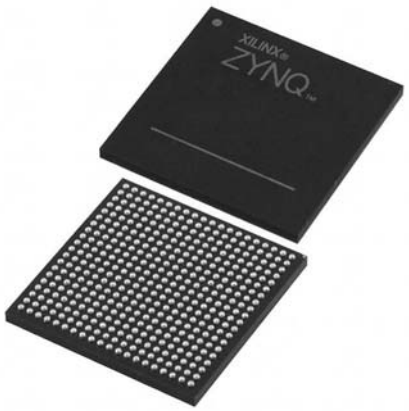


Introducción a la Tecnología Xilinx a través de la Práctica 1

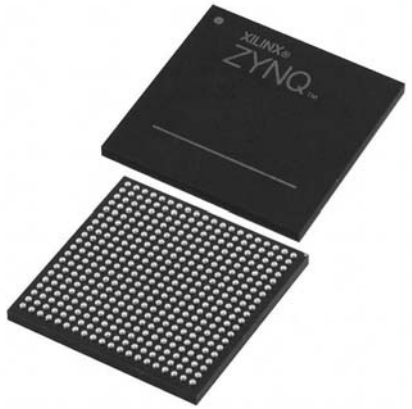
Parte 1: Introducción

Eduardo Boemo
Universidad Autónoma de Madrid

eduardo.boemo@uam.es

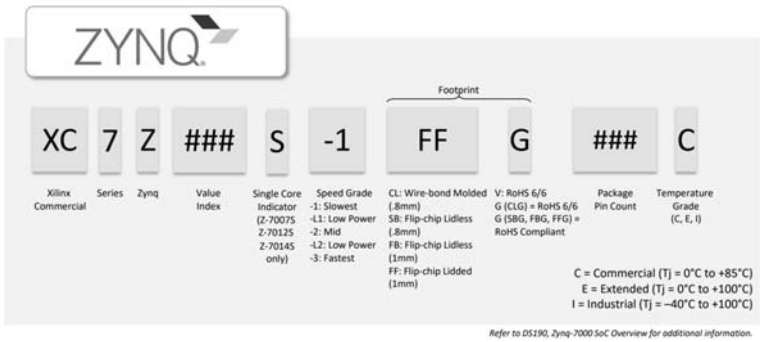
¿Cuál de todos es el chip del Lab 2021?

- XC7Z010CLG400-1C
  - XC = Xilinx Commercial
  - 7Z = Familia Zynq
  - 010 =“Tamaño” (cantidad de diversos bloques internos). También se usa:
    - “capacidad” (no la de los pF)
    - “puertas equivalente”
  - CLG400 = encapsulado y número de “patas”
  - -1 = “Velocidad”
  - C = “Comercial”



Nomenclatura Xilinx

Zynq®-7000 Family Device Ordering Information



Zynq®-7000 SoC Family

		Cost-Optimized Devices						Mid-Range Devices			
	Device Name	Z-7007S	Z-7012S	Z-7014S	Z-7010	Z-7015	Z-7020	Z-7030	Z-7035	Z-7045	Z-7100
	Part Number	XC7Z007S	XC7Z012S	XC7Z014S	XC7Z010	XC7Z015	XC7Z020	XC7Z030	XC7Z035	XC7Z045	XC7Z100
Processing System (PS)	Processor Core	Single-Core ARM® Cortex™-A9 MPCore™ Up to 766MHz			Dual-Core ARM Cortex-A9 MPCore Up to 866MHz			Dual-Core ARM Cortex-A9 MPCore Up to 1GHz[1]			
	Processor Extensions	NEON™ SIMD Engine and Single/Double Precision Floating Point Unit per processor									
	L1 Cache	32KB Instruction, 32KB Data per processor									
	L2 Cache	512KB									
	On-Chip Memory	256KB									
	External Memory Support[2]	DDR3, DDR3L, DDR2, LPDDR2									
	External Static Memory Support[2]	2x Quad-SPI, NAND, NOR									
	DMA Channels	8 (4 dedicated to PL)									
	Peripherals	2x UART, 2x CAN 2.0B, 2x I2C, 2x SPI, 4x 32b GPIO									
	Peripherals w/ built-in DMA[2]	2x USB 2.0 (OTG), 2x Tri-mode Gigabit Ethernet, 2x SD/SDIO									
	Security[3]	RSA Authentication of First Stage Boot Loader, AES and SHA 256b Decryption and Authentication for Secure Boot									
	Processing System to Programmable Logic Interface Ports (Primary Interfaces & Interrupts Only)	2x AXI 32b Master, 2x AXI 32b Slave 4x AXI 64b/32b Memory AXI 64b ACP 16 Interrupts									
Programmable Logic (PL)	7 Series PL Equivalent	Artix®-7	Artix-7	Artix-7	Artix-7	Artix-7	Artix-7	Kintex®-7	Kintex-7	Kintex-7	Kintex-7
	Logic Cells	23K	55K	65K	28K	74K	85K	125K	275K	350K	444K
	Look-Up Tables (LUTs)	14,400	34,400	40,600	17,600	46,200	53,200	78,600	171,900	218,600	277,400
	Flip-Flops	28,800	68,800	81,200	35,200	92,400	106,400	157,200	343,800	437,200	554,800
	Total Block RAM (# 36Kb Blocks)	1.8Mb (50)	2.5Mb (72)	3.8Mb (107)	2.1Mb (60)	3.3Mb (95)	4.9Mb (140)	9.3Mb (265)	17.6Mb (500)	19.2Mb (545)	26.5Mb (755)
	DSP Slices	66	120	170	80	160	220	400	900	900	2,020
	PCI Express®	—	Gen2 x4	—	—	Gen2 x4	—	Gen2 x4	Gen2 x8	Gen2 x8	Gen2 x8
	Analog Mixed Signal (AMS) / XADC[2]	2x 12 bit, MSPS ADCs with up to 17 Differential Inputs									
	Security[3]	AES & SHA 256b Decryption & Authentication for Secure Programmable Logic Config									
	Speed Grades: Commercial	-1			-1			-1			-1
	Speed Grades: Extended	-2			-2,-3			-2,-3			-2
	Speed Grades: Industrial	-1, -2			-1, -2, -1L			-1, -2, -2L			-1, -2, -2L

Product Attributes

XC7Z010-1CLG400C (Lab UAM)

TYPE	DESCRIPTION
Category	Integrated Circuits (ICs) Embedded - System On Chip (SoC)
Mfr	Xilinx Inc.
Series	Zynq®-7000
Package	Tray
Part Status	Active
Architecture	MCU, FPGA
Core Processor	Dual ARM® Cortex®-A9 MPCore™ with CoreSight™
Flash Size	-
RAM Size	256KB
Peripherals	DMA
Connectivity	CANbus, EBI/EMI, Ethernet, I²C, MMC/SD/SDIO, SPI, UART/USART, USB OTG
Speed	667MHz
Primary Attributes	Artix™-7 FPGA, 28K Logic Cells
Operating Temperature	0°C ~ 85°C (TJ)
Package / Case	400-LFBGA, CSPBGA
Supplier Device Package	400-CSPBGA (17x17)
Number of I/O	130
Base Product Number	XC7Z010

¿2 ARMs? ¿No era una FPGA?

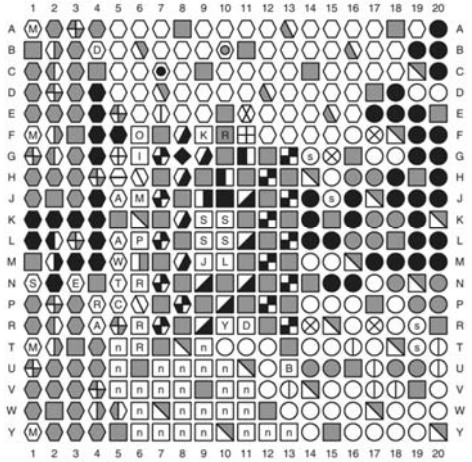
¿Poca memoria?

¿Reloj? ¿Sistema?

¿Qué es Tj? ¿Funcionará el día que en Madrid lleguemos a 85° en Julio? ¿Carnot?

¿400 patas y sólo 130 I/O? ¿Y el resto?

Encapsulado CLG400



User I/O Pins	Dedicated Pins	Other Pins	PS MIO Pins	PS DDR Pins
IO_LXXY_#	CFGBVS_0	GND	PS_POR_B	PS_DDR_CKP
IO_XX_#	DONE_0	VCCAUX_IO_G#	PS_CLK	PS_DDR_CKN
Multi–Function Pins	DXP_0	VCCAUX	PS_SRST_B	PS_DDR_CKE
PUDC_B	DXN_0	VCCINT	PS_MIO_VREF	PS_DDR_CS_B
ADSP/AD0N–AD15P/AD15N	GNDADC_0	VCCO_#	PS_MIO	PS_DDR_RAS_B
VRN	INIT_B_0	VCCBRAM		PS_DDR_CAS_B
VRP	PROGRAM_B_0	VCCPINT		PS_DDR_WE_B
VREF	TCK_0	VCCPAUX		PS_DDR_A
DQS	TDI_0	VCCO_MIO0		PS_DDR_BA
MRCC	TDO_0	VCCO_MIO1		PS_DDR_A
SRCC	TMS_0	VCCO_DDR		PS_DDR_ODT
	VCCADC_0	VCCPLL		PS_DDR_DRST_B
	VCCBATT_0	RSVDVCC[3:1]		PS_DDR_DQ
	VP_0	RSVDGND		PS_DDR_DM
	VN_0	NC		PS_DDR_DQS_P
	VREFP_0			PS_DDR_DQS_N
	VREFN_0			PS_DDR_VRP
				PS_DDR_VRN
				PS_DDR_VREF

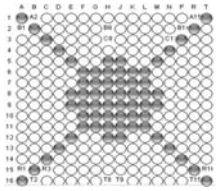
Migración para aumentar lógica manteniendo I/O (nº patas constante)

Zynq®-7000 Family Device Migration Table

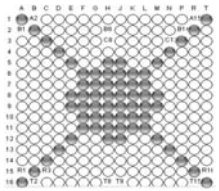
Pkg	mm	Zynq®-7000 Family									
		Z-7007S	Z-7012S	Z-7014S	Z-7010	Z-7015	Z-7020	Z-7030	Z-7035	Z-7045	Z-7100
CLG225	13	■			■						
CLG400	17	■		■	■		■				
CLG484	19			■			■				
CLG485	19			■			■				
SBG485	19							■			
FBG484	23							■			
FBG676	27							■	■	■	
FFG676	27							■	■	■	
FFG900	31								■	■	■
FFG1156	35										■

- Ejemplo:
- Un diseño procesa 16 bits de entrada y saca 16 de salida.
- PCB fabricado y de pronto se requiere más procesamiento.
- ¿Qué hacer?

Migración para aumentar lógica manteniendo I/O (nº patas constante). Concepto de circuito pad / core limited



FT256 package



200 K System Gates

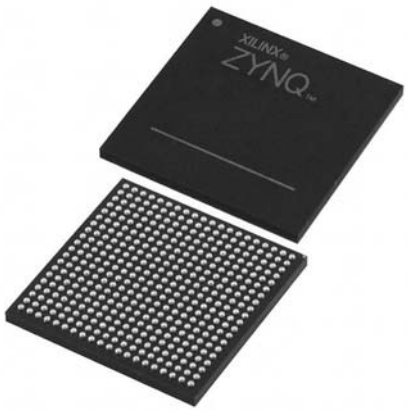
5X Density Range

1 Million System Gates

VCC & GND ○User I/O

- Los encapsulados son compatibles entre sub-rangos de FPGAs para facilitar el diseño de sistemas “core-limited”.

← Figura: ejemplo Spartan 3



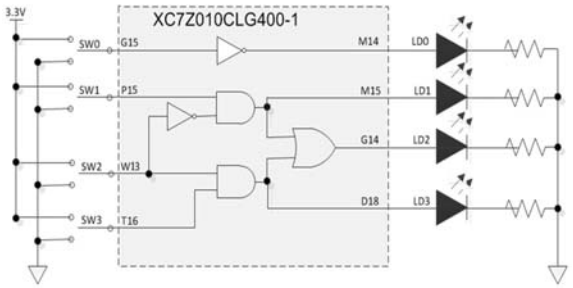
Introducción a la Tecnología Xilinx a través de la Práctica 1

Parte 2: La Práctica 1

Eduardo Boemo
Universidad Autónoma de Madrid

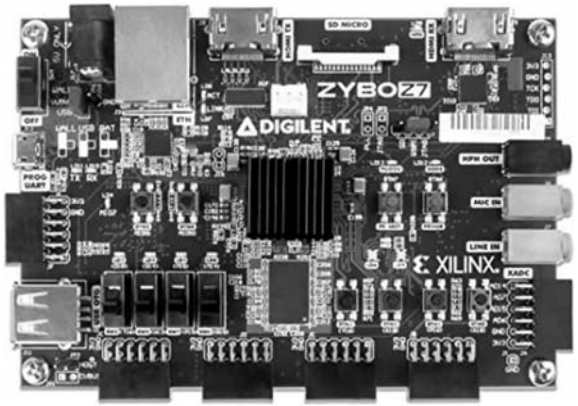
eduardo.boemo@uam.es

Primer diseño FPGA



- ¿Vcc?
- ¿SW?
- ¿G15, P15, M14…?
- ¿LD?

Tarjeta de desarrollo EPS 2021



Tarjeta de desarrollo:

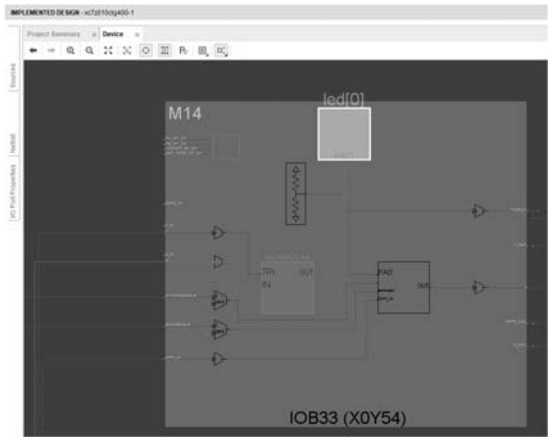
- Bajo coste.
- Prototipo para facilitar pruebas
- Diseños de referencia
- Incluye planos.

- Digilent: originariamente orientada al mercado universitario

XDC = **X**ilinx **D**esign **C**onstraints (file)

D:/ISE/2021_DIE_P1/2021_DIE_P1.srcs/constrs_1/imports/P1_EIB/lab1.xdc

```
# ZYBO Pin Assignments
############################
# On-board Slide Switches  #
############################
set_property PACKAGE_PIN G15      [get_ports swt[0]]
set_property IOSTANDARD  LVCMOS33 [get_ports swt[0]]
set_property PACKAGE_PIN P15      [get_ports swt[1]]
set_property IOSTANDARD  LVCMOS33 [get_ports swt[1]]
set_property PACKAGE_PIN W13      [get_ports swt[2]]
set_property IOSTANDARD  LVCMOS33 [get_ports swt[2]]
set_property PACKAGE_PIN T16      [get_ports swt[3]]
set_property IOSTANDARD  LVCMOS33 [get_ports swt[3]]

############################
# On-board led             #
############################
set_property PACKAGE_PIN M14      [get_ports led[0]]
set_property IOSTANDARD  LVCMOS33 [get_ports led[0]]
set_property PACKAGE_PIN M15      [get_ports led[1]]
set_property IOSTANDARD  LVCMOS33 [get_ports led[1]]
set_property PACKAGE_PIN G14      [get_ports led[2]]
set_property IOSTANDARD  LVCMOS33 [get_ports led[2]]
set_property PACKAGE_PIN D18      [get_ports led[3]]
set_property IOSTANDARD  LVCMOS33 [get_ports led[3]]
```



XDC = Xilinx Design Constraints (file)

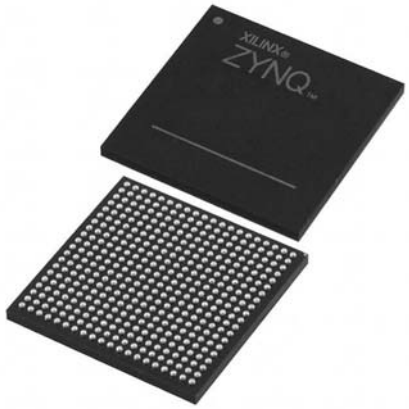

Vivado Design Suite Tutorial

Using Constraints

UG945 (v2019.2) December 6, 2019

This tutorial was validated with 2019.1. Minor procedural differences might be required when using later releases.

XILINX



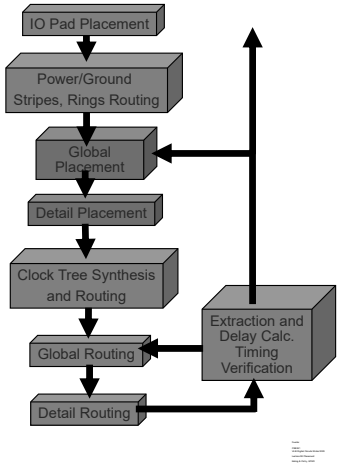
Introducción a la Tecnología Xilinx a través de la Práctica 1

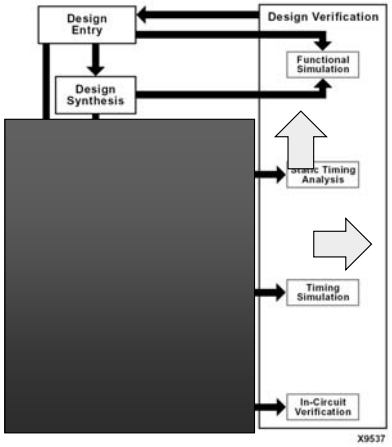
Parte 3: EDA Tools

Eduardo Boemo
Universidad Autónoma de Madrid

eduardo.boemo@uam.es

Masked-ASICs & FPGA Design Flows





FPGA ≠ Masked-ASIC

FPGAs = *platform chips.*

Es decir, chips donde están prediseñados muchos bloques tales como:

- VCC y GND (core y pines)
- Árbol de reloj.
- Memorias.
- Drivers.

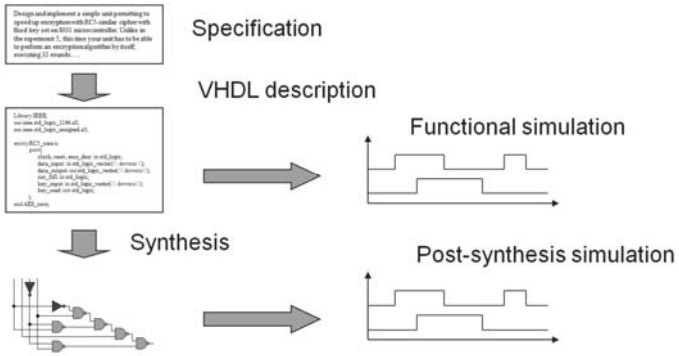
Xilinx Vivado

https://reference.digilentinc.com/vivado/getting_started/start

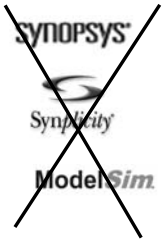
The Navigator is broken into seven sections:

- *Project Manager*
  - Allows for quick access to project settings, adding sources, language templates, and the IP catalog
- *IP Integrator*
  - Tools for creating Block Designs
- *Simulation*
  - Allows a developer to verify the output of their design prior to programming their device
- *RTL Analysis*
  - Lets the developer see how the tools are interpreting their code
- *Synthesis*
  - Gives access to Synthesis settings and post-synthesis reports
- *Implementation*
  - Gives access to Implementation settings and post-implementation reports
- *Program and Debug*
  - Access to settings for bitstream generation and the Hardware Manager

Síntesis: un texto VHDL (Verilog) → Primitivas HW (transistores, puertas, LUTs, etc.



- Xilinx incluye síntesis y simulación



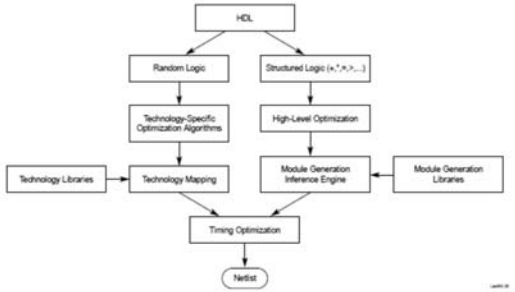
Pasos de la Síntesis

Fuente: https://forums.xilinx.com/t5/Synthesis/what-exactly-is-elaborating-a-design/td-p/682043

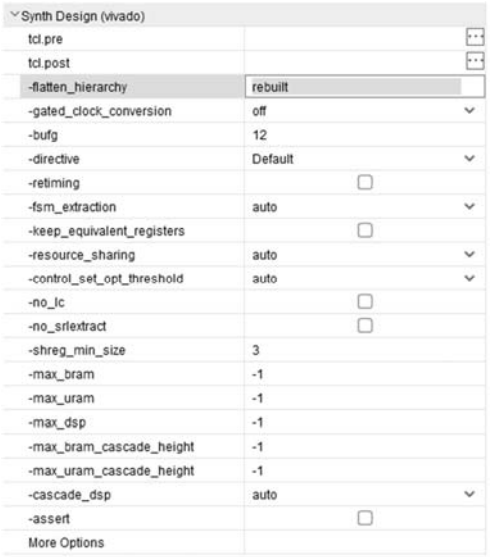
- ***Elaboration:*** *"Reading in your RTL file (which is text) and recognizing bits of code that represent real hardware structures. Once recognized, these are converted (in Vivado synthesis case) into "generic technology cells" - abstract things like registers, adders, comparators, multiplexers, arbitrarily wide gates, etc...*
- ***Apply constraints to the design:*** *"This step is necessary since the next steps (high and low level optimizations) are timing driven, and hence need constraints. But, constraints cannot be applied to your RTL (which is text) - they need to be applied to a netlist. So elaboration creates the netlist of generic technology cells."*
- **Do high level optimizations of the design.**

Pasos de la síntesis

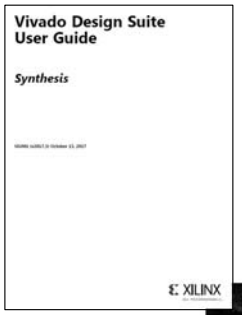
1. Load technology library into database
2. Analyze design
   Load HDL models into database, check for synthesizable models
3. Elaborate design
   Technology-independent circuit (random & structured logic)
4. Specify design constraints (timing, area)
5. Compile/optimize design
   Optimize for the loaded technology library
   Repeat as necessary to meet constraints
6. Generate technology-specific netlist(s)
7. Generate simulation timing data (SDF file)
8. Generate reports (cells, area, timing)



Source: http://www.eng.auburn.edu/~nelson/

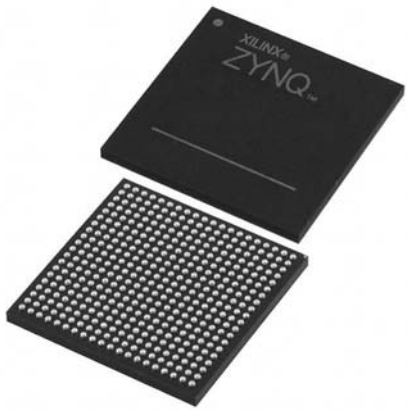


Opciones de síntesis



Ejemplo de flujo de diseño sobre una FPGA educativa (pizarra)

```
library ieee;
use ieee.std_logic_1164.all;

entity lab1 is
  port (
    swt : in  std_logic_vector (3 downto 0);
    led : out std_logic_vector (3 downto 0)
  );
end lab1;

architecture rtl of lab1 is
  signal ledSig : std_logic_vector (3 downto 0);
begin

  ledSig(0) <= not swt(0);
  ledSig(1) <= swt(1) and not swt(2);
  ledSig(2) <= (swt(1) and not swt(2)) or (swt(2) and swt(3));
  ledSig(3) <= swt(2) and swt(3);

  led <= ledSig;

end architecture;
```

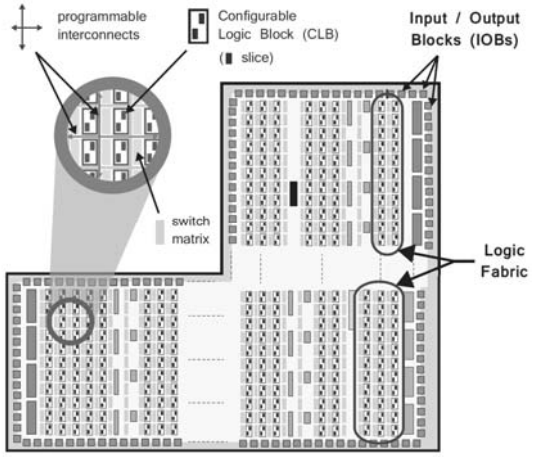
Introducción a la Tecnología Xilinx a través de la Práctica 1

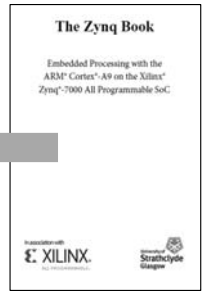
Parte 4: CLB e IOB Zynq

Eduardo Boemo
Universidad Autónoma de Madrid

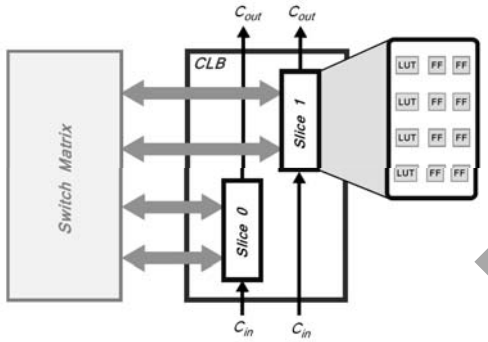
eduardo.boemo@uam.es

PL (program. logic = FPGA) Zynq: Componentes básicos

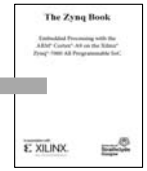




Zynq Configurable Logic Block (CLB)

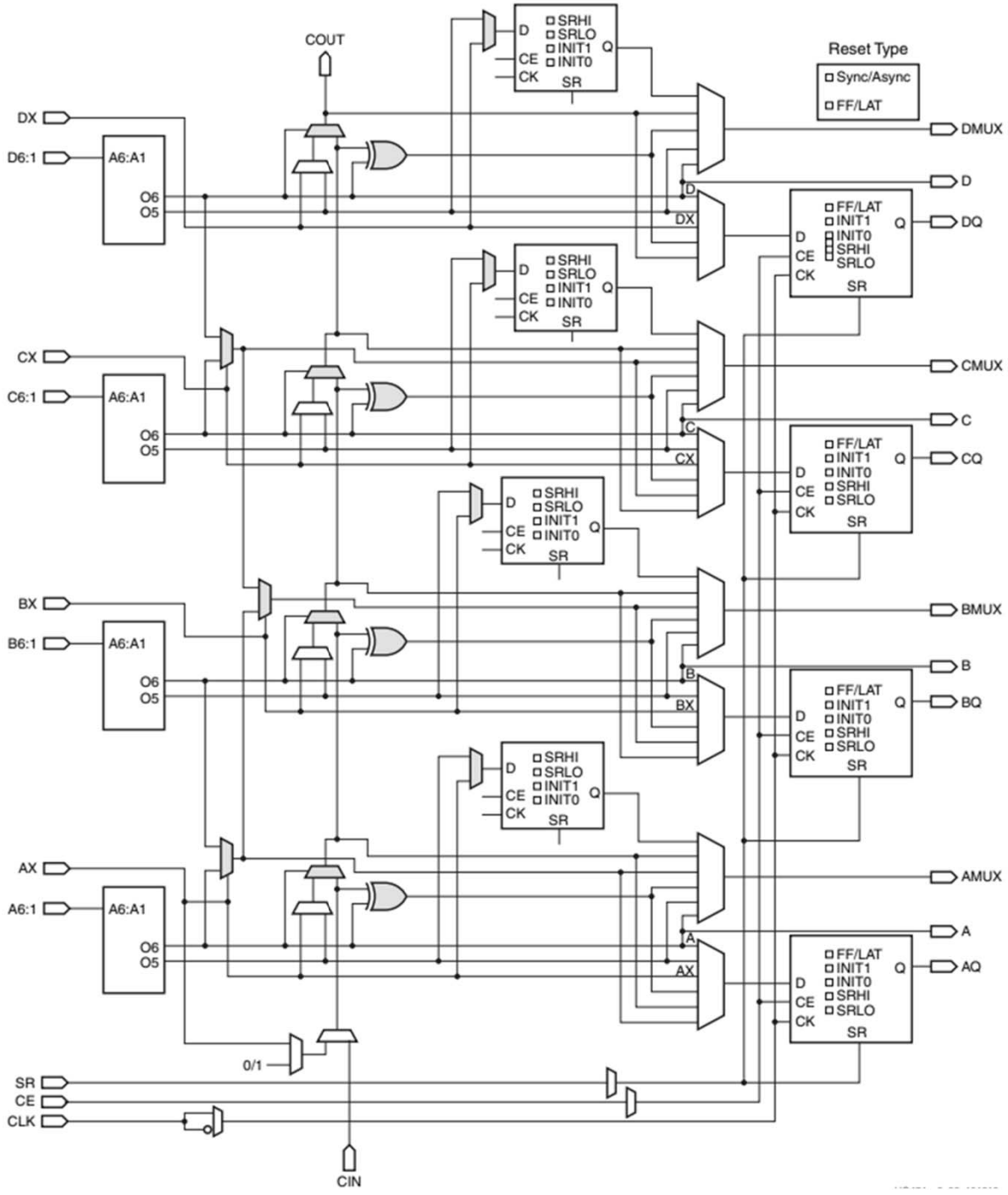


Source:



- Estructura repetitiva y jerárquica.
- Similar a Virtex 7 (y a V6)
- LUTs de 6 entradas (MUX de 64 entradas)

LUT → LE=LOGIC ELEMENT (LUT + FF) → *SLICE* (Conjunto de LEs) → CLB (Conjunto de *Slices*)


COUT
Reset Type
Sync/Async
FF/LAT
DX
D6:1
A6:A1
O6
O5
D
SRHI
SRLO
INIT1
INIT0
CE
CK
SR
Q
DMUX
D
DQ
FF/LAT
CX
C6:1
CMUX
C
CQ
BX
B6:1
BMUX
B
BQ
AX
A6:1
AMUX
A
AQ
0/1
SR
CE
CLK
CIN