Raffaele Giordano

A VLSI System-on-Chip for Particle Detectors

Ph.D. Thesis

Università degli Studi di Napoli "Federico II"

November 2010

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

DOTTORATO IN FISICA FONDAMENTALE ED APPLICATA - XXIII CICLO

PH.D. THESIS

A VLSI SYSTEM-ON-CHIP FOR PARTICLE DETECTORS

Defended by

Raffaele Giordano

Coordinator: Prof. Lorenzo Marrucci
Advisor: Prof. Alberto Aloisio
Co-Advisors: Prof. Sergio Cavaliere,
Dr. Giuseppe Osteria

Academic Year 2009-2010

A Maria Teresa ed alla mia famiglia.

Abstract

In this thesis I present a System-on-Chip (SoC) I designed to offer a self-contained, compact data acquisition platform for micromegas detector monitoring. I carried on my work within the RD-51 collaboration of CERN. With a companion ADC, my architecture is capable to acquire the signal from a detector electrode, process the data and perform monitoring tests. The SoC is built around on a custom 8-bit microprocessor with internal memory resources and embeds the peripherals to be interfaced with the external ADC. Peripherals implement in hardware threshold checking, pedestal suppression, waveform recording and histogram building. The CPU has some attractive features for real-time applications: high working frequency, constant instruction execution time, short and fixed interrupt latency and a tiny logic footprint. The processor makes it possible to execute in software additional off-line data processing, such as averaging, FWHM calculation and peak finding. It includes an 8-bit IO bus for interfacing with external logic and a UART for RS232 communications. The architecture is fully portable to any technology for the implementation of digital circuits (e.g. VLSI CMOS or FPGAs). In fact, I implemented and tested a prototype in a Virtex-II Xilinx FPGA and I completed the layout of a standard-cell CMOS 90nm version of system. The performance in terms of clock frequency and logic resource occupation are discussed in the view of the deployment of the system with the detector and of the development of a multi-channel version of the system.

Contents

Introduction **13**

1 The RD-51 Collaboration **17**

1.1 The RD-51 Collaboration 17

1.2 Motivation and Objectives 18

1.3 Scientific Organization 19

1.3.1 WG1 - Technological Aspects and Development of New Detector Structures 19

1.3.2 WG2 - Common Characterization and Physics Issues . 21

1.3.3 WG3 - Applications 21

1.3.4 WG4 - Simulations and Software Tools 22

1.3.5 WG5 - MPGD related electronics 22

1.3.6 WG6 - Production 23

1.3.7 WG7 - Common Test Facilities 23

1.4 Outcomes from RD-51 24

1.4.1 GEMs 24

1.4.2 Micromegas 27

1.4.3 The Scalable Readout System 33

2 Micromegas Detectors **37**

2.1 Working Principle 37

2.2 Electric Field Configuration 39

2.3 Gain 40

2.4 Gas Mixture 42

2.5 Signal Development 43
2.6 Rate Capabilities 45
2.7 Spatial Resolution 46
2.8 Some Experiments Using Micromegas 46
2.9 Requirements for a Compact Data Acquisition System 47

3 Defining the System-on-Chip Architecture 51
3.1 System-on-Chip Planning 51
3.2 The LiloBlaze Microprocessor 55

4 SoC Architecture 63
4.1 ParticleBlaze 63
4.2 ADC Interface 67
4.3 Subtractor/Discriminator 69
4.4 Multi-Channel Analyzer 71
4.5 Sample Counter 73
4.6 XPROM Reader 74

5 FPGA Prototype 77
5.1 Field Programmable Gate Arrays 77
5.2 FPGA Implementation 80
5.2.1 Simulation 82
5.2.2 Layout 87
5.3 Laboratory Test 89
5.3.1 Test Bench 89
5.3.2 Test Results 92

6 VLSI Implementation 97
6.1 VLSI Digital Systems 97
6.2 Primitives of the Faraday Library for the UMC CMOS 90nm process 101
6.3 RAM Hard Macro 107
6.4 Design Flow 108
6.5 Implementation of the ParticleBlaze ASIC 110

6.5.1 Synthesis and Map 111
6.5.2 Floorplan 113
6.5.3 Placement 115
6.5.4 Clock Tree Synthesis 119
6.5.5 Routing 120
6.5.6 Back-Annotation and Post-Layout Simulation 121
6.5.7 Sign Off 123
6.5.8 Tape out 124

Conclusions **125**

Biliography **127**

Introduction

The RD51 collaboration is a CERN research and development program on Micro Patterned Gas Detectors (MPGD) and their associated electronic read-out systems. Among the detectors developed in the program there are the Gas Electron Multiplier (GEM), the Micro-Mesh Gaseous Structure (Micromegas) and more recently other micro pattern detector schemes. The collaboration is divided in seven working groups (WGs), each one defined by certain tasks and objectives. For instance the WG1 is dedicated to the optimization of fabrication methods for MPGDs and the development of new multiplier geometries and the WG5 is dedicated to development of the read-out and data acquisition electronics for the detectors.

My thesis work is within WG5 and consists in a study of feasibility of a microprocessor-based data acquisition Application Specific Integrated Circuit (ASIC) with embedded real-time peripherals to be used with micromegas. The realization of an ASIC, with respect to a discrete component system, allows more integration, a lower power dissipation and better frequency performance, thank to the reduction of parasitic elements. Moreover designing an ASIC allows to integrate all the interface logic (serial interfaces, program bootstrap logic, parallel IO buses, ADC handling logic, LCD displays, etc.) on the chip.

The processing tasks required to the system are:

1. basic real-time operations performed by dedicated logic (pedestal subtraction, threshold crossing check, data storing or histogramming);

2. advanced software operations performed by the on-chip microprocessor

(FWHM and mean calculation, maximum and minimum search, bin re-size, serial IO).

Other main requirements I had to satisfy in designing the SoC were: minimum logic resources for monitoring a single channel, in order to easily scale the system to multiple channels; the digital data acquisition, in order to be able to process and transmit the data to other systems (e.g. a PC) and finally the re-programmability of the system in order to guarantee the possibility to add initially-not-foreseen functionalities. These requirements had suggested me to design a microprocessor-based integrated system (or System-on-Chip) with a tiny logic footprint. Specifically, I designed a custom RISC microprocessor equipped with two small on-chip RAM banks and an application specific instruction set. I interfaced the CPU with dedicated custom peripherals for real-time data acquisition. I built and tested a prototype of the SoC realized with a Field Programmable Gate Array (FPGA), then I designed the proper Very Large Scale Integration (VLSI) device.

FPGAs are commercial off-the-shelf programmable logic devices based on a matrix of logic resources placed in a regular layout on the device surface. Logic resources are available in the form of configurable logic blocks able to implement combinational and sequential functions. The devices also include RAM banks and dedicated IO resources allow the logic to read/write data toward external devices. In order to connect the logic resources among them, there are dedicated programmable interconnection resources.

Standard cell VLSI devices are integrated circuit realized by interconnecting basic elements ("standard cells"), which are collected in dedicated libraries and which realize basic logic functions. Each cell executes a simple combinational or sequential operation (e.g. AND, NOT, flip-flop). Libraries may include macro-cells which realize more complex logic functions (e.g. RAM banks or PLLs) and IO cells. The cells are interconnected by means of ad-hoc designed metal segments. On the contrary of FPGAs, standard cell VLSI devices are not re-programmable, but they have the advantage of reaching higher operation frequencies and circuit densities. After the completion of a VLSI design, the layout is sent to a foundry, which actually

produces the device.

I designed the SoC by means of a hardware description language (HDL). HDLs, similarly to programming languages, which allow the programmer to write hardware independent code, permit to describe a digital electronic system in a technology-independent way. By means of this feature I implemented the prototype of the SoC and I designed a hardware test bench in the FPGA in order to verify on-field the correct behavior of the circuit and validate the HDL description I had written. After the successful test of the HDL description I moved to the implementation of the design in VLSI technology.

The SoC is meant to be interfaced to the detector through a front-end chip and/or an ADC. The system allows to monitor the pulse height (ΔV) or collected charge (Q) on one detector channel. The SoC acquires data from the detector, executes user-defined processing (programmed in the firmware of the CPU) on the data and periodically sends the histogram of the collected charge per each ionizing event plus relative information (e.g. FWHM, mean, maximum, minimum) to a serial terminal emulator running on a PC. All the acquired data is also collected in a file on the PC, in order to perform further analysis and processing. The study of the distribution of the acquired parameter (ΔV or Q) allows to assess whether or not operation of the detector (on the monitored channel) is correct.

Structure of This Document

This thesis is divided in six chapters.

In the first chapter, I introduce the RD51 collaboration, its research objectives, structure and recent outcomes.

In the second chapter, I describe the Micromegas detectors, which the SoC designed in this thesis is dedicated to.

In the third chapter, I plan the SoC architecture and I introduce the microprocessor I developed.

In the fourth chapter, I present in detail the internal architecture of the system also by showing some simulations of its key features.

In the fifth chapter, I describe the FPGA prototype and I illustrate the results of the tests performed.

In the sixth chapter, I introduce VLSI digital systems, I outline the design flow for standard cell designs and I show all the design steps of the SoC integrated circuit.

Chapter 1

The RD-51 Collaboration

In this chapter, I present RD-51, a research collaboration approved by CERN, and I describe its working groups.

1.1 The RD-51 Collaboration

The RD51 collaboration [1] aims at facilitating the development of advanced gas-avalanche detector technologies and associated electronic-readout systems, for applications in basic and applied research. The main objective of the R&D program is to advance technological development and application of Micropattern Gas Detectors (MPGDs). The invention of MPGDs (Fig. 1.1), in particular the Gas Electron Multiplier (GEM) [2], the Micro-Mesh Gaseous Structure (Micromegas) [3], and more recently other micro pattern detector schemes, offers the potential to develop new gaseous detectors with unprecedented spatial resolution, high rate capability, large sensitive area, operational stability and radiation hardness. In some applications, requiring very large-area coverage with moderate spatial resolutions, more coarse Macro-patterned detectors, e.g. Thick-GEMs (THGEM) or patterned resistive-plate devices could offer an interesting and economic solution. The design of the new micro-pattern devices appears suitable for industrial production. In addition, the availability of highly integrated amplification and readout electronics allows for the design of gas-detector systems with channel

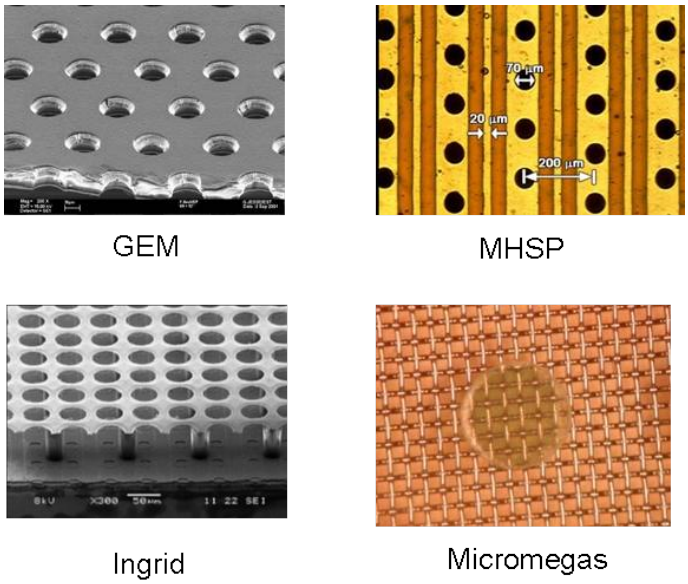


Fig. 1.1: Micro-patterned structures of four MPGD types.

densities comparable to that of modern silicon detectors. Modern wafer post-processing allows for the integration of gas-amplification structures directly on top of a pixelized readout chip. Thanks to these recent developments, particle detection through the ionization of gas has large fields of application in future particle, nuclear and astro-particle physics experiments with and without accelerators. The RD51 collaboration involves 430 researchers, 73 Universities and Research Laboratories from 25 countries in Europe, America, Asia and Africa. All partners are already actively pursuing either basic or application-oriented R&D involving a variety of MPGD concepts. The collaboration established common goals, like experimental and simulation tools, characterization concepts and methods, common infrastructures at test beams and irradiation facilities, and methods and infrastructures for MPGD production.

1.2 Motivation and Objectives

The main motivation and objectives of the collaboration are:

- foster collaboration between different R&D groups - optimize communication and sharing of knowledge/experience/results concerning MPGD technology within and beyond the particle physics community;
- investigate world-wide needs of different scientific communities in the MPGD technology;
- optimize finances by creation of common projects (e.g. technology and electronics development) and common infrastructure (e.g. test beam and radiation hardness facilities, detectors and electronics production facilities);
- steer ongoing R&D activities but not direct the effort and direction of individual R&D projects;
- applications area will benefit from the technological developments developed by the collaborative effort; however the responsibility for the completion of the application projects lies with the institutes themselves.

1.3 Scientific Organization

The scientific organization is structured in seven working groups (WG) (Fig. 1.2) each being defined through a set of tasks. Working-group conveners coordinate the R&D tasks of the respective working groups.

1.3.1 WG1 - Technological Aspects and Development of New Detector Structures

The objectives of this WG are both the optimization of fabrication methods for MPGDs and the development of new multiplier geometries and techniques. These objectives are pursued via selected tasks: (1) Development of techniques to manufacture large area modules with reduced material budget and minimum dead regions; new materials, including low-radioactivity ones for rare-event detectors; (2) Design optimization including fabrication procedures and the development of new MPGD geometries for bulk Micromegas,

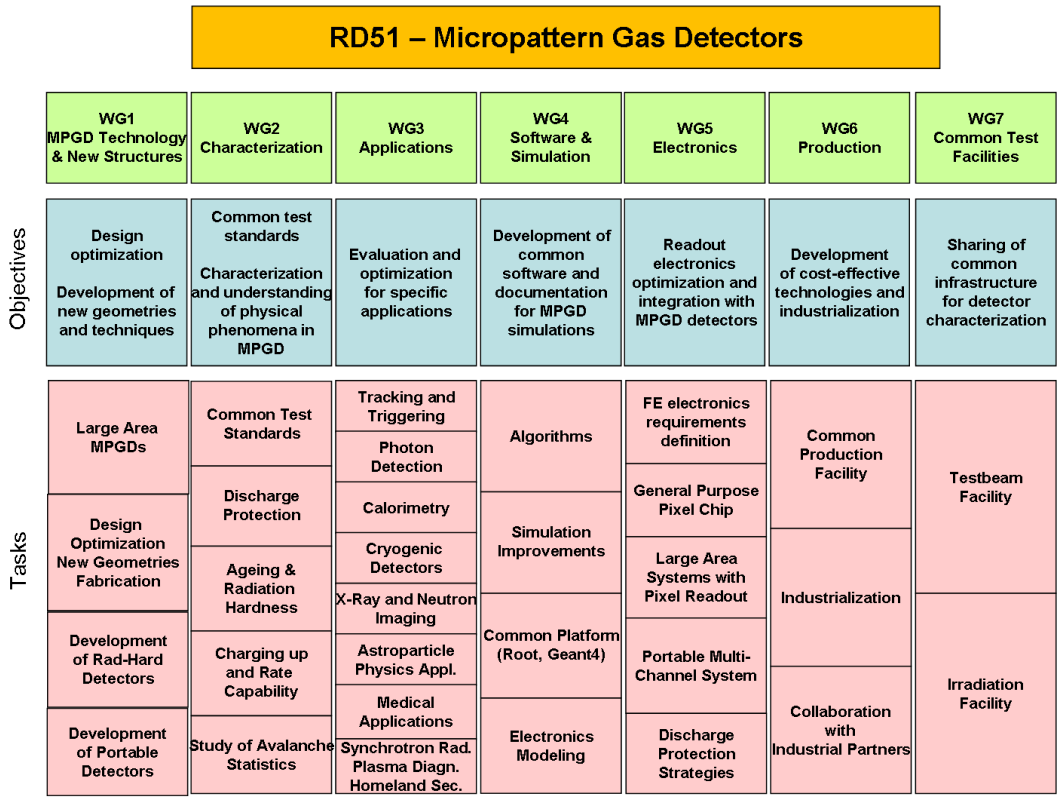


Fig. 1.2: RD51 working groups and their tasks and objectives.

micro bulk Micromegas and single-mask GEMs, Thick GEMs (THGEM), Resistive Electrode Thick GEMs (RETGEM), Micro-Patterned Resistive Plate Chambers (MPRPC), Micro Hole And Strip Plates (MHSP), charge-dispersive readout and integration of gas-amplification structures on top of a CMOS readout chip by wafer post-processing (InGrid); (3) Development of radiation-hard detectors; and (4) Design of portable sealed detectors.

1.3.2 WG2 - Common Characterization and Physics Issues

In this WG, a common effort towards the development of common standards for the characterization and comparison of different technologies will be made. The collective knowledge on the physics of discharges in MPGD detectors will be bundled and solutions towards more efficient prevention of and protection against discharge will be made. Systematic studies on ageing and radiation hardness of MPGDs will be performed and a common database on radiation hardness and ageing properties of materials will be created in order to arrive at radiation-hard detectors capable of operating beyond the limits of present devices. The tasks in the WG are: (1) Development of common test standards (comparison of different technologies in different laboratories); (2) Discharge studies and spark-protection developments for MPGDs; (3) Generic aging and material radiation-hardness studies (creation of database of "radiation-hard" materials & detectors depending on application, commercially available materials, cleanliness requirements, validation tests for final detector modules, gas system construction, working remedies); (4) Charging up (gain stability issues) and rate capability; (5) Study of avalanche statistics: exponential versus Polya (saturated-avalanche mode).

1.3.3 WG3 - Applications

Several applications impose specific new requirements and challenges on the production and properties of MPGDs. While the development of the applications itself is carried out in the individual laboratories, the collaboration will collect the requirements coming from these specific applications. The

individual tasks are structured according to these applications: (1) MPGD based detectors for tracking and triggering; (2) MPGD based Photon Detectors (e.g. for RICH); (3) Applications of MPGD based detectors in Calorimetry; (4) Cryogenic Detectors for rare events; (5) X-ray and neutron imaging (6) Astroparticle physics applications; (7) Medical imaging and diagnostics applications; (8) Synchrotron Radiation, Plasma Diagnostics and Homeland Security applications.

1.3.4 WG4 - Simulations and Software Tools

In this WG, a common, open-access, maintainable software suite for the simulation of MPGD detectors will be developed. The existing tools for the simulation of primary ionization, transport and gas amplification will be extended, in particular to improve the modeling at very small scales. An effort will be made in order to integrate the tools into the Geant 4 package to make them easier to maintain and directly applicable within arbitrary geometry and field configurations. Also the modeling of the electronics response to the detector signals has to be improved. This will also make the simulation applicable to systems with very high granularity such as CMOS pixel readout. The tasks are: (1) Development of algorithms (in particular in the domain of very small scale structures); (2) Simulation improvements; (3) Development of common platform for detector simulations (integration of gas-based detector simulation tools to Geant 4, interface to ROOT); and (4) Development of simplified electronics modeling tools.

1.3.5 WG5 - MPGD related electronics

The availability of highly integrated electronics systems for the charge readout of high granularity MPGD systems poses a non-trivial problem to many of the modern MPGD applications. The specifications of such systems for the different fields of application will be collected. For the classical configuration of charge collecting pads or strips an easy-to-use portable readout solution will be developed. Ultimate granularity is achieved by using the inputs of a CMOS pixel readout chip directly as a charge collecting anode. The

specifications of such a readout chip will be worked out and a common effort will be made towards a next-generation pixel chip for MPGD readout. The tasks are: (1) Definition of front end electronics requirements for MPGDs; (2) Development of general purpose pixel chip for active anode readout; (3) Development of large area detectors with pixel readout; (4) Development of portable multichannel data acquisition systems for detector studies; and (5) Discharge protection strategies.

1.3.6 WG6 - Production

In this working group cost-effective, industrial technology solutions will be developed and transferred to industry. A common "production facility" based on the MPGD workshop at CERN will be developed and maintained and procedures for industrialization will be set up. The tasks are: (1) Development and maintenance of a common production facility; (2) MPGD production industrialization (quality control, cost-effective production, and large-volume production), (3) Collaboration with Industrial Partners.

1.3.7 WG7 - Common Test Facilities

The development of robust and efficient MPGDs entails the understanding of their fundamental properties and performance at several stages of their development. This implies a significant investment for detector test beam activities to perform the R&D needed, to test prototypes and to qualify final detector system designs, including integrated system tests. The measurements in test-beam facilities cover efficiencies, noise, time, position and energy resolutions - basically all the critical performance parameters for new detector systems. Additionally, characterization of specific detector behaviors operated in large particle background demands some targeted aging tests in irradiation facilities. A common effort in this direction is needed because the number of groups involved in MPGD development has grown very significantly and will still do so during the coming years. As members of the RD-51 collaboration, research groups will get easier access to the facilities inside RD-51 collaborating institutes and at CERN, and, most important, share re-

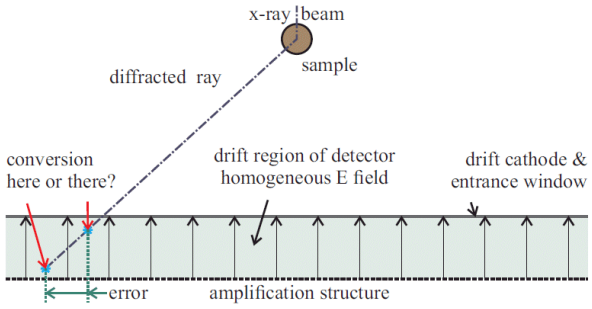


Fig. 1.3: The cause of a parallax error in a gas detector with a homogeneous drift field.

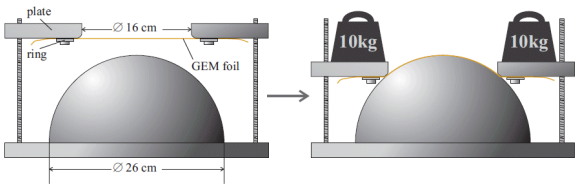


Fig. 1.4: The setup built to remold a flat GEM into a spherical one.

sources, make common requests and group experiments. The two tasks are (1) Development and maintenance of common "Test-Beam Facility"; and (2) Development and maintenance of common “Irradiation Facility”.

1.4 Outcomes from RD-51

The RD-51 collaboration is scientifically very active in all its working groups and their tasks. Without the aim of covering all the results from RD-51, here I briefly present some interesting achievements of the collaboration in the field of GEMs, Micromegas and electronics systems for MPGD read-out.

1.4.1 GEMs

Position sensitive radiation detectors with a spherical geometry can be attractive for various applications. The main reason in the case of gas detectors

is to eliminate the parallax error arising from the uncertainty of how deep radiation penetrates the sensitive volume before causing ionization. If the electric field in the conversion region of a gas detector is not parallel to the direction of irradiation, an uncertain conversion depth will give rise to an error in position reconstruction, see Fig. 1.3. Methods that have been used to suppress parallax error include:

- arranging small area flat detectors in such positions as to approximate a spherical shape [4];
- creating an almost spherical conversion region with foils and meshes, then transferring the charge to a planar wire chamber [5, 6].

All the previously used methods have some limitations. In all cases the challenge of making a fully spherical detector, however desirable, has been avoided. Within RD-51, a group started an effort that should lead to the first fully spherical GEM-based gas detector. They developed a method that permit to make a spherical GEM [7] from a flat standard GEM, apparently without affecting its properties significantly. Starting with a flat GEM foil they use a method similar to thermoplastic heat forming; the foil is forced into a new shape by stretching it over a spherical mold, see Fig. 1.4. First performance tests of single spherical GEM will be done in a setup with a spherical entrance window (which serves as drift electrode as well) and a flat readout structure. The electric field in the conversion region is truly radial. The amplitude of signals from conversions in the non-radial induction region will be suppressed by the gain of the GEM.

Another aspect of GEMs being studied within RD51 is the possibility to build large area (few m^2) detectors. In 2008, a triple GEM detector prototype with an area of $\sim 2000\ cm^2$ has been constructed, based on foils of $66{\times}66\ cm^2$. GEMs of such dimensions had not been made before, and innovations to the existing technology were introduced to build this detector [8].

The application of GEMs as gates for stopping the drift of ions in Time Projection Chambers (TPCs) is also under study. Large volume gaseous

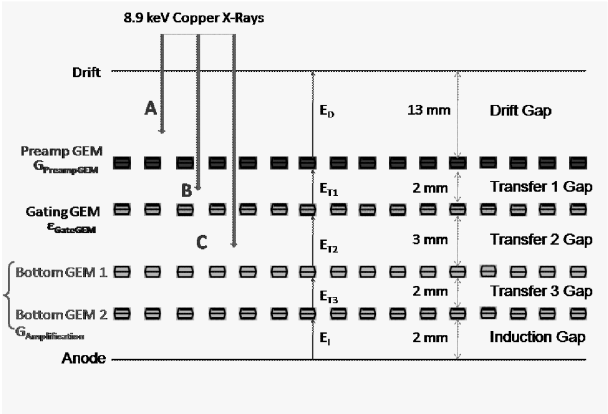


Fig. 1.5: Small prototype schematics. The readout anode was a full metal plane.

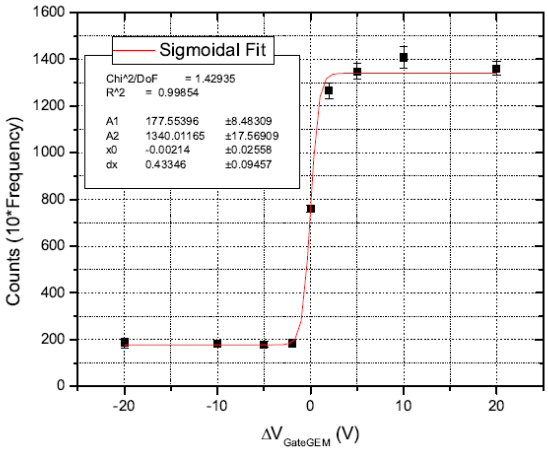


Fig. 1.6: $\Delta V_{GateGEM}$ scan with pre-amplification GEM activated. Negative potential sign means closed gate.

TPCs suffer from the problem of ion-built space charge. The amplification structures (nowadays still MPWC) amplifies the primary electrons created by the interacting particle and consequently generate ions that slowly move back to the drift volume where they modify the electric field thus changing the time-space properties of the next track. A voltage-controlled GEM can be used to block the re-injection of positive ions in large volume TPCs (Fig. 1.5). Through an accurate choice of proper geometry, gas filling and external fields it is possible to obtain a sufficient level of electron transmission at very low GEM voltages (Gating GEM) despite the degradation of energy resolution due to the loss of primary electrons. The addition of a pre-amplification GEM in front of the Gating GEM causes an improvement in energy resolution while keeping the ion feedback at the level of primary ionization. There are results showing that a small pulse of about 40 V completely closes the gate, stopping the ions produced in the amplification stage [9]. The tests have been performed with a GEM exposed to 8.9 keV X-Rays. In order to prove the gating properties of the Gating GEM, a counting mode voltage scan was made and the corresponding photon interaction rate was recorded. As it is shown in Fig. 1.6 the potential difference applied to the Gating GEM ($\Delta V_{GateGEM}$) varied from -20 V up to +20V (where the positive sign means correct polarization to get electrons drifting in the holes): this step demonstrated that the gate is completely closed with a very small potential difference, from -10 V to -5 V. This is a very promising feature for high rate pulsed gate in TPCs. The closed gate plateau is not on the zero level because of X-Rays conversion in the C region.

1.4.2 Micromegas

With the luminosity upgrade of the LHC machine (s-LHC, Super-LHC), the Muon spectrometer of the ATLAS experiment at CERN will also need a detector upgrade in the highest rapidity region. The detectors of the spectrometer, Monitored Drift Tubes and Cathode Strip Chambers (used for tracking), Resistive Plate Chambers and Thin Gap Chambers (dedicated to fast triggering) are able to sustain counting rates at least up to five times

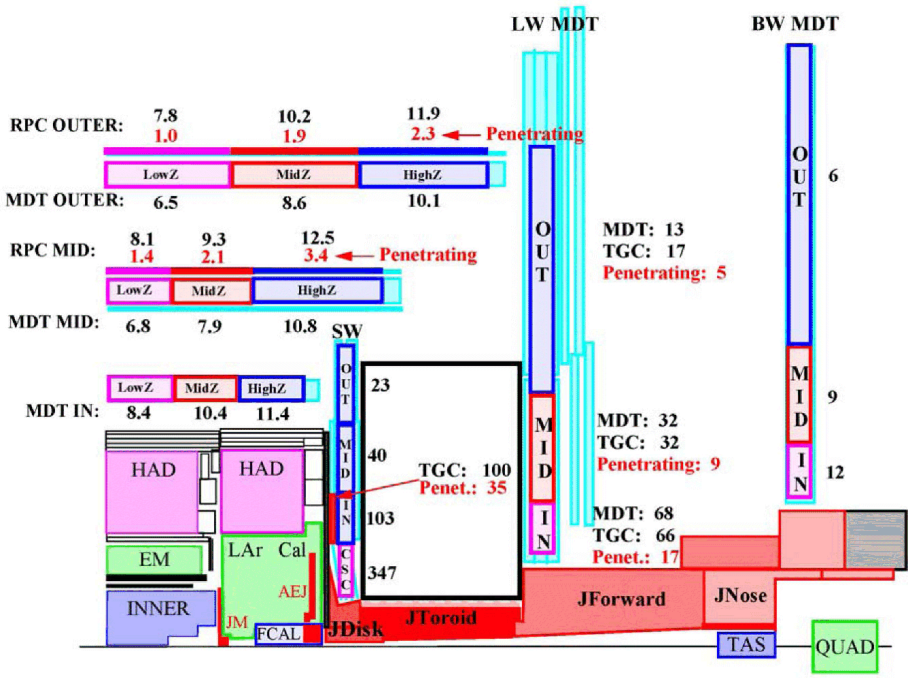


Fig. 1.7: Counting rates ($\frac{Hz}{cm^2}$) in the muon spectrometer at the LHC nominal luminosity ($10^{34} cm^{-2} s^{-1}$). Mid stands for Middle, In stands for Inner.

the expected rates at nominal LHC luminosity. The counting rates in each part of the spectrometer are graphically reported in Fig. 1.7.

For the s-LHC, delivering ten times higher instantaneous luminosity (i.e. $10^{35} cm^{-2} s^{-1}$), an increment of the expected particle fluxes from muons and background sources (i.e. photons, neutrons and protons) is expected. In order to maintain the detector coverage and a good detecting performance under these new conditions, it is most likely that an upgrade of the present trigger and tracking chambers in the forward region, $|\eta| > 2$, will be necessary. The Muon ATLAS Micromegas Activity (MAMMA), is an ongoing RD-51 activity with the aim at developing large detectors based on the bulk-Micromegas technology for use in the ATLAS Muon Spectrometer [10]. Micromegas is a good potential candidate for the construction of large muon chambers that combine trigger and tracking capability and can sustain high particle rates expected at the s-LHC. A medium size Micromegas prototype, in scale 1:10 of the final chambers, has been built and evaluated in the laboratory and in beam tests at CERN. The final chambers to be used for the upgrade of the ATLAS Muon Spectrometer should be of approximate size 1m x 2m, with the following characteristics:

- high-rate capability ($\geq$5 kHz/cm^2);
- spatial resolution $\sim$100 μm for impact angles $\leq$45°;
- transverse coordinate resolution $\sim$ 1 cm;
- time resolution: $\leq$ 5 ns;
- high efficiency $\geq$ 98%;
- level-1 triggering capability;
- radiation hardness and good ageing properties.

The R&D activity has been launched in 2007 with the final aim to build a detector of appropriate size, complying with the listed requirements, by using the Micromegas technology. Assuming a single measurement station in the inner muons wheel and three stations in the middle wheel, the total

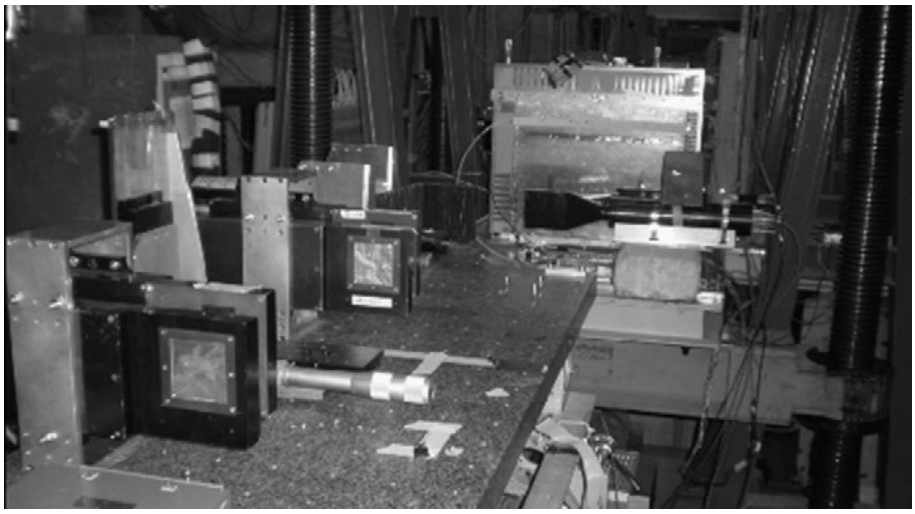


Fig. 1.8: Photograph of the test-beam setup, Micromegas is visible at the far right of the picture.

active area to be covered with the new detectors is $\sim 400\ m^2$. Micromegas offers good performance and high rate capability as well as competitive cost, compared to other detector technologies; however, they have never been used as muon detectors in large experiments at colliders.

A medium size bulk-Micromegas prototype, with an active area of 450mm x 350mm, was built at CERN in summer 2007. It was one of the largest chambers built at that time with this technology. In 2008 I participated in the tests of the the chamber at the CERN H6 beam line with 120 GeV pions beam from the SPS. Fig. 1.8 shows the experimental setup described in the following:

- Gas system: Three different gas mixtures were used: Ar:CO2:iC4H10 (88:10:2), Ar:CF4:iC4H10 (88:10:2) and Ar:CF4:iC4H10 (95:3:2) with a gas flow of 5 L/h corresponding to a gas volume exchange every 40 min.

- Trigger system: Trigger was provided by the coincidence of the signals coming from two scintillators vetoed with a third scintillator having a 3-cm diameter hole.

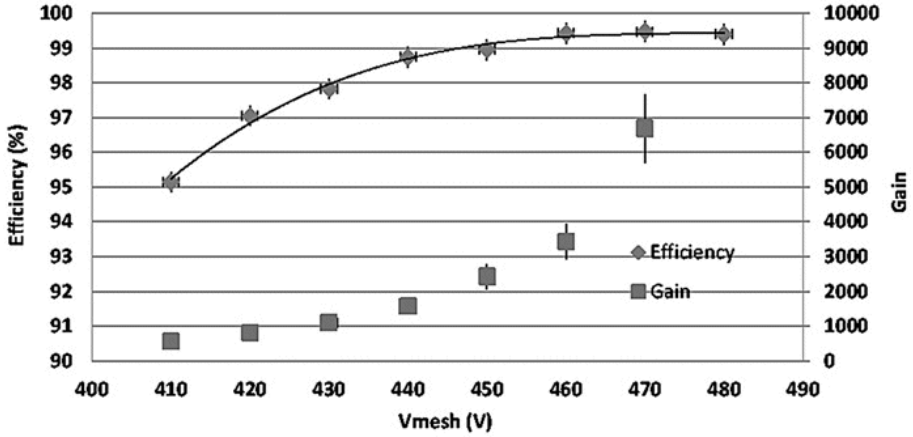


Fig. 1.9: Efficiency vs. Vmesh, on the same plot the gas gain is also displayed (right scale). Gas mixture Ar:CF4:iC4H10 (88:10:2).

- Tracking system [11]: Three Si-trackers 3cm x 3cm in size, provided each a double coordinate (x–y) measurement. The pitch of the silicon strips was 50mm. Both the tracking and the triggering systems were downstream the beam with respect to the Micromegas under test (Fig. 1.8).

- DAQ: The data acquisition system was the DATE system [12] which was configured to read out FEE cards [13] with Altro [14] chips.

The charge collected was studied as a function of the HV applied on the mesh and on the drift cathode. From these distributions the gas amplification was computed as a function of the mesh voltage. In this configuration the detector can stably operate at $HV_{mesh} = 455$ V corresponding to a gas amplification of about 3×10^3. The detector efficiency (shown in Fig. 1.9 together with the gain) was measured by identifying a hit on the chamber with a proximity cut of two strips with respect to the reconstructed track. The required performance is fully satisfied: efficiency bigger than 98% at the operating voltage.

For the spatial resolution, the residuals distribution of the Micromegas hits with respect to the reconstructed tracks with the Si-tracker is computed.

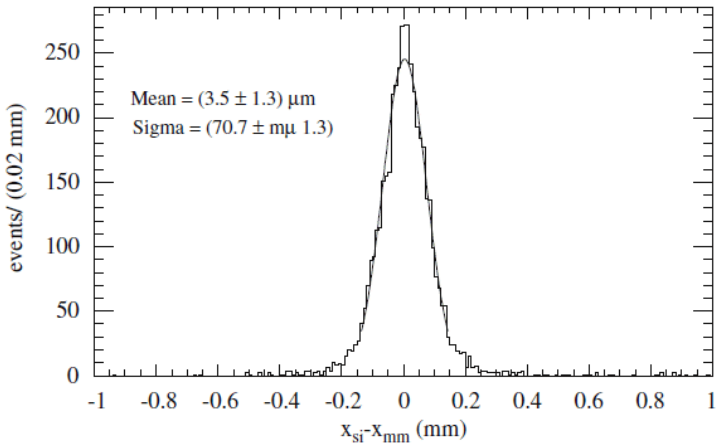


Fig. 1.10: Residuals distribution for strip pitch of 500 mm, gas mixture Ar:CF4:iC4H10 (88:10:2) and drift field of 200 V/cm.

The resolution is expected to be a Gaussian with three different contributions: the intrinsic Micromegas resolution, the Si-tracker resolution and the uncertainty of the track extrapolation. The last contribution is mainly from multiple scattering in the plastic scintillators (3 cm and 1 cm thick) located in between the Micromegas detector and the Si-tracker. Fig. 1.10 shows the distribution of residuals measured for strip pitches of 500mm and pion beam perpendicular to the chamber. The combined contribution of the Si-tracker resolution and multiple scattering was estimated to be (61±5) μm leading to an intrinsic spatial resolution of the Micromegas detector of σ_{MM} = (36±5) μm for strip pitch of 500mm and σ_{MM} = (24±7) μm for strip pitch of 250mm. These results completely satisfy the requirement for the upgrade of the ATLAS Muon system.

Techniques to reduce the sparks in micromegas are being investigated. Within RD51 there is a study aimed at demonstrating that it is possible to reduce the discharge probability and intensity, and protect the electronics, in a high flux hadrons environment by using a resistive anode plane or a segmented mesh. Several prototypes of 10 × 10 cm^2, with different pitches (0.25 to 2 mm) and different resistive layers (2 to 20 MΩ/cm^2) have been

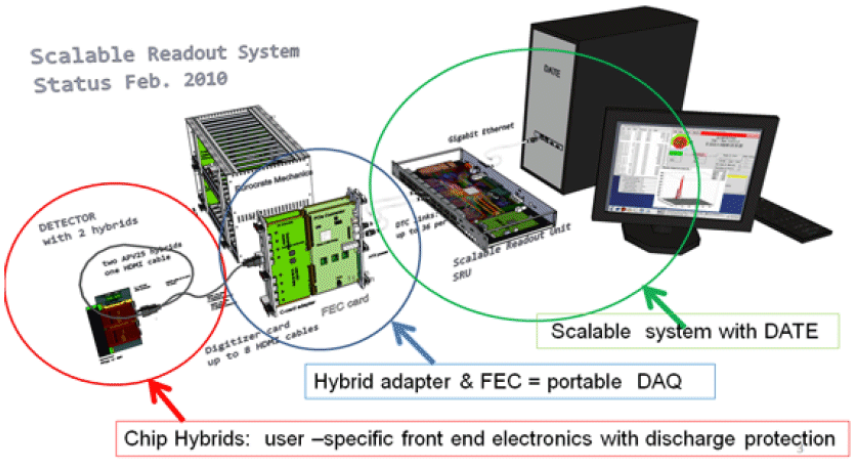


Fig. 1.11: Simplified physical representation of a SRS acquisition chain.

tested at CERN (on a 120 GeV $\pi+$ beam at SPS). Preliminary results show that the spark rate is lower at least of a factor of ten in the case of resistive strips. These results have to be confirmed in the future beam tests and more measurements have to be performed. This study will involve the topology of the sparks, the spatial resolution and efficiency of the prototypes and the radiation hardness and ageing of resistive layers.

1.4.3 The Scalable Readout System

The RD51 WG5 activities on common electronics for a multichannel readout system started its design phase in 2009 with the compilation of a chip knowledge base as a common base for electronics requirements for RD51 users. The large variety in readout requirements for MPGD detectors (signal polarity, trigger concept, timing resolution, radiation tolerance, analogue versus digital, choice of readout bus, number of channels, power and packaging, bandwidth, data formats, on-line software) does not allow for a simple common solution, unless one designs a system with the following general properties:

1. common chip link interface for different readout chips on detector-resident hybrids;

2. scalability from a small to a large system based on a common readout back-end with link interfaces for specific font-ends;

3. integration of commercial standards for a minimum of custom hardware modules between the chip front-end and the on-line system;

4. default availability of a very robust and supported data acquisition package;

5. flexibility to implement different readout architectures and trigger schemes.

The Scalable Readout System (SRS) [15] implements the above requirements in the following practical framework:

- test systems need a few channels/chips that can be connected to a turnkey and low-cost readout system with a standard on-line system;
- large system, like an LHC experiment, use the same readout back-ends but with a much larger number of front-end chips and front-end cards;
- typically detectors need some special features, for this SRS allows for adapter cards to be integrated into SRS with application specific logic.

In a simplified SRS acquisition chain (Fig. 1.11) we can identify three main blocks:

1. The DATE on-line system from the Alice experiment (DAQ PC). Optionally, this block is equipped with the Scalable Readout Unit (SRU) as a 40-fold Data Trigger and Control (DTC) link concentrator between the front-end system and the 10Gigabit port of DATE (the SRU is not needed for small systems).

2. Programmable Front End Cards (FECs). These cards are based on FPGAs and programmed with an application specific firmware. There are DTC link cables (CAT6) for transferring Data Trigger and Control

between the FECs and the SRU (or the DATE system directly). Also application-specific adapter cards may be interfaced to FECs for user-defined purposes like bias voltage control etc.

3. Detector specific components. These include chip hybrids with standard connectors (HDMI) on the detector and on the readout side. There are readout links (cables, fibers) for distances of tens of meters between the on and off-detector hybrids (which are connected to the FECs).

The only application-dependent components are the firmware running on the FECs and the detector specific hybrids, all the rest of system is fully portable. The SRS is under development and subject to change, however its portability makes it a promising common solution for all the measurements on MPGDs within the RD51 community and outside.

Chapter 2

Micromegas Detectors

2.1 Working Principle

Micromegas (Fig. 2.1) is a gas-based parallel plate detector with built-in amplification and a tolerance to high particle fluxes [16]. Inside the structure two conducting electrodes are at a distance of the order of $100\mu m$ and they define a region called "amplification gap". Since the Micromegas gap is small, it only requires moderate bias (e.g. few hundreds of volts) to achieve reasonable gas gain, whereas in other a parallel-plate gas detectors, the bias voltage required is higher (e.g. few tens of kV) because the amplification gap used is typically deeper (e.g. few cm). The cathode is made of a thin

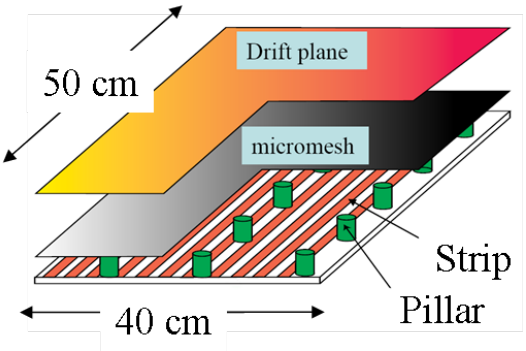


Fig. 2.1: Structure of a micromegas detector.

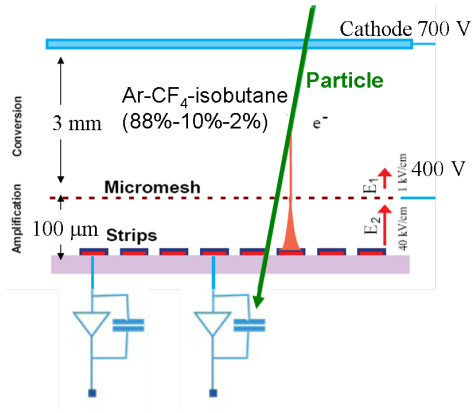


Fig. 2.2: Working principle of a micromegas detector.

metallic micromesh, few microns thick, while the anode consists of microelements (strips or pads) of a conductor, printed on a circuit board (made of an insulator). The mesh and the anode are kept separated by small non-conductive pillars, deposited by standard photo-lithographic methods on the anode. The pillars usually cover just a small part (1%) of the surface and therefore result in a small dead area of the detector. This technical solution permits the construction, at low cost, of large chambers up to 40x40 cm^2 and can be extended to larger surfaces. In many applications, an additional electrode (cathode) is placed parallel to the mesh in such a way to define two regions (Fig. 2.2) :

1. the conversion region, between the cathode and the micromesh, where electrons, released by any conversion process (ionizing particle or photon conversion) drift towards the mesh;

2. the amplification gap, between the micromesh and the anode elements, where electrons coming from the conversion region are multiplied in an avalanche process, resulting in a larger number of electron-ion pairs.

Detectable signals are induced on the anode elements and on the cathode mesh. Thanks to the small gap, the positive ions, created during the avalanche process, are collected in a short time (about 100 ns) on the micromesh and this results in a low dead time of the channel and allows the

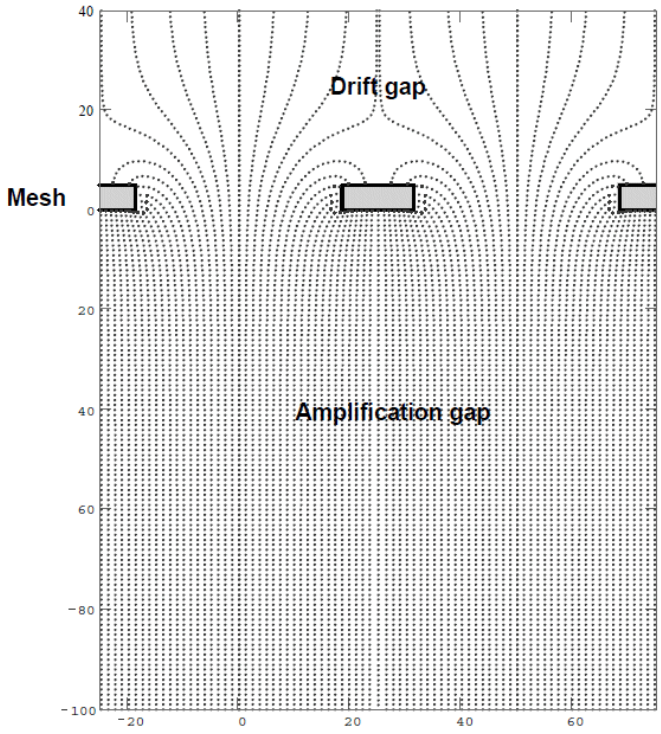


Fig. 2.3: Electric field in a micromegas detector.

detection of events with high track multiplicity. The read-out electronics can be either a charge pre-amplifier or a current pre-amplifier, respectively if one is interested in the total charge (e.g. for estimation of the energy loss of the particle) or to the timing of the signal (e.g. for triggering).

2.2 Electric Field Configuration

The shape of the electric field lines close to the micromesh has an impact on the performance of the detector and especially on the efficiency of the transfer of electrons to the amplification gap [17]. The electric field is almost homogeneous in both the conversion and the amplification gap. It exhibits a funnel-like shape around the openings of the micro-mesh: field lines are compressed towards the middle of the openings, into a small pathway, a

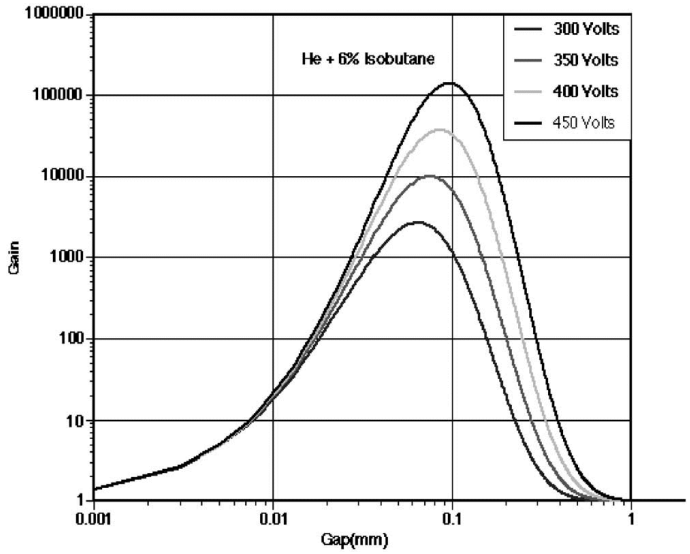


Fig. 2.4: Gain of a micromegas detector versus the amplification gap size.

few microns in diameter (for the usual grid sizes). The compression factor is proportional to the ratio of the electric fields in the two gaps. As an example in Fig. 2.3, I report the the field lines near the grid (with a 50 μm opening pitch) . The electrons generated in the conversion gap by ionization follow these lines and are focused into the multiplication gap where amplification process takes place. The ratio between the electric field in the amplification gap and that in the conversion gap must be set to suitably high values (>5) to permit an efficient electron transmission, and to reduce a part of ion cloud, produced in the avalanche, to escape into the conversion gap.

2.3 Gain

An interesting property of micromegas is that, thanks to its narrow gap, locally small variations of the amplification gap, due to, for instance, mechanical defects, do not induce gain fluctuation; they are compensated by an inverse variation of the amplification coefficient. In fact, it can be shown that:

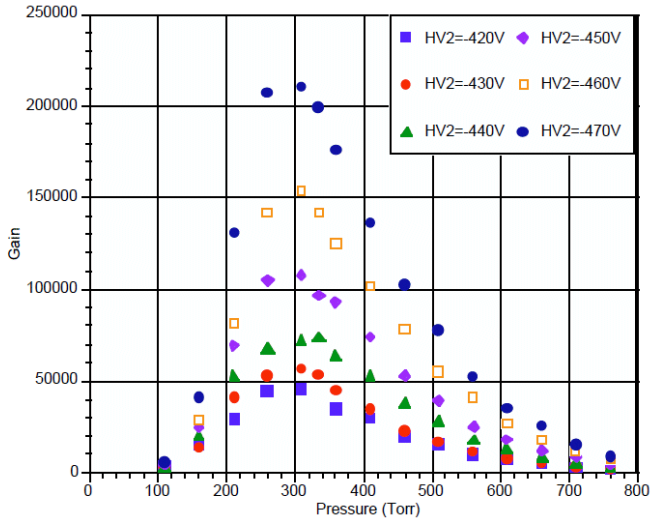


Fig. 2.5: Gain of a micromegas as a function of the pressure for various potentials applied on the mesh.

$$Log(M) = Apde^{\frac{-Bpd}{V}} \tag{2.1}$$

where M is the gain, A and B are parameters dependent on the gas mixture, d is the gap size, p is the gas pressure and V is the voltage at the mesh. Fig. 2.4 shows the gain as a function of the gap for a mixture of He + 6% Isobutane, for V=300, 350, 400, 450 Volts and $p = 1$ bar. One can see that M rises as d increases, then it reaches a maximum and finally falls at large values of d. The maximum gain is obtained by a differentiation of the equation (2.1), resulting in $\frac{d(Log(M))}{Log(M)} = (\frac{1}{p} - \frac{Bd}{V})dp$. The maximum value is for $d = \frac{V}{B}$ at $p = 1$ bar. The amplification gap chosen in this way depends slightly on the gas mixture; for a given applied potential, the multiplication factor is at the maximum in the range of gaps between 30 and 100 microns. This is the range currently used by micromegas detectors, because around the maximum fluctuations due to defects of flatness of the two parallel electrodes are minimized. It is quite difficult to verify experimentally the previous calculation, since one would need to measure the gain for a number of very narrow gaps. Fig. 2.5 shows the multiplication factor as a function of the

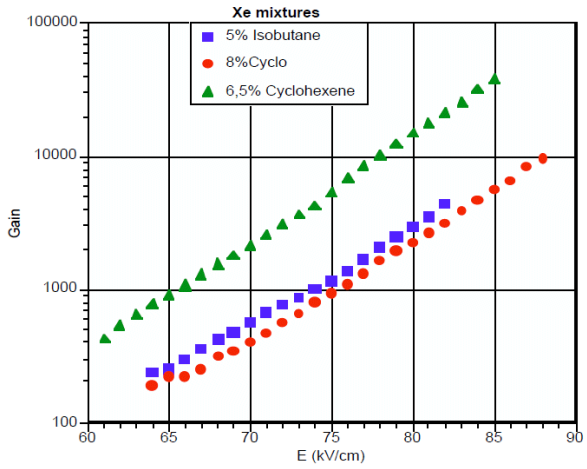


Fig. 2.6: Gain of a micromegas as a function of the electric field for various gas mixtures.

pressure for He + 6% Isobutane for a gap of 50 microns. The curve clearly shows that there is a maximum of multiplication at p = 500mbar, on the contrary the optimal operation of a conventional parallel plate avalanche chamber (d = 4 mm) is at pressures of the order of 10 mbar.

2.4 Gas Mixture

The maximum gain of a gaseous detector is a key parameter in a large number of applications. In particular, a detection of minimum ionizing particles (MIPs) requires a large dynamic range because of the Landau fluctuation of the deposited energy and the emission of heavy ionizing particles. The goal of a "good" detector is to achieve a stable operation before the breakdown, which corresponds to a total charge per avalanche approaching $10^7 - 10^8$ electrons (so called Rather limit). Micromegas has been tested with many gas mixtures. Mixtures of Argon-hydrocarbons with 5-10% addition of Isobutane achieve a maximum gain close to 10^5 [18, 19], and it grows three times higher with a small amount of Cyclohexane. The presence of CF4 to the mixture improves the time resolution and the total deposited energy [20].

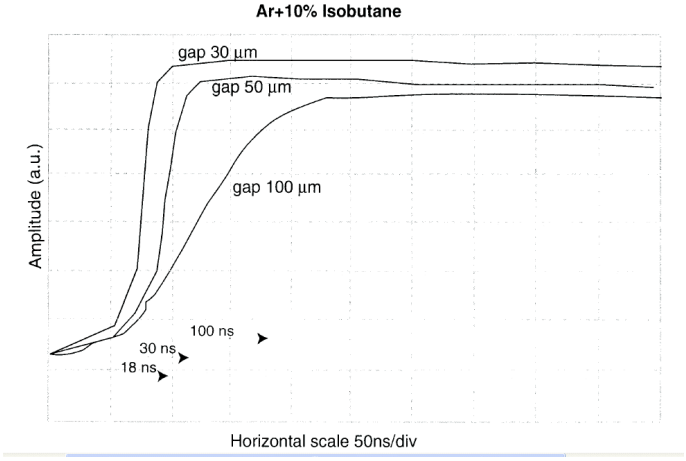


Fig. 2.7: Simulated signal at the output of a charge preamplifier connected to a micromegas for three amplification gaps: 100, 50 and 25 μm.

Neon or He mixtures plus hydrocarbons permit increasing the total charge per single avalanche to the highest Rather values (about 10^8 e^-). Many tests have been performed in order to optimize the operation of micromegas as a function of the gas mixture. One of the results is that the maximum achievable gas gain increases with heavier hydrocarbon quenchers with lower ionization potentials. As an example, Fig. 2.6 shows the gas gain measured as a function of the applied voltage and for various quenchers added to Xenon. The maximum achievable gas gain is increasing as one goes from the Isobutane (4500) to Cyclohexane (10^4), and finally to Cyclohexene ($3 \cdot 10^4$). Such high gas gain gives the required margin factor when a detector has to cope with very-high X-ray environments, or at high-pressure operations.

2.5 Signal Development

The signal is induced on the read-out strips by the movement of electrons and ions. The charge signal is mainly due to the ions drifting toward the micromesh electrode, which takes place typically within 100ns, depending on the amplification gap and the gas mixture. Figure 2.7 shows the preamplifier response for an Argon + 10% Isobutane gas mixture and for various gaps.

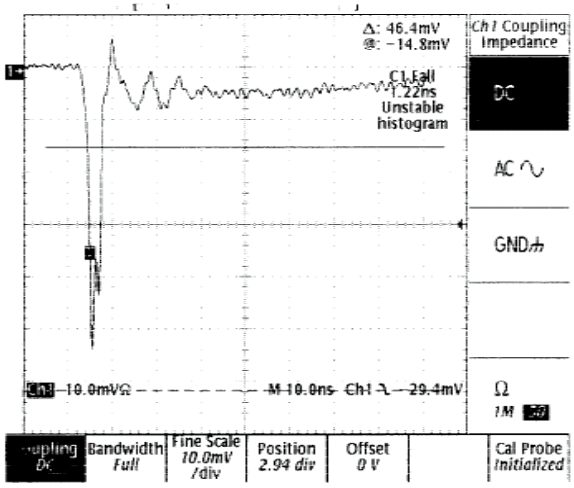


Fig. 2.8: Signal at the output of a fast current-preamplifier connected to a micromegas.

The rise time of the signal can be reduced by shrinking the gap (the rise time decreases from 100ns to 18 ns as the gap decreases from 100μm to 30μm) and using a higher ion mobility gas carrier (Ne or He). Therefore micromegas can be used with low-noise charge pre-amplifiers without loss due to ballistic deficit, which occur in other micro-strip devices; choosing the right amplification and the right gas mixture, the rise of the detector can be compatible with the shaping of the charge amplifier.

Due to the higher drift velocity, the electron current is larger and faster (about 1ns instead of 100ns for the ion signal). In fact, there is a very fast rise of the signal, followed by a tail due to the ion drift. Such a fast edge is quite difficult to follow, but is within reach with present fast current amplifiers.

Fig. 2.8 shows the signal from a micromegas through a current preamplifier with a rise time smaller than 1 ns. The fast signal has a rise of 1ns and amplitude ten times higher than the ion tail. Such fast signals will allow the development of novel drift chambers or small time projection chambers with a time resolution below 1ns. The time jitter of micromegas depends on several parameters: the fluctuations in the time of arrival of the electrons at

the entrance of the holes of the micromesh, the ionization density and the longitudinal diffusion of the electrons. The intrinsic resolution due to the amplification structure is less than 1 ns [21].

2.6 Rate Capabilities

Investigations, using high flux of incident beta particles, low energy protons or high energy muons, have shown that micromegas can cope with very high rates of these particles. However, the detector show a high discharge rate when the incident beam is composed by high energy hadrons. The rate is proportional to the hadron flux. A possible cause might be that large ionization deposits trigger discharges. These deposits are probably released by recoil nuclei produced by charged particles, especially hadrons, traversing the detector. Ionization losses are proportional to Z^2, so the recoil nuclei resulted from elastic or quasi-elastic interactions, with energies in the MeV region, are quite efficient to produce heavy ionization charge in the gas. For example a nuclear interaction on the Argon nuclei might produce fully ionized Argon nuclei having a 1 MeV kinetic energy. The whole energy will be lost within 100 microns producing about 10^5 electron-ion pairs in the conversion gap. This huge quantity of charges is again multiplied by the detector gain in the amplification gap, exceeding the Rather limit (a few 10^7 e^-), thus triggering a breakdown. Taking into account the nuclear collision length for hadrons, which is about 10^4 cm in Argon, the probability to produce such process in the conversion gap is of the order of 10^{-6}. This spark probability produces a serious limitation when the detector has to deal with very high hadron fluxes. In the case of muons the corresponding cross section is several order of magnitudes lower, therefore the probability to induce sparks is negligible. Micromegas have been tested with $\sim 5 \cdot 10^7$ muons in a small area (a few cm^2) without serious loss of its performance. It is quite important to remark that in micromegas the sparks do not propagate in the whole detector, instead they are limited in area to maximum of a few mm^2, and their duration does not exceed 100 ns. The possible consequences of discharges vary from a high dead time to the destruction of the electronics

or even the damage of detector itself.

2.7 Spatial Resolution

For the tracking of charged particles in high energy physics (HEP), a detector should offer a spatial resolution as good as possible. The conversion gap appears then to be the main source of degradation of the detector performances. Several groups [22], using various Micromegas configurations in terms of strip pitch, amplification gap and operating gas, have investigated the space resolution of the detector, with a conversion gap of 3 mm. The spatial resolution of particles at normal incidence is always limited by the transverse diffusion of the drifting electrons which depends on the gas mixture used. All the results stand under 100 μm (σ): The best accuracy, 12 mm (σ) was obtained using an amplification gap of 100 μm; a strip pitch of 100 μm and a gas mixture of CF4-isobutane (80–20%). The conclusion is that the space accuracy of Micromegas can satisfy the needs of most of the HEP experiments for tracking purposes. There are two different configurations of the detector which are proposed in HEP experiments: the most common one has a small conversion gap and the second one in TPC mode has a large conversion space. The choice of the gas mixture together with the anode elements and the associated front-end electronics depends on the space and/or time resolution needed and on the particles environment (nature, flux, multiplicity). The use of the amplification "a la Micromegas" allows safe operation in a high rate environment. In the mode with a small conversion gap, the detector is placed in such a way that most particles cross the detector plane at a small incidence angle.

2.8 Some Experiments Using Micromegas

The usual competitors for micromegas are silicon microstrips detectors, since they have similar characteristics. However, the Micromegas detector comes in some cases to be competitive with the silicon micro-strip detector with quite a few advantages: higher radiation resistance, lower cost and lower

detector mass. An an example, the Alice experiment at future LHC envisages to implement Micromegas detectors in the front part of the calorimeter. This pre-shower detector consists of a sandwich of two Micromegas chambers surrounding a passive lead converter [23]. A more advanced project is the tracking of particles in the COMPASS experiment at CERN [24]. In this experiment, the detected particles are mainly muons at high rate $2 \cdot 10^7$ Hz with a relatively small amount of hadrons $5 \cdot 10^4$ Hz. As the limitation in gain with muons is three order of magnitude lower than with hadrons, the palette of gas mixtures, leading safe operation of the detector, is quite broad. Large area Micromegas chambers 40x40 cm^2 have been constructed and successfully tested in the COMPASS beam [25]. In the mode with a large conversion space, some tests have been made in laboratory using a small TPC chamber coupled to a Micromegas amplification structure, in view to the charged particles detection in the future TESLA accelerator. In such a detector, the longitudinal diffusion of the gas becomes a crucial limitation of the space resolution. Another great motivation of using Micromegas detector is the reduction of ions escaping into the drift volume. Using large ratio between the amplification and drift field, this reduction can be of the order of 99%.

2.9 Requirements for a Compact Data Acquisition System

My thesis work is in the framework of a study of feasibility of a data acquisition system (DAQ) for the measurement of ionization charge Q and of pulse height ΔV of signals from micromegas. The system to be built must be able to acquire, elaborate and store the signal coming from a micromegas detector. The system must offer the following features:

1. digital data acquisition, in order to permit a simple storing, elaboration and consecutive transmission;

2. low logic footprint, in order to scale the system for multi-strip monitoring;

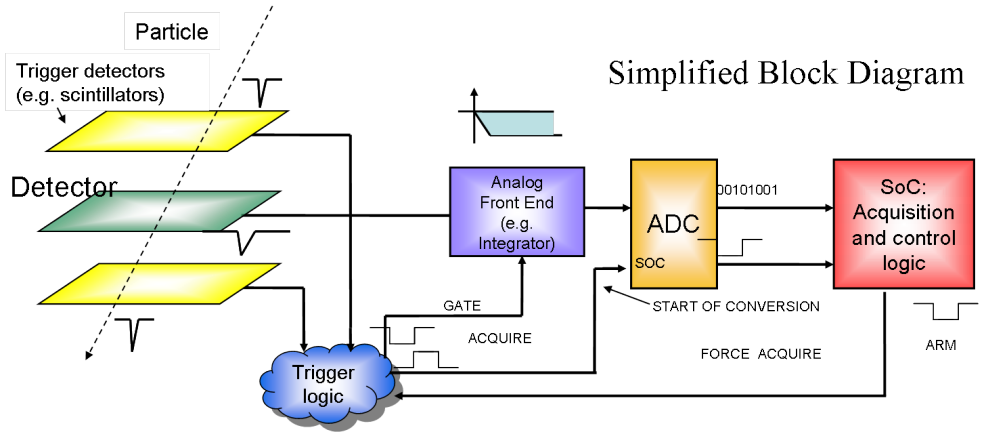


Fig. 2.9: Data acquisition system (DAQ) for the study of ΔV and Q of a pulse coming out from a micromegas detector.

3. autonomy, the system must require a minimum number of external electrical components;

4. simple management of the system and access to acquired data, for instance through the interface to a PC;

5. possibility to update the set of functions offered by the system, in order to fix bugs or add not-initially-foreseen functionalities.

A classical scheme for the architecture of a data acquisition system from a detector is shown in Fig. 2.9. The signal from a strip of the detector is pre-amplified and sent to a a peak detector, in the case one wants to measure the pulse height, or sent to an integrator for measuring Q. The output of an analog circuit is connected to the input of an ADC. Some trigger detectors, placed over and below the micromegas, are connected to a trigger logic. This produces a signal (GATE) which enables the peak-detector (or the integrator) and a signal (ACQUIRE) which makes the conversion start by the ADC. The output of the ADC is connected to a digital acquisition and control logic, which provides a veto signal (VETO) for the trigger logic and can force the start of the conversion from the ADC with a calibration purpose (FORCE_ACQUIRE signal). The control logic manages the inter-

facing with the ADC and the trigger logic. The acquisition logic executes simple arithmetic elaborations on the acquired data, like the pedestal subtraction, the discrimination and the calculation of a mean value. In fact, in general, the digital data from an ADC in absence of signal is not zero, and is called pedestal. In order to correct this effect it is necessary to subtract the pedestal from the data after its digitization. During a calibration phase, the value of the pedestal is acquired and stored by the system, afterwards when running an acquisition, it is possible to subtract the pedestal from the digitized data. The type of operations to execute pushes toward the use of a microprocessor, which in addition to a programmable memory allow to re-define the functions of DAQ system, even after its realization. The use of a commercial microprocessor would lead to the construction of a board with memories and I/O interfaces, but this would not be compatible with the scalability requirement. On the contrary, the development of an integrated system dedicated to the application, would allow to reduce the size. Moreover, a dedicated system would include some auxiliary peripherals optimized for a specific function, which would allow the elaboration of the data in real time and would allow to save precious software resources. For example, it would possible to include peripherals for the histogramming of the acquired data, the pedestal subtraction and the ADC control. For this reason, in this thesis work I designed a dedicated System-on-Chip (SoC).

According to the previous remarks about the SoC functionalities, the main elements of the SoC are :

1. a microprocessor;
2. memory banks for the executable code and for the acquired data;
3. peripherals for the histogram-building, the pedestal subtraction and the ADC management, in such a way to free the CPU from these operations and optimize the performance.

The internal architecture of the SoC realized in this thesis work is discussed in the next chapter.

Chapter 3

Defining the System-on-Chip Architecture

In this chapter I define the System-on-Chip architecture and I introduce the custom microprocessor I have designed.

3.1 System-on-Chip Planning

I recall that the initial experimental problem is the study of feasibility of a DAQ system for the measurement of the signal coming from a Micromegas detector.

Considerations on size, connectivity, re-programmability and on the type of operations that the system must execute indicated that the most suitable solution for the acquisition and control logic is a dedicated, microprocessor-based System-on-Chip.

A possible architecture is the one in Fig. 3.1. In the block diagram there are the three fundamental elements enumerated at the end of the previous chapter: microprocessor, RAM and dedicated peripherals. From now on, we will refer to these peripherals with the term "DAQ peripherals". For the sake of simplicity, in this block diagram the RAM is aside a logic which allows the interfacing with the different DAQ peripherals. The microprocessor accesses the RAM through an internal bus and it is able to read the program code, the

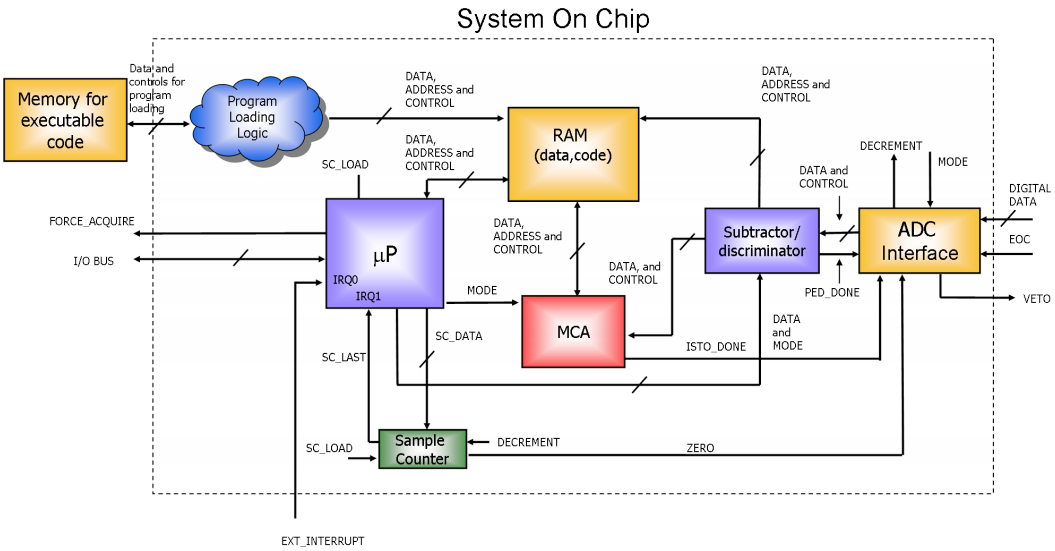


Fig. 3.1: Simplified block diagram of the System On Chip.

data acquired from the ADC and the data from the multi-channel analyzer (MCA). The microprocessor can be interfaced to external circuits through its IO bus.

An example of operation of the SoC is the following: the microprocessor starts the data acquisition; after having acquired N digital samples, which the MCA histogrammed into the memory, it calculates the mean and the standard deviation. At this point, it restarts the acquisition and in the meanwhile it shows the calculated quantities on an LCD display or it transmits to a PC trough its I/O bus. If it is necessary, the user from the PC can request to the processor all the acquired data together with the calculated values. Moreover, it is possible to read the data and store them as they are read from the ADC, without executing any processing.

The interfacing of the system with the external devices can be summarized as follows. The system receives the digital data from the ADC and when it receives an EOC (End Of Conversion), it processes and then stores the data.

During the storing, the microprocessor flags the completed reading of

the data to the trigger logic, by asserting the ARM output for one clock cycle. The system can force the ADC to start converting with the signal FORCE_ACQUIRE. The input EXT_INTERRUPT allows the integrated microprocessor to receive interrupt requests from the outside and the I/O bus allow the connection with external electronic devices.

The ADC Interface is the subsystem devoted to the reading from the external ADC. The output of the ADC interface is connected to the Subtractor/Discriminator, which subtracts a pedestal to the data or suppresses it, if it is below to the previously set threshold.

The subtractor/discriminator contains two registers: TH (**TH**reshold) which stores the minimum threshlod for the discriminator and OF (**OF**fset) storing the pedestal to subtract to the data. The subtractor is connected both to the MCA and to the RAM, this way the data can be written directly into the RAM or histogrammed in real-time. The selection between the two modes, is performed through the MODE signals. The multichannel analyzer composes an histogram in memory, considering every location as a bin of the histogram. In order to histogram the X data, the multichannel reads the memory location X, stores its content (Y) in a register, increments it and finally writes the result ($Y + 1$) back to the location X. The ***S**ample **C**ounter* (SC) is pre-loaded by the microprocessor with the number of data to be acquired. Each time a sample is read, the ADC interface asserts a signal (DECREMENT) in order to decrement the content of SC. When SC equals zero, the signal SC_LAST generates an interrupt request for the microprocessor. By loading via the value N in SC, the acquisition stars and N samples are acquired. In order to stop the acquisition before the reading of all the data, it is sufficient to to load zero in SC. In this case, the signal ZERO becomes high and inhibits the acquisition of further samples. Every time the ADC interface receives an EOC, it asserts the signal ARM for one clock cycle in order to flag that the reading of the data completed. The *program loading logic* is dedicated to the loading of the microprocessor program in memory; the program to be loaded comes from the outside through a serial connection, the logic parallelizes it and loads it into the memory. The choice to execute a serial loading is very useful in order to minimize the number of IO pins of

the device, which otherwise could set a minimum limit on the VLSI device surface, with a corresponding growth of the cost.The microprocessor can receive interrupt requests, can communicate with the outside trough an IO bus and can force trigger-less acquisitions through the FORCE_ACQUIRE signal.

Many measurements performed with oscilloscopes, which have an 8-bit vertical dynamic range, have shown that such a range is sufficient for the digitization of the signal from the front-end charge or current preamplifier. Also, the conversion time of 8-bit ADCs is low enough to be used for our application and achieve conversion rates of the order of a few MHz. Moreover, 8-bit ADCs are the cheapest and the ones which there is more choice on the market for. This argument suggests the use of a microprocessor with an 8-bit data-path.

The SoC realized in this work foresees the reading of the data from one channel. Anyway, in case of future developments, it could be very useful to monitor several electrodes of the detector. In order for this to be possible, it is necessary that the SoC would be clonable several times in the same ASIC, with each channel monitored by its dedicated CPU. This modular approach is scalable to any number of channels, the only limit being the size of the integrated circuit. In order for this to be feasible, the CPU logic footprint must therefore kept as low as possible. In fact, a system requiring too many resources would lead to an increase of the required area, with a correspondent increase of the cost. This reasoning is a further incentive toward the development of an 8-bit microprocessor, which could be implemented with a reduced amount of logic resources with respect to larger data-path architectures.

Given that the heart of the SoC consists of the microprocessor and that the organization of all the interface logic depends on it, the next paragraphs will be dedicated to define its architecture. The modern design of digital systems includes the use of Hardware Description Languages (HDLs). In this work, the whole SoC has been described and simulated by means of an HDL.

3.2 The LiloBlaze Microprocessor

For the SoC I am designing a general purpose microprocessor would have not been adequate. I needed to design a custom architecture with integrated memory providing direct memory access to the DAQ peripherals and suited for their control. I prepared a fully synthetizable, and therefore portable, VHDL description of a CPU, which I called LiloBlaze.

When designing the microprocessor, although I needed a fully custom architecture for my processor LiloBlaze, I decided to make part of the instruction set and of the architecture compatible (wherever possible) with the one of an existing CPU. This approach allowed me to use all the code developed for that machine, included its software tools, and to have part of the documentation already available. I found the CPU most suited for this purpose to be the PicoBlaze CR [26] processor designed by Xilinx. The LiloBlaze microprocessor I have designed has more advanced hardware resources and a larger instruction-set than PicoBlaze CR, but I preserved full forward compatibility with it (i.e. a PicoBlaze program runs flawlessly on LiloBlaze).

LiloBlaze is an 8-bit RISC micro-controller and its main features are:

- timing predictability, all the instructions, under any condition require two clock cycles to be executed, therefore the execution of a constant number of instruction per second is guaranteed, which is a very desirable feature in real-time systems;
- maximum working frequency $\sim$ 100 MHz (50 MIPS[1]) in Virtex II V1000 -4 FPGA, with the XST synthesizer;
- very low logic footprint $\sim$ 4% of a Virtex II V1000 -4 FPGA.

Internal Architecture

The main features of the LiloBlaze architecture are:

- 8 general purpose 8-bit registers;

[1] **M**illions of **I**nstructions **P**er **S**econd

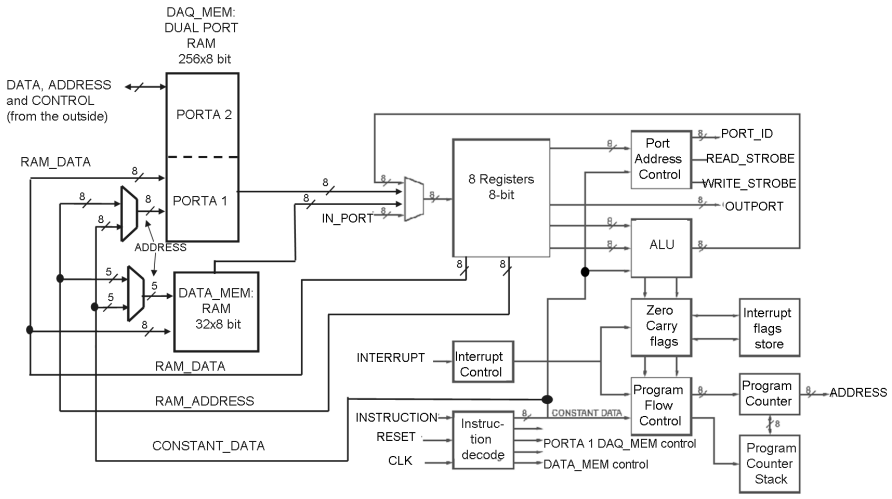


Fig. 3.2: Internal architecture of the LiloBlaze microprocessor.

- 34 instructions;
- 16-bit wide instruction word;
- 256 instructions maximum program length;
- 8-bit program counter;
- 8-bit ALU with zero and carry flags;
- 256 input/output ports directly and indirectly addressable;
- 16-level hardware call/return stack;
- one interrupt request input (maskable);
- two integrated RAM banks
  - DAQ_MEM bank, 256 8-bit words, dual-port, used for storing the histogram of the acquired data or the data directly (depending on acquisition mode)

  - DATA_MEM bank, 32 8-bit words, which serves as a support memory for the program running on microprocessor.

Fig. 3.2 shows a block diagram of the internal architecture. The 8 8-bits registers (s0..s7) are all equivalent, no one of them is dedicated to a specific function.

The 8-bit ALU performs the following operations:

- arithmetic
  - addition, with possibility to include the carry bit into the sum;
  - subtraction, with possibility to include the carry bit into the subtraction;
  - arithmetic rotate;
  - arithmetic barrel shift;
- logic
  - LOAD;
  - AND;
  - OR;
  - XOR;
  - COMPARE;
  - logical rotate;
  - logical barrel shift;
- Memory access (with direct and indirect addressing)
  - FETCH;
  - STORE.

All the operations have at least one register as an operand, which after the operation stores its result. The other operand can be an 8-bit numerical

constant or another register. The possibility to specify a constant as an operand of the instruction expands the simple instruction set. For instance the instruction XOR Sx, FF is equivalent to one-complement dedicated instruction (where Sx indicates of the registers from S0 to S7). Thanks to the possibility of including the carry bit in addition to the two operands, the additions (subtractions), in extended precision, are easily realized by performing several additions (subtractions) in sequence. Operations executed by the ALU affect the carry and zero flags. Their state in turn affects the conditional jumps, in this way it is possible to control the program flow. Jump instructions allow to jump to a memory location only directly.

In order to enable an efficient coding, through the division of the program in nested sub-routines, I have designed an automatic 16-level hardware stack. The program counter is automatically saved on a CALL instruction and restored on a RETURN instruction.

LiloBlaze has 256 8-bit IO ports. It has two 8-bit buses for data IO: IN_PORT for the input and OUT_PORT for the output. The number of IO port is specified on the PORT_ID output bus. The WRITE_STROBE and READ_STROBE outputs notify respectively the completed writing and reading of a data from the IO port.

The processor has only one maskable interrupt input (INTERRUPT). If interrupts are enabled, a high level[2] on the INTERRUPT input forces the jump to the last program location, the 256th. Before the jump, the only context information being automatically saved are the program counter and the carry and zero flags. At the return from the interrupt they are automatically restored. The latency for an interrupt request can be of 5 or 6 clock cycles.

The access to the DATA_MEM memory is controlled by the instruction decode block: the address to the memory can be the content of a register[3] or a 5-bit constant decoded in the instruction word. The input data is read by a microprocessor register, while the output data is stored there. An analogous

[2]In order for the interrupt to be detectable, the input must be steadily high for at least two clock cycles.

[3]Since the memory has 32 locations, I consider only the 5 less significant bits.

topic is valid for port 1 of DAQ_MEM memory, with the difference that the address space is 8-bit wide. I have chosen to use a double port memory for the DAQ_MEM, one of the two ports is used inside LiloBlaze (port 1), while the other one is accessible from the outside of the processor by the DAQ peripherals (port 2). My choice is preferable with respect to the use of a single port memory, because in this case the access to the memory should have been performed by means of the microprocessor, using its resources for each read/write of each DAQ peripheral. In order to allow the access to the memories, I included dedicated instructions: FETCH0 and STORE0 for the reading and writing in DATA_MEM and analogously FETCH1 e STORE1 for DAQ_MEM. The FETCH0 supports two different syntax-es:

```
(1) FETCH0 sX,k
(2) FETCH0 sX,sY
```

where sX and sY are registers internal to the microprocessor (from s0 to s7) and k is a 5-bit numerical constant. Syntax (1) allows to copy the content of the k location of DATA_MEM to the sX register. By means of syntax (2), the address of the memory to read from is specified in the content of sY register. The STORE0 instruction is identical to FETCH0, with the only difference that it writes into the memory instead of reading. FETCH1 and STORE1 instructions are analogous to FETCH0 and STORE0, with the difference that they access the DAQ_MEM and consequently they foresee an 8-bit address. All the instructions for accessing the RAMs support both direct and indirect addressing, therefore it is possible to program using arrays and pointers in the assembly code. The LiloBlaze ALU includes a barrel shifter[4], such a circuit can be very useful since it allows to multiply and divide by powers of 2 in a single clock cycle. For this reason, I integrated a barrel shifter in the LiloBlaze ALU and its functionality are accessible trough:

- BSHR0 and BSHL0 instructions, which execute respectively the logical

[4] A "barrel shifter" is a digital circuit for shifting an n-bit data word of a number of positions within 0 and n in a single clock cycle.

shift toward right and left and they clear as many bits as the number of the shifted bits, respectively starting from the msb or from the lsb;

- BSHR1 and BSHL1, analogous to the previous ones, but they set the bits instead of clearing them.

The possible syntax-es of the BSHR0 instruction are:

```
(3) BSHR0 sX,k
(4) BSHR0 sX,sY
```

with symbols analogous to the syntax of the FETCH0 instruction. In case (3) the sX register is shifted by k-places toward right and the k most significant bits are cleared. In case (4) the content of sY specifies the amount of bits to shift. The BSHL0 instruction is analogous in the functionality and in the syntax to BSHR0, but shifts toward left, instead of right. BSHR1 and BSHL1 have a syntax identical to the one of BSHR0 and BSHL0.

A compare instruction for the microprocessor is not absolutely necessary, provided that there is some other way (e.g. the subtraction instruction) to alter the microprocessor flags and therefore control the program flow. However, the lack of a compare instruction may lead to problems similar to the one shown in the following example. Let us suppose we would like to output the content of the first 10 locations of the DATA_MEM memory:

```
; Program for the output
; of the first 10 locations of the DATA_MEM memory
main:

     load s1,0A          ; load 0A in s1
     load s0,00          ; load 00 in s0
     continue:


          add s1,s0        ; compensate the subtraction executed before the jump nz
          fetch0 s2,s0     ; reads a location from DATA_MEM at the address pointed
                           ; by s0 and stores it in the s2 register
          out s2,00        ; outputs the data on the 00 port
          add  s0,1        ; adds 1 ad s0
          sub s1,s0        ; subtracts s0 from s1
```

```
        jump nz,continue    ; if s1 ≠ s0 the zero flag is '0' and the jump is taken,
                            ; else the instruction continues with the next instruction

    jump main
```

Note that it is necessary to restore the contents of s1 after the subtraction instruction `sub s1,s0`, by executing the sum `add s1,s0`. To avoid using this expensive, in terms of clock cycles, and counter-intuitive way to program, I introduced the COMP instruction which allows the processor to compare two operands without changing the contents of a register. Using the following syntax:

```
(5) COMP sX,k
```

the instruction compares the content of a register with a constant. The instruction affects only the zero and carry flags. The carry flag takes the logical value '1' if sX <k otherwise it takes the logical value '0'. The zero flag takes the logic value '1' if sX = k and the logical value '0' otherwise. Using the syntax:

```
(6) COMP sX,sY
```

in a completely analogous way to the previous case, the COMP instruction compares the content of two registers sX and sY. From an implementation perspective, subtraction and comparison instructions are identical except for the fact that in the second case, the CPU does not enable writing to the destination register.

Other than the VHDL description, in order to facilitate the generation of executable code, I developed an assembler program (in C language). The assembler is a command line program that allows the programmer to move from a program in text format (assembler code) to an executable code for the microprocessor. The assembler also performs an analysis of the syntax of instructions and flags errors, thus facilitating the debugging of the code. If one modifies the VHDL description of the processor, for instance by adding an instruction, it is possible to modify the assembler to support it.

Chapter 4

SoC Architecture

The present chapter describes in details the architecture of the SoC designed for this thesis work. First, I present the overall architecture of the system, illustrating which components it includes and how they interact, in the second place I discuss the internal structure of each single peripheral. In the following descriptions, for the sake of simplicity, the reset and clock inputs are not mentioned, however their presence is assumed.

4.1 ParticleBlaze

I named the system-on-chip designed in this thesis "ParticleBlaze" (Fig. 4.1), from the name of the custom microprocessor I embedded in the SoC. As mentioned earlier, the LiloBlaze processor includes two RAM banks: DATA_MEM and DAQ_MEM. During data acquisition, the pedestal subtractor (or the multi-channel analyzer) writes into the DAQ_MEM memory. In this phase, the processor can only read from memory, but when the acquisition is stopped it can also write. The "Program memory" is a RAM containing the executable program for the microprocessor. At start up, the program code is read from an external PROM and loaded into the RAM by the XPROM reader (more details will be provided in section 4.6).

During the execution of the program, the processor sends the address of the instruction to be executed to the program memory on the ADDRESS

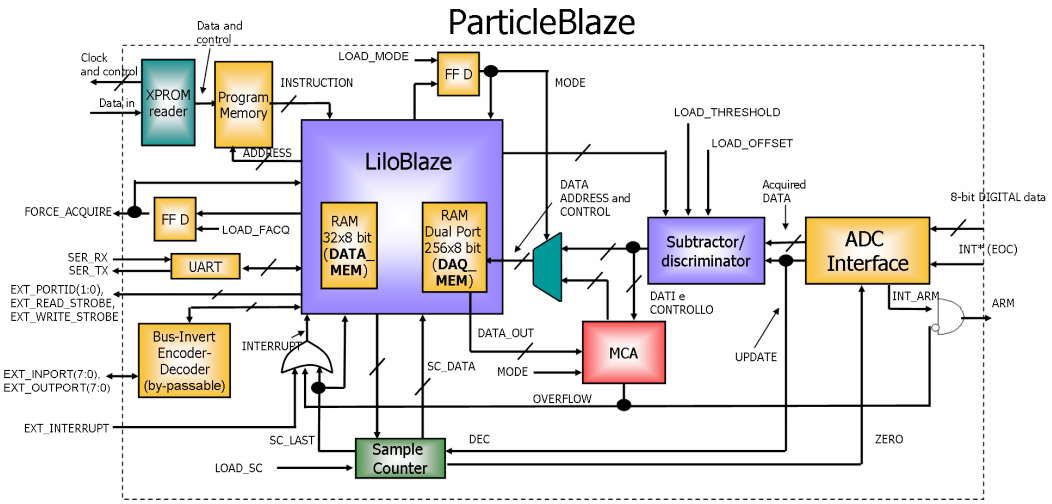


Fig. 4.1: Simplified block diagram of the architecture of ParticleBlaze.

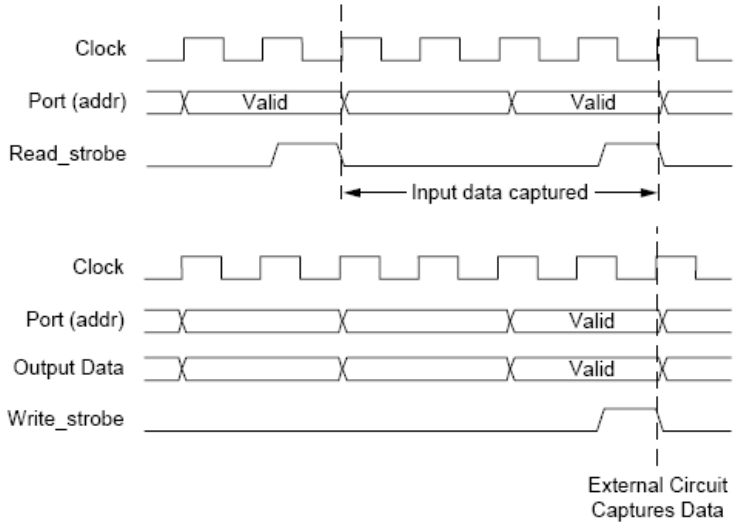


Fig. 4.2: Up: timing diagram of a read operation from an IO port. Down: timing diagram of a write operation.

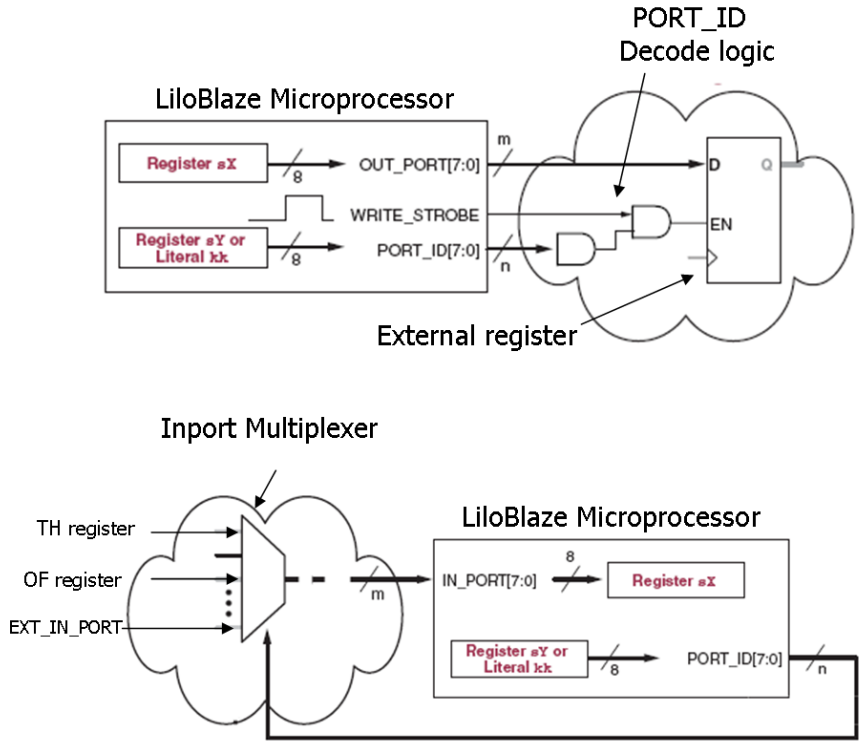


Fig. 4.3: Up: logic for writing into an external register by means of the output port. Down: logic for reading from external registers by means of the input port.

bus and receives the actual op-code on the INSTRUCTION bus.

LiloBlaze provides two IO instructions: INPUT and OUTPUT. The INPUT instruction allows the CPU to transfer an 8-bit word from the INPORT bus to one of the registers. The OUTPUT instruction allows the CPU to transfer the content of a register to the output bus. For both the instructions, the port identifier (PORT_ID) can be specified directly, with a constant, or indirectly, with the content of a register. Figure 4.2 shows the timing of IO signals for INPUT and OUTPUT instructions. The management of the DAQ peripherals is achieved by means of some reserved port identifiers (from $(04)_{16}$ to $(FF)_{16}$). Ports from $(00)_{16}$ to $(03)_{16}$ are available for the interfacing with peripherals external to the SoC. A 3-bit RID (Register Identifier) is assigned to each register of the DAQ peripherals. All the registers have the input connected the OUTPORT bus. The signal for the loading of each register is active if the PORT_ID equals the RID of the register and if the WRITE_STROBE signal is asserted (Fig. 4.3). Analogously, reading from the register is performed through a multiplexer on the IN_PORT, which has the inputs connected to the outputs of the registers and the selection signal connected to the PORT_ID (Fig. 4.3). Therefore, reading (writing) a register is performed by means of a INPUT (OUTPUT) instruction with a specific PORT_ID.

Reserved ports allow the programmer to set the status of control signals of the DAQ peripherals (for instance MODE) and of writing and reading in their internal registers, in such a way to set the acquisition parameters. For example, an IO port allows to read and write into the **S**ample **C**ounter (SC), in order to set the number of samples to acquire and starting the acquisition.

In order to allow the microprocessor to receive an interrupt request at the end of the acquisition (SC_LAST), without losing the possibility to receive requests from the outside, I added the support for an additional interrupt input. In order to disambiguate if the request came from the outside, the processor can read the status of the SC_LAST flip-flop. An interrupt request is also generated in case of overflow by the MCA: this happens when a bin in the histogram overcomes 255 events. The processor can distinguish this request from the previous other two by reading the number of samples to be

acquired in the SC register.

ParticleBlaze exports the non-reserved IO ports toward the outside, i.e. it makes the input and output buses and the 1 and 0 bits of PORT_ID bus (EXT_PORT_ID(1:0)) accessible from the outside. On the external IO buses I have added an optional bus-invert encoder/decoder. This block can be enabled or by-passed by writing at a dedicated PID.

The bus-invert coding [27] is a very simple and effective technique for the reduction of the switching activity of parallel buses and therefore of the power dissipated by the drivers. A bus-invert encoder conditionally inverts the word in the attempt to minimize the Hamming distance between the word on the bus and the next one to transmit. In order to mark whether or not the transmitted word is inverted, the encoder uses one line of the bus (the "invert" line). The decoder uses this line to restore the original form of the word. The coding provides a 50% reduction of the peak switching activity, while the average reduction depends on the bus size, narrower buses perform better than wider ones. The bus-invert coding has been demonstrated to be effective also in the reduction of the noise generated by the drivers [28, 29] rather than just in lowering the dissipated power.

I designed and integrated a very simple UART in the ParticleBlaze architecture which is accessible via a LiloBlaze reserved PID. The UART I designed is inspired to a open source design [30]. It allows ParticleBlaze to be interfaced for instance with a Personal Computer or with a serial terminal, a very useful feature to send commands to the SoC or for it to display information.

In the following paragraphs, I describe the architecture of the peripherals of the SoC shown in fig 4.1.

4.2 ADC Interface

As mentioned in the previous chapter, the management logic of the ADC is an internal peripheral that reads data from the ADC and communicates the completion of the reading operation to the external trigger logic.

In order to design the interface I have selected a "standard" 8-bit ADC,

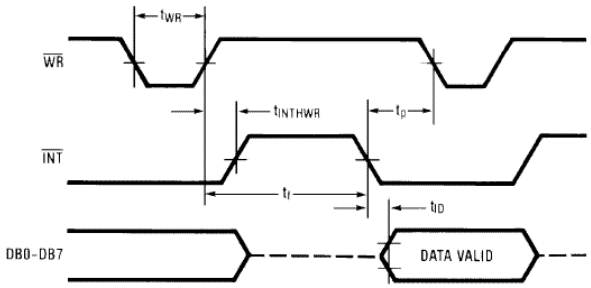


Fig. 4.4: Timing diagram of a conversion with the ADC0820.

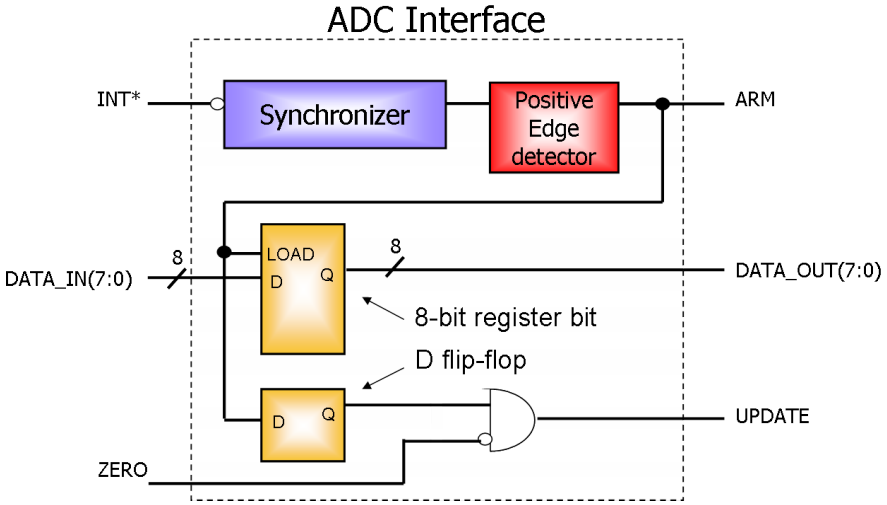


Fig. 4.5: Block diagram of the ADC interface. All the represented blocks are positive edge-triggered with a synchronous reset.

which I used as a reference to develop the interface. Among the different models available on the market I selected the ADC0820 from National Semiconductors. This ADC is one of the most common and has a simple interface adopted by many other models. The ADC0820 has different operation modes, however for my thesis I just used the WR-RD mode.

In this mode the ADC needs just one input signal (WR*) and provides the 8 digital bits representing the digital data (DB0-DB7) and strobe validating the data (INT*). The falling edge on WR* starts the conversion (Fig. 4.4), around 800ns after WR* rises back to high, the ADC outputs the digital data and flags the end of the conversion by asserting INT*. Reading from the micromegas detectors, subject of the present work, needs the generation of the WR* signal, a task for the trigger logic (which is external to ParticleBlaze) while the ADC interface is dedicated to the data input. The ADC interface (Fig. 4.5) has three inputs: an 8-bit bus for the data from the ADC, the INT* signal, as described below, the signal ZERO, which if asserted inhibits the reading of new data from the ADC. The interface registers the INT* signal on the local clock, and in the presence of a negative edge stores the input data to a register, by asserting the ARM signal, which flags the completion of a data reading. After one clock cycle, the interface outputs the read data on the DATA_OUT bus and asserts the UPDATE signal. DATA_OUT and UPDATE are used for the interfacing with the other DAQ peripherals of the SoC. As anticipated in a previous chapter, the ARM output is routed to a pin of the SoC and can be used to flag the completion of the acquisition of an event to the trigger logic.

4.3 Subtractor/Discriminator

The Subtractor/Discriminator (SD) subtracts a pedestal to the data from the ADC (DATA_IN(7:0)), outputs the result of this subtraction (DATA_OUT(7:0)) and if it is greater or equal to a pre-defined threshold it asserts the WEN* strobe. The SD is a fully pipe-lined system with a 3 clock cycles latency.

I designed the SD to interface it to the DAQ_MEM memory, in fact it provides the address (on the ADDRESS(7:0) bus) and the data to write

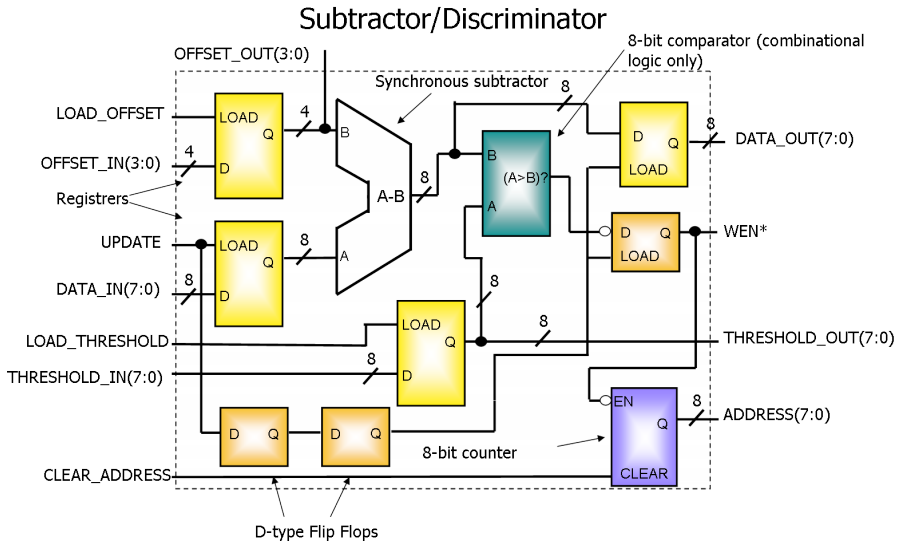


Fig. 4.6: Block diagram of the SD. All the represented blocks (with the exception of the comparator) are positive edge-triggered with a synchronous reset.

(on the DATAOUT(7:0) bus). When the UPDATE input is asserted, the DATA_IN(7:0) bus is written into an internal 8-bit register. After a clock cycle a synchronous subtractor removes the pedestal from the register. At next clock cycle, the result is compared with the threshold and if the result is higher than that, after one clock cycle the SD asserts the WEN* output.

When WEN* is asserted, an 8-bit counter storing the address for the RAM is enabled. The counter can be cleared by means of the CLEAR_ADDRESS input. Each time an UPDATE is received, after a latency of two clock cycles the 8-bit output register is loaded with the result of the subtraction to be written in memory. The subtractor is such that if the subtraction would result in a negative number, the output is set to zero. The setting of the pedestal is performed by writing the pedestal value on the OFFSET_IN(3:0) bus and by asserting the LOAD_OFFSET input. It is possible to read the pedestal value from the outside thank to the OFFSET_OUT(7:0) bus. An analogous discussion is valid for the threshold (THRESHOLD_IN(7:0), LOAD_THRESHOLD and THRESHOLD_OUT(7:0) signals).

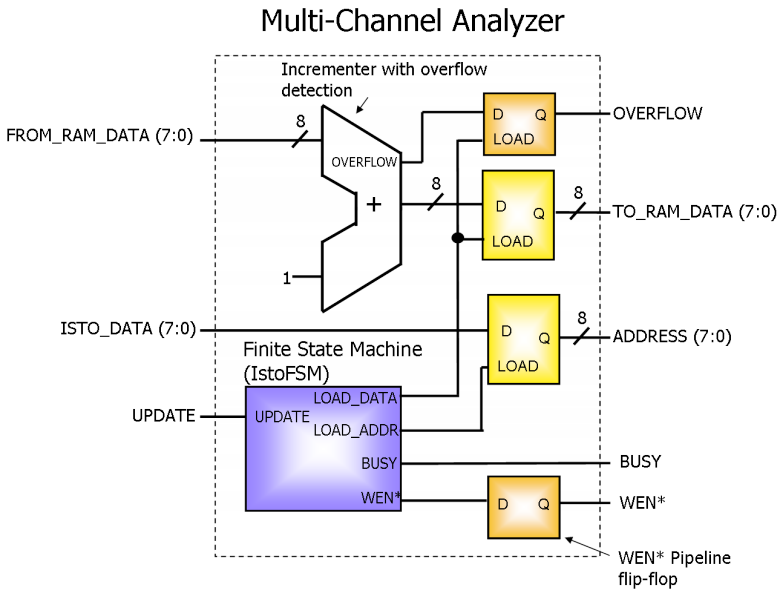


Fig. 4.7: Block diagram of the MCA. All the blocks, but the incrementer, are synchronous and positive edge-triggered.

4.4 Multi-Channel Analyzer

The MCA composes an histogram of the acquired data in the memory, by considering each location as a bin of the histogram. It is designed to use the DAQ_MEM memory of LiloBlaze and allows to realize histogram with 256 bins and a maximum of 255 events per bin.

It has three inputs:

1. the FROM_RAM_DATA(7:0) bus, for receiving the data from the RAM;
2. the ISTO_DATA(7:0) bus, for receiving the data to add to the histogram;
3. the UPDATE signal, which indicates that the data on the ISTO_DATA(7:0) bus must be added to the histogram.

The outputs are:

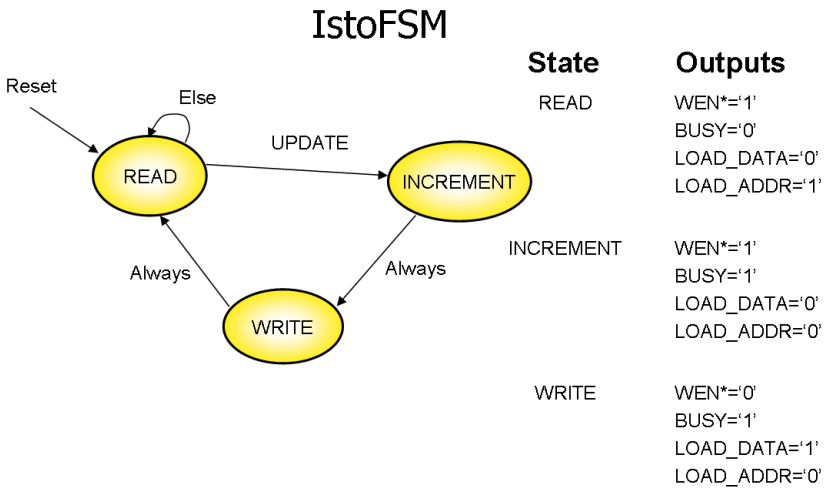


Fig. 4.8: Bubble diagram of the Moore finite state machine controlling the operations on the histogram.

1. the TO_RAM_DATA(7:0) bus, storing the data to write in memory;
2. the ADDRESS(7:0) bus, specifying the address of the RAM to write to;
3. the WEN* signal, used for enabling the writing to the RAM;
4. the BUSY signal, which if asserted flags that the histogram is being updated;
5. the OVERFLOW signal, which flags that an overflow happened during the increment of a data read from the RAM (this happens if the bin of the histogram to be incremented contains already 255 events).

The MCA consists of a control part (realized with some Moore finite state machines) and an operative part, which increments the data read from the RAM and writes it back into memory. The operative part (Fig. 4.7) consists of an 8-bit increment-er which is also able to detect an overflow condition. If the result of the sum exceeds 255, the output is set to 255 and

the OVERFLOW output is asserted. The remaining components of the operative part are the DATA, ADDRESS and OVERFLOW registers, which loading is handled by the IstoFSM finite state machine and a pipeline flip-flop. The IstoFSM machine has only the UPDATE input, which if asserted starts the addition of the current data to the histogram in memory, unless the MCA is already busy. The outputs of the FSM are:

1. BUSY, set if an operation on the histogram is in progress;
2. WEN*, the write enable for the RAM;
3. LOAD_DATA and LOAD_ADDR, the load enable for the DATA and ADDRESS registers.

The IstoFSM has three possible states (Fig. 4.8). After the reset, the FSM is in the READ state, which enables the writing in the ADDRESS register. The FSM persists in the READ state until the UPDATE input becomes high, in this case the FSM goes in the INCREMENT state. During the transition to this state, the address is written into the ADDRESS register. At the next clock cycle, the data (Y) is read from the location pointed by the ADDRESS register and it is incremented (Y+1) and the FSM jumps into the WRITE state. The outputs are set in such a way to store Y+1 in the DATA register and the WEN* signal is asserted. In order to write Y+1 in the RAM, it is necessary to wait for the loading of the DATA register, therefore I placed a D flip-flop to generate a one-clock cycle latency from the LOAD_DATA signal to the WEN* for the RAM. From the WRITE state, the machine goes always back to the READ state and it is ready for a new operation on the histogram.

4.5 Sample Counter

The sample counter is a down counter, used in ParticleBlaze to start and count the acquisition of N consecutive data from the ADC. Its inputs are:

1. the LOAD signal, used to initialize the counter;

2. the DATA(7:0) bus, the data to initialize the counter with;

3. the DEC signal, which if asserted enables the count.

The outputs are:

1. the COUNT(7:0) bus, which stores the current value for the counter;

2. the ZERO signal, which if asserted indicates that COUNT(7:0)=0;

3. the LAST signal, if asserted indicates that COUNT(7:0) went from 1 to 0 in the latest clock cycle.

The ZERO and LAST signals are used inside ParticleBlaze to start and stop acquisitions and to generate interrupt requests for the LiloBlaze CPU. By loading an 8-bit data in the counter and enabling the count, the COUNT(7:0) output is decremented at each clock cycle, going from N to 0. During the count, the ZERO output is de-asserted and being connected to the ZERO input of the ADC interface it allows the reading from the ADC. When COUNT goes from 1 to 0, the ZERO and LAST signals are both activated. The ZERO signal stops further reading from the ADC, while the LAST signal, which only lasts for two-clock cycles causes an interrupt request to the CPU. This way, the processor is informed of the end of the acquisition.

4.6 XPROM Reader

At the device power up, the "XPROM Reader" transfers an executable code from an external Xilinx PROM (XC18V04 in the prototype) to the internal program memory. I recall that the program memory is a 256 by 16 bits dual-port RAM. The width of the memory is determined by the instruction word size (16 bits), while its depth is related to the size of the program counter and to the maximum program length (256 instructions). One of the two ports is used for the loading of the program,while the other is used by LiloBlaze to fetch the instructions to execute.

I designed the XPROM Reader architecture according to the requirements for the serial reading of Xilinx PROMs [31]. My XPROM reader

drives the clock for the PROM and receives the data serially at low frequency ($\sim$ 1 MHz). I have chosen a serial transfer in order to reduce the number of IO pins for the VLSI device, with a consequent decrease of its area.

My XPROM Reader directly handles the writing to the program memory and initializes it at the power-up. It provides the bit de-serialization in order to obtain 16-bit words from the serial input. I also included the option to retrieve user data (in the format of 8-bit words) from the PROM and read them on a LiloBlaze input port. This feature can be used to store calibration data on the PROM and restore it at the start up.

Chapter 5

FPGA Prototype

In this chapter I present a prototype of the ParticleBlaze SoC implemented in a Field Programmable Gate Array (FPGA) and I discuss its performance and logic foot-print. The realization of the SoC in the FPGA allowed me to verify that the HDL description of the system works correctly before implementing it with an ASIC. At the end of the chapter I discuss the test strategy used, which is based on using the same FPGA to implement the SoC and the hardware for testing it.

5.1 Field Programmable Gate Arrays

FPGAs are electronic devices based on a matrix of on-field programmable logic resources. Logic resources are homogeneously distributed in blocks on the whole surface of the chip. Blocks can work in combinational (executing boolean functions) or sequential mode (executing boolean functions whose result may depend on the history of the inputs).

In order to connect the blocks among them and with the IO logic there are dedicated interconnection resources, which are also programmable. Interconnection resources are a grid of conductive segments, which can be in contact with each other and with the logic in such a way to make the connect the blocks.

FPGAs can have a “volatile” or “non-volatile” configuration, according

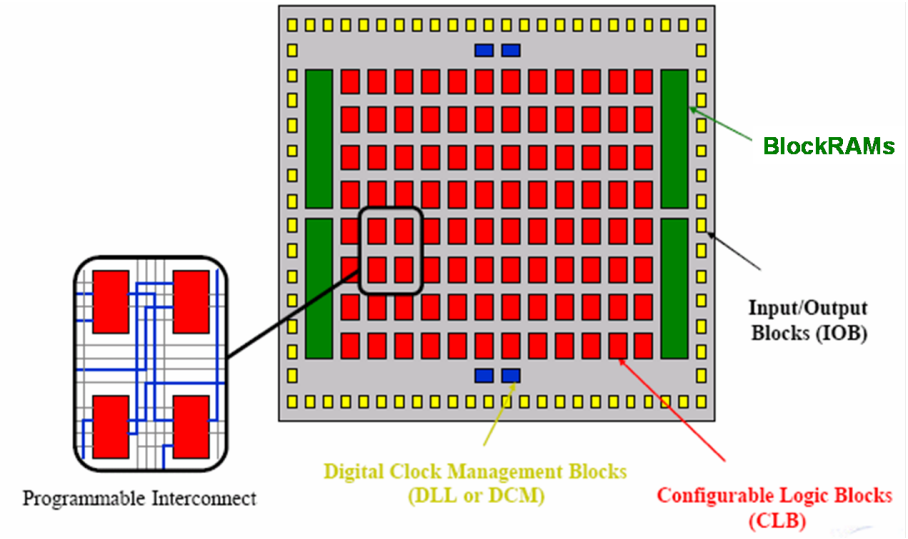


Fig. 5.1: Layout of the logical resources in a Virtex-II FPGA.

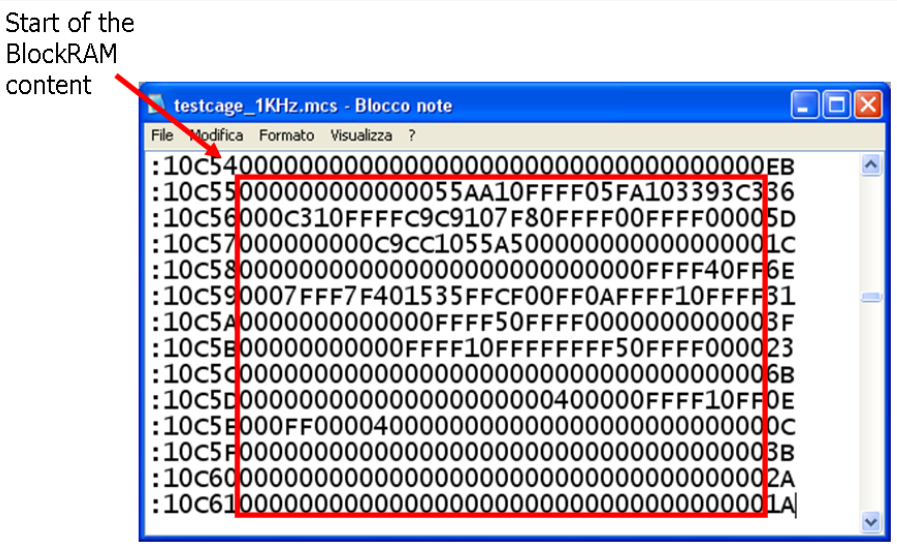


Fig. 5.2: Part of a bitstream. The rectangle encloses a section specifying the content of a BlockRAM.

to whether or not the configuration is lost each time the device is switched off. There are also devices which can be programmed only once ("one-time" configuration).

Designing with FPGAs can be divided in three steps:

1. development and optimization of the logic according to the available resources;
2. logic partitioning, allocation and interconnection of the resources;
3. physical programming of the device.

There are several FPGA manufacturers and each one produce several models, grouped into families. In order to realize the ParticleBlaze SoC, I used an FPGA of the Virtex-II family [33] produced by Xilinx. The main elements of a Virtex-II FPGA are:

1. Configurable Logic Blocks (CLBs),
2. BlockRAMs,
3. Input/Output Blocks (IOBs),
4. interconnection resources.

CLBs are programmable elements which can execute combinational and/or sequential operations and therefore constitute the heart of the device: the designer may program CLBs and then connect them together in such a way to build the required logic. BlockRAMs are configurable RAMs which can be used as a support for the logic implemented with the CLBs: in fact, they allow to store information inside the device, without using the sequential resources of the CLBs. BlockRAMs can be configured as double-port or single-port RAMs or as ROMs. The IOBs permit interfacing the internal logic with external devices and the interconnection resources are the "wires" connecting CLBs, BlockRAMs and IOBs.

The specific task executed by an FPGA changes according the configuration of its logic elements, and this translates into a good versatility of the

device, which enables the user to emulate a big number of possible fixed-logic circuits. An FPGA includes an internal memory (the "configuration memory"), used for storing the programming code of all the logic elements (the bitstream). Usually, the FPGA configuration memory is volatile, therefore the bitstream is loaded into a non-volatile external memory (for instance a PROM, a flash memory or the hard-disk of a computer) and at the power-up the configuration is copied to the configuration memory. The bitstream defines also the content of each BlockRAM at the device power-up. Fig. 5.2 shows an example of bitstream, where I highlighted part of the content of a BlockRAM.

5.2 FPGA Implementation

I described ParticleBlaze with the VHDL[1] language [32], which other than allowing an easy behavioral description of the circuit functionality, allows also the portability of the project from the FPGA to an ASIC implementation. My purpose, in realizing the FPGA prototype, was to validate the VHDL description of ParticleBlaze, in such a way to start the VLSI implementation having a proven description, at least with sequences of input stimuli tested in the laboratory.

Moreover, there is another strong reason which suggested me the realization of a prototype. In the testing phase, it is necessary to simulate the code execution, this may happen within a VHDL simulation of the system, but it could require many clock cycles and therefore long simulation times. The following code illustrates this problem with an example. Let us suppose we have to realize a program for LiloBlaze which makes a led flashing at 1 Hz frequency:

```
; Program for making a LED flash
; at 1 Hz frequency
```

[1] VHDL is the acronym of **V**HSIC **H**ardware **D**escription **L**anguage, while VHSIC is acronym of **V**ery **H**igh **S**peed **I**ntegrated **C**ircuits. VHDL was born in the first half of the '80s, in the framework of a project of the U.S. department of defense. It has then been made a standard by the IEEE (**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers) in 1987 and it has been revised in 1993.

```
; (if the clock frequency is 100 MHz)
constant led_port,00
load s4,00
; Main loop
main:
        xor s4,01             ; inverts the LED status
        out s4,led_port       ; outputs on the led_port
        load s2,64            ; calls a wait sub-routine,
        call wait             ; in order to wait for
                              ; 2 · (2 + (64)16 · 4 · 256^2) ≃ 5 · 10^7
                              ; clock cycles
       jump main
; ***********************************************
; Function Wait
; Purpose: wait
; Input parameters: s2
; Description:
; waits (2 + wait2 * 4 * 256^2 ) * 2 clock cycles
; i.e. around wait2 * 0.005 s at 100 MHz
; ***********************************************
wait:

        load s1, 00
        load s0, 00

wait_sub_0:
        sub s0,01
        subcy s1,00
        subcy s2,00
        jump nz, wait_sub_0
        ret
```

The instruction `call wait` takes $5 \cdot 10^7$ clock cycles to be executed. In order to observe a single flashing of the LED in the VHDL simulation, on a modern PC[2], there would be necessary around 8 hours and 20 minutes. Of course the presented example is extremely simplified, but important because there are often cases, when the code execution requires many clock cycles. This

[2] As a reference let us consider a Personal Computer with a 3 GHz Pentium IV, 2 GB RAM, Windows 7 operating system and ModelSim SE simulator.

issue is easily overcome by realizing an FPGA prototype and interfacing it with a system which permits to analyze the output (e.g. a PC). This way, it is possible to execute the software in real-time and perform the debugging. Moreover one is protected against possible bugs of the simulator. It is important to keep in mind that the prototype, even if it is very useful in the development and the debug phase of the software, does not substitute the simulation of the VHDL description, but it is complementary for that. The simulation is necessary during the development of the hardware since it permits to obtain precious information on the circuit behavior, for instance the logical state of the internal nodes.

In order to move from the VHDL code to its physical implementation on FPGA, I used a proprietary software provided by the vendor: integrated synthesis environment (ISE). ISE is a computer aided design (CAD), namely a set of programs which cover all the design phases: from the synthesis of the logic to the definition of the layout of the resources into the device. This CAD includes also a simulator which allows the user to check the correct the working of the circuit in the various design phases (for instance after the synthesis and after the layout definition).

5.2.1 Simulation

In this section, I illustrate the simulations of the three key aspects of the system: the storing of a piece of data from the ADC, its histogramming and the handling of an interrupt request generated at the completion of an acquisition. In order to understand the simulations, it is important to remember how the ADC Interface, the SD, the MCA and the DAQ memory are interfaced. I recall the interface architecture in Fig. 5.3.

Fig. 5.4 shows a detail of the acquisition and storing in memory of a ADC data. The DATA_IN and INT* inputs are stimulated from the outside using a stimuli file coded in VHDL and they emulate the ADC. In the simulation, I also simulated the fact that the INT* signal in not synchronous with the clock, as it happens during the actual operation of the device. The ADC flags the end of the conversion with a falling edge on the INT* input and

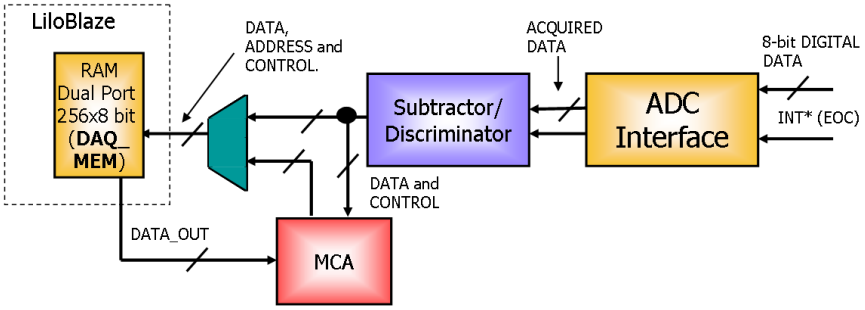


Fig. 5.3: Detail from the ParticleBlaze architecture block diagram showing the DAQ peripherals connections.

asserting a valid data on the output. After five clock cycles from the INT* edge, due to the latency of the synchronizer and of the edge detector internal to the ADC interface, the latter captures the data ($(26)_{16}$) and transfers it on the output. The assertion of the UPDATE signal flags that the data on the output is valid and enables the processing of the data by the SD. The SD subtracts a pedestal ($(3)_{16}$) from the datum and compares the difference $(23)_{16}$) with a pre-defined threshold ($(4)_{16}$). The elaboration of the data lasts 3 clock periods, at the end of those, the SD asserts the WEN* output in order to write the data into the DAQ_MEM at the location pointed by the ADDRESS output ($(00)_{16}$). After the writing into the RAM, the counter inside the SD is incremented and the ADDRESS output increments from $(00)_{16}$ to $(01)_{16}$. The next data storing will then happen at the address $(01)_{16}$.

The simulation in Fig. 5.5 shows an example of histogramming of a data into the memory. Please note that the MCA receives the input data from the SD and not from the ADC Interface directly. Specifically, the MCA receives the data to histogram from the SD/DATA_OUT output and starts the histogramming when SD/WEN* is asserted. In the considered example, the MCA histograms the $(23)_{16}$ data. At the very next clock cycle after the assertion of SD/WEN*, the internal register ADDRESS of the MCA is loaded with $(23)_{16}$ (see the MULTI/ADDRESS output) and the internal

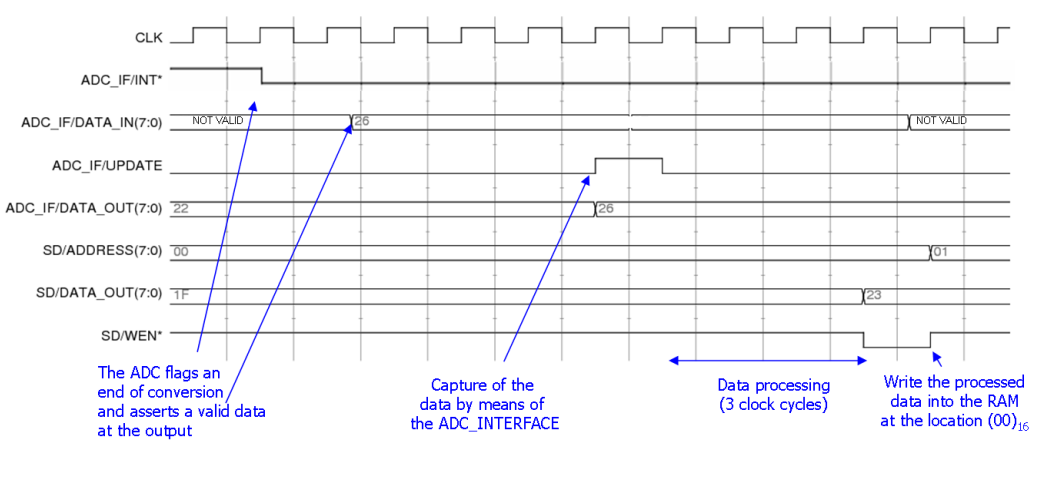
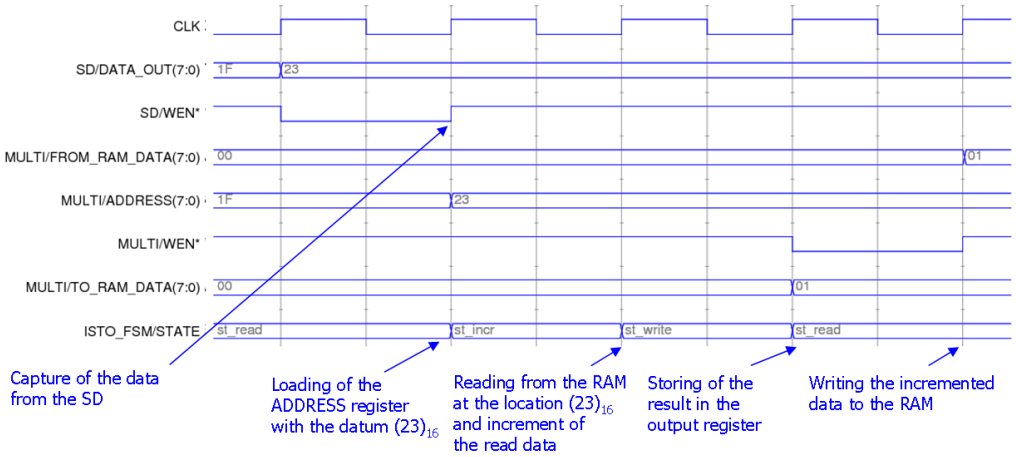


Fig. 5.4: Simulation of the reading from the ADC. Inputs and outputs of the ADC_INTERFACE have the "ADC_IF/" prefix, while those of the SD have the "SD/" prefix.

finite state machine (FSM) enters the INCREMENT state. After a clock cycle, the DATA register of the MCA is loaded with the data read from the RAM at the address $(23)_{16}$ ($(00)_{16}$) and incremented by one ($(01)_{16}$), and the FSM enters the WRITE state. After an additional clock cycle the MCA outputs the content of the DATA register ($(01)_{16}$) on the TO_RAM_DATA output and writes in the memory by asserting the MULTI/WEN* output. The result of the operation is therefore the addition of one event into the bin number $(23)_{16}$.

An interesting aspect of the SoC operation is the detection of the end of an acquisition. Let us suppose we wish to realize a program which composes into the memory the histogram of all the data read from the ADC, finds the maximum and the full width half maximum (FWHM) and transmits them, together with the acquired data through the output port. Let us suppose we wish to acquire 100 consecutive events. Let us start the acquisition by loading the 100 value in SC. The LiloBlaze processor can detect the end of the acquisition in two ways:

1. by reading the SC register (if its content is greater than zero and the



i

Fig. 5.5: Simulation of the operation of the MCA. Inputs and outputs of the SD have the "SD/" prefix and those of the MCA have the "MULTI/" prefix. The ISTO_FSM/STATE signal indicates the state of the IstoFSM state machine internal to the MCA.

acquisition is running, otherwise the acquisition has been terminated);

2. by enabling interrupts and writing a service routine to handle the end of the acquisition.

The first method (called "polling") is easier to realize, but has the disadvantage of using the microprocessor for the reading of the SC register and therefore of blocking the execution of the calculation and output routines, during the data acquisition. The second method relies on the interrupt service routine, which is called at the end of the acquisition. It is clear then, that the microprocessor can execute the search for the maximum, the calculation of the FWHM and the output of the data while the acquisition is running. The second one is a more efficient method, because it allows to reach higher operation frequency and it does not waste processor resources.

The simulation if Fig 5.6, which refers to the second method, shows the answer of LiloBlaze to an interrupt request generated by the Sample Counter reaching to zero. Fig. 5.7 shows the code executed by the mi-

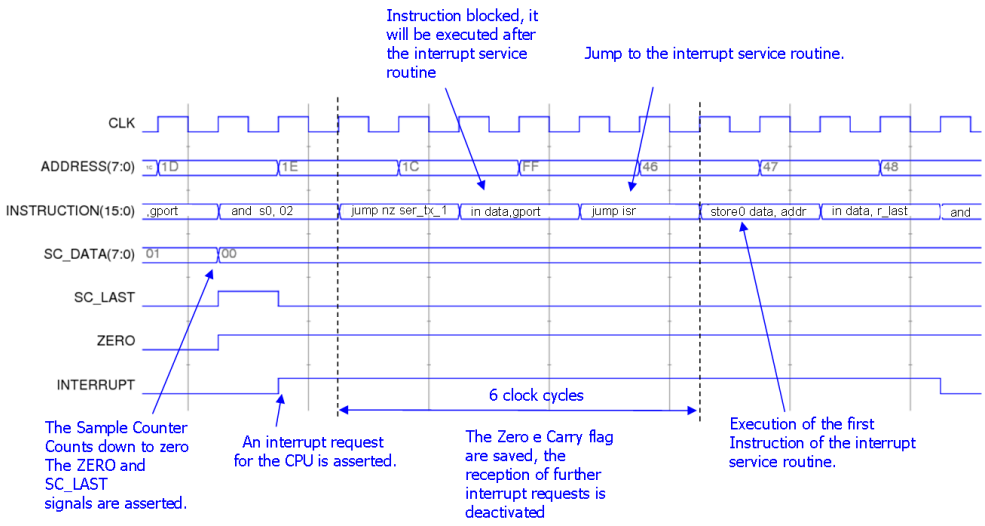


Fig. 5.6: Simulation of the response to an interrupt request generated by the Sample Counter reaching zero. The ADDRESS(7:0) bus shows the content of the Program Counter and the INSTRUCTION(15:0) bus shows the instruction being executed.

croprocessor. During the execution of the main program, SC counts down to zero (see the SC_DATA signal). The ZERO signal is asserted and inhibits further readings from the ADC (the ZERO signal will be asserted until $SCDATA = (00)_{16}$). The SC_LAST is asserted for a clock cycle and makes the INTERRUPT signal go to '1'. This signal is kept high by a dedicated logic until the reading of the SC_LAST register. After two clock cycles of latency, the processor detects the logic '1' on the INTERRUPT input, blocks the execution of the `in data,gport` instruction, saves its address $(1C)_{16}$ into the stack and loads the program counter with $(FF)_{16}$. At this address there is a jump instruction toward the interrupt service routine `jump isr` (Fig. 5.7). The processor executes the jump and starts the execution of the routine. The second instruction of this routine reads the status of the SC_LAST register and causes the de-assertion of the INTERRUPT signal. At the end of the ISR, a dedicated instruction `reti enable`, restores the execution of the main program starting from the instruction at the address

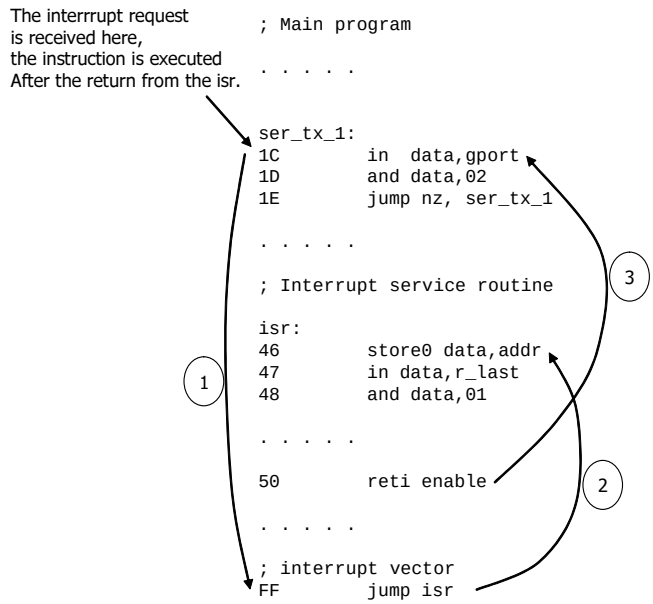


Fig. 5.7: Code segments being executed while the interrupt requests shown in Fig. 5.6 is received. The suspensive dots indicate omitted portion of the code. The numbered arrows show the jumps in the code in the same order as they are taken.

saved into the stack ($(1C)_{16}$).

5.2.2 Layout

In order to realize the project on the FPGA I used a V2MB1000 [34] demo board from Memec (Fig. 5.8). Demo boards are designed for the evaluation and testing of digital electronic circuits; specifically this board is built around a Xilinx Virtex II 1000 FPGA, which is interfaced to buttons, dip-switches, LEDs, seven-segments displays and other auxiliary hardware. Also, the board includes an RS-232 interface, which can be used for connecting the FPGA to a PC. On the board there are two quartz oscillators, providing two clocks: the first at 24 MHz and the second at 100 MHz. The loading of the bitstream on the FPGA is performed trough a dedicated serial programming

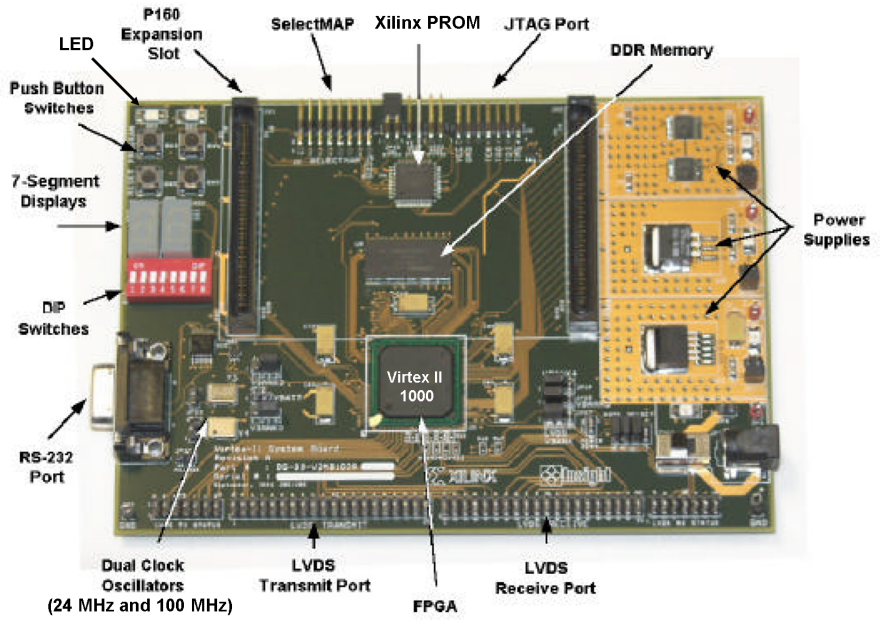


Fig. 5.8: Photograph of the Memec V2MB1000 demo board.

port (JTAG).

In Fig. 5.9 it is possible to observe the layout of ParticleBlaze implemented in the FPGA. The device includes 40 18-kbit BlockRAMs, which are configurable with different address space and word width. The DAQ_MEM memory included in the microprocessor and the double port program memory are realized by suitably configuring two BlockRAMs inside the device. The occupation of logic resources of the SoC is the 8% of the available resources, this allows to realize several instances of ParticleBlaze inside the same FPGA, in full agreement with the scalability requirement mentioned in paragraph 3.1. The maximum operation frequency is the same as the one of LiloBlaze processor ($\sim$ 100 MHz). Therefore, we can state that the frequency performance did not diminish, even after the addition of internal hardware to the microprocessor and the addition of the peripherals (SD, MCA, etc.)

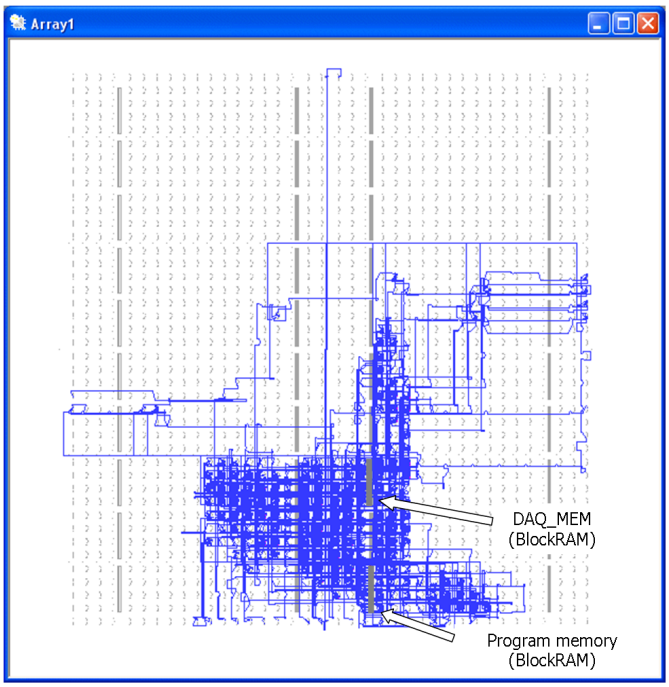


Fig. 5.9: ParticleBlaze SoC layout on a Xilinx Virtex 2 V1000 FPGA.

5.3 Laboratory Test

After the realization phase of the prototype in the FPGA, I found a strategy to test the system which puts it into conditions very similar to those of the actual operation. This way the probability of finding bugs in the design is maximized.

5.3.1 Test Bench

The test strategy, I followed, consisted into integrating in the same FPGA the circuit to be tested and other digital circuits which emulate the behavior of external elements, in this specific case the ADC.

I described in VHDL a complete hardware test-bench and I implemented it inside the Virtex II V1000 FPGA hosted on the Memec board. A possible alternative choice would have been to realize a dedicated test board hosting

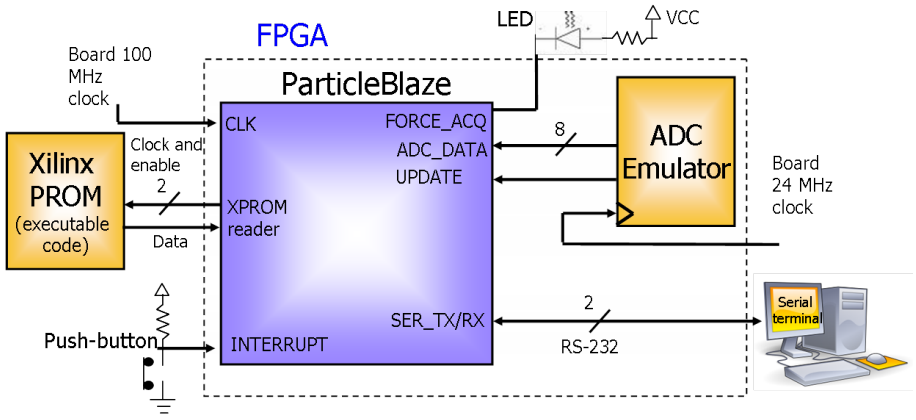


Fig. 5.10: Simplified block diagram of the hardware test bench.

the FPGA and the external elements needed for the testing. That choice would have allowed to obtain a testing environment even more similar to the real working conditions, but it would have been much more complex and time-consuming to design.

Fig. 5.10 shows a simplified block diagram of testing system. The "ADC Emulator" block simulates the behavior of the external ADC. At the power-up the XPROM reader in the SoC retrieves the program from the PROM and after it has completed, the program execution starts. In the meanwhile the "ADC Emulator" provides input data to the system following a pre-defined sequence and with a fixed frequency (it can be modified in the VHDL description). In order to use the test bench it is necessary to write the sequence for the ADC Emulator inside a dedicated BlockRAM and to prepare programming file with the program code for the Xilinx PROM.

By means of terminal emulation software (such as Putty) it is possible to visualize information about the program execution.

Test Bench Components

The "ADC Emulator", as suggested by its name, simulates the behavior of the external ADC which will be connected to the ParticleBlaze ASIC implementation. I realized this system by means of a BlockRAM configured

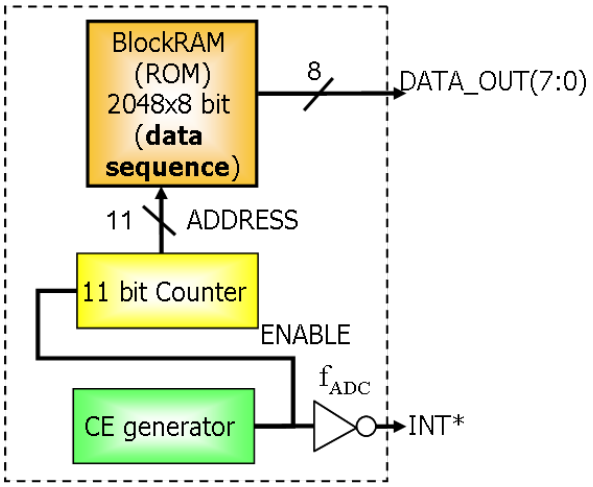


Fig. 5.11: Simplified block diagram of the "ADC Emulator".

as a ROM (2048 8-bit words). The internal architecture is shown in Fig. 5.11. A counter provides the input address for the ROM. At every clock cycle, the content of the location pointed by the counter is transferred to the output and simulates the 8 bits of digitized data. A clock generator enables a periodic signal, high for 1 clock cycle, at configurable frequency f_{ADC} (from 100 Hz to 24 MHz), which is used both for enabling the counter and for generating the INT* signal. By programming the BlockRAM with desired word sequence, the emulator asserts INT* and outputs a word from sequence with a frequency f_{ADC}. In order to make an example, I realized a simple C language program which generates a 2048 8-bit word sequence with a gaussian distribution having a mean $x \in [0; 255] \cap \mathbb{Z}$ and a σ set by the user.

By loading the generated sequence into the ROM, I realized an ADC emulator providing gaussian distributed data. In order to simulate the fact that the ADC is asynchronous with respect to the SoC, I used two uncorrelated clocks for the two systems. The ADC Emulator works with a 24 MHz clock, while ParticleBlaze with a 100 MHz clock. Thank to the possibility of varying f_{ADC} up to 24 MHz, I can emulate an external ADC which provides data almost at the maximum frequency allowed by the system (50 MHz in

storing mode, 20 MHz in histogramming mode). This kind of emulation allows to have scenario even "worse" than the real one, in fact the maximum frequency of a ADC0820 is of the order of 1 MHz.

The UART embedded in ParticleBlaze is used to interface it with a PC. The UART is very useful for the debug of the hardware and of the software. IN fact, it is possible to display the acquired data an compare them to the expected ones and to check the correct operation of the system with several acquisition parameters (threshold, pedestal,etc.) and acquisition mode (histogram building or waveform recording).

5.3.2 Test Results

I realized a demonstration program which allows the user to: set the acquisition mode, set the threshold and the pedestal to be subtracted by the SD and start the acquisition of N consecutive samples. The end of an acquisition causes the execution of a dedicated ISR which restarts it. Every second, the processor transfers all the content of the DAQ_MEM to the PC through the UART. By means of an ASCII serial terminal emulator, it is possible to visualize the acquired data. If we are in acquisition mode, the program transmits also the bin number of the maximum and the FWHM of the histogram.

By pushing a button connected to the input of the microprocessor it is possible to change the acquisition mode, which is flagged to the outside with a LED on the FORCE_ACQUIRE output. Using this program I tested the system with several acquisition parameters (threshold, pedestal and number of samples per acquisition) and in two acquisition modes.

Fig. 5.12 shows the output produced by the program after histogramming the data read from the ADC Emulator programmed with a gaussian word sequence.

In order to make the program debug more efficient, I included a dedicated feature in the LiloBlaze assembler program. In fact, the assembler I developed produces an output file which can be used together with an external utility (provided by Xilinx) for the update of the executable code, even

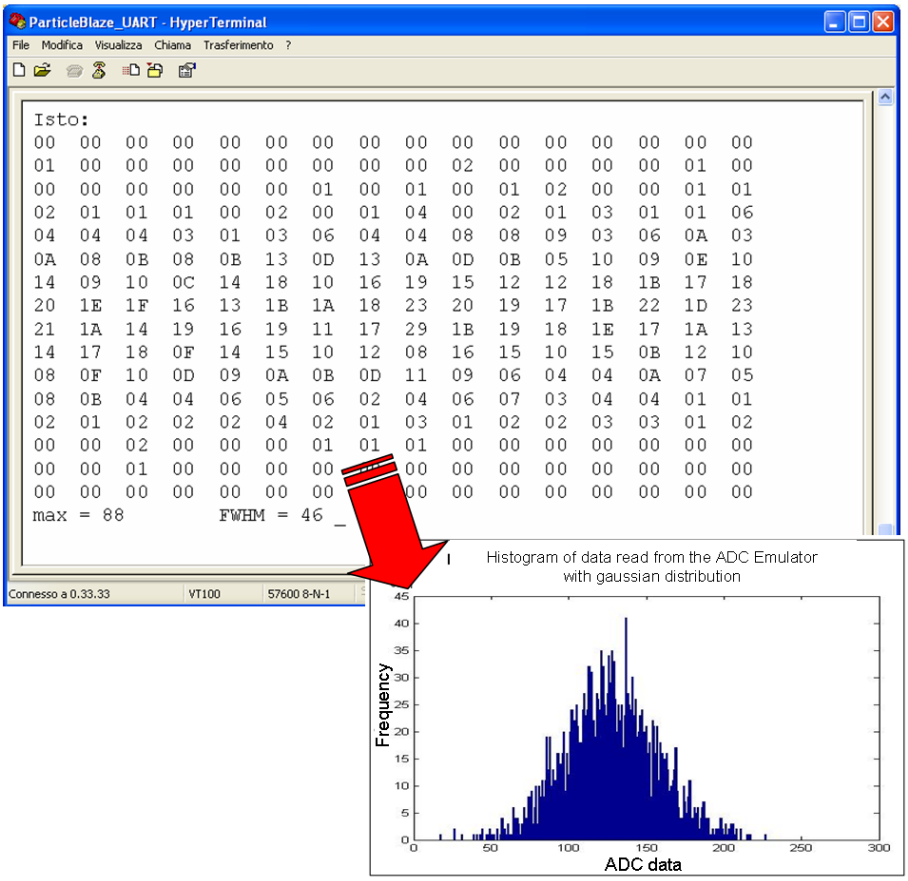


Fig. 5.12: Output of the demonstration program running in histogram mode. On the serial terminal it is displayed: the content of each bin of the histogram (DAQ_MEM memory), its FWHM and the number of the bin containing the maximum.

after the definition of the layout on the FPGA, without requiring a new implementation of the project. The external utility operates on the bitstream, modifying the part corresponding to the content of the program BlockRAM. During the debug phases this feature of the assembler translated into a good time save.

Following I present an oscilloscope screen-shot, which I acquired while the system was in histogram building mode and with a zero threshold. In

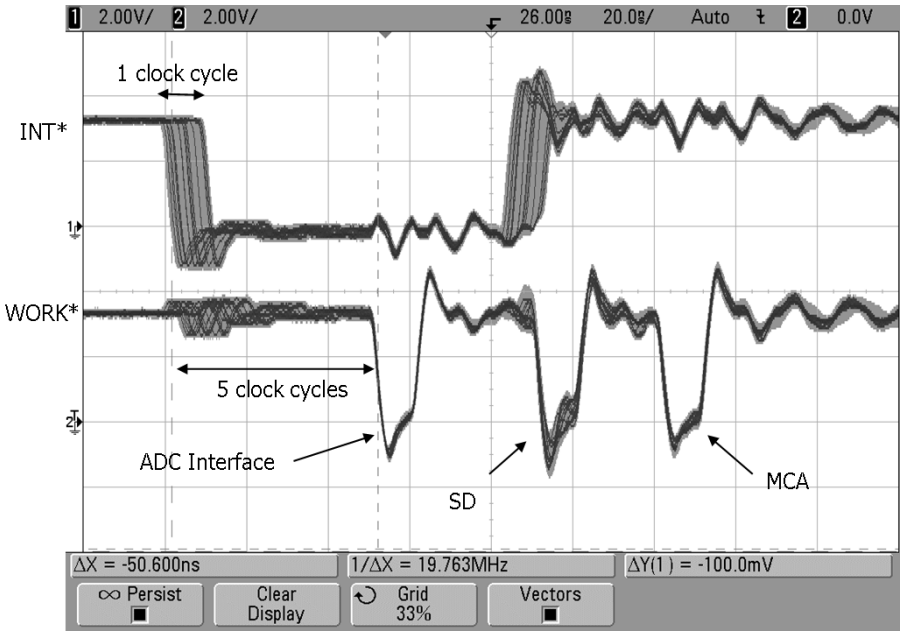


Fig. 5.13: Oscilloscope screen-shot about the processing of data acquired from the ADC. The oscilloscope is triggered on the WORK* signal.

order to understand the screen-shot, it is necessary to define the WORK* signal, which is asserted low each time one of the three DAQ peripherals (ADC Interface, MCA or SD) asserts a valid data on its output bus and de-asserted otherwise. The WORK* is the logical AND of the signals validating the data from the three DAQ peripherals (ADC_IF/UPDATE, SD/WEN* e MULTI/WEN*) :

$$WORK^* = ADCIF/UPDATE^* \cdot SD/WEN^* \cdot MULTI/WEN^* \quad (5.1)$$

Fig. 5.13 is an infinite persistence screen-shot acquired with an oscilloscope, where the INT* generated from the ADC Emulator and the WORK* signals are available. A negative edge on the INT* signal flags a new valid data at the output of ADC Emulator. The first negative pulse on the WORK signal indicates the "ADC Interface" captured the data from the ADC, the second

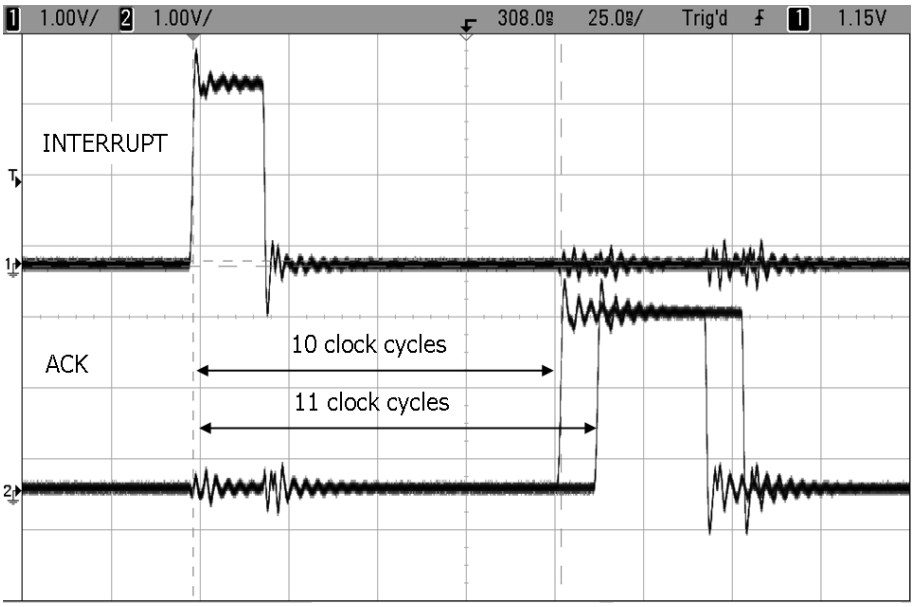


Fig. 5.14: Latency of the execution of the ISR by the LiloBlaze processor. The oscilloscope is triggered on the INTERRUPT signal.

that the SD has completed the elaboration, while the third flags the writing in memory by the MCA. Please observe the band formed by the positive and negative edges of the INT*. This band exists because INT* is asynchronous with respect to the system clock.

As discussed in the paragraph 5.2.1, the use of interrupts for the detection of end of acquisition events allows to have no impact on the microprocessor, which is therefore free to execute other functions when it does not handle an event. In general the use of interrupts allows to have an answer to IO requests from the peripherals with a predictable timing. Moreover, a low interrupt answer latency is a crucial requirement for the good operation of a real-time system. In the case of ParticleBlaze, the latency varies from 5 to 6 clock cycles and allows to have a low dead-time from an acquisition and the next one. The infinite persistence picture in Fig. 5.14 shows the interrupt answer latency of LiloBlaze. The pulse on the INTERRUPT input generates an interrupt request for the CPU. The ACK signal is connected to the bit

0 of the OUT_PORT and switches from '0' to '1' at the execution of the second instruction in the ISR. On this signal, we can observe two pulses, with a phase difference of a clock period, this is due to the uncertainty of interrupt answer latency (1 clock cycle).

Chapter 6

VLSI Implementation

6.1 VLSI Digital Systems

Very Large Scale Integration systems allow to implement on the same silicon die millions of active and passive electronics devices [38]. There are many advantages in realizing a full digital system in a single integrated circuit. First, there is a reduction of the system size with a consequent reduction of the parasitic elements (resistances, inductances, capacitances). The reduction of the parasitic capacitance in turn allows an increase of the maximum clock frequency of the system and a reduction of the dissipated power [36]. The deployment of a VLSI ASIC permit to reduce the costs with respect to a discrete-component system for large production volumes ($\gtrsim 10^3$ chips). Finally in the VLSI devices it is possible to obtain circuit densities un-reachable with any other technology.

There are different technological processes for the realization of VLSI circuits, the most used is the Complementary Metal-Oxide-Silicon (CMOS). Its key feature is the realization of PMOS and NMOS devices on the same silicon substrate. Usually a N-doped well is realized in a P-doped substrate, in order to have NMOS devices in the well and PMOS ones in the substrate. Other processes are based on the realization of P-wells in N-doped substrates or P-wells and N-wells in the neutral substrate. The CMOS technology, on the contrary of bipolar ones, allows to have a significant power dissipation

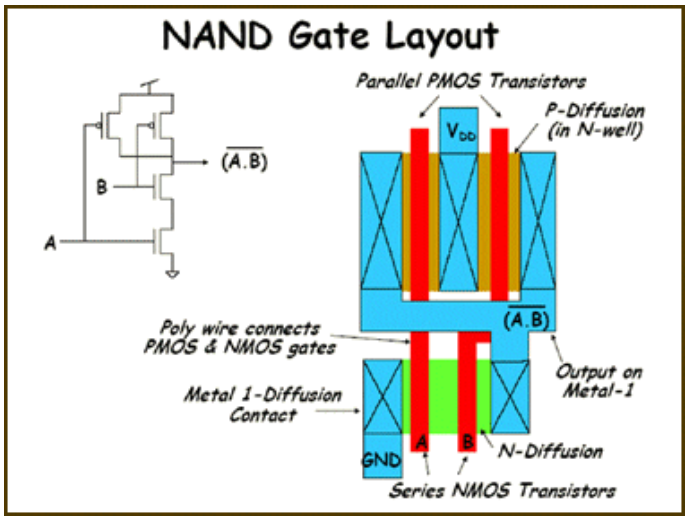


Fig. 6.1: Layout of a NAND port in CMOS technology.

only during the transition of the outputs (dynamic power dissipation) and to have a minimum static power dissipation.

A reference feature for identifying the integration scale of CMOS devices is the minimum channel length of metal-oxide-silicon field effect transistor (MOSFETs), which generally follows the CMOS acronym (e.g. CMOS 0.35 μm)[42]. CMOS technologies require many fabrication steps of the silicon substrate, which is suitably doped in such a way to implement the active devices, several metal layers are then deposited in order to realize the interconnections among the devices. The fabrication of integrated circuits takes place into dedicated factories, called foundries.

There are different approaches for the ASIC VLSI design: some foresee a manual realization of the circuit layout (full-custom) other use some software which automates this procedure (semi-custom). In a full-custom approach the designer manually draws the layout of transistor and of interconnections on the chip surface, by defining rectangles where, after the fabrication of the die, there will be specific dopings or metal layers [35]. This approach allows to realize devices highly optimized for a specific performance (power consumption, operation frequency, etc.) and with a reduced area occupation,

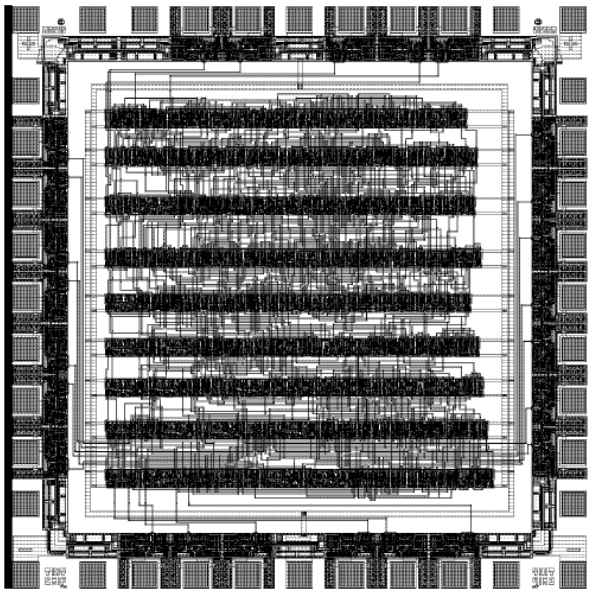


Fig. 6.2: Example of a layout of a standard-cell integrated circuit. The structure of the cells is hidden by the interconnection metal layers. On the perimeter of the die, there are bonding pads for the interconnection with the package pins.

but it requires more design time with respect to the semi-custom approaches and it is mainly used in three cases:

1. for the realization of analog circuits, because in this case it is the only possible approach [37, 39];

2. when the main target is the performance, instead of the design time;

3. for the design of a circuit to be used many times (e.g. a library element).

It is in fact possible to realize libraries of logical blocks (cells) which can be instantiated inside more than one project. For instance, it is possible to design a cell implementing a two-input AND gate, one which implements an inverter and use them together in the design of a multiplexer. Fig. 6.1 shows the layout of a NAND cell. Among the semi-custom approaches, the

"standard-cell" one is based on this concept. Cells implement basic logic functions (e.g. AND, NOT, flip-flops) and they follow well-defined geometrical rules regarding the position of the power and ground contacts and their size. Cells are designed for a specific technological process: a change of foundry (or even just a change of process) requires their re-design. Foundries usually provide some cell libraries for their technological processes. The user bases his design on the available cells. Other than the layout, for each cell the foundry provides models describing its electrical, logical and timing characteristics. At the end, the user's design is built by connecting some library cell instances among them, chosen according to the cited models.

Analogously to the FPGA design-flow, the one for a digital VLSI system consists in three main steps:

1. development and optimization of the logic depending on the available cells;

2. logic partitioning in sub-sets, each one to be implemented by a single cell, placement on the chip surface and interconnection of the cells;

3. static timing analysis and post-layout simulation.

The design is based on softwares for the automatic logic synthesis and for the placement and routing of the cells. Cells are placed in rows on all the die surface. The interconnection of the cells is based on some conductive segments realized with the metal layers offered by the technological process (Fig. 6.2). In addition to the cells which implement basic boolean functions, there are macro-cells (hard macros) which implement more complex blocks, such as RAM memory banks, multipliers, ALU or even microprocessors.

With respect to FPGAs, VLSI devices have additional issues for the designer: for instance it is necessary to design the power and clock distribution networks, which are critical elements for a digital electronic system. Moreover in order to guarantee the production of a chip respecting the defined layout, the designer must respect some geometrical rules in the layout definition, the "design rules" (DRs). DRs are a set of constraints which define for instance the minimum distance between two rectangles on the same metal

Table 6.1: Setup time (T_s), hold time (T_h) and clock to output (T_{co}) of a D-type edge-triggered in the 90nm Faraday library and in a Xilinx Virtex-II FPGA.

	90nm Faraday library	FPGA Xilinx Virtex 2
T_s(ps)	88	370
T_h(ps)	-32	-20
T_{co}(ps)	350	570

layer, the maximum size of a metal rectangle, the minimum distance between two diffusions, the minimum width of the channel of a transistor, etc. DRs depend on the technological limits which allow to realize the rectangles on the layers with a finite accuracy. There are usually thousands of DRs and their verification is performed with some dedicated tools (Design Rule Checkers). DRCs take as an input the layout and the set of design rules to be checked and return all the (possible) violations.

The design methodology I used for this work is a standard-cell approach and the technology is a CMOS 90nm process of the United Microelectronics Corporation (UMC). The provider of the standard cell library is not the foundry, but the Faraday company. It does not provide the layout of the cells, but only some geometrical information (abstract view) defining the unusable surfaces on the different metal layers (blockages). The designer therefore does not access the internal layout of the cells and must use models provided by Faraday in order to know the propagation delays or the power dissipation.

6.2 Primitives of the Faraday Library for the UMC CMOS 90nm process

The library used for this thesis work is the Faraday 90nm standard cell library [40, 41] for the CMOS 90nm process offered by UMC. In order to compare the switching performance of this library with the one of an FPGA, I have chosen some sample elements (a combinational cell, a sequential one, an input

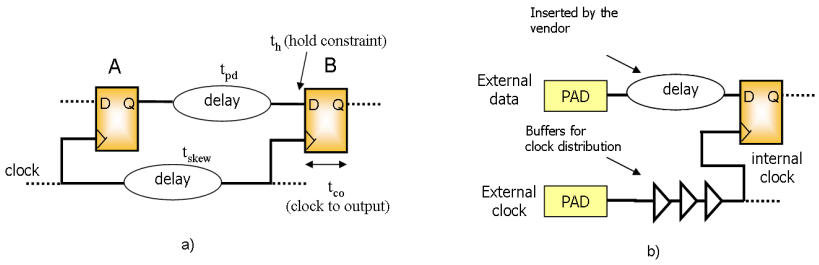


Fig. 6.3: a) Pipeline with skew on the clock network; b) delay on an input pad inserted by the FPGA vendor.

Table 6.2: Output delay (T_O) and input delay (T_I) for the IO pads in the Faraday 90nm library and Virtex-II FPGAs. The presented delays refer to an input slew-rate of 0.1 $\frac{ns}{V}$.

	Faraday 90nm library	FPGA Xilinx Virtex 2
T_O(ns)	3.095	1.74
T_I(ns)	0.19	2.43

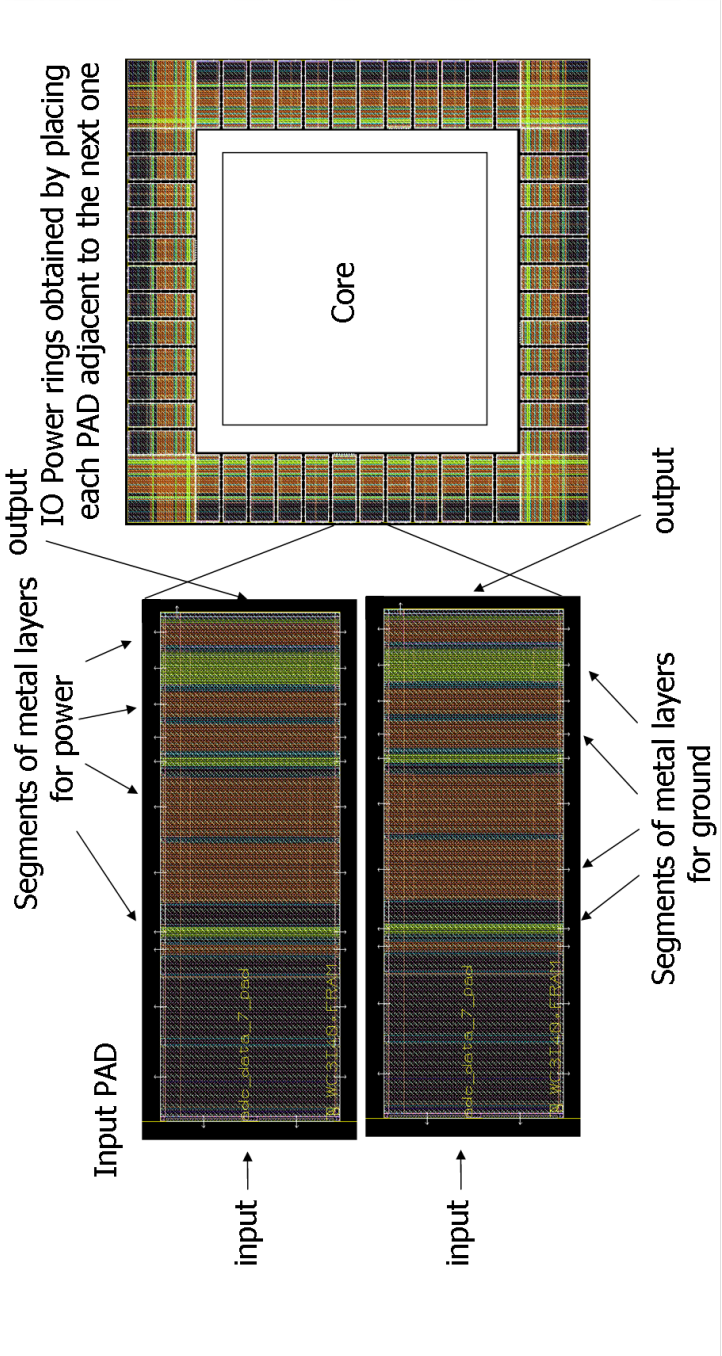


Fig. 6.4: Bottom, metal regions used by an input pad. Top, IO power rings realized by placing the IO cells on the perimeter of the die.

and an output cell) and I compared their performance with the equivalent elements which are in a Xilinx Virtex 2 1000 FPGA with a -4 speed-grade. It is important to underline that since the two technologies are very different, the elements from the library do not have a direct correspondence within the FPGA. For instance, a flip-flop in FPGA is realized by suitably configuring a CLB, which is a much more complex configurable logic block and therefore has inferior performance with respect to a bare flip-flop. On the contrary, in a VLSI standard cell library a flip-flop is implemented with a cell highly optimized for that specific function only.

Faraday provides maximum, minimum and typical propagation delays for each cell (with respect to the temperature, the power voltage and the process corner), while Xilinx provides only the maximum delays. In order to make a fair comparison, I compared the maximum delays for both the technologies. Moreover, since the propagation delays of the standard cells of the Faraday library are given as a function of the load capacitance, I selected the maximum load they are given for ($C_o = 213\ fF$ for the combinational cells and $C_o = 120\ fF$ for flip-flops).

For each logic function the Faraday library offers several standard cells with different drive-strengths. A higher drive-strength (lower output impedance) of the cell implies a lower propagation delay. In order to make a comparison between equivalent elements I selected the cells with the lowest drive-strength (highest output impedance), among the available ones. In general propagation delays of a logic port depend also on the input slew-rate, reported data refer to a slew rate of 80 $\frac{ps}{V}$. The cell I used for the comparison of the propagation delays is a 4-input NAND. The technology I used offers up to nine metal layers; for this cell, the metal regions used are all on the first metal layer, therefore the reduction of the available resources for the interconnection is minimum.

I selected a boolean function with 4 inputs because that is the number of inputs of the look up table (LUT) used to implement combinational logic into the considered FPGA. The maximum cell delay is $T_{NAND4} = 0.33\ ns$, while the maximum delay of a LUT in the FPGA is $T_{ILO} = 0.44\ ns$. In this case there is an increment of the order of 33% by moving from the ASIC

implementation to the Virtex-II FPGA. If we consider a more complex function, such as a 3-input multiplexer (plus 2-select inputs), with the Faraday 90nm library we have a delay $T_{MUX3} = 372\ ps$, while in order to realize the same function on a Virtex-II FPGA, we would need a 5-input LUT with a $T_{LUT5} = 720\ ps$ propagation delay. This time the gain in performance is even more noticeable. Let us now consider a sequential element, for the Faraday 90nm library we use a positive edge triggered D flip-flop, while on the FPGA we consider the FF included in a CLB. Table 6.1 compares the propagation delays and timing constraints of the two circuits. From now on I will refer to setup and hold times respectively with T_s and T_h , and to the clock to output with T_{co}.

Let us now consider two flip-flops connected together as in Fig. 6.3a. We define T_{pd} the propagation delay from the output of the A flip-flop to the input of the B flip-flop and T_{skew} the delay from the arrival of the clock at A and at B. In order for the system to work properly, the setup and hold time constraints must be satisfied. The first one can always be respected by suitably lowering the operation frequency. Instead, in order to satisfy the hold constraint it is necessary that, independently of the frequency, the following inequality holds:

$$T_{co} + T_{pd} > T_h + T_{skew} \tag{6.1}$$

Chosen T_{co} and T_h which depend on the flip flop and set $T_{pd} = 0$, which represents the worst case, in order to satisfy the constraint the designer can adjust the T_{skew} parameter. The inequality can be re-written $T_{skew} < T_{co} - T_h$. The difference $T_{co} - T_h$ defines how the hold constraint is strict, i.e. what is the maximum tolerable skew[1] T_{skew} on the clock distribution network. The fact that T_h and T_{co} for the two technologies are similar implies the difficulty of the designing of the clock distribution network is the same.

[1]Let $\{t_i\}_{i=1..n}$ be the delays from the input clock pin and the n destinations it has to reach. We define input delay of the clock distribution network the quantity $max\{\{t_i\}_{i=1..n}\}$.

Table 6.3: Clock to output, setup and hold times for a dual port hard-macro RAM of the Faraday 90nm library and for a dual port BlockRAM of a Virtex-II FPGA.

signal or bus	parameter	hard-macro RAM	Virtex-II BlockRAM
address	T_s/T_h (ns) min	0.65/0.00	0.36/0.00
input data	T_s/T_h (ns) min	0.55/0.00	0.36/0.00
write enable	T_s/T_h (ns) min	1.90/0.00	0.72/-0.25
output enable	T_s/T_h (ns) min	asynchronous	1.20/-0.58
output data	T_{co} (ns) max	1.68	2.65

However, in the FPGA device the clock tree has been implemented by the vendor, while in VLSI technology this is a designer's task.

Fig. 6.4 shows the internal layout of an input pad (it is just an example and does not refer to the Faraday library). The internal structure of the pads is such that by placing the pads on the perimeter of the die some rings on the metal layers are formed. The rings are used for the power and ground distribution to all the IO pads. Table 6.2 finally proposes a comparison between the IO pad delays of the Faraday library and of the considered FPGA. The propagation delay of an output pad of Faraday library is around the double of the one on FPGA, while for the input pads the situation is inverted. The difference of delay on the output pads is due to the fact that internal structures of FPGA pad are not significantly more complex than the one of the Faraday library, therefore we observe a performance improvement consequent of an improvement of the technological process.

The delay difference on the input pad is due to the fact that the FPGA vendor adds some delay for compensating the propagation delay of the clock inside the device, in such a way not to increase the hold time for the external data (Fig. 6.3b).

6.3 RAM Hard Macro

Faraday provides some macro cells (hard-macros) which implement RAM banks with different modes (dual port, single port, etc.). If there is the need for a RAM bank in the design it is necessary to request the corresponding macro cell by specifying the bank characteristics (number of words, number of bits per word, mode, form-factor, etc.). The RAM macros are automatically generated by means of a software (the memory compiler), which generates the layout and the timing models for the specific RAM bank configuration requested by the user. In the ASIC implementation of ParticleBlaze, I implemented the RAMs embedded in the processor by means of two RAM macros.

In Table 6.3, I compare the delays and the timing constraints of a dual port RAM hard macro of 256 words of 16 bits with those of a 18kbits BlockRAM of a Virtex-II. I selected these memories for the comparison because they are used to store the executable program respectively in the ASIC and FPGA implementation of the SoC. The memories I considered have an input bus, an address bus, a write enable an output enable and an output bus. The maximum delays reported for the hard macro refer to a load of 10 fF and an input slew rate of 7.5 $\frac{ps}{V}$. As the reader can see in Table 6.3, the clock to output delays and the setup times differ from 30% to 60%, in the two technology. The highest difference is for the setup time of the write enable input. The hold times are always zero for the Faraday RAM, while they are zero or negative in the Virtex-II BlockRAM. We notice that the timing constraints are tighter for the hard-macro RAM, but it has the smaller clock to output.

As we will see later on, by comparing the frequency performance of VLSI version of ParticleBlaze with the FPGA one, we obtain an improvement of the maximum operation frequency, which is 250 MHz in VLSI technology. Anyway the difference in the propagation delays of the basic elements (standard cells for the ASIC and CLBs for the FPGA) does not explain such a sensitive performance improvement. The performance gain is due to the lower propagation delay on the interconnecting nets.

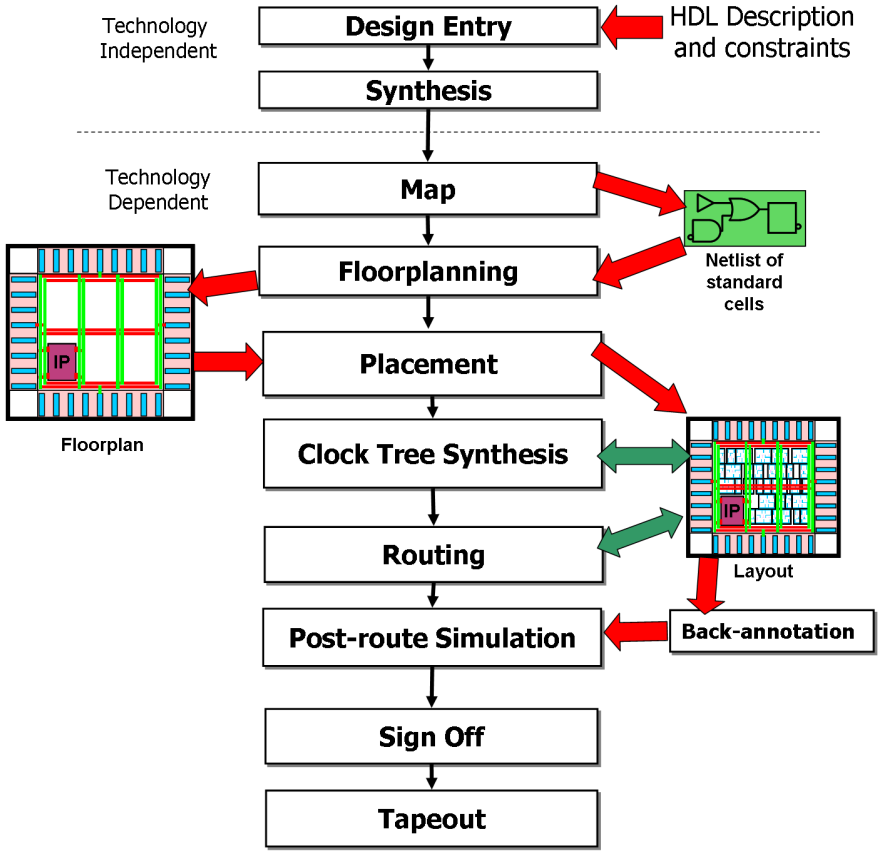


Fig. 6.5: Design flow for a standard cell VLSI device.

6.4 Design Flow

Fig. 6.5 proposes a general diagram of the design flow for ASIC devices with the standard cell approach. One or more softwares (also called "tools") are used in each design phase. I refer to this set of softwares with the acronym CAD. The CAD I used includes tools from the Synopsys and Cadence companies.

The first phase in the flow, "design entry", consists of defining the design logic, usually this is performed by writing a behavioral description with an HDL. As I said in a previous chapter, I used VHDL to describe the Parti-

cleBlaze logic. Other than describing project logic it is necessary to provide timing and area constraints. Timing constraints must include the maximum desired operating frequency, the maximum skew on the clock distribution network, the clock to output delay on the output pads and the setup time on the input ones. In all the subsequent phases I describe it is important to keep in mind that every tool in the CAD takes into account the design constraints. The HDL description is synthesized over a library of logic primitives (logic gates, flip-flops, adders, multipliers), which do not coincide with the standard cells and do not have a bi-unique correspondence with them. The result of the synthesis is a netlist of logic primitives.

During the "map" phase, the netlist is transformed (or better re-synthesized) in a netlist of standard cells, each logic element in the input netlist is implemented by means of the standard cells. Up to this point, the layout of the circuit is unknown yet, but since there is information about the geometry of the timing of the single used cells, we can already have an estimation of the propagation delays associated to the logical paths and the occupation of the system. If at this level we have not reached the timing and area constraint requirements yet, they will necessarily not be satisfied also at the end of the flow. In this case, it is necessary to go back to the design entry and modify the logic.

In the "floorplanning" phase, the designer defines some geometrical constraints for the placement and the routing of the logic on the die. For instance, we define the area dedicated for placing the standard cells (core area), the position of hard-macros, the placement of all the IO and power cells.

During the "placement" a software distributes standard cells in the regions in such a way to satisfy the assigned timing constraints, if possible. Some place and route tools, such as Cadence Encounter which I used in this work, can re-synthesize portions of the logic in such a way to have a more efficient implementation, with the aim of satisfying the assigned constraints, both on the timing and on the area. Completed the placement, we move to the construction of the clock distribution network. The design of digital synchronous devices is based on a fundamental assumption: the simultaneous arrival of the clock signals to all the registered elements. In order to balance

the delays to the different placed sequential cells, some buffers are inserted in between the point where the clock enters the die and all the destinations it has to reach. The "clock tree synthesis" (CTS) consists in adding these buffers to the design netlist and perform the placement in such a way to satisfy the constraints on the skew of the distribution network and on the maximum delay from the clock input pin to the internal destination.

The next step, "routing", consists in suitably connecting the standard cells. In this phase the connections among the cells are realized by defining the segments on the metal layers offered by the technological process. By exploiting different metal layers it is possible to route interconnections each over the other, by reducing the occupation and the area.

After each design phase, it is possible to execute simulations by extracting a netlist from the layout and the propagation delays associated to the cells and to the nets ("back annotation"). After the routing, the layout is design rule checked and a transistor level netlist is extracted from the layout and compared against the initial netlist (layout versus schematic, LVS). The LVS can be performed only in the case the internal netlist for the cells is known (which has been not the case for this work). Moreover a sign-off static timing analysis is performed in order to check that all the timing constraints are satisfied in the final layout. If all the checks are passed it is possible to move to tape-out, i.e. to the sending of the layout to the foundry for the device fabrication, otherwise it is necessary to go back in the flow and solve the problems.

6.5 Implementation of the ParticleBlaze ASIC

The design entry of the ASIC implementation of ParticleBlaze has been already completed for the FPGA prototype. In fact, I used in VLSI technology the same VHDL code of the FPGA version; that has been possible because the behavioral description I produced is independent of the technology. The only difference between the descriptions is in the fact that in order to realize the internal memories, on the FPGAs some BlockRAMs are instantiated while on the ASIC some hard-macros are used. I had to add some timing

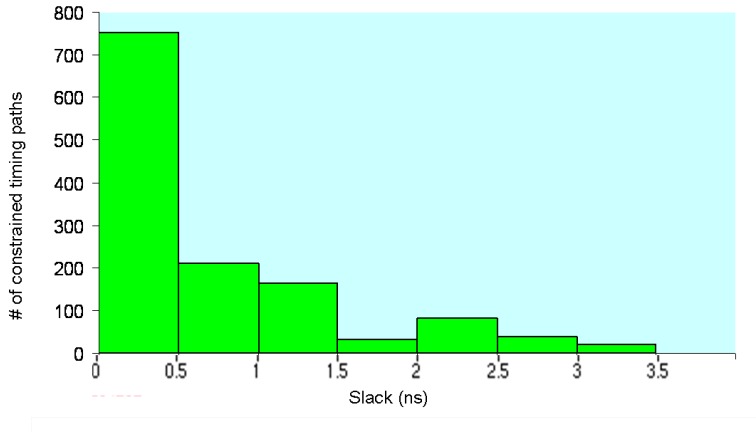


Fig. 6.6: Slack histogram estimated with a WLM.

constraints regarding the skew and input delay of the timing distribution networks, because the CTS phase is not present in the FPGA design flow.

Within the miniasic prototyping program, which I am going through for the fabrication of the ParticleBlaze ASIC prototype, the price does not scale proportionally with the area, instead it is fixed for a block of the size 1.875 mm x 1.875 mm. Therefore, those are actually the maximum dimensions for the die. The next paragraph presents the first design phase for the ParticleBlaze ASIC: the synthesis.

6.5.1 Synthesis and Map

The tool I used for the synthesis and map is Synopsys Design Compiler [43]. It takes as input an HDL description (VHDL or Verilog[2] [44].) and synthesizes it producing a netlist of logic primitives of the GTECH library, which does not have a direct representation in hardware. The next step is the map on the standard cell library. In order to execute the map I used an

[2]Verilog was developed in 1985 by Phil Moorby for Automated Integrated Design Systems (after renamed Gateway Design Automation). The Cadence Design Systems bought the Gateway Design Automation acquiring all the property rights of the language. Consequently, after the increasing diffusion of the VHDL, the company decided to liberalize Verilog in such a way to make it available for standardization. In 1995, it was standardized by IEEE and in 2001 it was revised.

interesting feature of the software which allows to optimize the propagation delay of the critical path[3] by automatically merging different hierarchical blocks. This way the hierarchy is automatically modified only where it is needed, therefore the internal nodes on logical paths which are not optimized in the post layout simulation.

It is possible to provide a list of timing and area constraints to be respect during the map phase. Among other parameters, I specified:

- the desired maximum operation frequency (277 MHz);
- the minimum T_s which the external logic provides the input data with (3 ns);
- the maximum T_{co} for the output data (3 ns);
- the skew and jitter and clock distribution network (respectively 100 ps and 200 ps);
- the maximum tolerable transition time at the input of the cells;
- the maximum design area (2.25 mm^2 = 1.5 mm x 1.5 mm to be conservative with respect to the miniasic block size).

All the specified constraints were satisfied after the synthesis. In fact, ParticleBlaze reaches the maximum specified operation frequency of 277 MHz (more than the double of the one in the FPGA) and the actual used area for the logic is 0.8 mm^2.

After defining the constraints for ParticleBlaze I executed the synthesis and the map of the project and I obtained a netlist of standard cells to be used to define the floorplan. The software allowed me to execute a timing analysis after the map and check that the timing was met before proceeding with the floorplan definition. At this stage, since the layout is not available yet, this analysis can only provide estimates of the propagation delays, but it is very useful for understanding if there are any chances of satisfying the

[3]The critical path is the timing path having the minimum difference between the maximum required delay and the actual delay.

constraints after the layout has been completed. In order for the propagation delays to be estimated the propagation delay of each cell is multiplied by a factor dependent of the fan-out of the cell $f(n)$ (with $n \in \mathbb{N}$). The function $f(n)$, called "wire load model" (WLM), is provided by the foundry and is also dependent on the device operating conditions such as temperature, power supply and process corner. The foundry estimates the WLM with statistical methods based on the comparison between the fan-out and the measured delay, or estimated after the routing, for each cell and for several sample projects.

Fig. 6.6 shows the histogram of the differences between the maximum required delays of ParticleBlaze and the delay estimated by means of the WLM (slack histogram) for the constrained timing paths. Most paths are within 500ps from their maximum delay, this means the design has an optimized timing after the synthesis. Instead, a distribution where most of the paths are far away from the minimum would have meant the timing could be improved by optimizing just a few paths.

6.5.2 Floorplan

The software I used for the floorplan (and for all the rest of the layout design) is Cadence First Encounter (Encounter for short). In order to produce the floorplan I started from the standard cell netlist produced in the map step. First, I introduced in the design the power supply pads, which are not present in the initial netlist since they are standard cells which only have a physical representation and not a logic one. The power for the core cells is separated by the one of the IO cells, in order to minimize the chance of failures due to the ground bounce. I added 4 power pins and 4 ground pins for the core cells as well as 4 power and 4 ground pins for the IO. This lead to have a total of 55 pins for the devices. Let us recall that by placing the pads on the perimeter of the die rings for the IO power and ground are formed. I inserted a ground pad on each side in such a way to make the resistive voltage drop as homogeneous as possible on all the IO ring in the hypothesis of a uniform power consumption by the placed cells. Moreover, in order to minimize the

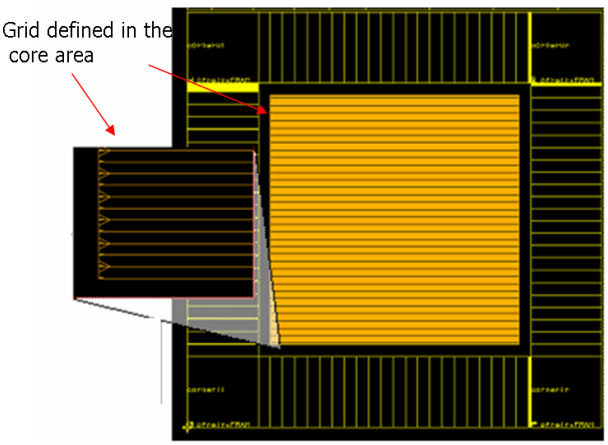


Fig. 6.7: Definition of the core area and of the grid for the placement of the standard cells.

ground bounce on the outputs, I equally distributed them on each side of the die and I placed them close to the power and ground pads in such a way to minimize the parasitic inductance of the bonding wires.

After defining the placement of all the IO pins, I defined the internal area to be used for the placement of the standard cells, the "core area". Inside the core area, I defined a virtual grid which is used for placing the standard cells of the project (Fig. 6.7). The grid defines also the vertical orientation that the cells must have. It is possible to place the cells in the grid by holding always the same orientation or invert the orientation on the even rows with respect to those on the odd rows, in such a way that cells of adjacent rows may share the power and ground rails (Fig. 6.8). I have chosen this option because it allowed to have a very high circuit density and consequently an optimal timing performance.

At this point I placed the hard macros in the core area and I defined some "placement blockages" around them. A placement blockage is a region where the placement of standard cells is forbidden. The blockages are useful for preventing the placement of cells too close to the macro cells, because that could cause congestion and routing problems. Finally, I tracked the

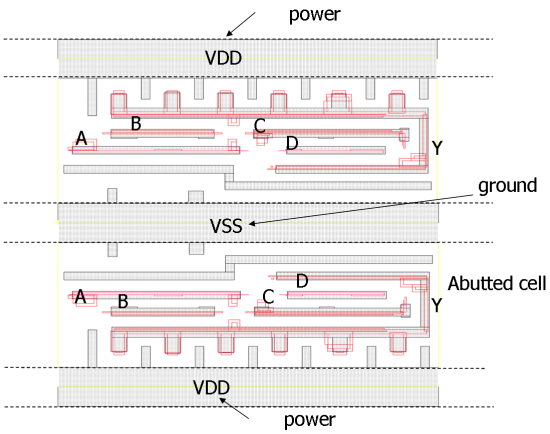


Fig. 6.8: Abutment of the cells on the even rows to share power and ground rails with the cells on the odd rows.

metal rings for the distribution of the power to the core cells and the power interconnections for the macro-cells. In order to size these interconnections I executed some simulations of the resistive voltage drop (or IR-drop) by assuming a uniform consumption of the cells. Fig. 6.9 shows the results of a simulation of the IR drop, the power ring and the power of the macro-cells have been coloured according to their resistive drop. The maximum voltage drop for the power grid and rings is 0.06 V. I checked that to be a tolerable drop for the standard cells from their data-sheet (assuming a power voltage at the pad of 1.1 V). Fig. 6.10 shows the ParticleBlaze floorplan: the die is square and has 14 pads per side (but the top side which has 13 pads), the length of each side is 1734 μm (or 1580 μm excluding the bond-pads). The die size is limited by the IO pads (pad-limited design) and in fact after the placement it is possible to see that most of the area in the core is free. The size of the chip fits in one miniasic block as required.

6.5.3 Placement

In the first place, Encounter starts a non-timing-driven placement. Since in this phase the interconnections are not realized yet, the software used some virtual interconnections (Virtual Routing) which are built by connecting

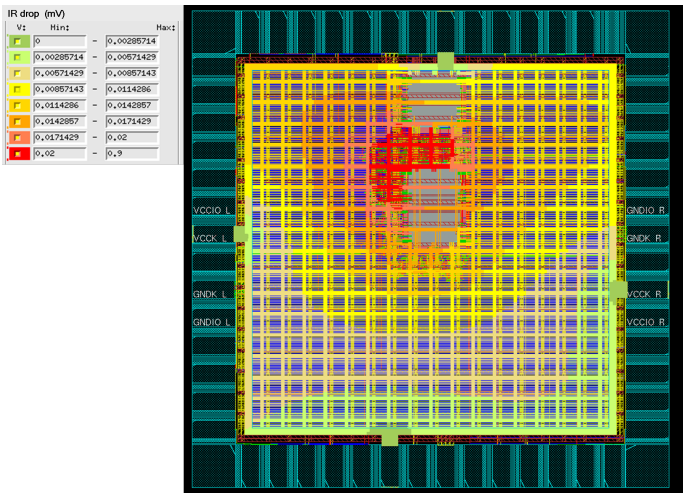


Fig. 6.9: IR drop on the core power rings and in the core area.

horizontal and virtual segments from the source to the destination, following the shortest allowed path Fig. 6.11a. There is a maximum number of the nets per unit area which can be realized respecting the design rules. Let us suppose to divide the surface of the die according to a grid and let n_{max} be the maximum number of nets which can cross a segment. Let n be the number of virtual nets crossing the segment, we then define the congestion on a segment as $r = \frac{n}{n_{max}}$ (Fig. 6.11b). Since they are virtual nets, it can happen that $r > 1$, therefore the estimate of the routing congestion is crucial to determine if it is possible to route the circuit or not. For example, if $r > 1$ on all the grid segments, the routing is impossible and then it is necessary to modify the floorplan. Moreover, for the segments where $r > 1$ some nets need to make a detour around the congested segment and therefore they cause an increase of the propagation delays. Knowing the more congested segments, the tool can distance cells which have reciprocal interconnections with an high value of r.

Once congestion problems are resolved, the placer starts an optimizing process in order to attempt to satisfy the timing constraints. The propagation delays on the logical paths depend on the logic and routing propagation delays. The logic delays are extracted from the timing library provided by

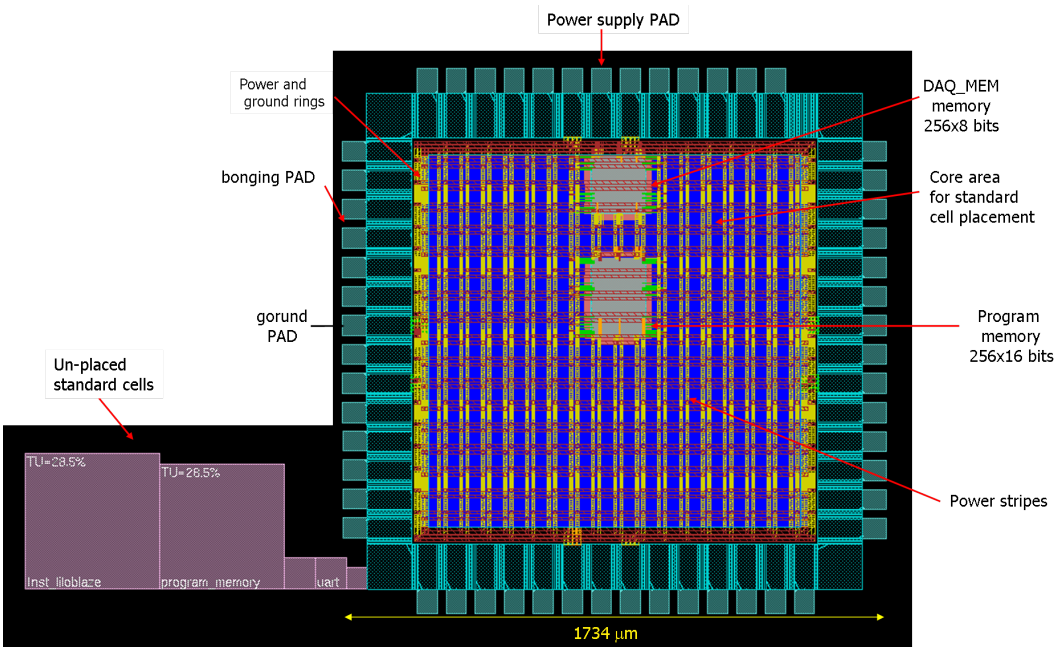


Fig. 6.10: Floorplan of ParticleBlaze.

the vendor (Faraday in our case), while for the routing an estimate is done for the associated resistances (R) and capacitances (C) according to the length of the virtual nets. This estimate is much more accurate than the one from the WLM used in the synthesis phase, in fact cells from the post-synthesis netlist might have an excessive or too small drive-strength. For this reason during the optimization, the placer can substitute automatically the cells instantiated in the project with equivalent cells having the same functionality but a drive-strength more suitable for the R and C estimated with the VR.

Fig. 6.12 shows the result of the ParticleBlaze placement. The RAM hard-macros are in the center top of the core area. The actual area occupied by standard cells is $A_{eff} = 0.087\ mm^2$, while the total core area available for them was $A_{avail} = 1.290\ mm^2$. The average standard cell density is then $d = \frac{A_{eff}}{A_{avail}} = 6.7\%$, while the core utilization percentage (which takes into account also the RAM macros) is 10%. In fact, most of the core area is free from cells. In the attempt to satisfy the timing constraints the tool placed

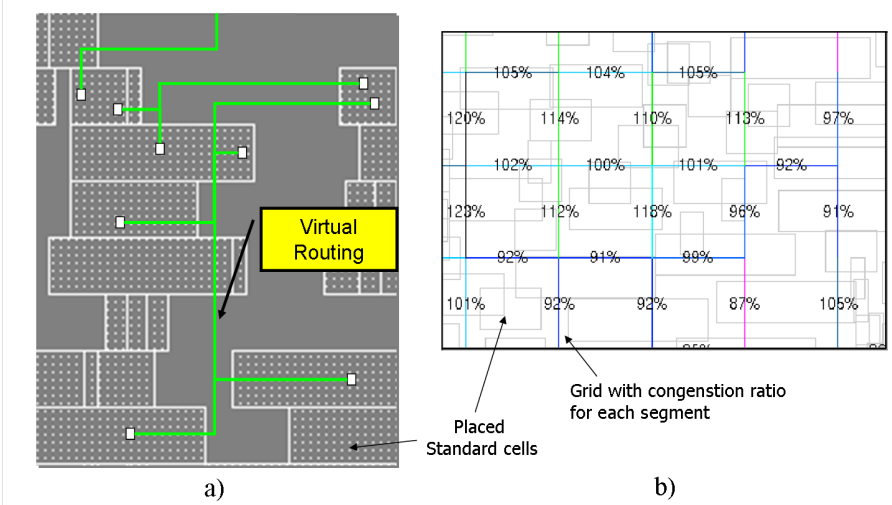


Fig. 6.11: a) Virtual Routing; b) congestion map.

all the cells close to each other and to the RAM banks.

During the optimization phase, Encounter re-maps portion of logic in order to satisfy the timing constraints. A key feature of the optimization is about high fan-out networks (HFNs), which by definition are all the networks with a single source and a number of destinations higher than a pre-defined threshold. I used this feature for the optimization of the reset network, which is a typical example of HFN. In the netlist from the map phase this network was simply a set of nets from the reset pin of the device to the reset of all the registered elements. Actually, in order to minimize the propagation delay from the source to the multiple destinations, the place&route tool inserts some buffers in the network. Also the clock distribution network is an HFN according to the definition given above, but that network is so critical and special that this task is performed as a separate step (the clock tree synthesis).

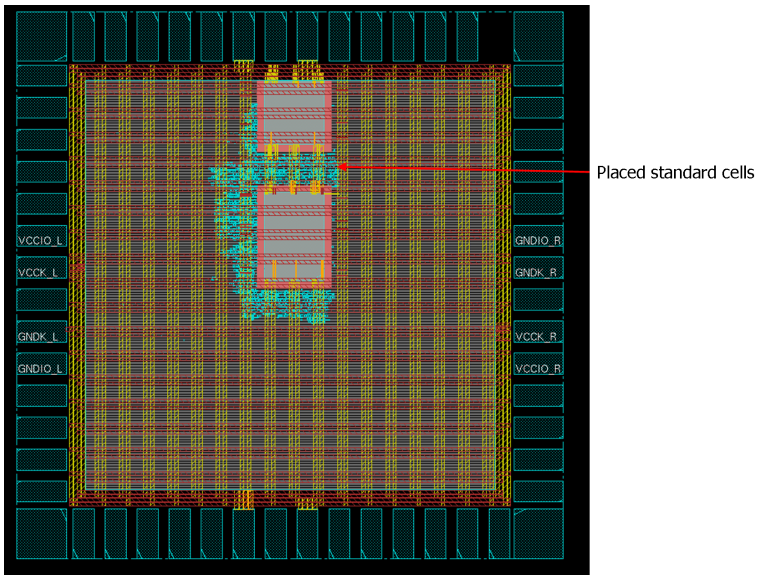


Fig. 6.12: Result of the placement of the design.

6.5.4 Clock Tree Synthesis

The clock distribution network is a very peculiar HFN, because it requires that the skew from the source to all the destinations is minimized, rather than the absolute propagation delay. This objective is reached by building an H-tree with buffers. In Fig. 6.13b, it is possible to see that the number of buffers from the source to any destination is constant. The buffers are identical by-design, as well as the delay of two homologous the branches in the tree. The placement of the buffers is optimized in such a way to minimize the differences among homologous branches. Therefore the H-tree structure allows to balance the propagation delay difference from the source to all the destinations. Fig. 6.14 shows the clock tree of ParticleBlaze. At this level, the delay on the clock tree is estimated since the nets are not routed yet. The tool estimated the maximum skew of the network to be 41.9 ps (the requirement I set was 50 ps), and the delay from the clock pad to the clock pin of the flip-flops (phase delay) to be 1.4 ns (the requirement I set was 1.5 ns). More details on the clock three synthesis results can be found in the

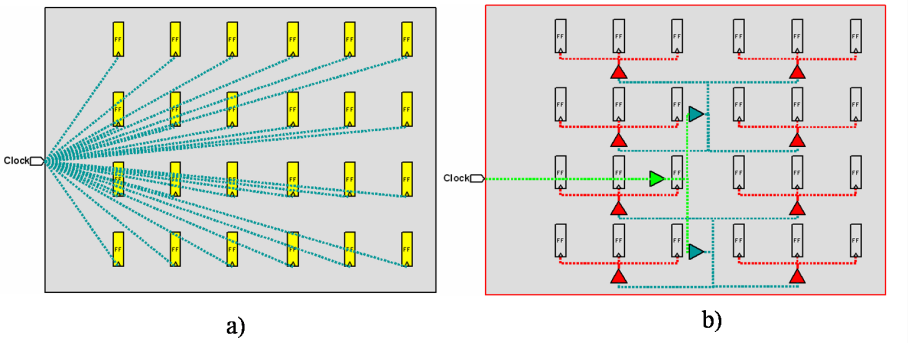


Fig. 6.13: Clock distribution network before and after the CTS: a) All the clock pins are driven by a single source b) H-tree synthesized in order to balance the load capacitances and on each node and minimize the skew.

report in Fig. 6.15.

6.5.5 Routing

In this phase Encounter routes the nets among all the cells by using the metal layers available in the technological process. Before starting the routing the estimated congestion must be acceptable and there must not be wide violations of the timing constraints. The routing of interconnections is performed by placing some metal segments on a grid, which step can be different for each layer. I completed the routing of the clock network first, in such a way for the router to be able to optimize the skew, thank to the fact the all the tracks for interconnections were free. Afterwards I routed the reset network - for similar reasons - and finally I routed all the rest of the signals. After the routing it is possible that the clock distribution network requires a new optimization to clean small timing violations (usually hold-time violations). Also in this case some routed nets might be re-routed, the drive strength of some buffers or their placement may be changed. Fig. 6.16 proposes the result of the routing; let us observe how the nets realized on the metal layers hide the structure of the standard cells, which are realized by using only the

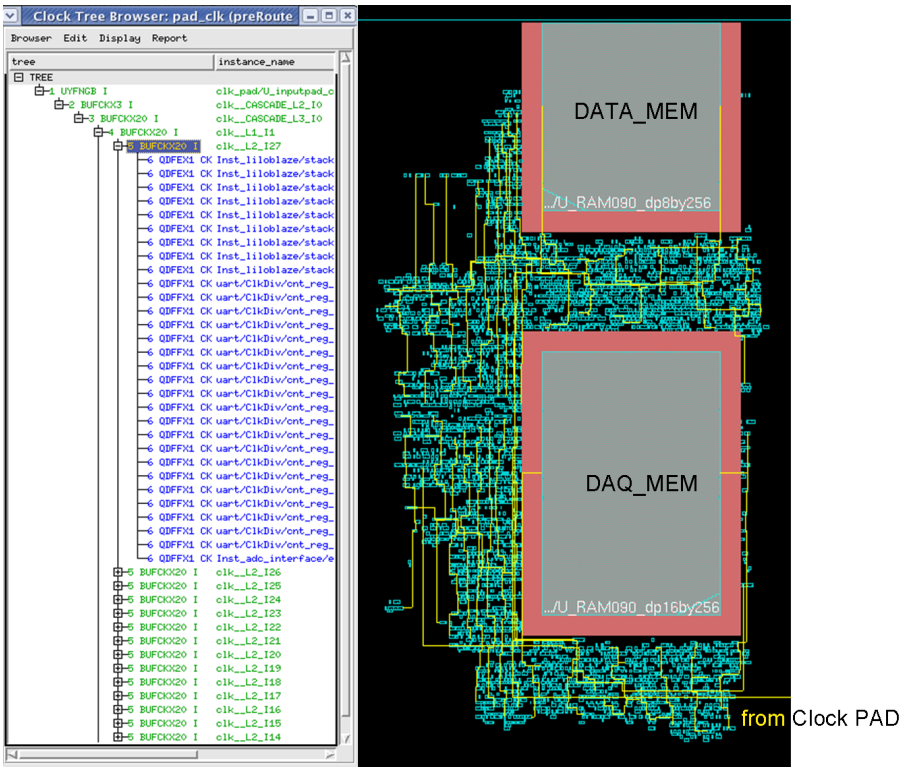


Fig. 6.14: Synthesized clock tree buffers.

lower metal layer. At this level, all the delays (included those of the clock tree) can be precisely estimated since the nets are actually routed. Only at this stage, in fact it makes sense to optimize the placement and the routing for fixing hold time violations. The results of the post-route timing analysis are reported in Fig.

6.5.6 Back-Annotation and Post-Layout Simulation

The design netlist can be modified in any layout design step (map, place, route, CTS). After the routing, I executed a post-layout simulation of the design netlist by also taking into account the delay extracted from the layout (net delays included). The simulator I used is Mentor Modelsim SE. The extraction of the delays is called "back-annotation." All the extracted delays

```
Post CTS timing report
Nr. of Subtrees                 : 2
Nr. of Sinks                    : 858
Nr. of Buffer                   : 71
Nr. of Level (including gates) : 6
Root Rise Input Tran            : 0.1(ps)
Root Fall Input Tran            : 0.1(ps)
Max trig. edge delay at sink(R): 1380.1(ps)
Min trig. edge delay at sink(R): 1338.2(ps)
                                  (Actual)                (Required)
Rise Phase Delay                : 1338.2~1380.1(ps)       1267~1473(ps)
Fall Phase Delay                : 1314.8~1358.2(ps)       1267~1473(ps)
Trig. Edge Skew                 : 41.9(ps)                50(ps)
Rise Skew                       : 41.9(ps)
Fall Skew                       : 43.4(ps)
```

Fig. 6.15: Timing report of the clock network after the synthesis.

Table 6.4: Timing report summary after the routing; `all` indicates all the constrained setup (or hold) paths, `reg2reg` the paths from the output of a register to the input of a register, `in2reg` the paths from a pad to the input of a register, `reg2out` the paths from the output of a register to a pad.

Setup mode	all	reg2reg	in2reg	reg2out
Worst Slack (ns)	0.015	0.015	8.578	0.869
Total Slack (ns)	0.000	0.000	0.000	0.000
Violating Paths	0	0	0	0
Num. of Paths	1332	1304	31	17

Hold mode	all	reg2reg	in2reg	reg2out
Worst Slack (ns)	0.033	0.033	N/A	N/A
Total Slack (ns)	0.000	0.000	N/A	N/A
Violating Paths	0	0	N/A	N/A
Num. of Paths	1304	1304	N/A	N/A

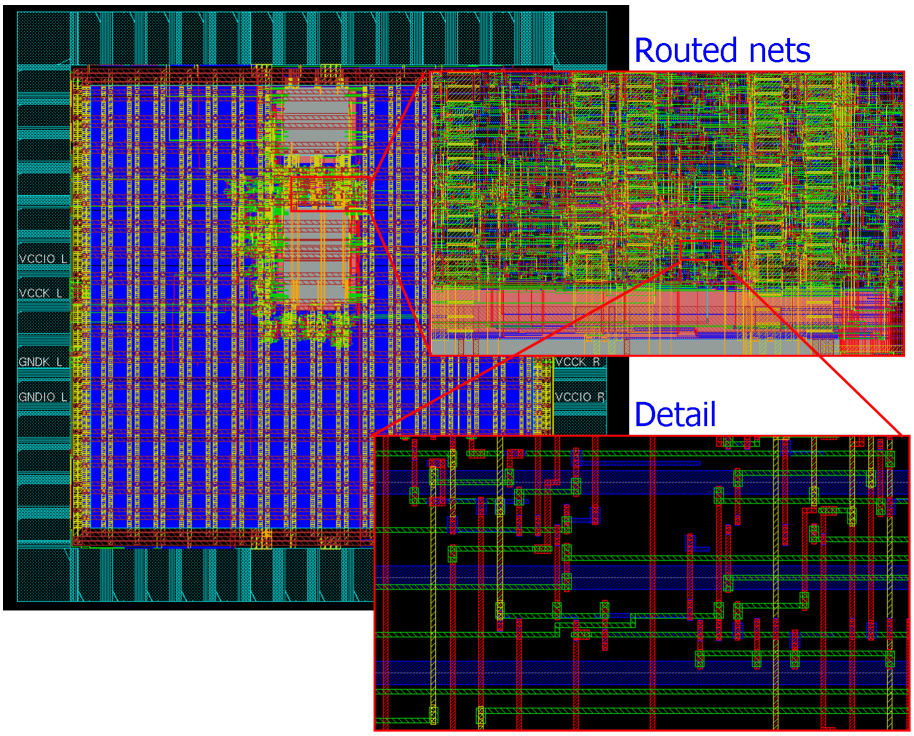


Fig. 6.16: Routed design.

are written in a standard delay format (SDF) file, which is then read by the simulator and used to assign a delay to each element of the netlist to simulate. The netlist extracted from the layout is a Verilog netlist. The simulator allows the user to make a mixed simulation with a VHDL test-bench and a Verilog netlist to simulate; this feature let me use the same VHDL test-bench I used for the logic synthesis.

6.5.7 Sign Off

Before sending the layout to the foundry it is necessary to perform some checks on the design in order to ensure no mistakes have been made during the whole design flow. The whole ensemble of checks is called Sign-Off. I verified, with a conformal check, that the final netlist was logically equivalent to the pre-layout one. Moreover, I performed a design rule check, in order

to ensure all the design rules were respected and there were not electrical errors, such as short or open circuits.

6.5.8 Tape out

The last phase is the tape-out, which consists in sending the layout to the foundry, which before starting the production executes additional checks similar to the Sign-Off, after having replaced the abstract view of standard cells with their complete layout. I will submit the chip in 2011. The device will be fabricated by the UMC foundry in the framework of the miniasic program of the Europractice consortium. The mini-asic program allows to have a very low-cost (few keuros) production of small-sized chips (few mm^2) by sharing of the same mask set for hundreds of different designs. The fabrication of ParticleBlaze will cost around 7 keuros and will provide around 20 samples.

Conclusions

In this Ph.D. thesis work I designed a System-on-Chip (SoC) for the acquisition and the real-time processing of data from micromegas detectors. I carried on my work within the framework of the RD51 collaboration of CERN. The SoC I designed includes a microprocessor and a set of peripherals for real-time data processing. Although my design is meant for the application to micromegas, it has a very flexible architecture could be used with a large variety of detectors with an analog output after the appropriate analog pre-processing of the signal (e.g. charge/current pre-amplification, sample and hold, integration). Moreover, I wrote the HDL description of the SoC to be completely technology-independent, therefore it is portable to any technology suited for the implementation of digital devices (e.g. VLSI CMOS or FPGA).

I realized a prototype of the system and I developed an hardware test bench in an FPGA order to verify on-field the logic description of the project. The test bench stimulates the system by emulating the behavior of external elements, which it will be interfaced to once it will be in real operation: the ADC and the trigger logic. This methodology allowed me to verify the successful operation of the system under conditions very similar to the real ones and permitted me to quickly develop and debug the firmware to be executed by the microprocessor. I used the same Xilinx Virtex-II V1000 FPGA for implementing both the SoC prototype and the hardware test bench. The maximum operation frequency of the prototype resulted to be $f_{FPGA} =$ 100 MHz and the percentage of the occupied device is 9 %.

For the development of VLSI chip, I used the HDL description tested on

the FPGA. I adopted a standard cell approach and the CMOS 90nm technological process from the UMC foundry. The chip I designed fulfills all the requirements in terms of logic footprint, re-programmability and functionality. The maximum operation frequency resulted to be f_{VLSI}= 277 MHz; the die takes around 1.734 x 1.734 mm^2 with a percentage of effectively used core area of 10%. The size of the device has been limited by the number of IO pins (55) but was anyway below the size limit imposed by the miniasic block (1.875 x 1.875 mm^2). Thanks to its very low logic footprint, it will be possible to develop a multi-channel version of the system simply by iterating in the same chip several instances of the SoC. Each instance will be dedicated to a detector strip. Moreover, since VLSI devices can host analog and digital circuits on the same die ("mixed signal" devices), it will be possible to integrate the ADC on the device to further reduce the number of the needed external components.

Bibliography

[1] The RD51 Collaboration, "RD51 2008-001 R&D Proposal Development of Micro-Pattern Gas Detectors Technologies," 2008, Available: [On-line] http://rd51-public.web.cern.ch/RD51-Public/Documents/RD51Proposal_21082008.pdf

[2] F. Sauli, "Gem: A new concept for electron amplification in gas detectors," Nucl. Inst. & Meth. A, vol. 386, issues 2–3, 21/2/1997, pp. 531–534.

[3] Giomataris et al., "MICROMEGAS: a high-granularity position-sensitive gaseous detector for high particle-flux environments," Nucl. Inst. & Meth. A, vol. 376, 1996, pp. 29-35.

[4] J.K. Black et al., "The imaging x-ray detector for Lobster-iss, "Nucl. Inst. & Meth. A, vol. 513, 2003, pp. 123–126.

[5] Z. Hajduk, G. Charpak, A. Jeavons, R. Stubbs, "The spherical drift chamber for x-ray imaging applications," Nucl. Inst. & Meth. I22, 1974, pp. 307–312.

[6] P.F. Christie, E. Mathieson and K.D. Evans, "An x-ray imaging proportional chamber incorporating a radial field drift chamber," Journal of Physics E: Scientific Instruments, vol. 9, 1976, pp. 673–676.

[7] S. Duarte Pinto et al., "Spherical gems for parallax-free detectors," arXiv:0911.3255v1 [physics.ins-det], 2009 [On-line] Available: http://arxiv.org/abs/0911.3255v1

[8] S. Duarte Pinto et al., 2009, JINST 4 P12009.

[9] G Croci et al., 2010 JINST 5 P03001.

[10] R. GIORDANO et al., "Development of large size Micromegas detector for the upgrade of the ATLAS Muon system," Nucl. Inst. & Meth. A, vol. 617 , 2010, pp. 161–165.

[11] J. Treis, et al., Nucl. Instr. and Meth. A 490 (2002) 112.

[12] The ALICE Collaboration, CERN-LHCC-2003-062, 7 January, 2004.

[13] H. Muller, et al., Nucl. Instr. and Meth. A 565 (2006) 768.

[14] R. Esteve Bosch, et al., IEEE TNS 50 (6) (2003).

[15] The RD51 collaboration, "Scalable Readout System," [On-line] Available: http://rd51-public.web.cern.ch/RD51-Public/Activities/Documents/WG5SRS.pdf

[16] G. Charpak, J. Derr, Y. Giomataris, Ph. Rebourgeard, Nucl. Instr. Meth. A478(2002).

[17] I. Giomataris, "MICROMEGAS: results and prospects" CEA/Saclay, DAPNIA,91191 Gif-sur-Yvette Cedex. France.

[18] G. Charpak et al., Nucl. Instr. Meth. A412(1998)47.

[19] G. Barrouch et al,, Nucl. Instr. Meth. A423(1999)32.

[20] J. Va'vra et al., Nucl. Instr. Meth. A324(1993)113.

[21] J. Derrè, et al., Nucl. Instr. and Meth. A 449 (1999) 554.

[22] J. Derrè, et al., Nucl. Instr. and Meth. A 459 (2001) 523.

[23] G. Martinez, A photon detector for relativistic heavy ion collisions, SUBATECH 99-10.

[24] D. Thers, et al., DAPNIA/SPHN-00-37, Nucl. Instr. and Meth., in preparation.

[25] A. Magnon, Micromegas for Compass, Presented at the Ninth Vienna Conference in Instrumentation, Vienna, February 19–23, 2001.

[26] "Picoblaze 8-bit Microcontroller for CPLD devices (XAPP387(v1.1) 9/1/2003)," Xilinx, 2003, online available.

[27] M. R. Stan and W. P. Burleson, "Bus-invert coding for low power I/O," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 3, no. 1, pp. 49–58, Mar. 1995.

[28] A. Aloisio, P. Branchini, F. Cevenini, R. GIORDANO, V. Izzo, S. Loffredo, "Bus-Invert Coding For Low Noise 2eSST VME64x Block Transfers," IEEE Trans. on Nucl. Sci., vol. 54, issue 3, pages 616-622, June 2007

[29] A. Aloisio, F. Cevenini, R. Cicalese, R. GIORDANO, V. Izzo, "Beyond 320 Mbyte/s with Bus-Invert and 2eSST on VME64x," IEEE Trans. on Nucl. Sci., vol. 55, issue 1, part 1, pages 203-208, February 2008

[30] "Minimal UART Core," May 5 2010, [On-line] Available: http://opencores.org/project,muart

[31] "XAPP694 (v1.1.1) - Reading User Data from Configuration PROMs," Xilinx, November 19, 2007 [On-line] Available: http://www.xilinx.com/support/documentation/application_notes/xapp694.pdf

[32] S. Sjoholm e L. Lindh, "VHDL for designer," 1996.

[33] Virtex-II Platform FPGA User Guide, Xilinx, 2004, online available.

[34] "Virtex-II V2MB1000 Development Board User's Guide," Version 3.0, 2002, Memec Design.

[35] Jan M. Rabey, Anantha Chandrakasan, Borijove Nikolic, "Digital Integrated Circuits, a design perspective, second edition," 2003, Prentice Hall, Chapter one.

[36] Yuan Taur, Tak H.Ning, "Fundamentals of Modern VLSI Devices," Cambridge University Press, 1998.

[37] Behzad Rezavi, "Design of Analog CMOS Integrated Circuits," Tata McGraw-Hill Edition, 2002.

[38] S. M. Sze, "VLSI Tecnology second Edition," Tata McGraw-Hill Edition, 2003.

[39] Philip E. Allen, Douglas R. Holberg, "CMOS Analog Circuits Design," 2003.

[40] "FSD0A_A_GENERIC_CORE UMC 90 nm Logic SP/RVT Low-K Process 90 nm GENERIC CORE CELL LIBRARY DataBook," Faraday, Rev. 0.3, February 2009.

[41] "FOD0A_B25_33VT_GENERIC_IO UMC 90 nm Logic SP/RVT Low-K Process 90 nm 2.5 V WITH 3.3 V TOLERANCE GENERIC I/O CELL LIBRARY," Faraday, Rev. 1.1, February 2009.

[42] David A Hodges, Resve Saleh, Horace G. Jackson, "Analysis and Design of Digital Integrated Circuits Third Edition", 2004.

[43] "Design Compiler User Guide," Synopsys, 2005, Chapter one.

[44] Samir Palnitkar, "Verilog Hdl : A Guide to Digital Design and Synthesis (2nd ED)," 2001.