy of dose for the middle-barrel detector layer in 10 years or for the B-layer in the first 3 years at LHC design luminosity.

All chips were successfully irradiated up to 300 kGy. After irradiation all chips were perfectly working but when tested in the laboratory, after a few weeks, four of them stopped responding to any command. We tried to anneal those chips by putting them into an oven at 100 °C for one week, but without being able to make them work again. We suppose that this problem is related to our full-custom LVDS I/O pad design.

7 Conclusions

The system architecture of the Pixel Detector module defined in the ATLAS Pixel Technical Design Report was implemented in a demonstrator MCC (MCC-AMS), which was successfully used to build 3 generations of detector modules. Extensive tests carried out show that the MCC architecture fits the needs of ATLAS. The need of a rad-hard version of the MCC brought us to select DMILL as the most promising solution for the chip to be used in the experiment. After two years to port the design and to test the chips, as the yields of both the MCC and FE chips were unsatisfactory, we decided to abandon the DMILL technology (February 2001). All efforts are now directed towards the implementation of the MCC in DSM technology, which in the last few years demonstrated to match much better the rad-hard needs of a detector like the ATLAS Pixel Detector. The advantage in the use of DSM technology will be the possibility of using larger FIFO's, which will drastically reduce the inefficiency in the B-layer at high luminosity. The chip size and yield will also improve, with a consequently lower unit cost.

8 Acknowledgments

The authors would like to thank D. Barberis for reading, commenting and editing this paper, INFN (Italy) for the financial support and all members of the ATLAS Collaboration who helped to operate the test beam and irradiation facility.

References

[1] ATLAS Pixel Collaboration, ATLAS Pixel Detector Technical Design Report, CERN/LHCC/98-13.

[2] F. Ragusa, Recent Developments in the ATLAS pixel detector, Proceedings of the "8^{th} International Workshop on Vertex Detectors - Vertex 99", Texel, The Netherlands, 20-25 June 1999, *Nucl. Instr. and Meth.* **A 447** (2000) 184-193.

[3] V. Vrba, The ATLAS Pixel Detector, Proceedings of the "International Workshop on Semiconductor Pixel Detectors for Particles and X-Rays - Pixel2000", Genova, Italy, 5-8 June 2000, *Nucl. Instr. and Meth.* **A 465** (2001) 27-33.

[4] C. Gemme, The ATLAS pixel detector, Proceedings of the "10^{th} International Workshop on Vertex Detectors - Vertex 2001", Brunnen, Switzerland, 23-28 September 2001, to be published on *Nucl. Instr. and Meth.* **A** .

[5] P. Netchaeva, Status and new layout of the ATLAS pixel detector, Proceedings of the "7^{th} International Conference on Advanced Technology and Particle Physics", Como, Italy, 15-19 October 2001, to be published on World Scientific.

[6] Hügging et Al., Prototype performance and design of the ATLAS pixel sensors, Proceedings of the "International Workshop on Semiconductor Pixel Detectors for Particles and X-Rays - Pixel2000", Genova, Italy, 5-8 June 2000, *Nucl. Instr. and Meth.* **A 465** (2001) 77-82.

[7] I. Gorelov et al., Electrical characteristics of silicon pixel sensors, Submitted to *Nucl. Instr. and Meth.*

[8] P. Skubic, Flex circuit for the ATLAS pixel detector, Proceedings of "International Workshop on Semiconductor Pixel Detectors for Particles and X-Rays - Pixel2000", Genova, Italy, 5-8 June 2000,

[9] G. Darbo, The ATLAS Pixel Detector System Architecture, Proceedings of the "Third Workshop on Electronics for LHC Experiments", London 22-26 September 1997, CERN/LHCC/97-60.

[10] G. Gagliardi, The ATLAS pixel detector electronics, Proceedings of the "4^{th} International Symposium On Development And Application Of Semiconductor Tracking Detectors", Hiroshima, Japan, 22-25 Mar 2000, *Nucl. Instr. and Meth.* **A 466** (2001) 275-281.

[11] P. Fischer, Pixel electronics for the ATLAS experiment, Proceedings of "International Workshop on Semiconductor Pixel Detectors for Particles and X-Rays - Pixel2000", Genova, Italy, 5-8 June 2000, *Nucl. Instr. and Meth.* **A 465** (2001) 153-158.

[12] G. Darbo, K. Einsweiler and P. Fischer, ATLAS Pixel Detector: Pixel Electronics Specifications,
http://www.ge.infn.it/ATLAS/Electronics/Demonstrator-2.0/MCMSpec.2.0.pdf

[13] Austria Micro Systems AG, Premstätten, Austria, www.austriamicrosystems.com.

[14] DMILL is a BICMOS technology from ATMEL, San Jose, California USA, www.atmel.com.

[15] R. Beccherle, The Module Controller Chip (MCC) of the ATLAS Pixel Detector, Proceedings of "IEEE Nuclear Science Symposium & Medical Imaging Conference", Toronto, Canada, 8-14 November 1998

[16] Specifications and test results on MCC chips can be found at the WEB page: www.ge.infn.it/ATLAS/Electronics

[17] The software packages: Cell Ensemble, Synergy, Verilog are from Cadence Design Systems Inc., California, USA, www.cadence.com.

[18] Synopsys is a software package from Synopsys Inc., California, USA, www.synopsys.com.

[19] P. Netchaeva et al., Results on 0.7% X0 thick pixel modules for the ATLAS Detector, Proceedings of "International Workshop on Semiconductor Pixel Detectors for Particles and X-Rays - Pixel2000", Genova, Italy, 5-8 June 2000, *Nucl. Instr. and Meth.* **A 465** (2001) 204-210. *Nucl. Instr. and Meth.* **A 465** (2001) 219-223.

[20] R. Beccherle, Design and Test of a DMILL Module Controller Chip for the ATLAS Pixel Detector, Proceedings of "7^{th} Workshop on Electronics for LHC Experiments", Stockholm, Sweden, 10-14 September 2001.