High Performance Logic Devices and Applications to CODAC systems

Bernardo Brotas de Carvalho
bernardo.carvalho@tecnico.ulisboa.pt

Instituto de Plasmas e Fusão Nuclear
Physics Department
Instituto Superior Técnico, Universidade de Lisboa
Portugal
http://www.tecnico.ulisboa.pt

Summary

- Applications of FPGAs
- FPGAs: Main benefits
- Evolution of PLDs
- Development Code Model
- Example Language: VHDL Introduction
- Applications to CoDAC Systems
- Nuclear Fusion Research

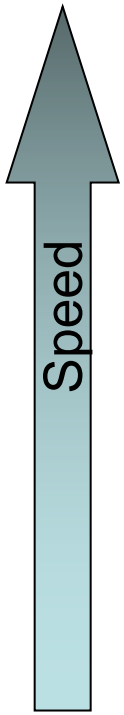
Applications of FPGAs

- Digital signal processing
- High Performance Computing
  - FFT or Convolution
  - Massive parallelism
  - Fast control systems
- Software-defined radio
- Aerospace, Defense systems
- Computer vision
- Medical imaging, Bioinformatics,
- Speech recognition, cryptography, Data compression
- Computer hardware emulation
- "Glue logic" for PCBs
- Full Systems on a Chip (SoC)
- Internet-of-Things
- High Energy Physics
- Nuclear Fusion

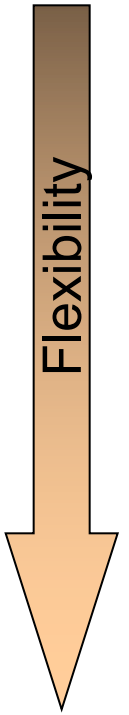
FPGAs: Main benefits

- Performance:
  - Massive parallelism
  - High Speed
  - High Logic Capacity
  - More rapid than Microprocessors and DSPs
- Easy long term Maintenance
  - Re-programmable at any time
    - In some case it can even program "itself"
  - New functions do not break existing code
- Fast development
  - FPGA is Commercial off-the-shelf (COTS)
  - Compilation made easy as there are a lot of libraries and "ip-cores" (some can cost money…)
- Lower initial cost and risk factor than ASICs
- No Operating System
  - Less Bugs
  - True deterministic Operation
  - Each "task" runs in dedicated hardware!

Comparison with other Technologies



Technology	Ratio Performance/ Cost	Developing Time to first operation	Developing Time to get High performance	Developing Time to change functionalities
ASIC	Very High (for large qtys)	Very Long	Very Long	Impossible
FPGA	Medium	Medium	Long	Medium
DSP	High	Long	Long	Long
General Purpose CPU	Low /Medium	Very Short	Medium	Very Short

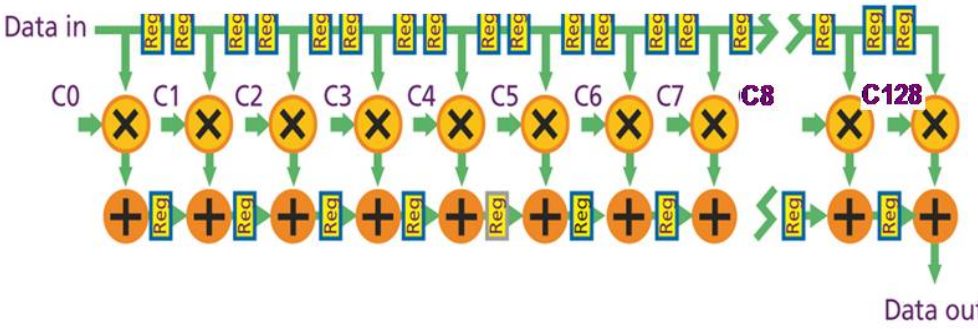


Digital Processing Systems: FPGAs vs DSP

- FPGA versus conventional DSP devices: FIR filters

$$y[n] = \sum_{i=0}^{N-1} c_i \cdot x[n-i]$$

FPGA-based DSP - Parallelism



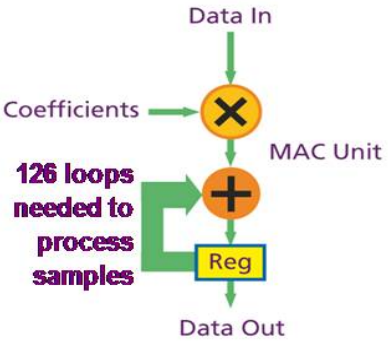
Virtex-DSP

600 MHz / 1 clock cycle = 600 MSPS

Spartan-DSP

250 MHz / 1 clock cycle = 250 MSPS

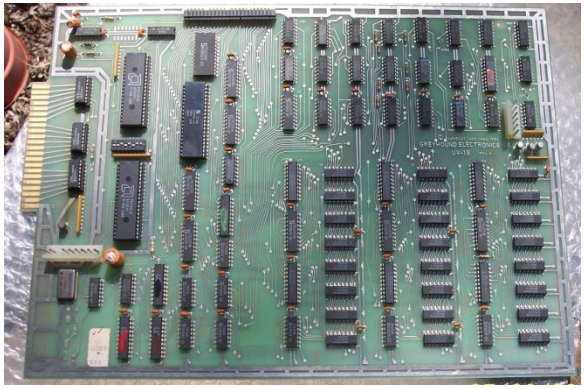
Conventional DSP Processor - Serial

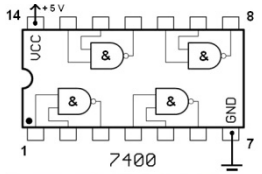


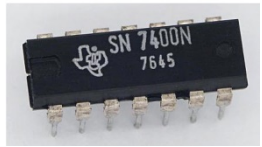
1 GHz / 126 clock cycles = 8 MSPS / MAC unit

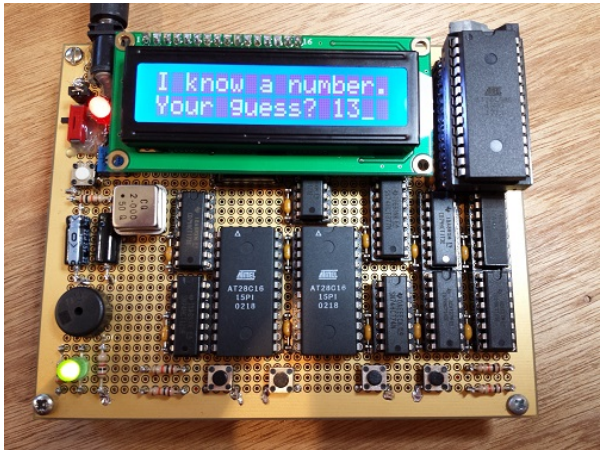
Before the Age of Programmable Logic Devices

- Digital Logic Circuits
  - state machines, automation, controllers, counters, decoders, glue logic, etc..
- Implemented
  - Several gates / flip-flops
  - discrete devices SSI (small-scale integration) chips (e.g 7400-series parts)
- Complex systems (e.g.. CPU) may require an excessive number of SSI chips

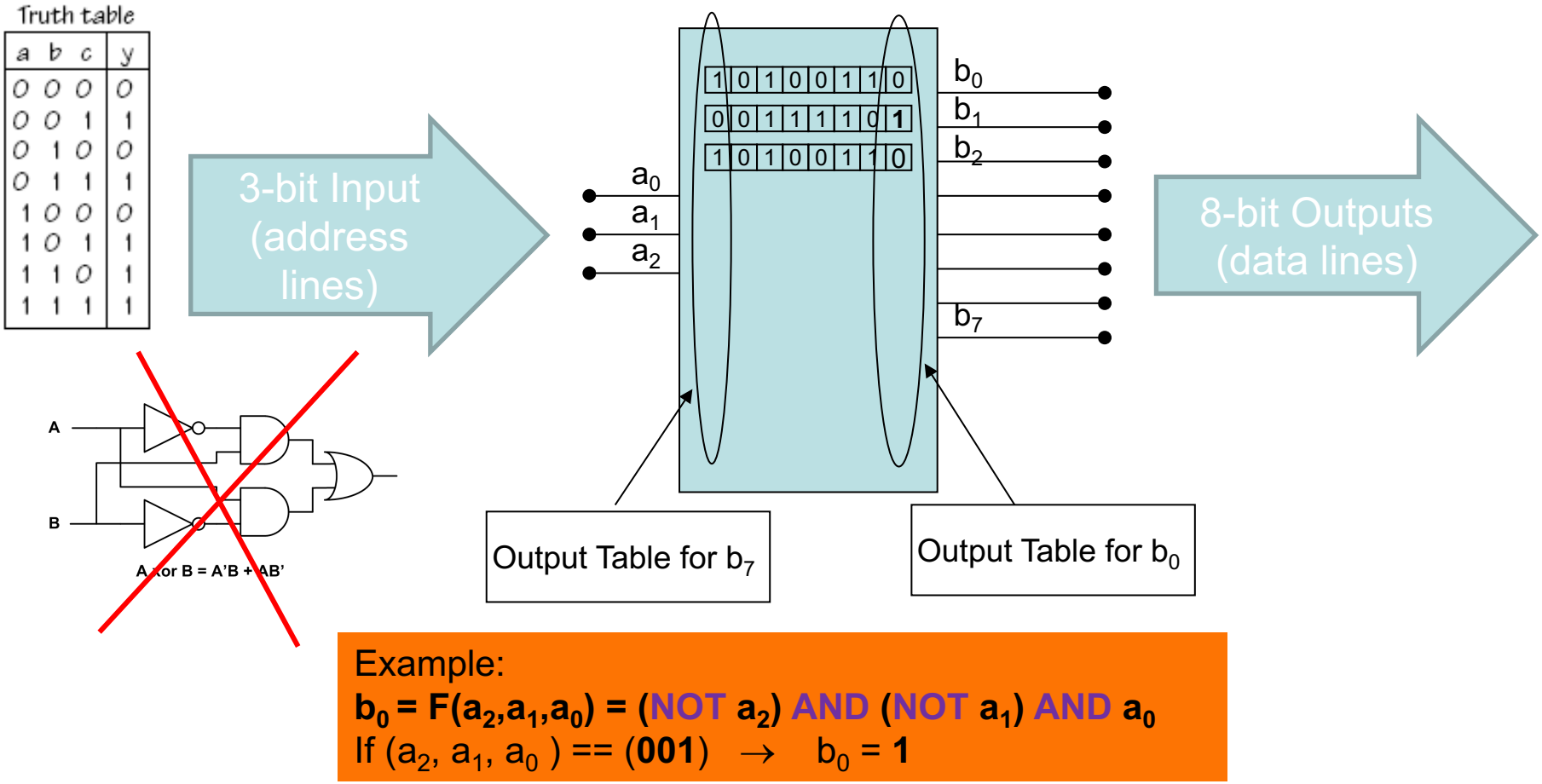








First PLDs: Combinatorial Logic Circuit implemented with a Memory (ROM / RAM)

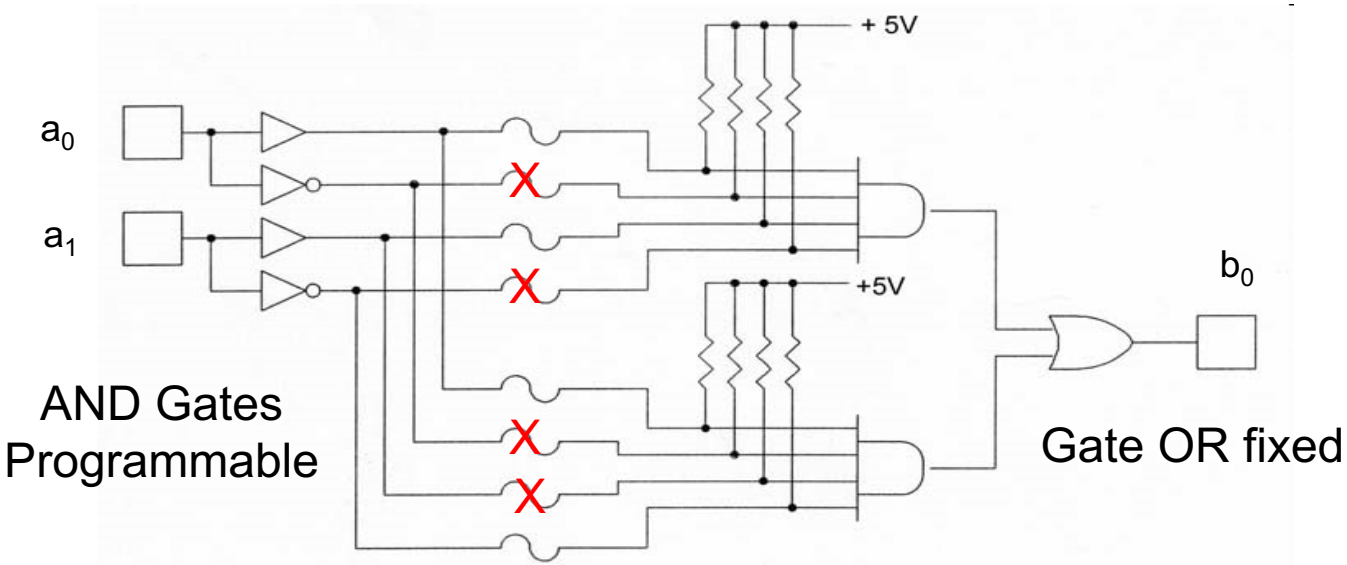


Example:
$b_0 = F(a_2,a_1,a_0) =$ (NOT a_2) AND (NOT a_1) AND a_0
If (a_2, a_1, a_0) == (**001**) → b_0 = **1**

Very inefficient! (however similar method is used on present FPGA LUT...)

First dedicated PLDs: Programmable Array Logic (PAL)

One time Programmable
by burning fuses

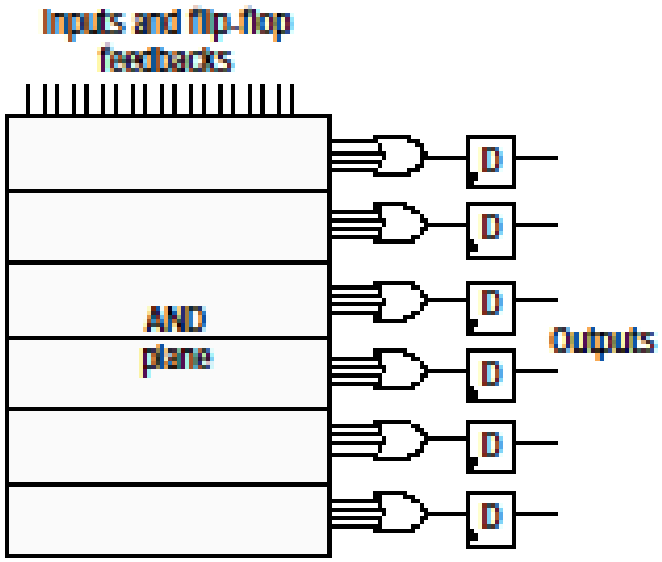


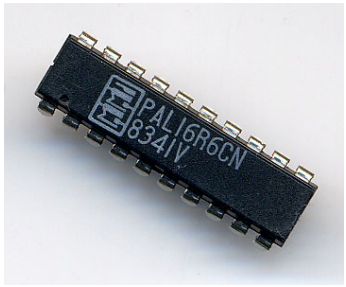
Simplified programmable logic device

Example:
$b_0 = F(a_1, a_0) = a_1 . a_0 + a_1 . a_0$

The PAL has TWO Logic Levels:
One column 'AND' and one column 'OR'

Typical PAL families

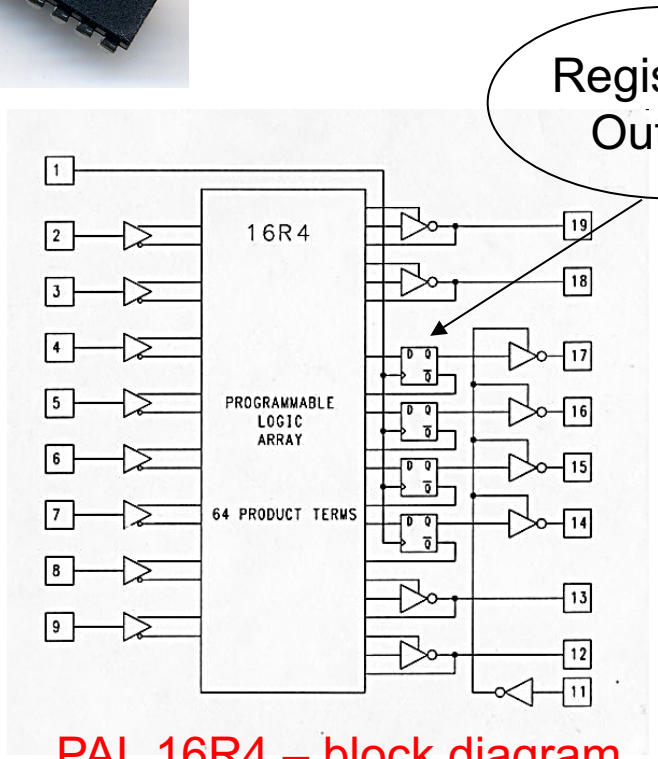




EXAMPLES

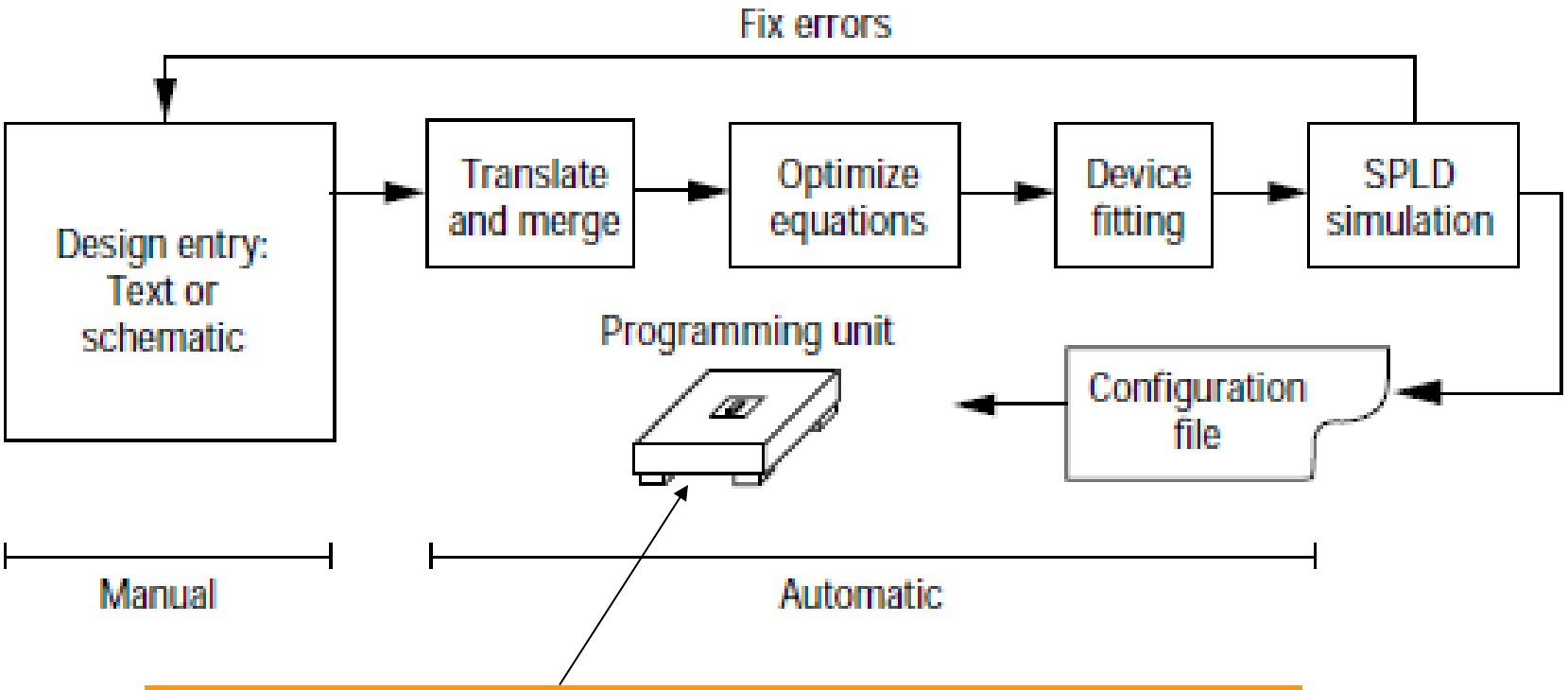
- PAL16L8 = 8 combinatorial outputs
- PAL16R8 = 8 registered outputs

- PAL16V8 = 8 “variable” outputs



PAL 16R4 – block diagram

PAL Design Flow



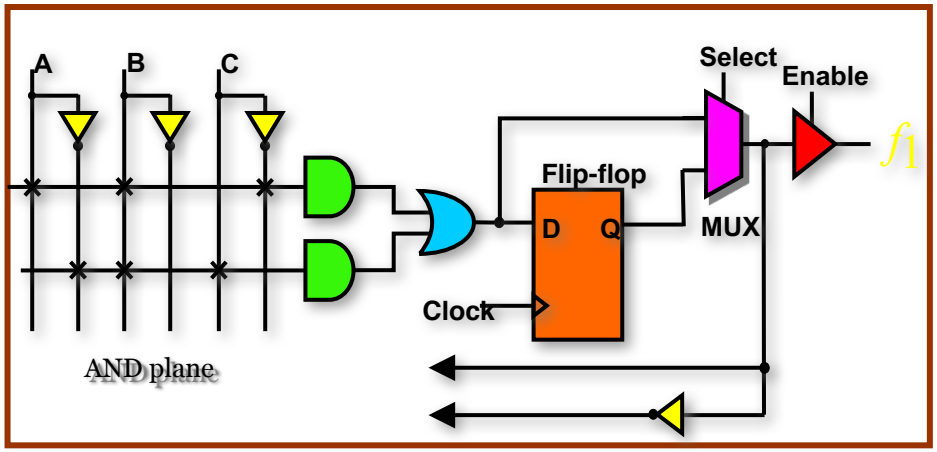
Program types:

- Devices PROM (burn fuses permanently)
- Devices E-PROM (UV - 'Erasable' PROM)

Next Levels: SPLD -> CPLD

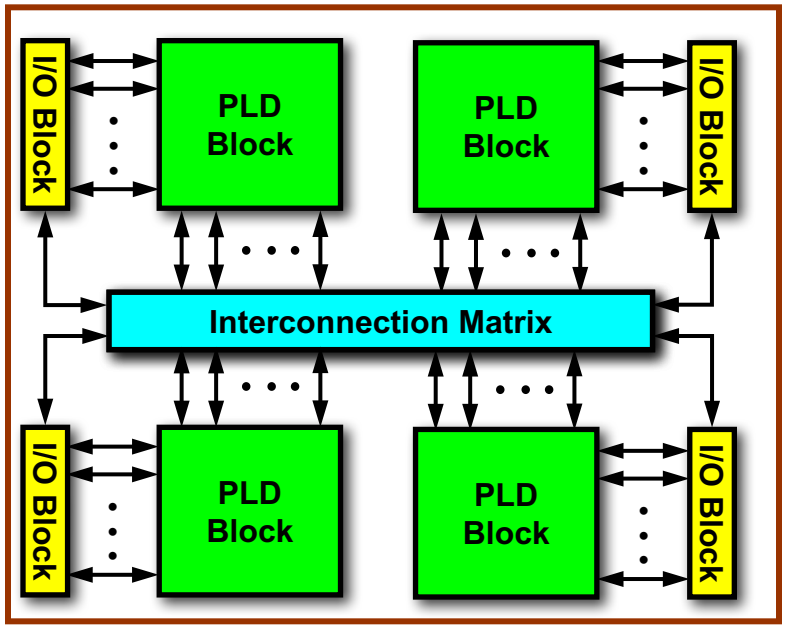
Simple Programmable logic device

- Only one-level AND
- Flip-Flops and
- Feedback”



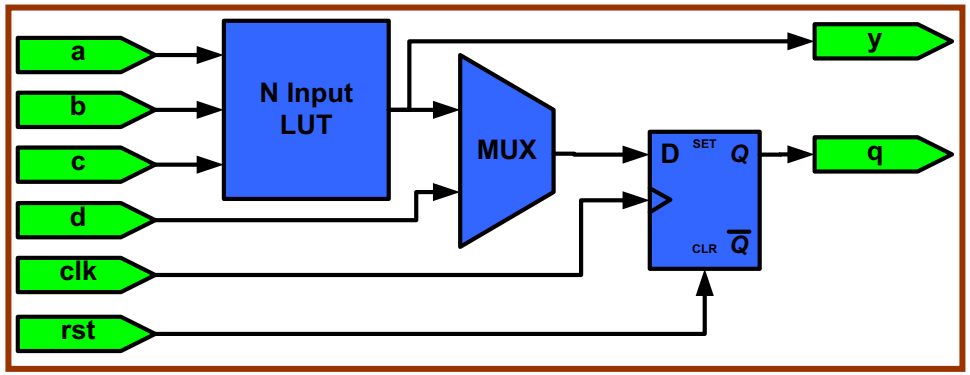
Complex Programmable logic device

- Several interlinked PLDs
- Logic capacity equivalent of up to 50 SPLD

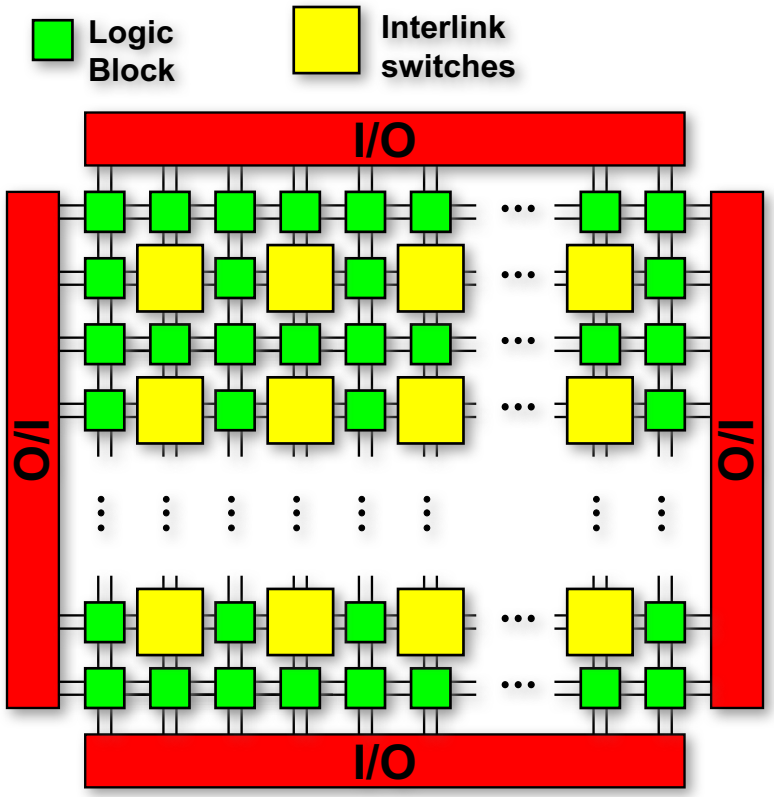


Field Programmable Gate Array

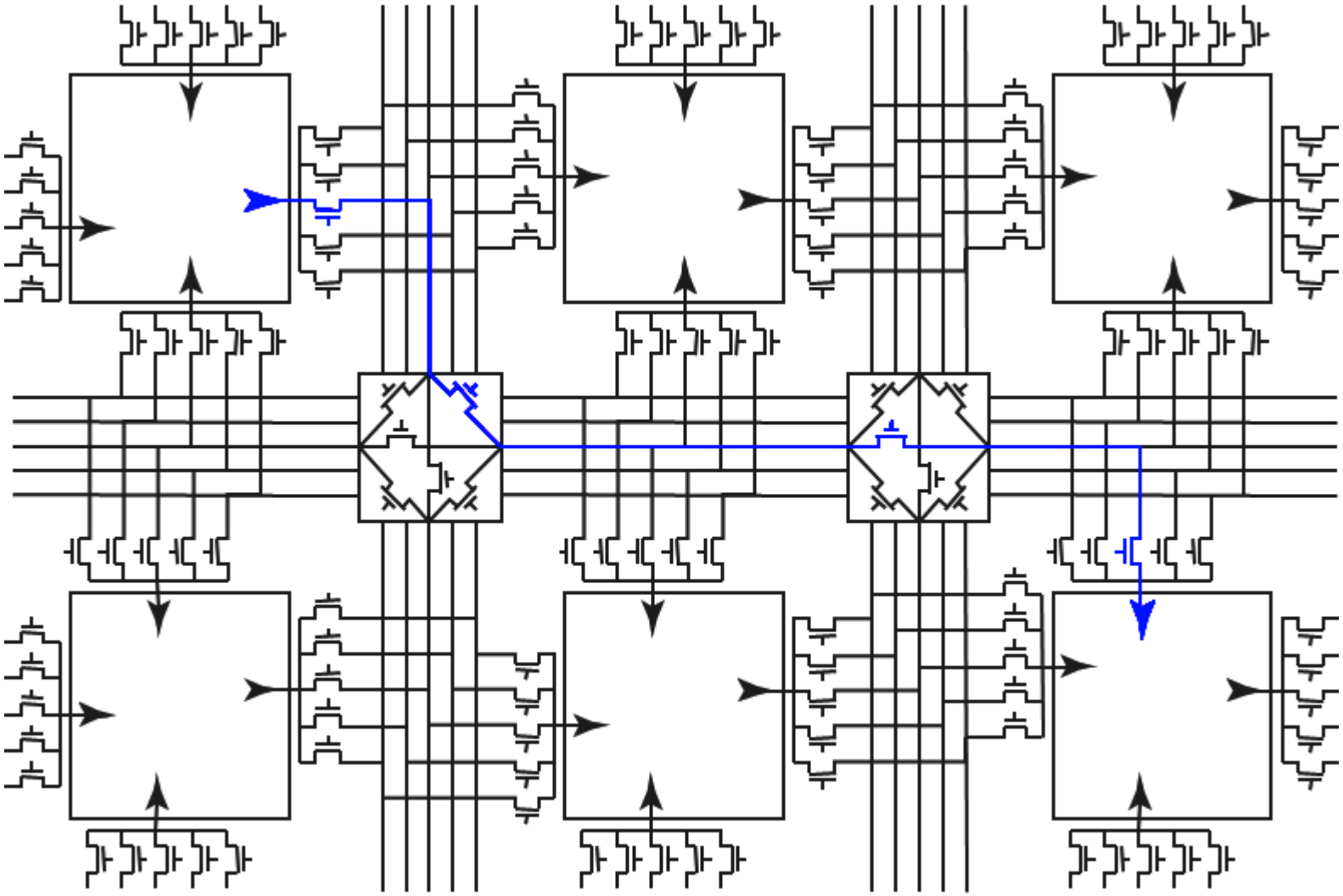
- The FPGA is a set of Configurable Logic Blocks (CLB), connected through a matrix.
- Interlinked by switches which allow to re-arrange the links.
- I/O Blocks configurable for external physical pins.



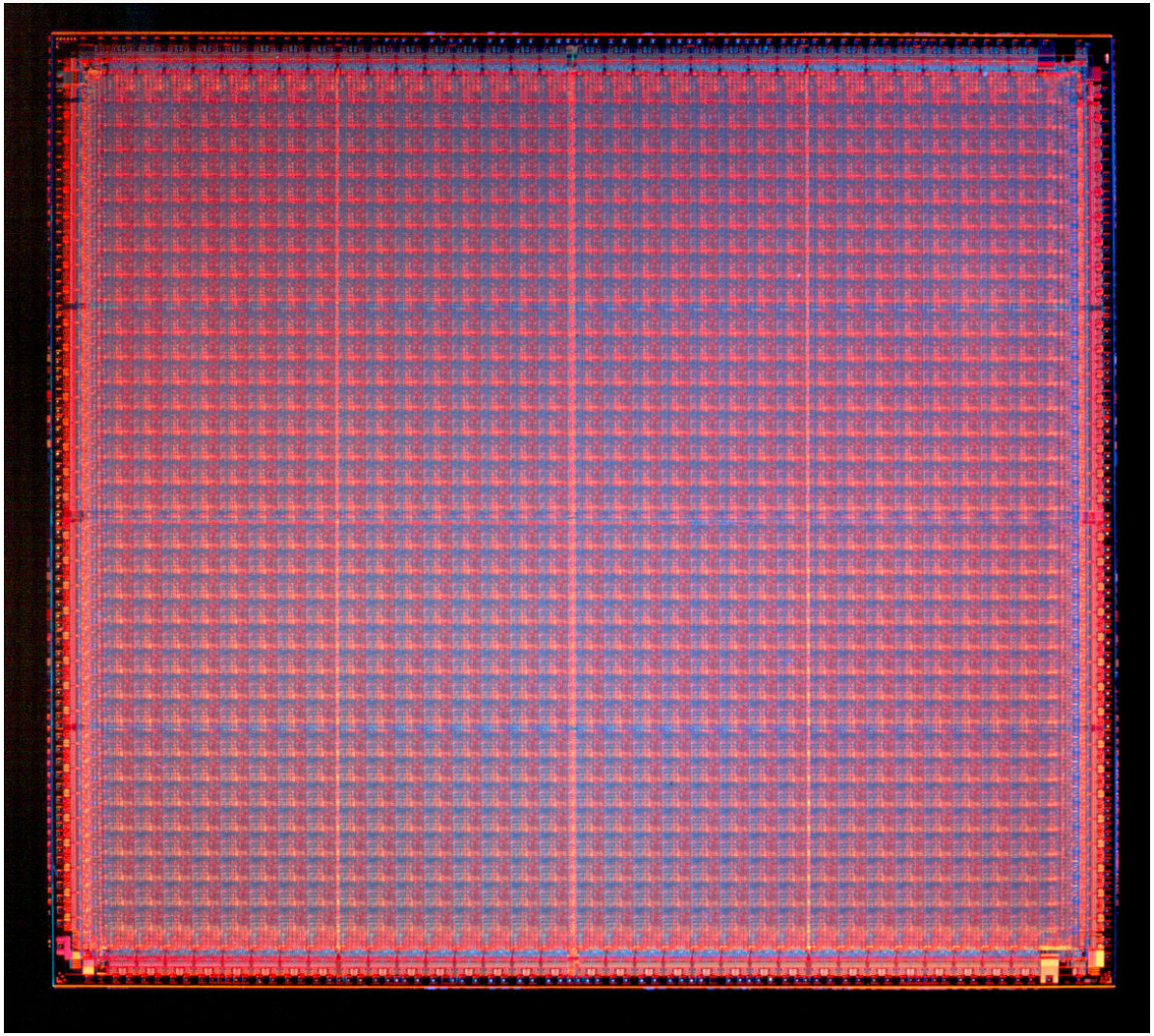
- Special Clock Lines to make synchronous transitions across device
- Additional Logic Resources such as ALUs, Memory RAM blocks, DSP, CPU, etc.



FPGA switching Matrix



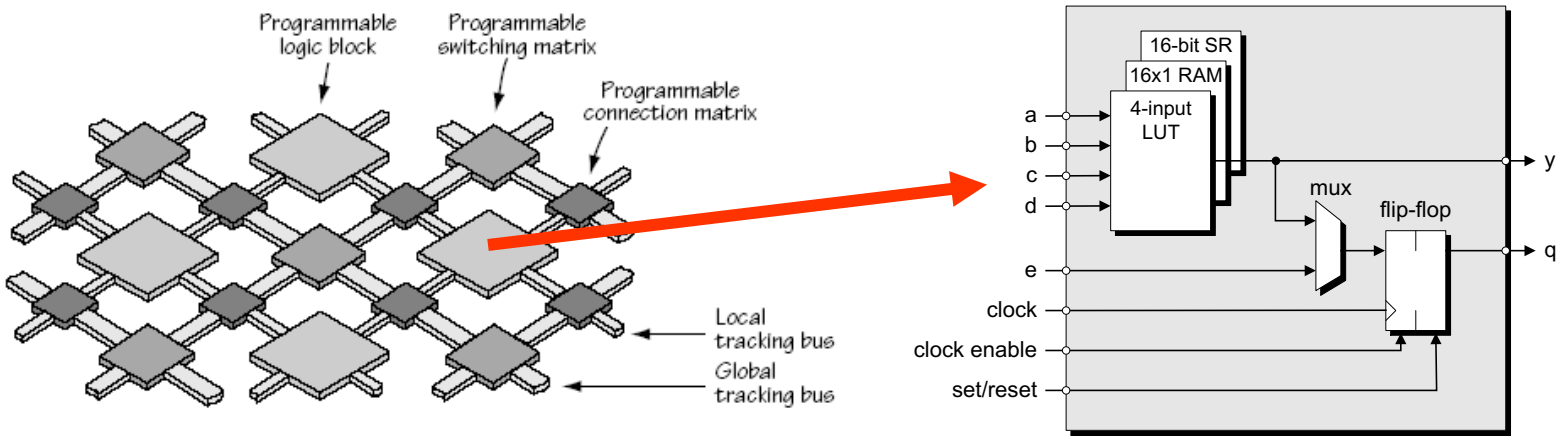
FPGA – Microscope photo



Xilinx Virtex 4 XC4000

Configurable Logic Blocks

- Implement Logic Functions in Look-up-Table (LUTs) (same as primordial memory based PLD…)
- Multiplexers (Select 1-of-N lines)
- “Flip-Flops”. Synchronous Registers.

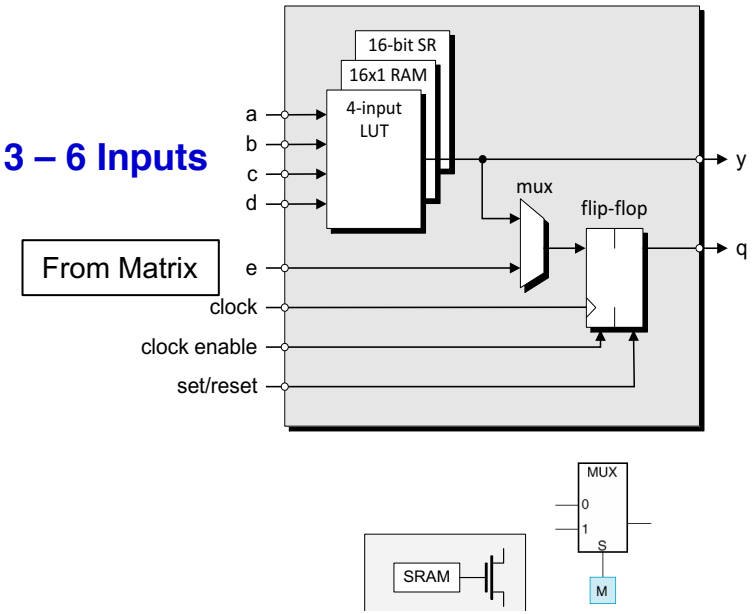


FPGA Grid **CLB Block**

Lookup Tables LUTs

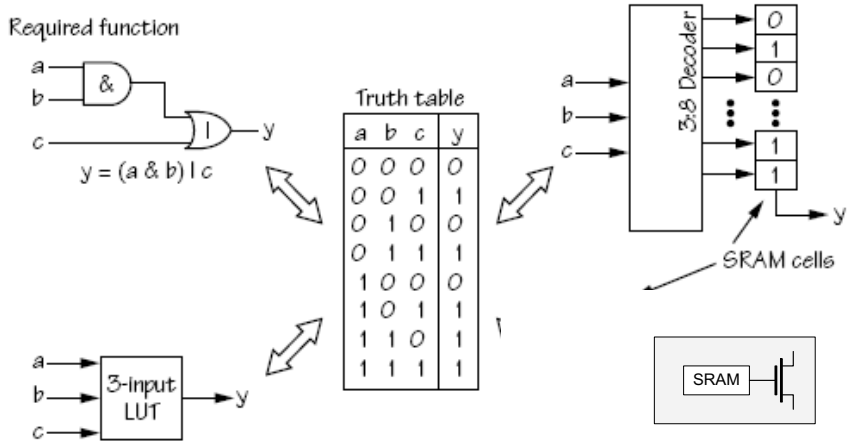
- The LUT contain Memory Cells which implement small combinatorial Logic Functions, using.
- LUT is programmed with a a Boolean Table
- LUT can implement *any n* input Function!

3 – 6 Inputs



Multiplexer (MUX)

3 input LUT -> 8 mem cells

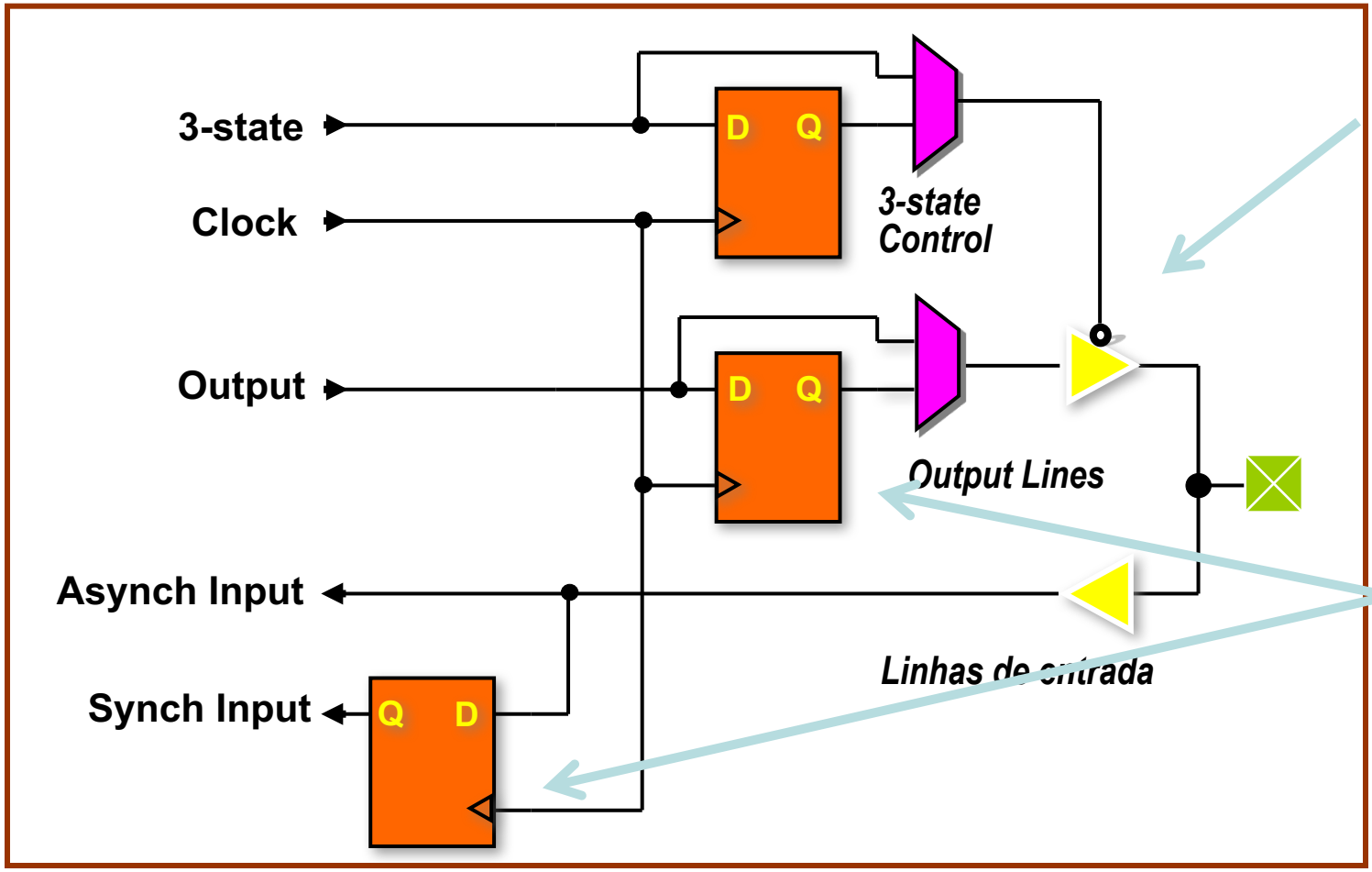


Static Random Access Memory (SRAM) Cells

Input Output Blocks

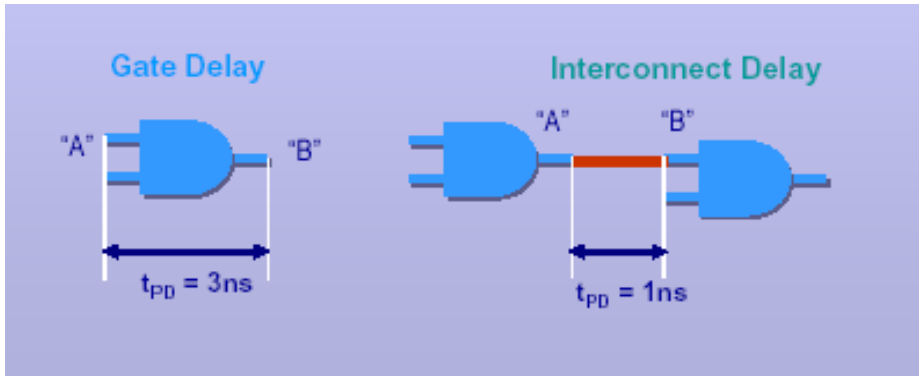
Makes connections between physical pins and CLBs.

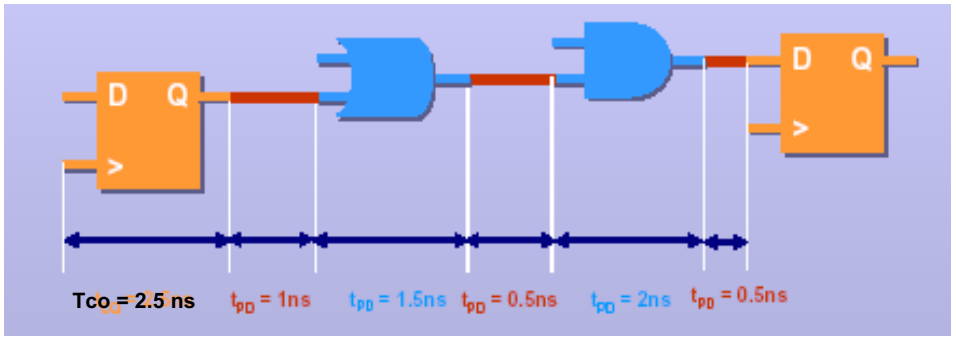
Each physical pin may be configured as input or output.



Delays and Maximum Operating Frequency

- Logic cell Delay: t_{PDL}
- Flip Flop Delay: t_{CO}
- Interlink Delay : t_{PDI}



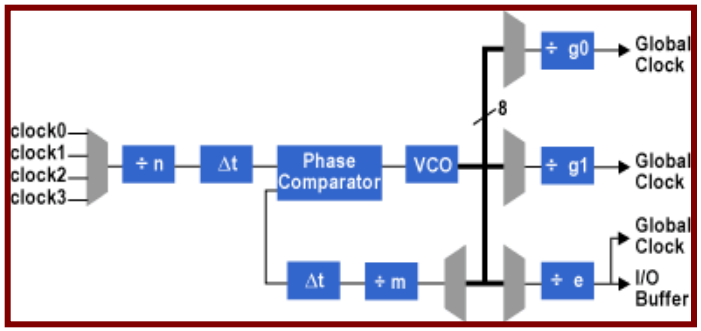


$$1/F_{max} = Tco + Tpd_{logic} + Tpd_{link}$$

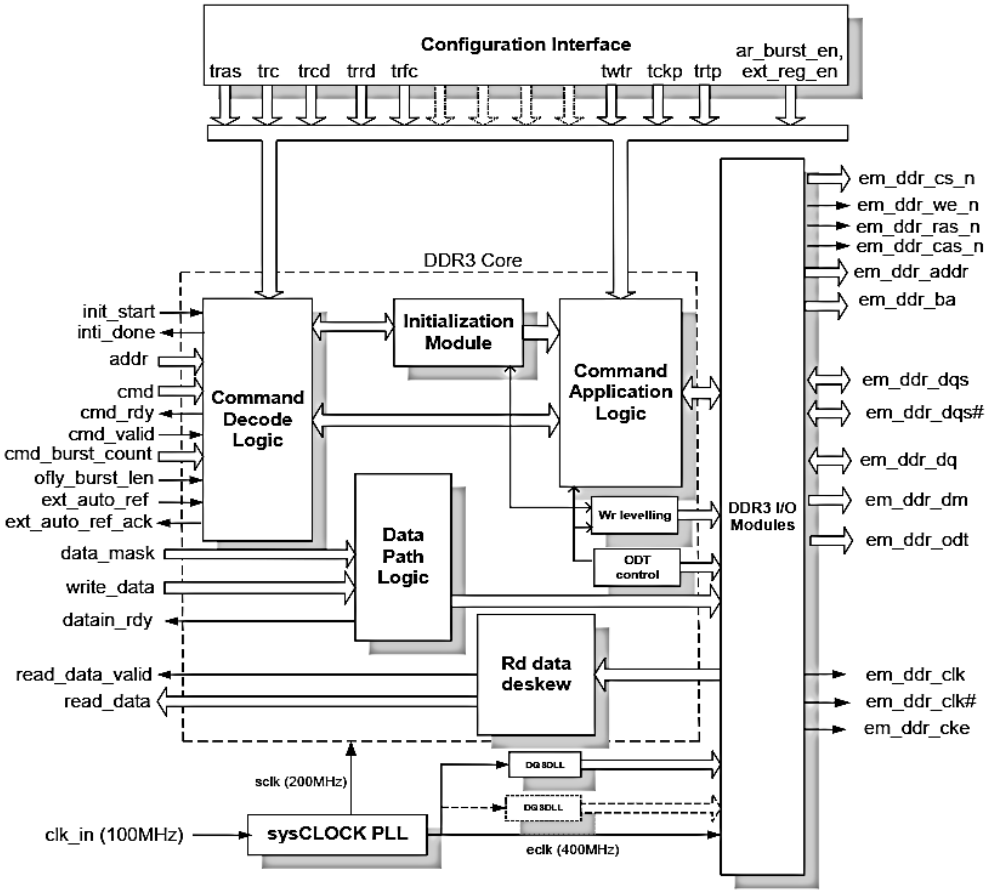
Maximum Frequency-> Maximum speed a logic circuit can operate.

Other FPGA Special Hardware Blocks

- Internal SRAM
- External DRAM controller
- DSP blocks, Multipliers;
- Embedded Logics Analyzer
- Embedded CPUs (Power PC, NIOS, ARM)
- Very High Speed I/Os (~10GHz)
- PLLs and Delay Blocks

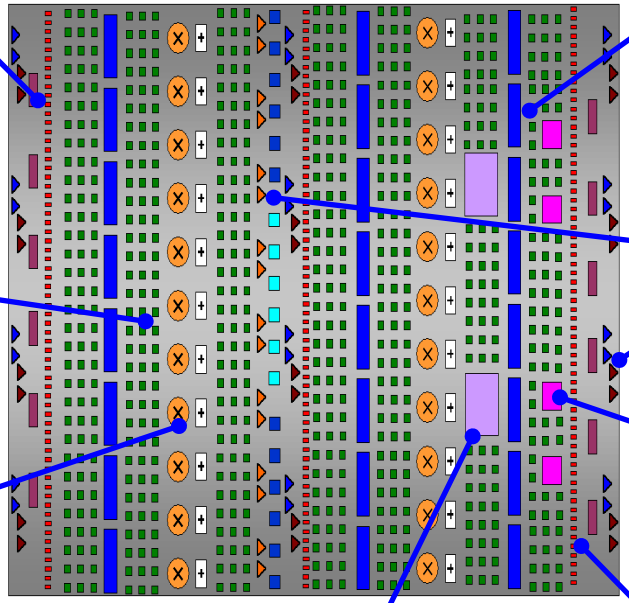


PLL



DDR3 SDRAM controller

Example: XILINX Virtex-4



Multi-Gigabit Transceivers
622 Mbps–10.3 Gbps

CLBs
200K Logic Cells

DSPs
256 18x18 GMACs

PowerPC™ 405
450 MHz, 680 DMIPS

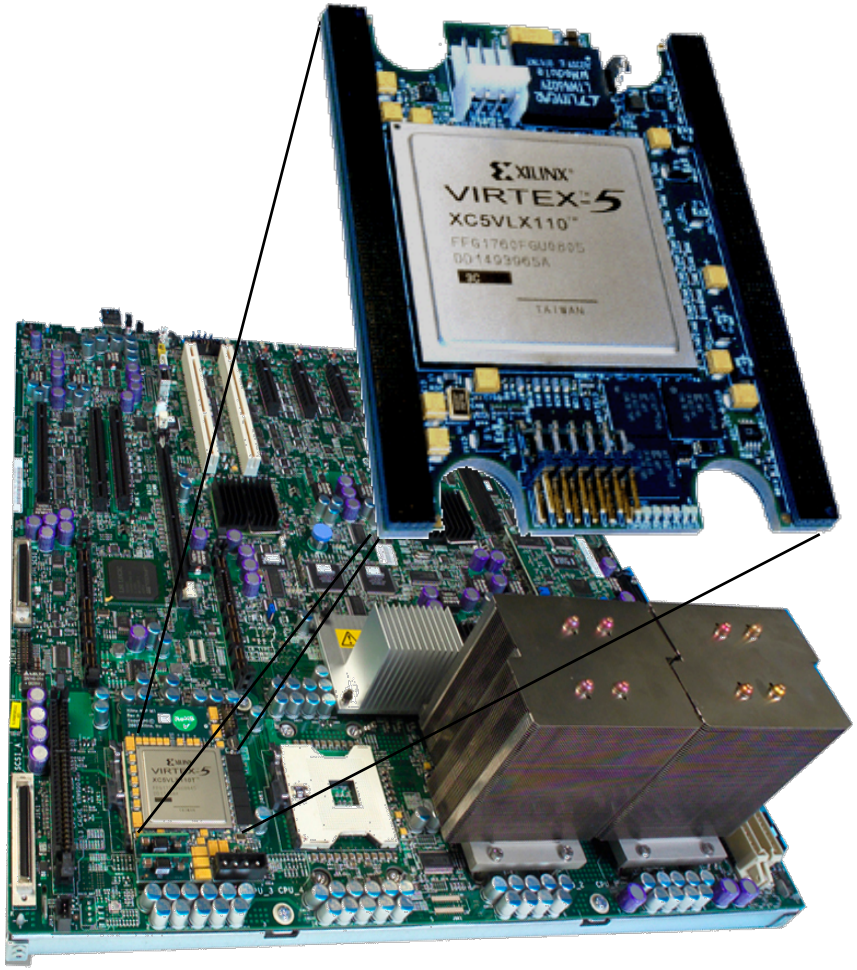
SRAM
Block RAM/FIFO

Clocks/PLL
500 MHz

Tri-Mode Ethernet MAC
10/100/1000 Mbps

1 Gbps IO

FPGA as a “co-processador“ for CPUs



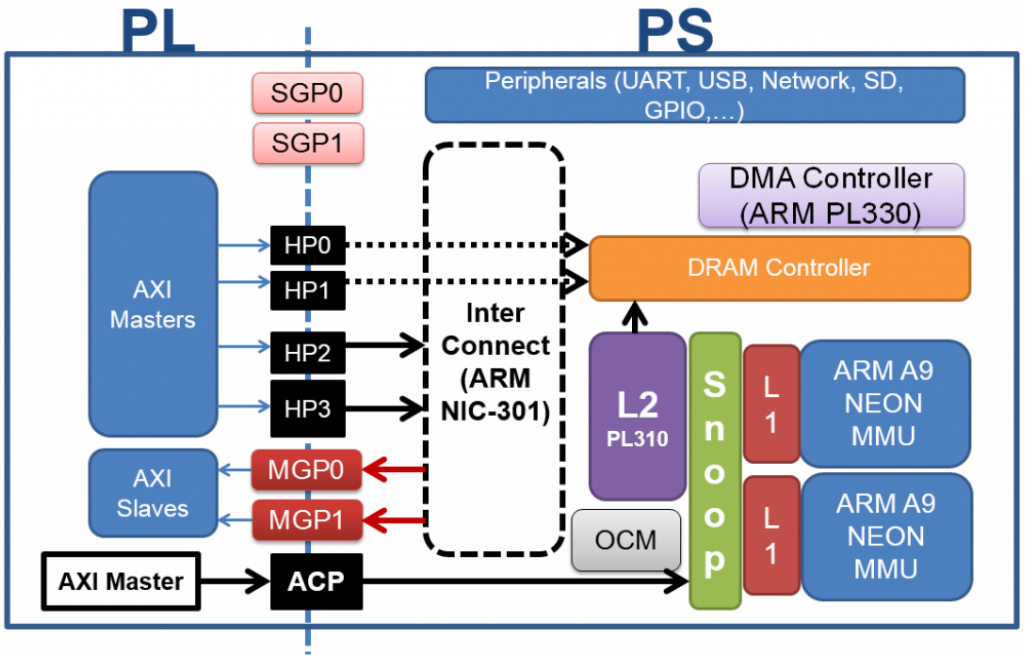
Example: Xilinx/Intel

- Multi-core
  - –x86 cores, FPGAs
- Performance
  - –8.5 GB/s bandwidth
  - –105 ns latency para 64 bytes
- Programming Model
  - –Shared Memory Space
  - –Always “Coherent” Memory

System on a Chip (SoC)

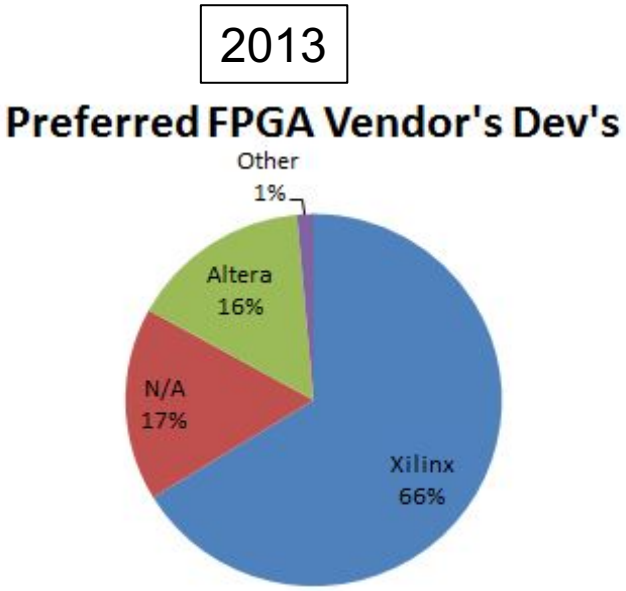
- One single FPGA chip:
  - Implements (almost) all functions of a complete computer (Processing System PS)
  - Additional Programmable Logic (PL) resources

ArduZynq
Arduino compatible
Xilinx Zynq
FPGA module



FPGA Market by Vendor

2013

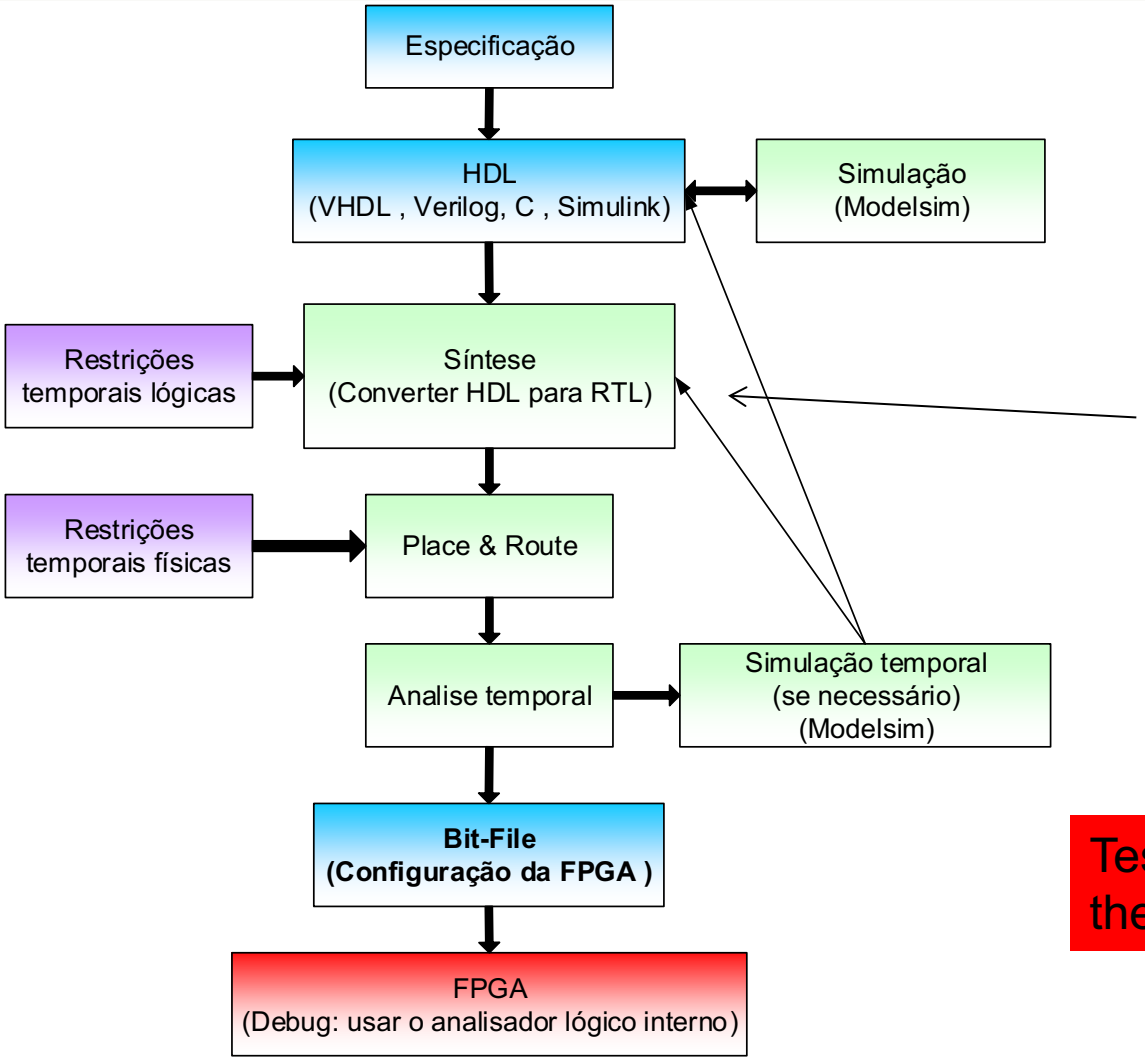


	2015		2016		
Vendor	FPGA Total	Market share	FPGA Total	Market share	Growth CY15-CY16
Xilinx	$2,044	53%	$2,167	53%	6%
Intel (Altera)	$1,389	36%	$1,486	36%	7%
Microsemi	$301	8%	$297	7%	-1%
Lattice	$124	3%	$144	3%	16%
QuickLogic	$19	0%	$11	0%	-40%
Others	$2	0%	$2	0%	0%
TOTAL	$3,879	100%	$4,112	100%	6%

* Numbers are 10^6 US$

- Two Main Manufacturers:
  - indicator of a mature competitive Market
  - Altera was been recently bought by Intel…

Development Code Model for PLD/FPGA



Register-Transfer Level (**RTL**)
An technology independent code model for synchronous state transition or combinatorial logic mod

Test & Simulation takes about the same time as Design

Hardware Description Languages: CPLD / FPGA

- Schematics
- Verilog
- VHDL

"Basic Level" (equivalent to 'C' or 'Assembly')

- System C
- System Verilog

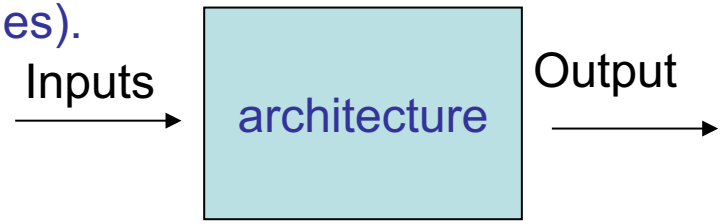
"High- Level" (equivalent to 'C++', Python)

VHDL Coding: Introduction

VHDL Allows to describe:
- Logic behavior and data flow
- Test procedures

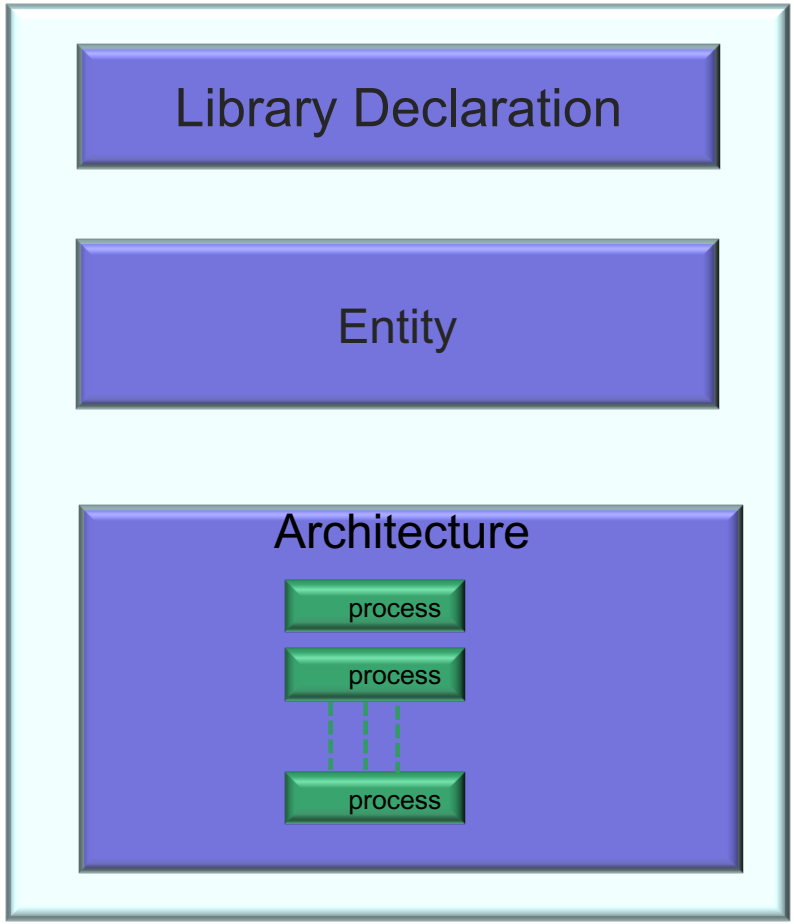
Each VHDL description hardware consist in a "entity/architecture" pair

- The "entity" is the interface, used to declare I/O pins (for the chip or for other modules).



- The "architecture" describes the modules behavior in one or more "processes".

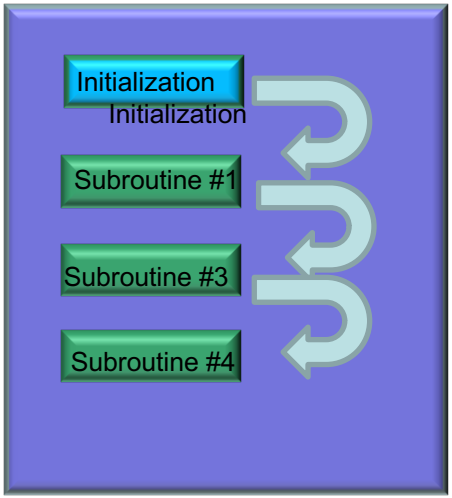
- The "process" is a block which encases a set of logic. Each process can be driven by zero or one only clock



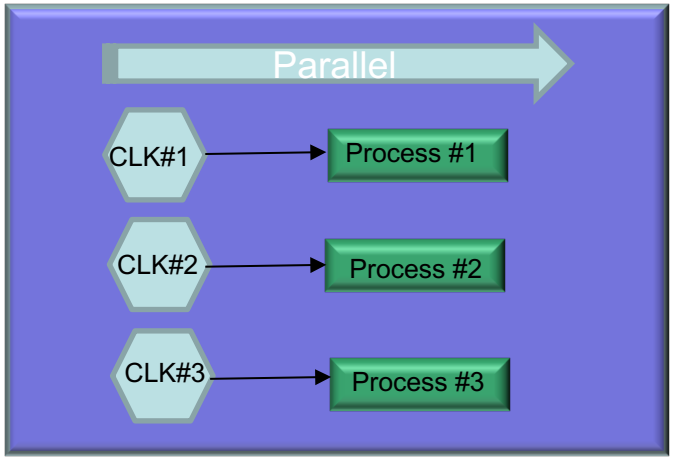
As in "C", standard libraries used are declared before the "entity".

VHDL: Process Execution

- CPU
  - Sequential Execution



- FPGA
  - Truly Parallel Execution



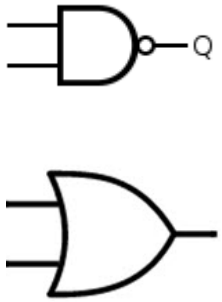
VHDL Combinatorial Code

```
-- (this a VHDL comment)

-- import std_logic from the IEEE library
library IEEE;
use IEEE.std_logic_1164.all;

--
entity ANDGATE is
   port (
        IN1 : in std_logic;
        IN2 : in std_logic;
        OUT1: out std_logic;
        OUT2: out std_logic);
end ANDGATE;

architecture RTL of ANDGATE is
begin
  OUT1 <= IN1 and IN2;
  OUT2 <= IN1 or IN2;
end RTL;
```



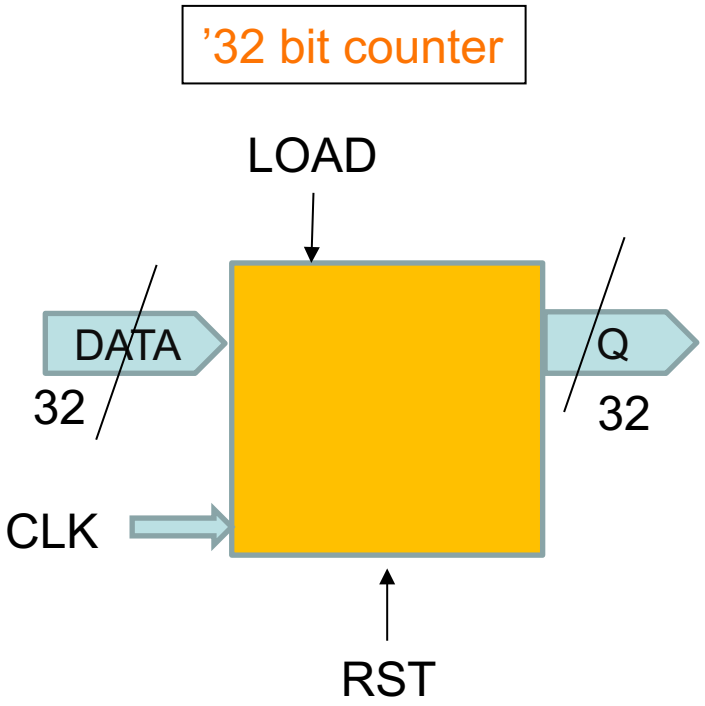
A	B	Q
0	0	1
0	1	1
1	0	1
1	1	1

Asynchronous instructions (always valid)

VHDL Code: Synchronous Example

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;    -- for the unsigned type
entity counter_example is
-- this is a parameter (can be overridden when 'instaciated'
generic (WIDTH : integer := 32);
port (
  CLK, RESET, LOAD : in std_logic;
  DATA : in  unsigned(WIDTH-1 downto 0); -- vector
  Q    : out unsigned(WIDTH-1 downto 0));
end entity counter_example;

architecture RTL of counter_example is
signal cnt : unsigned(WIDTH-1 downto 0); -- internal signal
begin – you can name a process:
 PROC_NAME: process (RESET, CLK, LOAD)
  begin
    if RESET = '1' then
      cnt <= (others => '0');
    elsif rising_edge(CLK) then
      if LOAD = '1' then
        cnt <= DATA;
      else
        cnt <= cnt + 1;
      end if;
    end if;
  end process PROC_NAME;
      Q <= cnt;
 end architecture RTL;
```

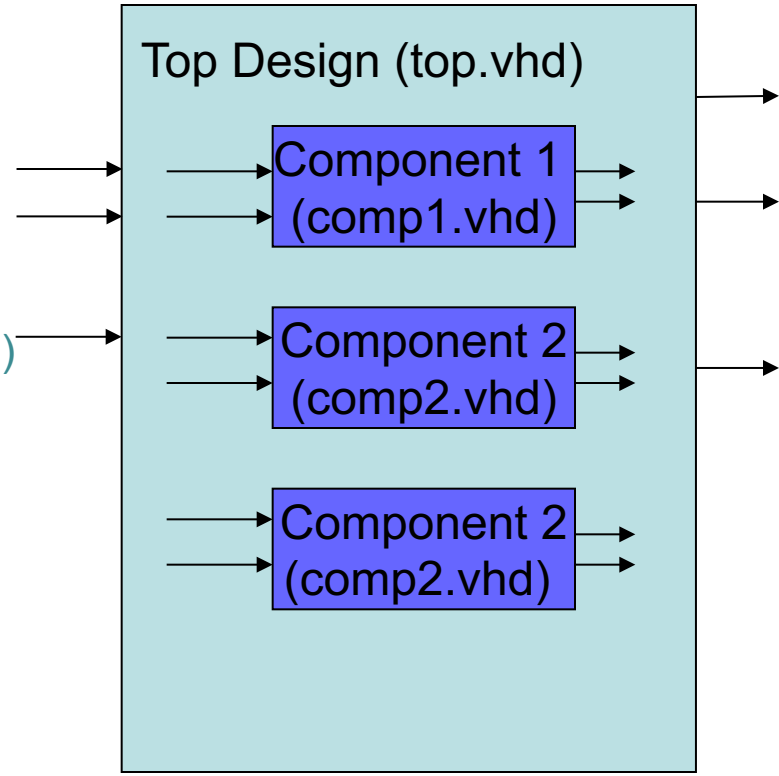


Second 'One-Line' Process (Asynchronous)

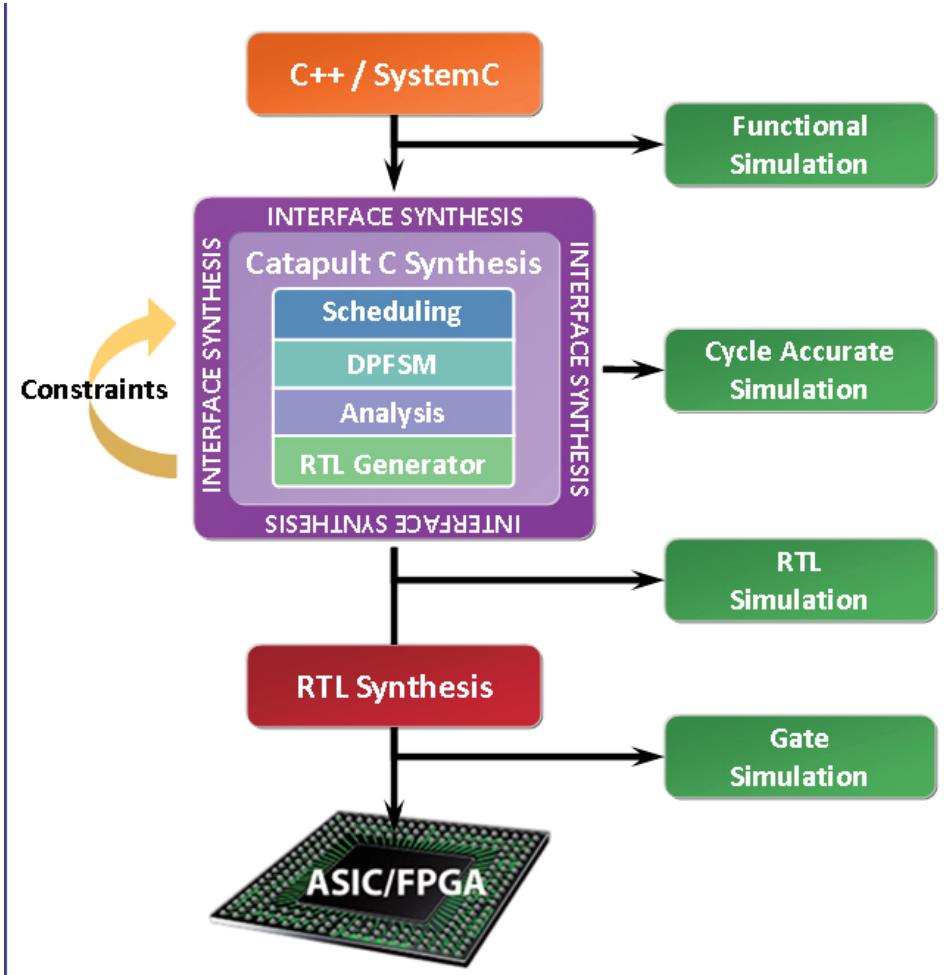
Hierarchical design

VHDL allows a hierarchy of entities containing components.

```
--Hierarchical design
entity Top is
Port(….);
end Top
architecture RTL of Top is
  signal X,Y,S,C : bit;
  component HALF_ADD --(defined in HALF_ADD.vhdl)
    port(A, B : in bit;
         SUM, CARRY : out bit);
  end component;
begin
  HA1: HALF_ADD port map (X,Y,S,C); --(instance)
 -- other statements
end RTL;
```



Other programming models: "SystemC"



- The idea is to compile C++ code for FPGAs.
- Only involves a subset of C/C++.
- Permits the *quasi* direct transposition of C++ that runs is a CPU to a FPGA!

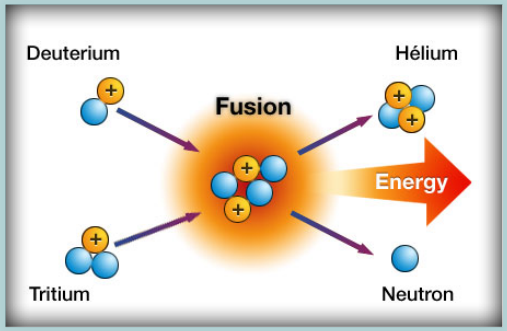
Example: Mentor Catapult C

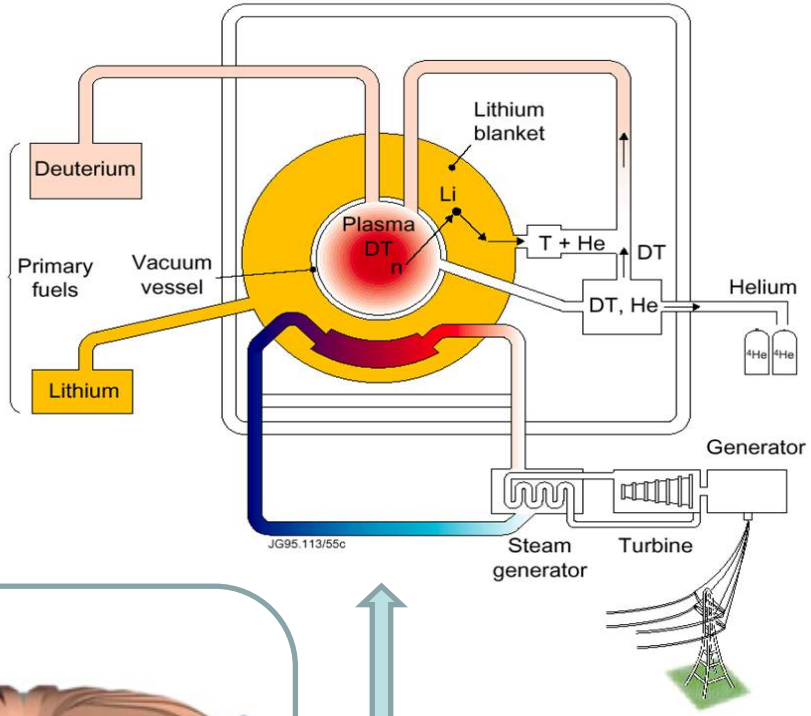
IPFN FPGA developments for Fusion

- JET VS Controller Hardware
  - ADC data decimation to 20kHz and Filtering (FIR)
- JET Gamma and Neutron Camera Real-Time Spectroscopy
- W7-X Stellarator Magnetic Diagnostics
  - Magnetics Integration of Signals
- W7-X Diamagnetic Interlock System
  - Calculates Energy from the plasma in 1 us

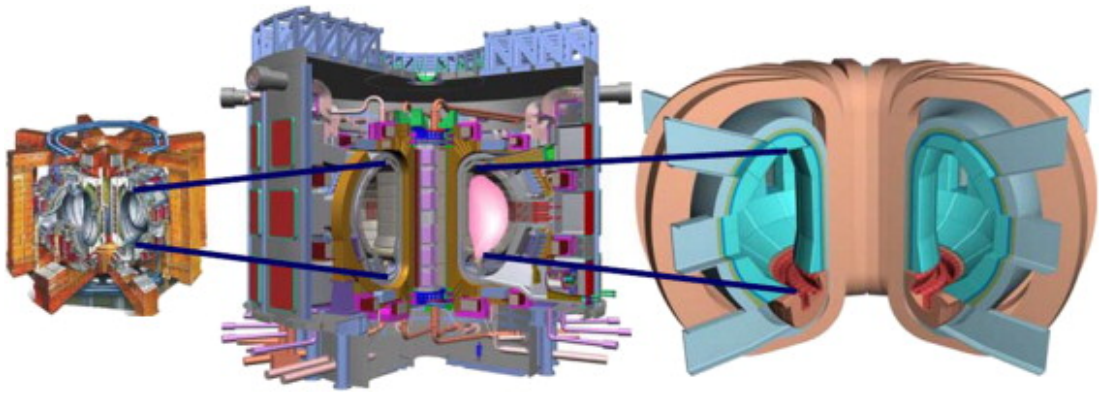
ATCA Nuclear Fusion

Aiming for self-sustainable Nuclear Fusion Power on Earth …



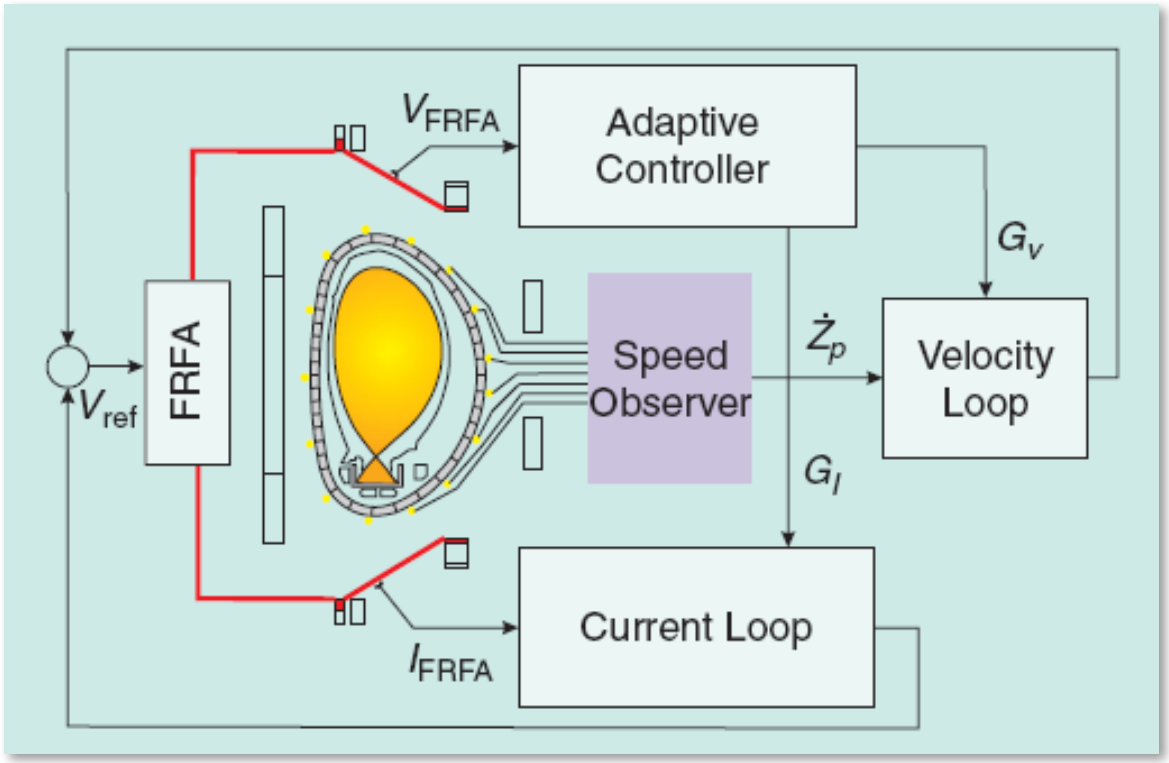


TOKAMAK



JET | 1983 | 16MW ITER | 2025 | 500MW DEMO | 2040 | 2GW

JET TOKAMAK Vertical Stabilization



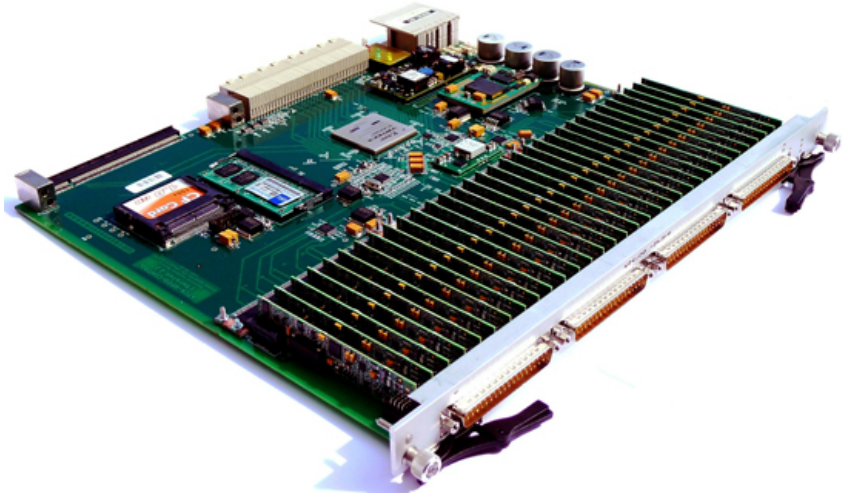
From F. Sartori, IEEE CONTROL SYSTEMS MAGAZINE, APRIL 2006

Elongated plasmas are vertically unstable

MIMO systems designed to make plasma vertically stable

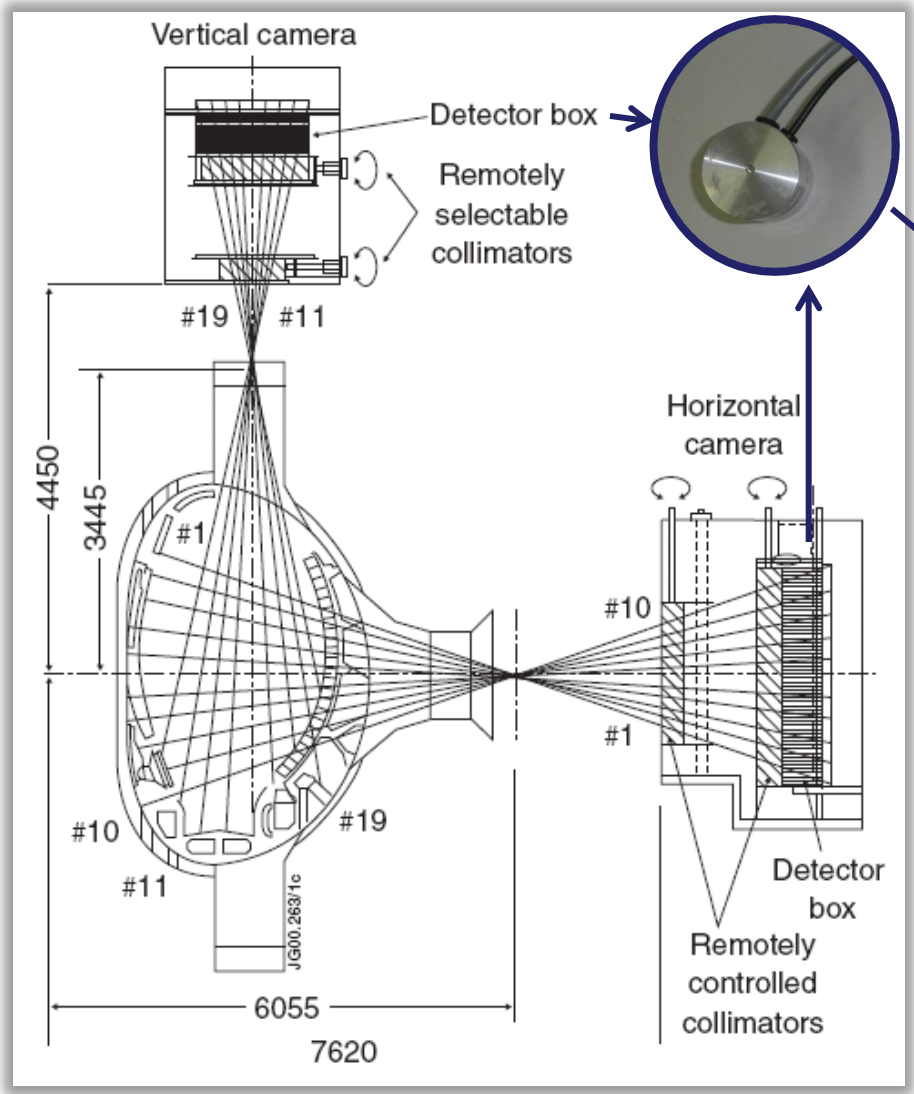
IPFN ATCA Hardware for CoDAC

- ATCA MIMO-ISOL board
  - 32 ADC channels
    - Isolated inputs
    - 18 bit resolution
  - 8 channel DAC
  - Xilinx Virtex 4 FPGA
  - PCIe channel 4x Interface
  - 512 MBytes of RAM
  - MARTe Software Drivers



http://atca.ipfn.ist.utl.pt/

JET Gamma Ray Diagnostic



19 LOS for neutron / gamma / hard X-ray signals
(**3 ≠ types of detectors**)

Gamma / hard X-ray detectors:

- **19 x CsI(Tl)** scintillator + PIN Photodiode + Pre-amplifier
- Energy Range: **0.2-6 MeV**

Detectors limitations

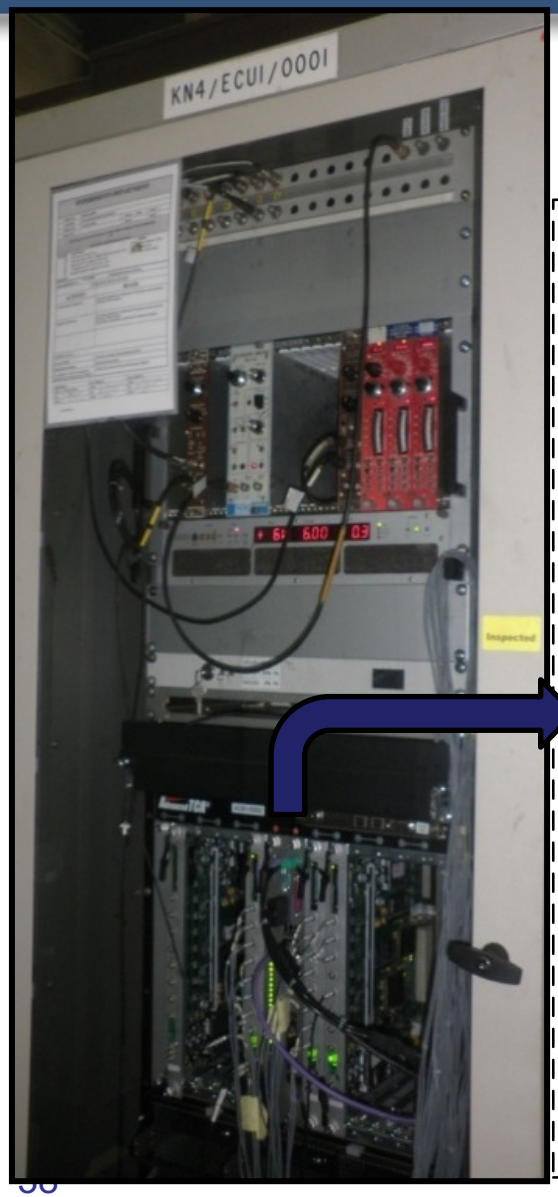
- **Poor SNR** (electronics, cabling)
- **Long pulse** decay

Former DAQ

- Shaping Amplifier + MCA

Former DAQ limitations:

JET ATCA Gamma Ray Real Time System



Diagnostic DAQ Cubicle

- 14-slot ATCA **Shelf**
- ATCA **Controller**
- **ATM** module
- 3 **DAQ** modules

KN4/ECU1/0001

Backplane

Digitizer — Virtual slot 6 — clk, trg

Digitizer — Virtual slot 4 — clk, trg

Controler — **MARTe** — ATM — 1000 Base-T

Digitizer — Virtual slot 3 — clk, trg

ATCA DAQ system

YN3/ECU1/0003

RTMC — ATM switch

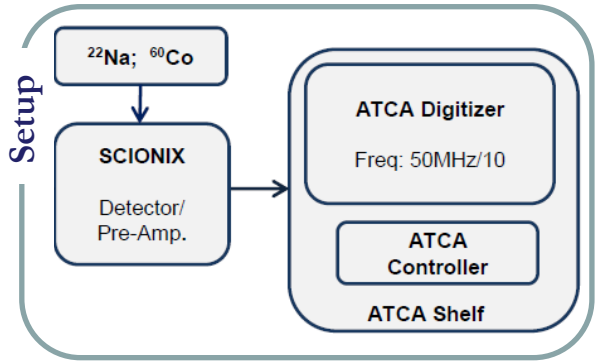
CODAS — Level-1, GAP

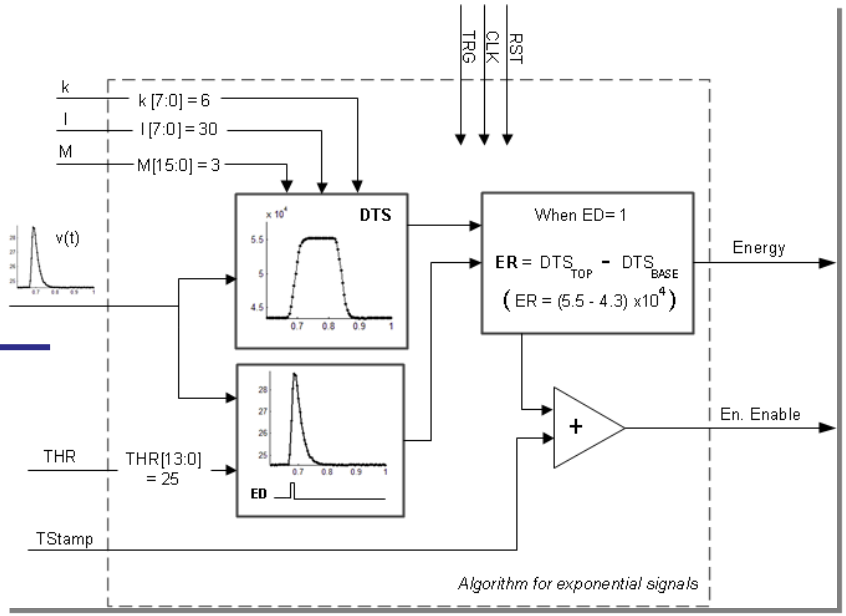
CTTS

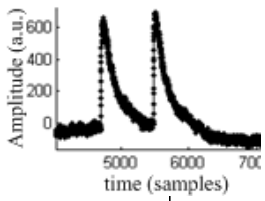
CODAS cubicle in J1D

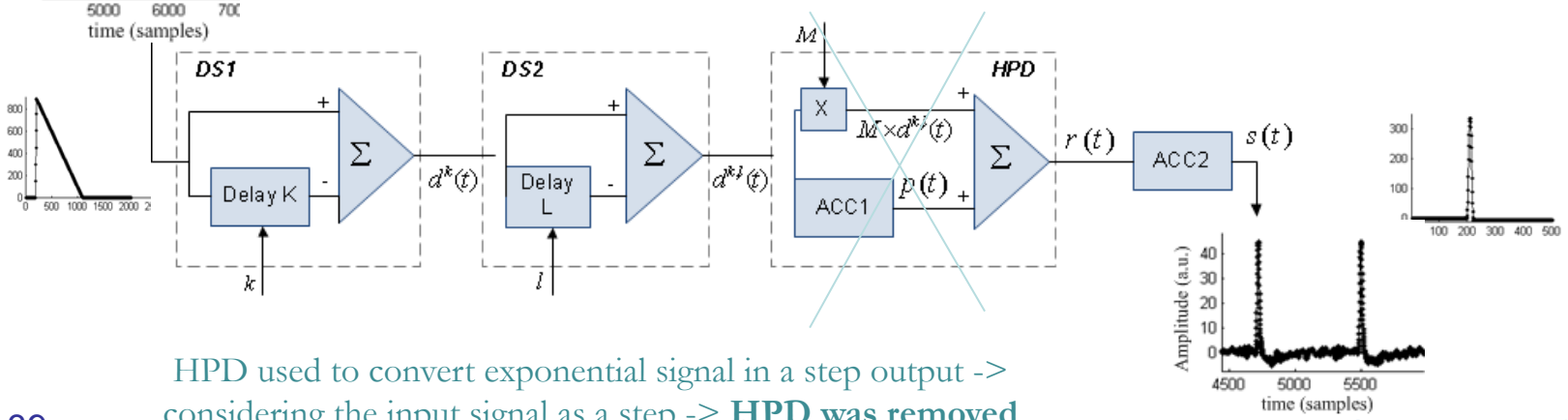
Real Time Algorithms @ FPGA

Algorithm for exponential / ramp-like pulses





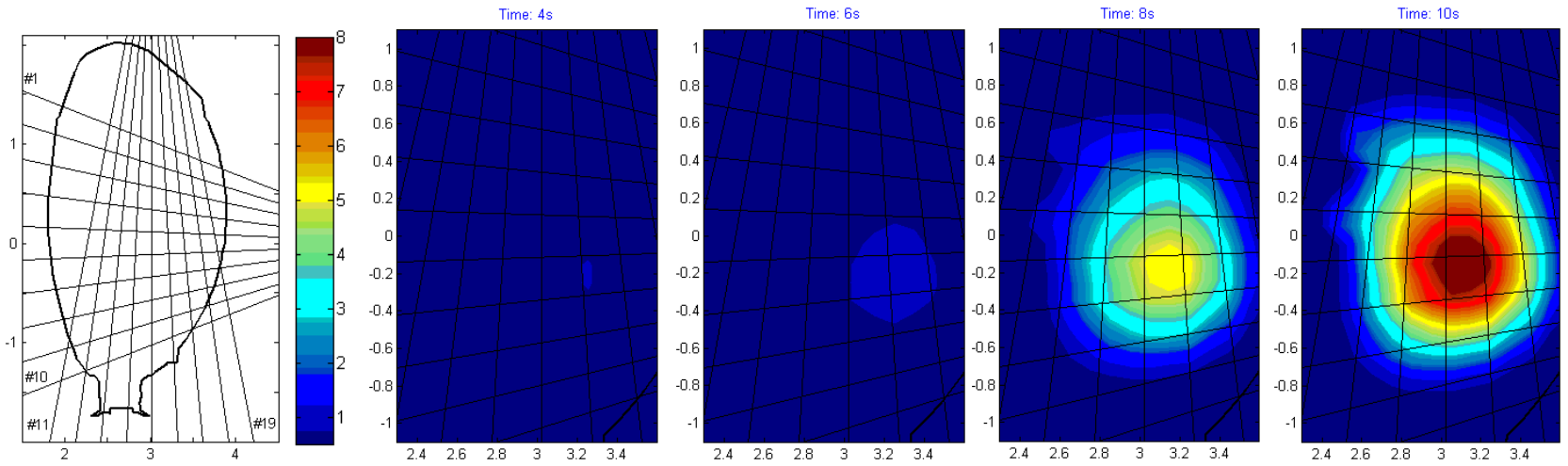




HPD used to convert exponential signal in a step output -> considering the input signal as a step -> **HPD was removed**

Tomographic reconstructions

shot #83786



Method:

- **Adapted-MFR**

Data:

- **Counts/time**

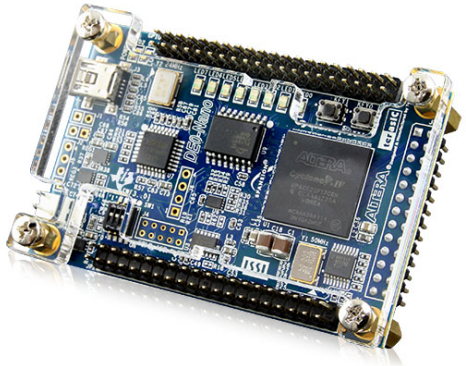
(real-time streamed data)

NBI:

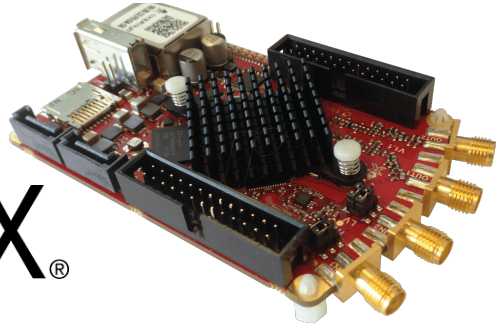
- $8 - 14s$ $(11.4MW)$
- $14 - 16s$ $(2MW)$

FPGA: Have Fun, Learn Yourself!

MachXO3 Starter Kit
~ €22



DE0-Nano Development
and Education Board
~ €80



STEM lab 125-10
Starter Kit
SoC ZINQ FPGA
~ €159

Mille Grazie!!

Prof. Gianmaria De Tommasi

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Funded by the Erasmus+
Programme of the European Union

FPGA Manufacturers

SRAM based FPGAs

- Xilinx Inc. – www.xilinx.com
- Altera Corp. – www.altera.com
- Atmel Corp. – www.atmel.com
- Lattice Semiconductor Corp. – www.latticesemi.com

” Antifuse/Flash based FPGAs

- Actel Corp. – www.actel.com
- QuickLogic Corp. – www.quicklogic.com