Systemy wbudowane

Współczesne technologie implementacji mikrokontrolerów i systemów wbudowanych

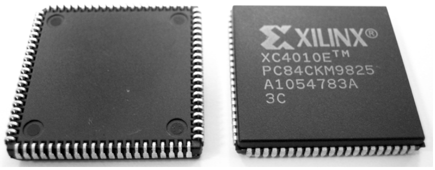
dr inż. Ignacy Pardyka, UJK Kielce

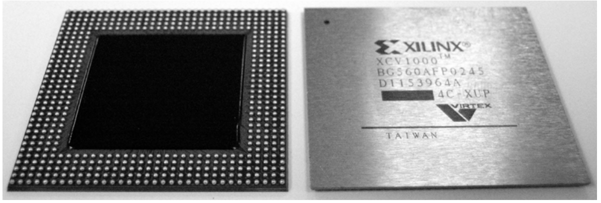
Materiały źródłowe:

1. Slides to accompany the textbook *Digital Design*, First Edition, by Frank Vahid, John Wiley and Sons Publishers, 2007, http://www.ddvahid.com
2. Slides to acompany the textbook *Embedded Systems: A Contemporary Design Tool*, by James K.Peckol, John Wiley and Sons Publishers, 2008.

Programmable IC Technology – FPGA

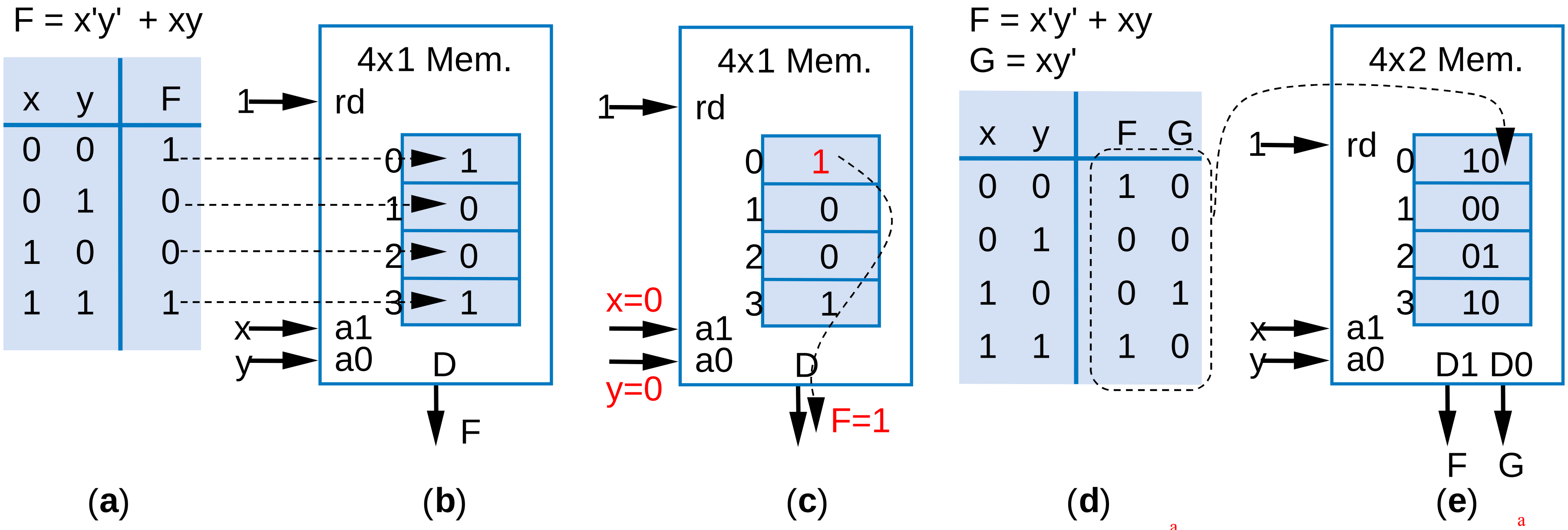
- Manufactured IC technologies require weeks to months to fabricate
  - And have large (hundred thousand to million dollar) initial costs
- Programmable ICs are pre-manufactured
  - Can implement circuit *today*
  - Just download bits into device
  - Slower/bigger/more-power than manufactured ICs
    - But get it today, and no fabrication costs
- Popular programmable IC – FPGA
  - "Field-programmable gate array"
    - Developed late 1980s
    - Though no "gate array" inside
      - Named when gate arrays were very popular in the 1980s
    - Programmable in seconds





FPGA Internals: Lookup Tables (LUTs)

- Basic idea: Memory can implement combinational logic
  - e.g., 2-address memory can implement 2-input logic
  - 1-bit wide memory – 1 function; 2-bits wide – 2 functions
- Such memory in FPGA known as Lookup Table (LUT)



F = x'y' + xy

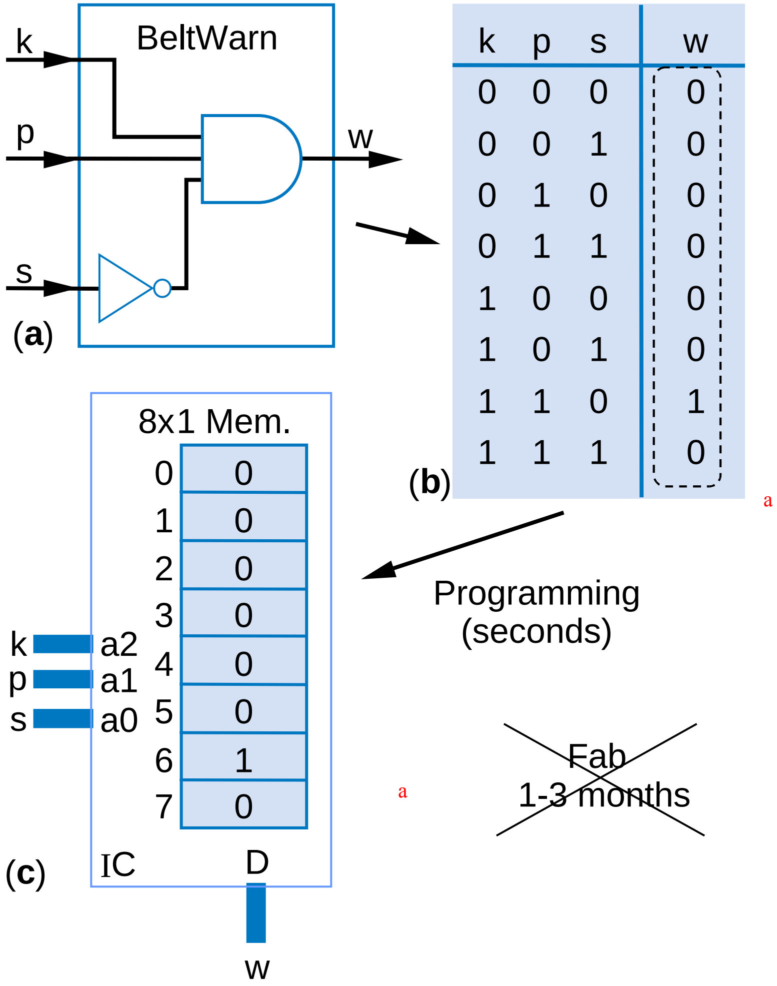
x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

F = x'y' + xy
G = xy'

x	y	F	G
0	0	1	0
0	1	0	0
1	0	0	1
1	1	1	0

FPGA Internals: Lookup Tables (LUTs)

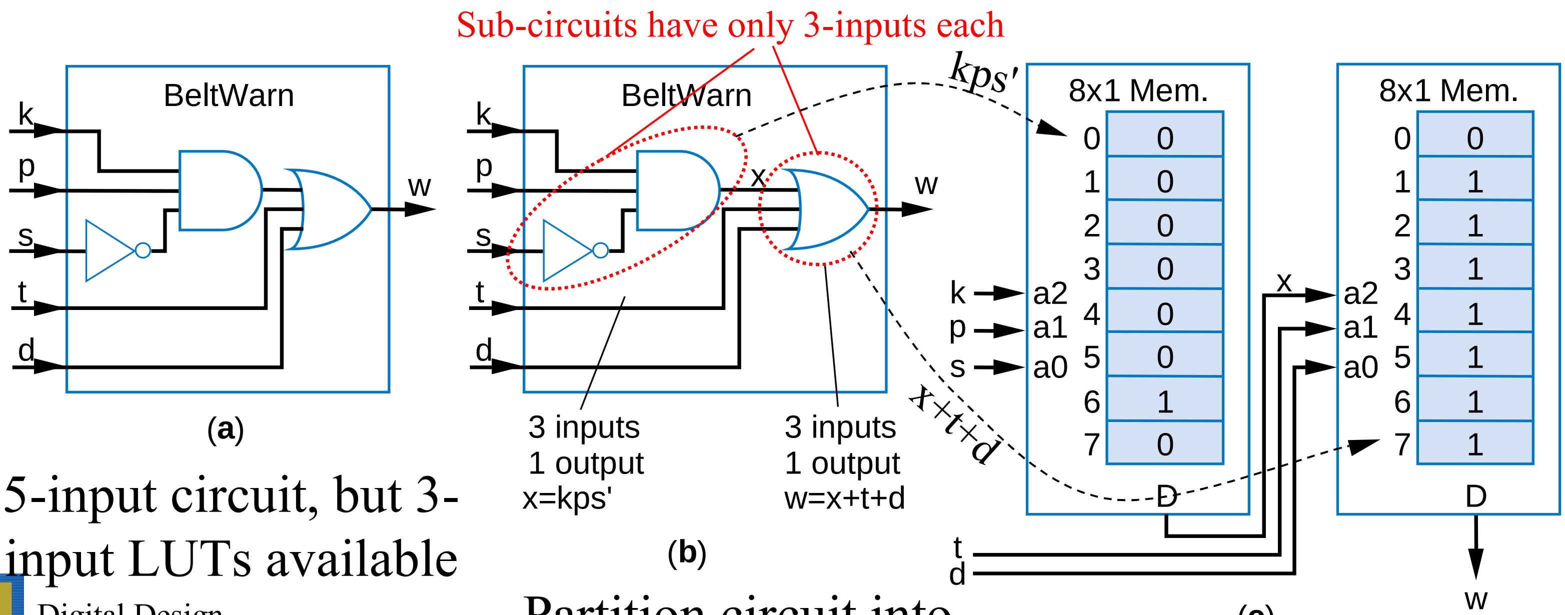
- Example: Seat-belt warning light (again)



k	p	s	w
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

FPGA Internals: Lookup Tables (LUTs)

- Lookup tables become inefficient for more inputs
  - 3 inputs → only 8 words
  - 8 inputs → 256 words; 16 inputs → 65,536 words!
- FPGAs thus have numerous small (3, 4, 5, or even 6-input) LUTs
  - If circuit has more inputs, must partition circuit among LUTs
  - Example: Extended seat-belt warning light system:



	8x1 Mem.
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	0

	8x1 Mem.
0	0
1	1
2	1
3	1
4	1
5	1
6	1
7	1

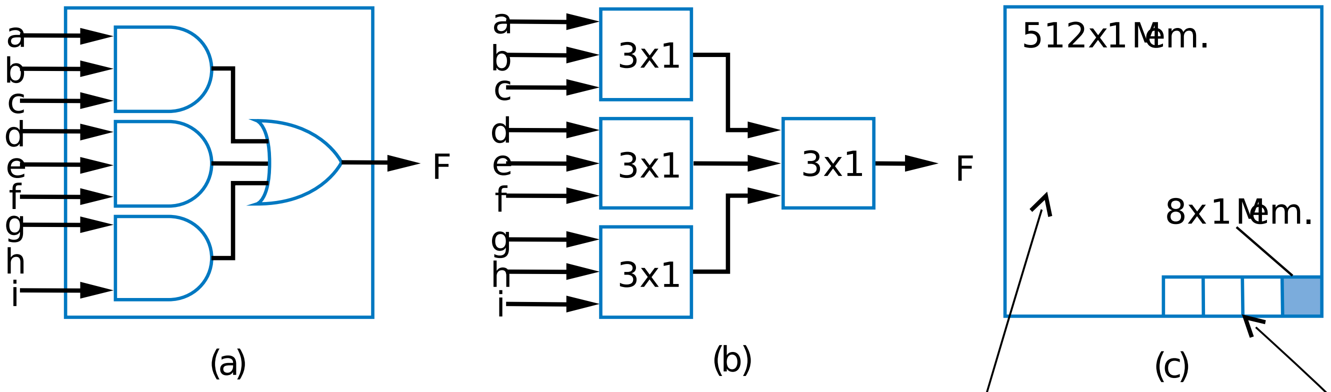
5-input circuit, but 3-input LUTs available

Partition circuit into 3-input sub-circuits

Map to 3-input LUTs

FPGA Internals: Lookup Tables (LUTs)

- Partitioning among smaller LUTs is more size efficient
  - Example: 9-input circuit



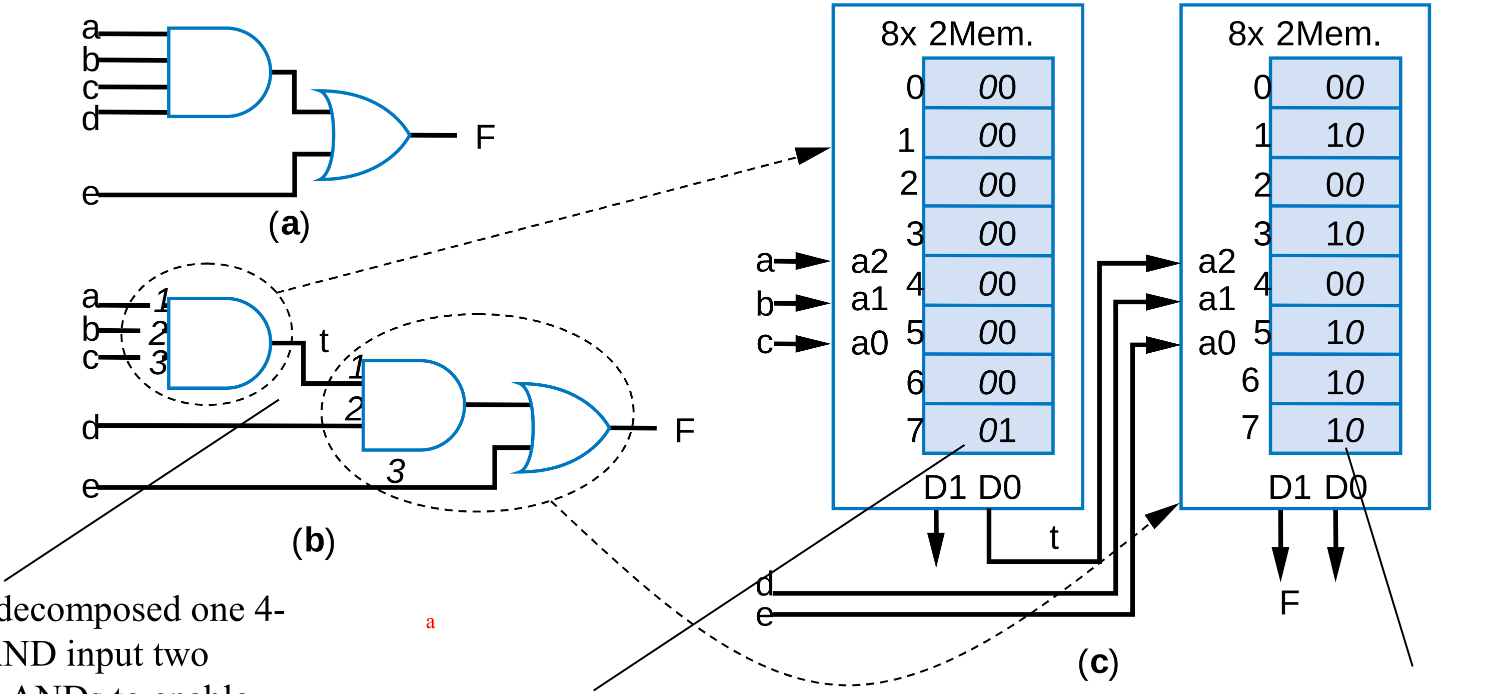
Original 9-input circuit

Partitioned among 3x1 LUTs

Requires only 4 3-input LUTs (8x1 memories) – much smaller than a 9-input LUT (512x1 memory)

FPGA Internals: Lookup Tables (LUTs)

- LUT typically has 2 (or more) outputs, not just one
- Example: Partitioning a circuit among 3-input *2-output* lookup tables



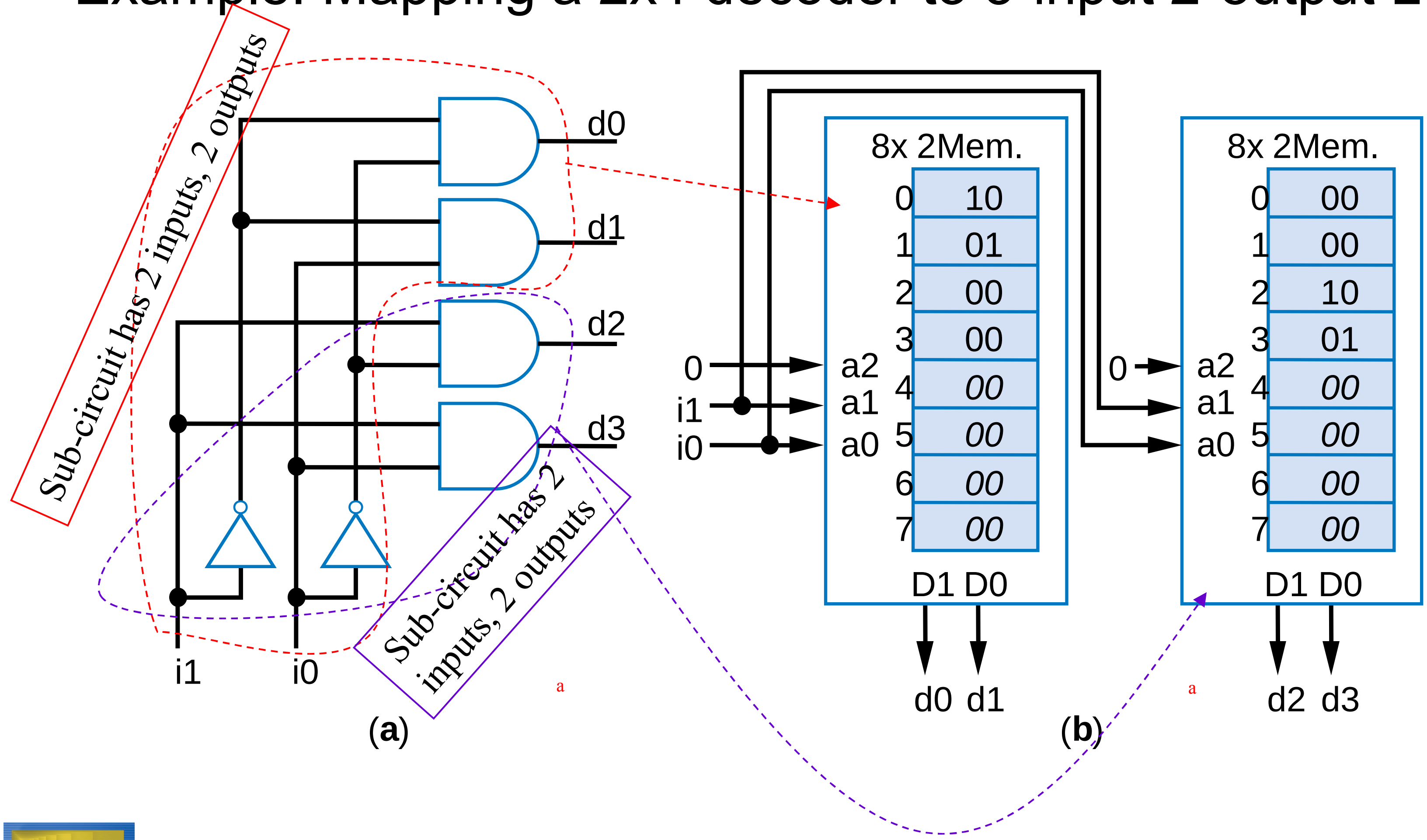
(Note: decomposed one 4-input AND input two smaller ANDs to enable partitioning into 3-input sub-circuits)

First column unused; second column implements AND

Second column unused; first column implements AND/OR sub-circuit

FPGA Internals: Lookup Tables (LUTs)

- Example: Mapping a 2x4 decoder to 3-input 2-output LUTs

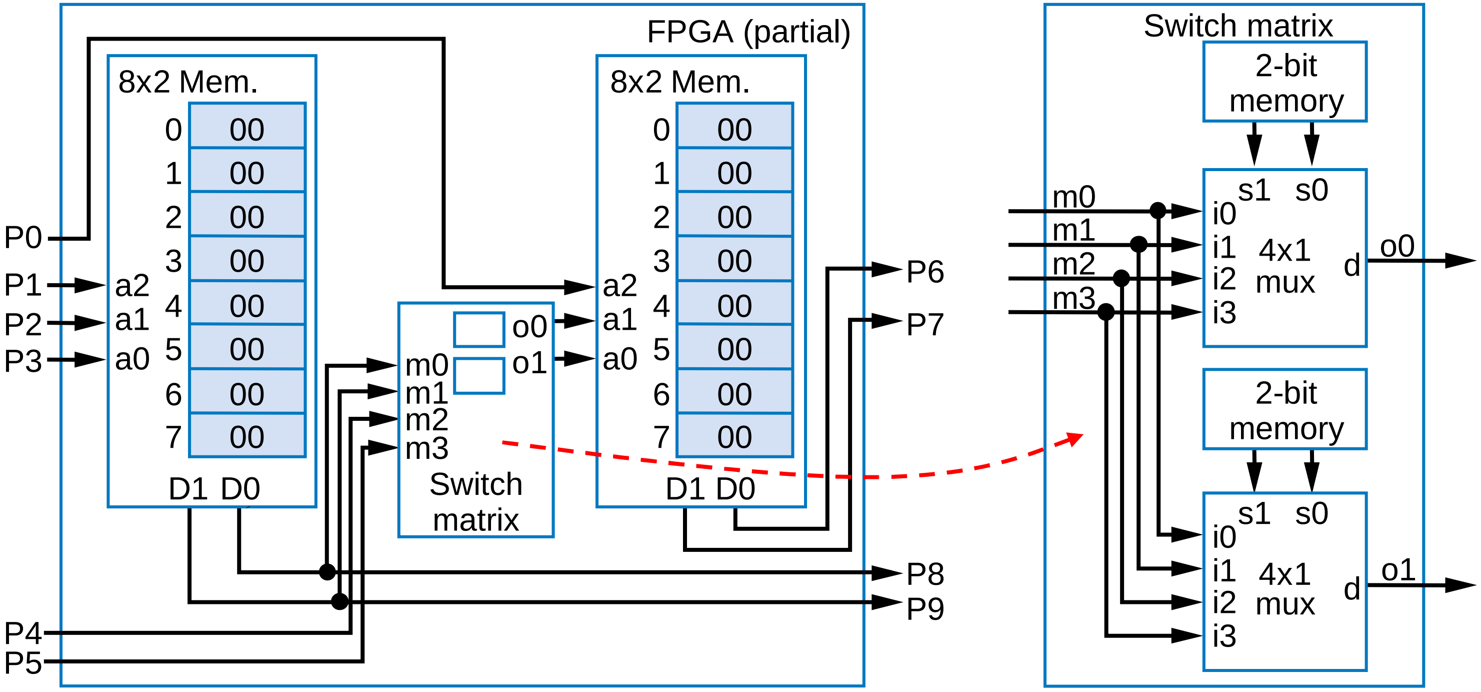


8x 2Mem.	D1 D0
0	10
1	01
2	00
3	00
4	*00*
5	*00*
6	*00*
7	*00*

8x 2Mem.	D1 D0
0	00
1	00
2	10
3	01
4	*00*
5	*00*
6	*00*
7	*00*

FPGA Internals: Switch Matrices

- Previous slides had hardwired connections between LUTs
- Instead, want to program the connections too
- Use switch matrices (also known as programmable interconnect)
  - Simple mux-based version – each output can be set to any of the four inputs just by programming its 2-bit configuration memory

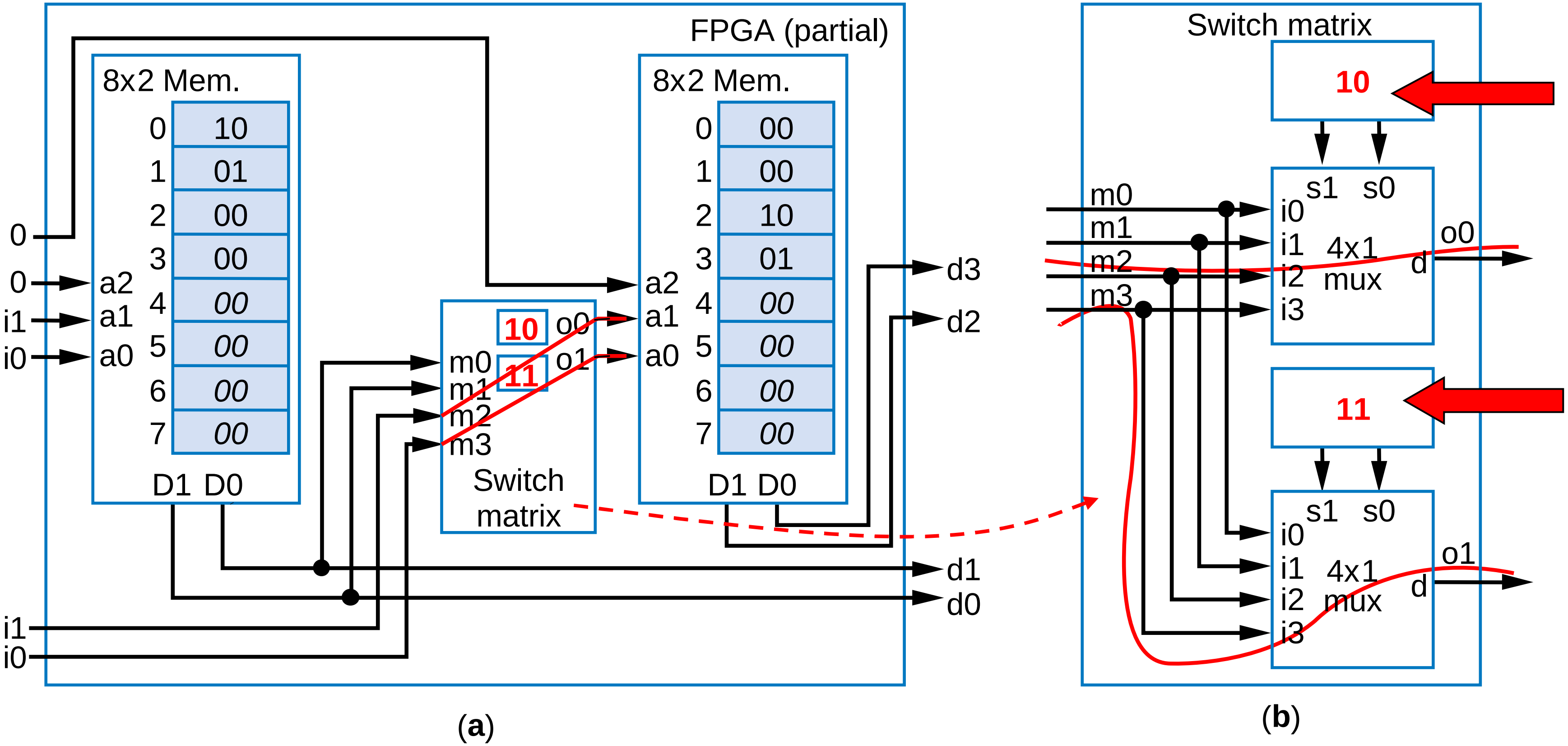


(a) (b)

a a

FPGA Internals: Switch Matrices

- Mapping a 2x4 decoder onto an FPGA with a switch matrix



(a)

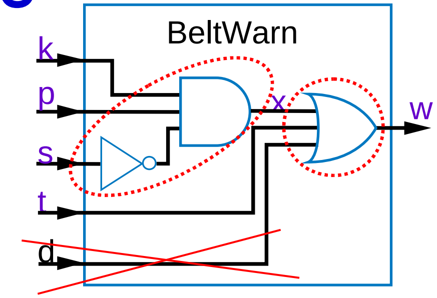
(b)

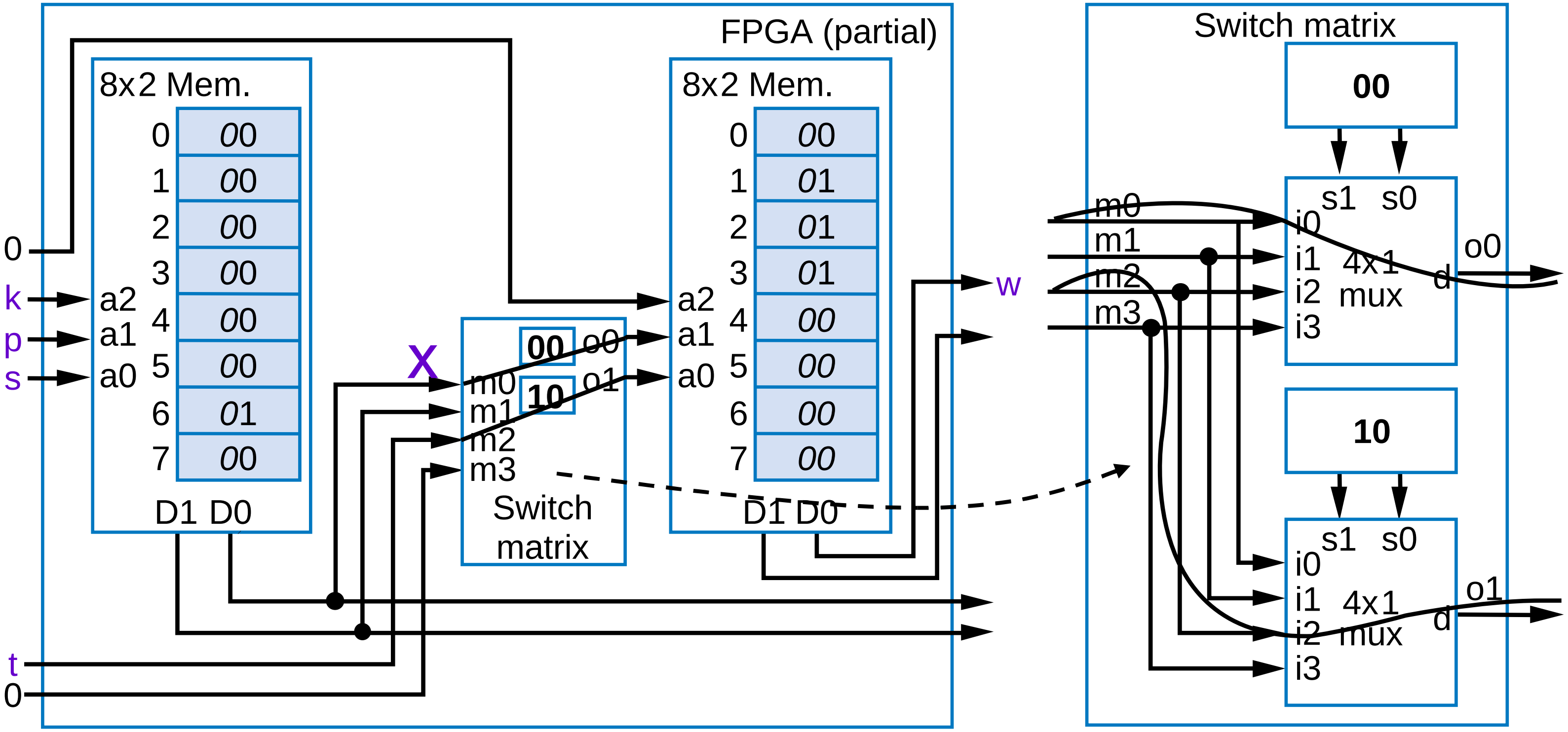
These bits establish the desired connections

a

FPGA Internals: Switch Matrices

- Mapping the extended seatbelt warning light onto an FPGA with a switch matrix
  - Recall earlier example (let's ignore d input for simplicity)



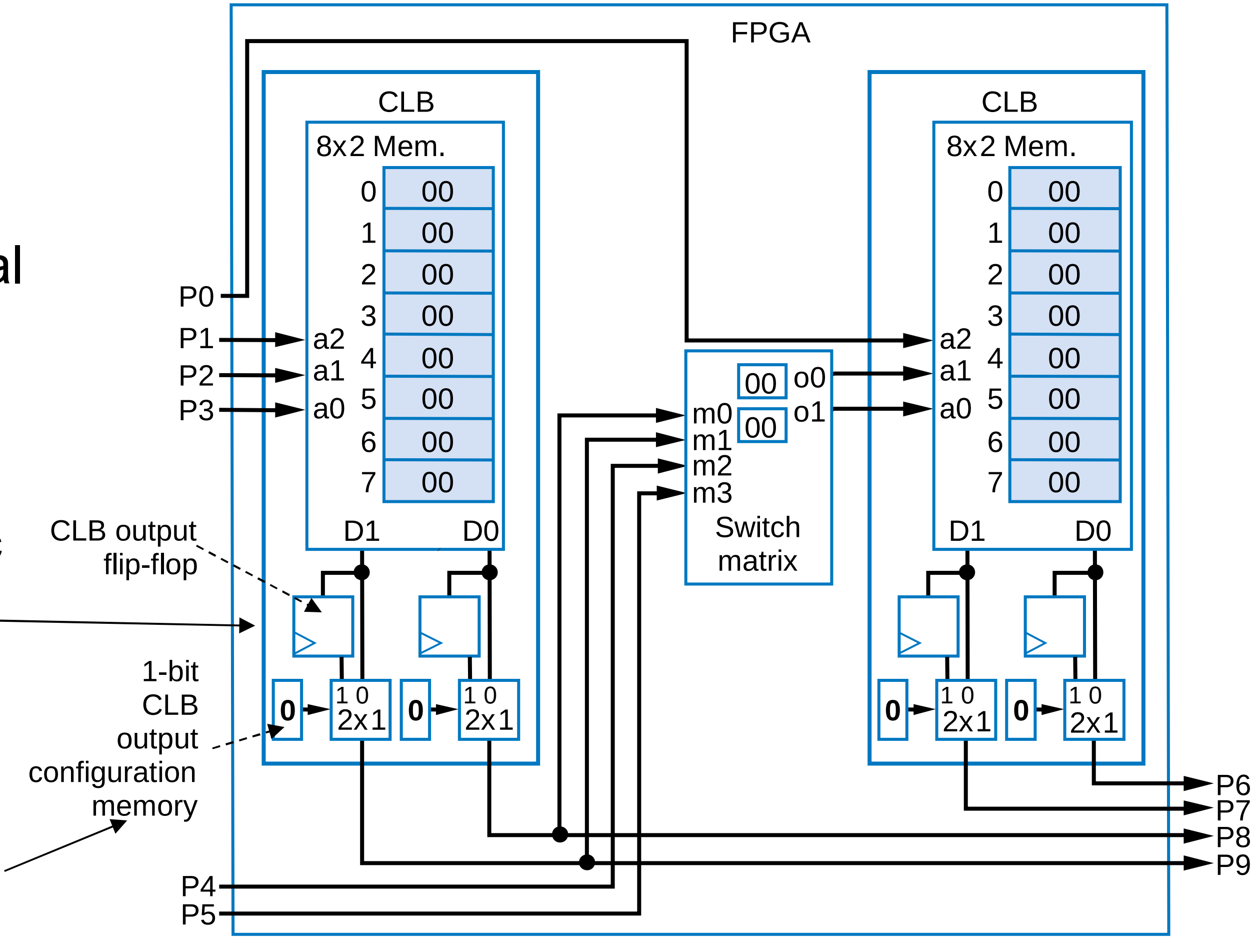


(a) **(b)**

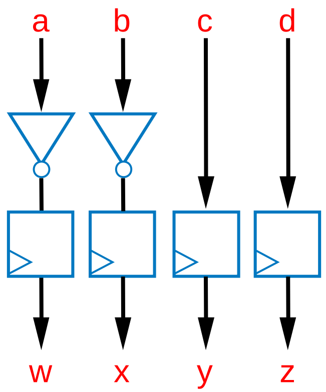
a

FPGA Internals: Configurable Logic Blocks (CLBs)

- LUTs can only implement combinational logic
- Need flip-flops to implement sequential logic
- Add flip-flop to each LUT output
  - Configurable Logic Block (CLB)
    - LUT + flip-flops
  - Can program CLB outputs to come from flip-flops or from LUTs directly



FPGA Internals: Sequential Circuit Example using CLBs



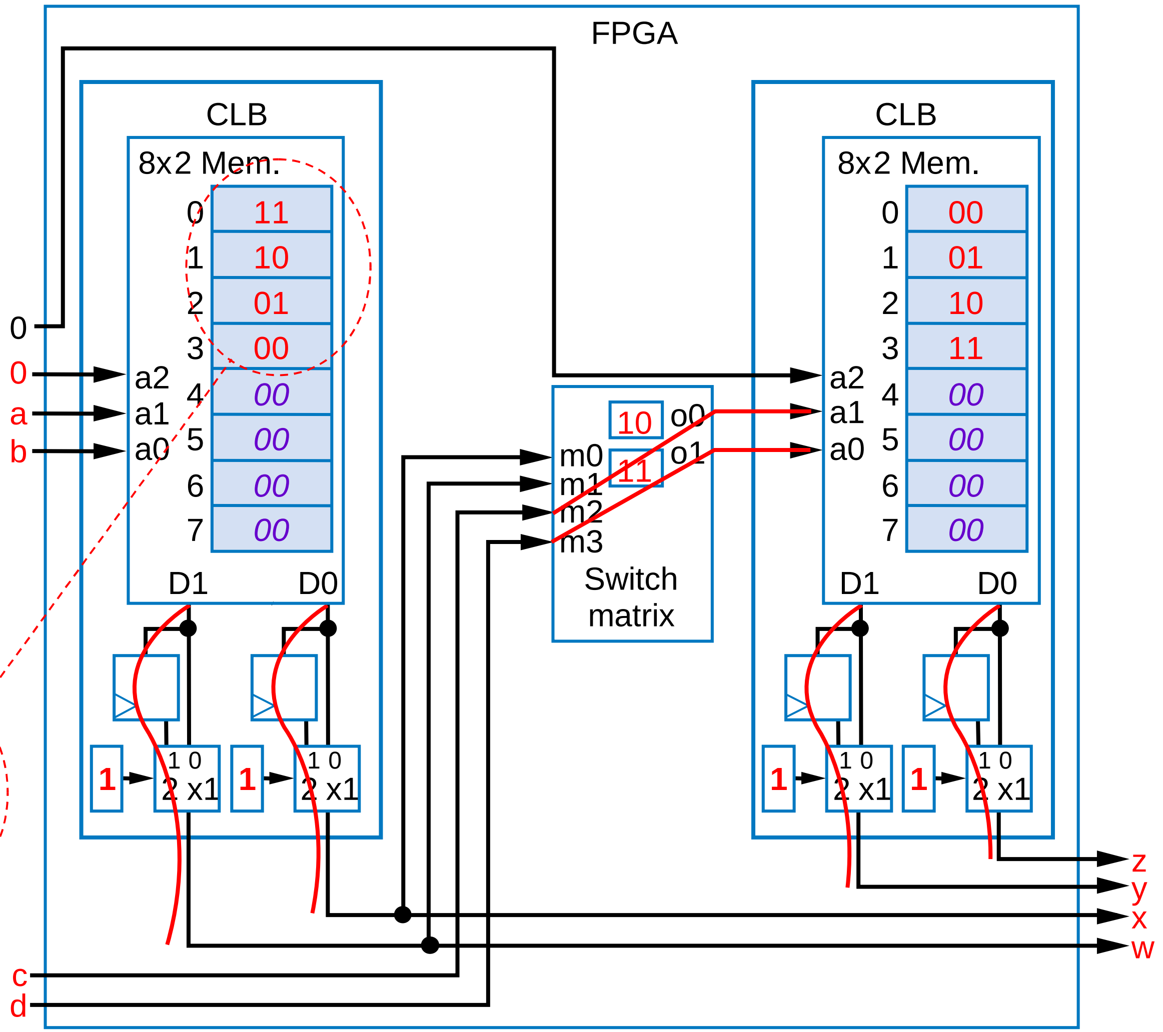
(**a**)

Left lookup table

a2	a1	a0	D1	D0
0	a	b	w=a'	x=b'
0	0	0	1	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	0

below unused

(**b**)



(**c**)

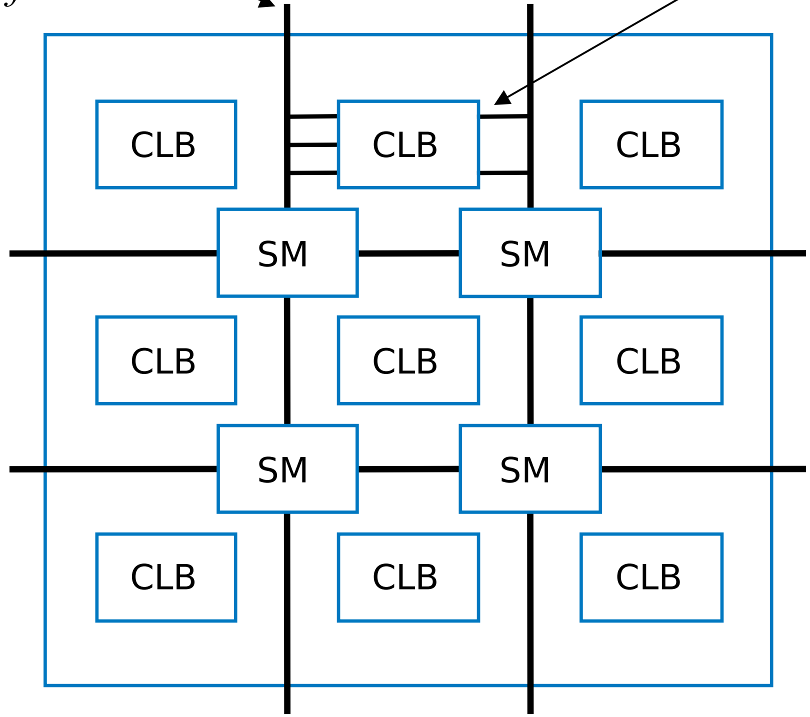
a

FPGA Internals: Overall Architecture

- Consists of hundreds or thousands of CLBs and switch matrices (SMs) arranged in regular pattern on a chip

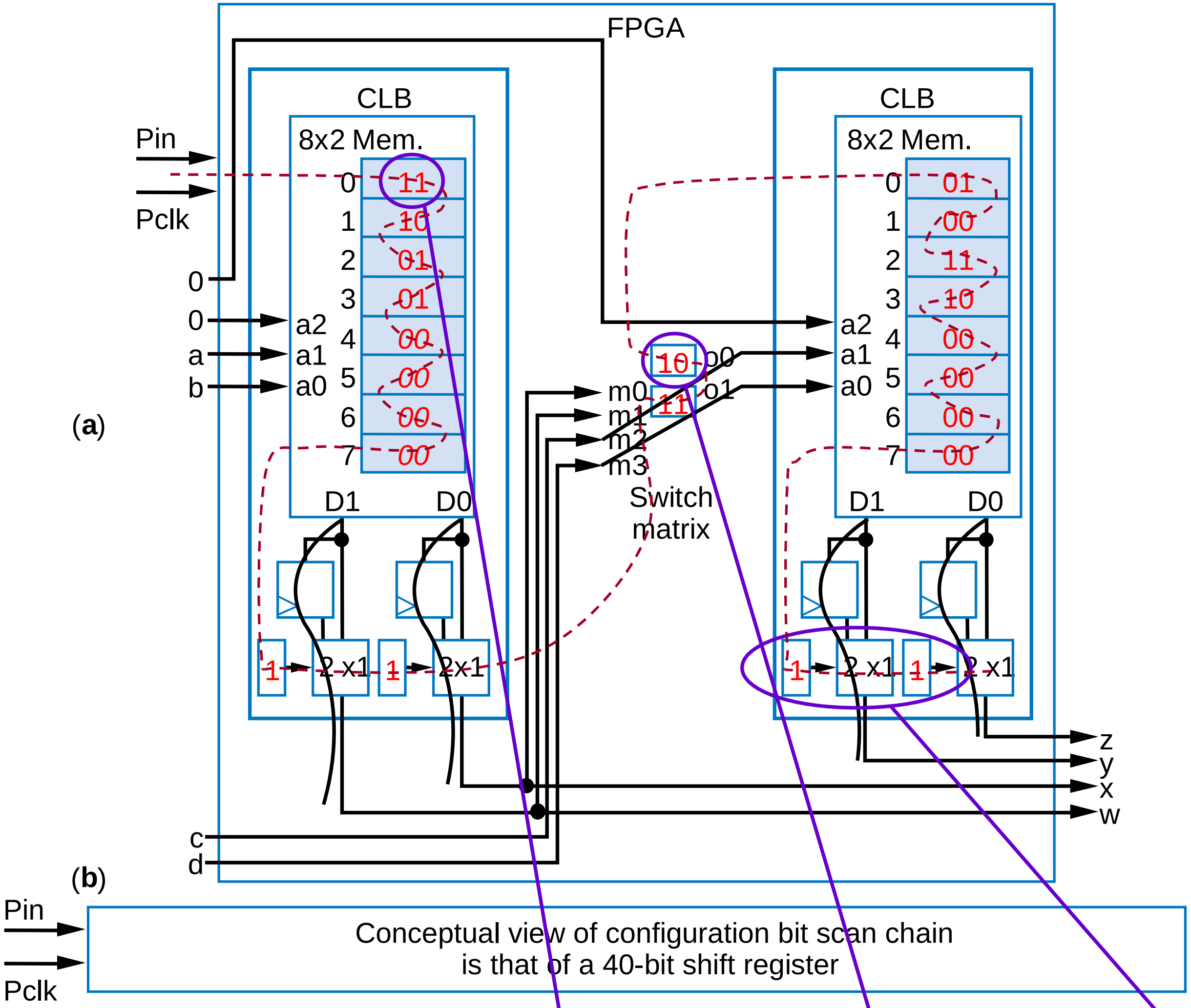
Represents channel with tens of wires

Connections for just one CLB shown, but all CLBs are obviously connected to channels



FPGA Internals: Programming an FPGA

- All configuration memory bits are connected as one big shift register
  - Known as scan chain
- Shift in "bit file" of desired circuit



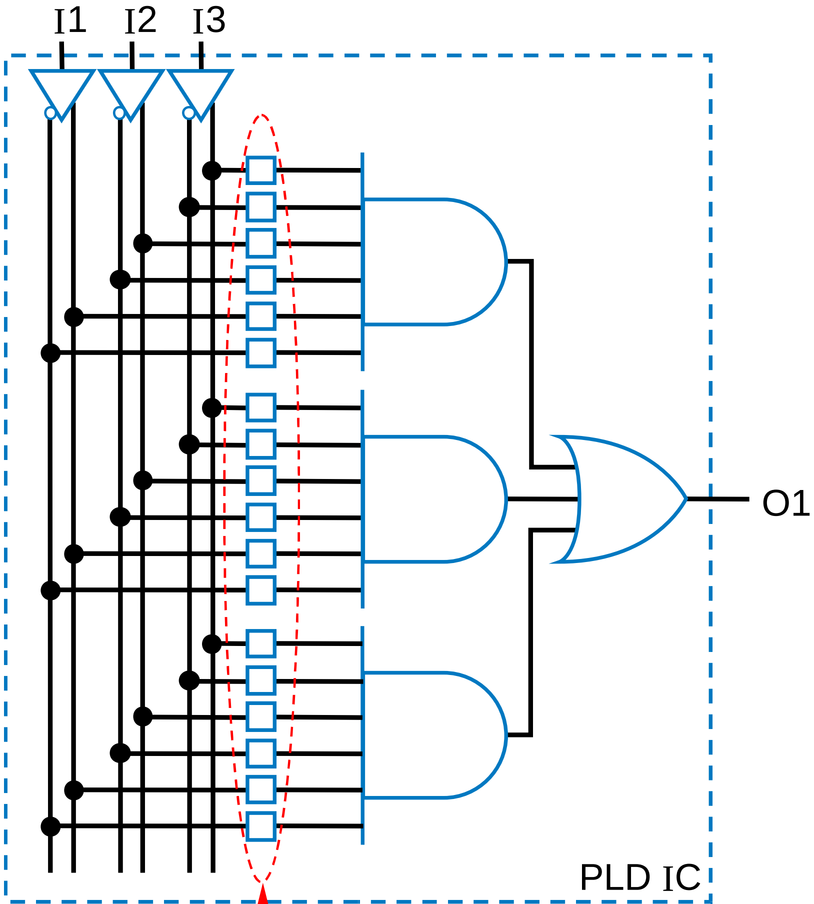
Conceptual view of configuration bit scan chain is that of a 40-bit shift register

(**c**) Bit file contents for desired circuit: 1101011000000000111101010011010000000011

This isn't wrong. Although the bits appear as "10" above, note that the scan chain passes through those bits from right to left – so "01" is correct here.

Other Technologies

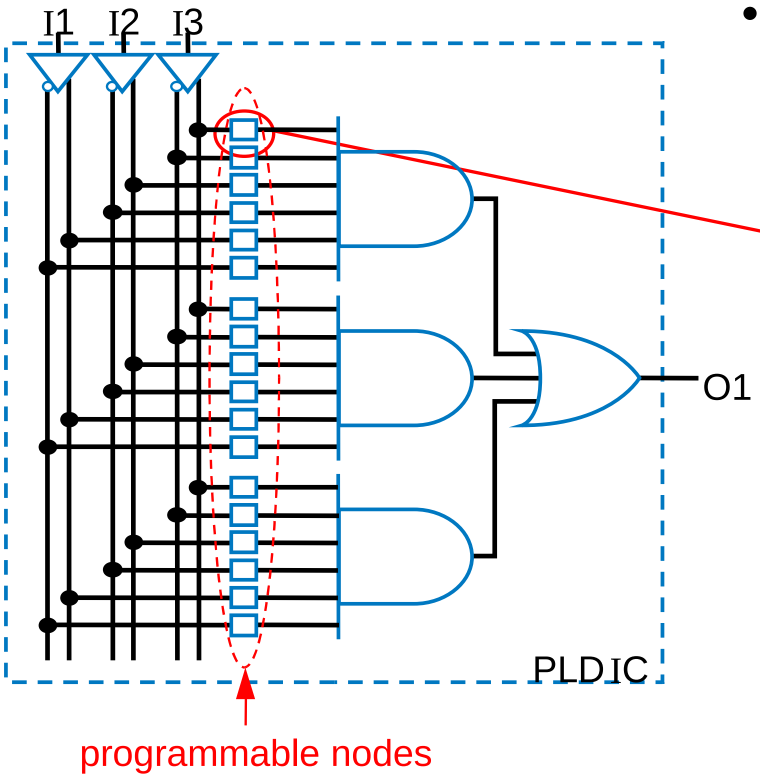
- Simple Programmable Logic Devices (SPLDs)
  - Developed 1970s (thus, pre-dates FPGAs)
  - Prefabricated IC with large AND-OR structure
  - Connections can be "programmed" to create custom circuit
    - Circuit shown can implement any 3-input function of up to 3 terms
      - e.g., F = abc + a'c'

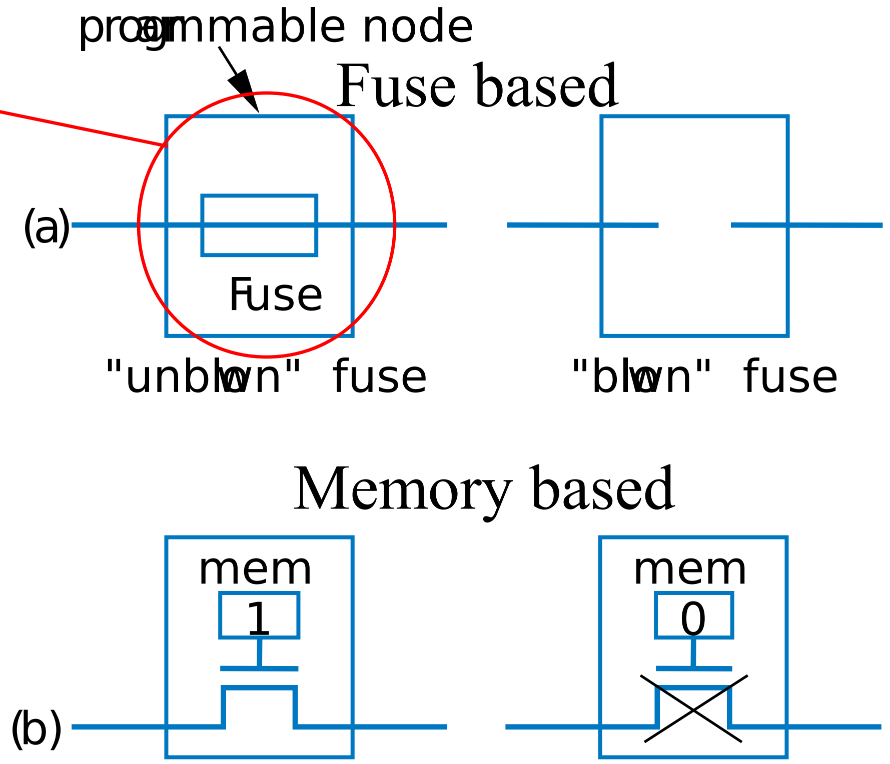


programmable nodes

Programmable Nodes in an SPLD

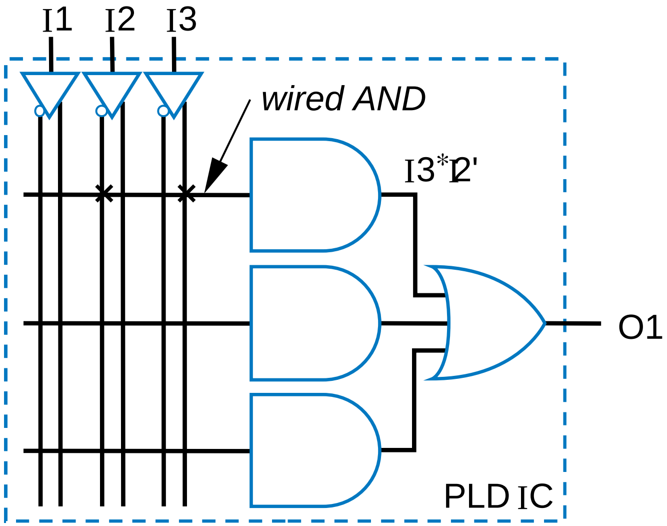
- Fuse based – "blown" fuse removes connection
- Memory based – 1 creates connection



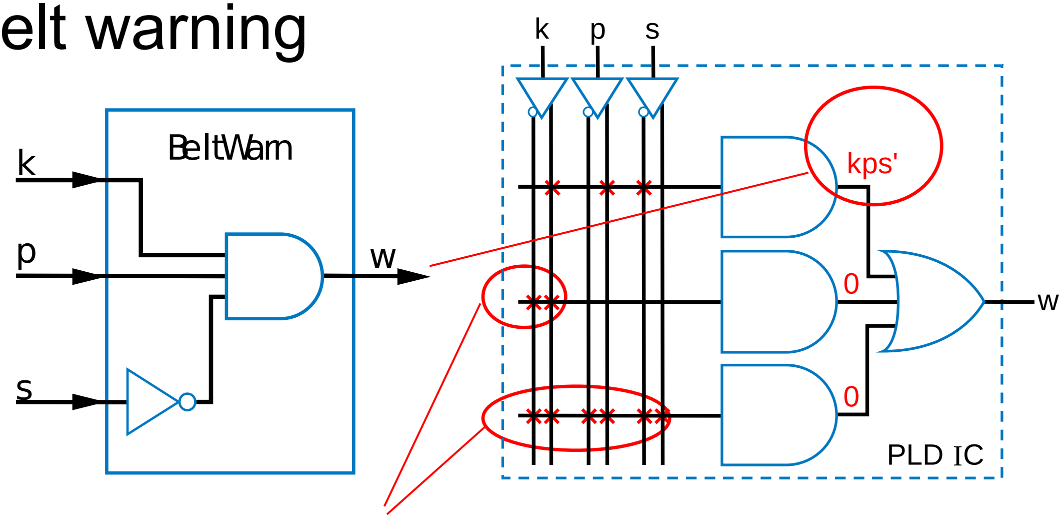


PLD Drawings and PLD Implementation Example

- Common way of drawing PLD connections:
  - Uses one wire to represent all inputs of an AND
  - Uses "x" to represent connection
    - Crossing wires are not connected unless "x" is present

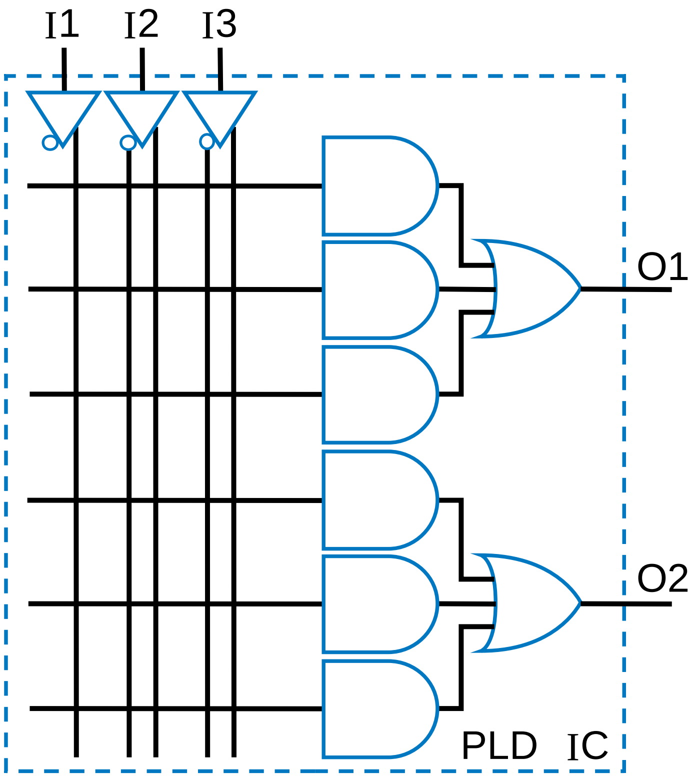


- Example: Seat belt warning light using SPLD



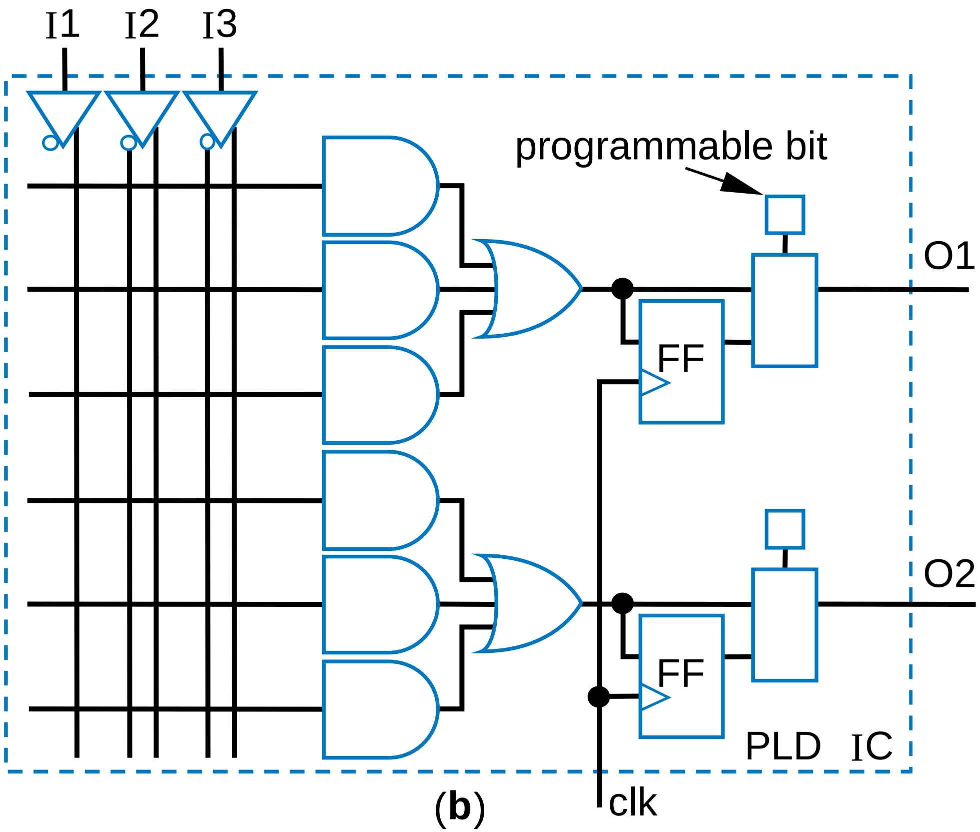
Two ways to generate a 0 term

PLD Extensions



(a)

Two-output PLD



(b)

PLD with programmable registered outputs

More on PLDs

- Originally (1970s) known as Programmable Logic Array – *PLA*
  - Had programmable AND and OR arrays
- AMD created "Programmable Array Logic" – "*PAL*" (trademark)
  - Only AND array was programmable (fuse based)
- Lattice Semiconductor Corp. created "Generic Array Logic – "GAL" (trademark)
  - Memory based
- As IC capacities increased, companies put multiple PLD structures on one chip, interconnecting them
  - Become known as Complex PLDs (CPLD), and older PLDs became known as Simple PLDs (SPLD)
- GENERALLY SPEAKING, difference of SPLDs vs. CPLDs vs. FPGAs:
  - SPLD: tens to hundreds of gates, and usually non-volatile (saves bits without power)
  - CPLD: thousands of gates, and usually non-volatile
  - FPGA: tens of thousands of gates and more, and usually volatile (but no reason why couldn't be non-volatile)

Technology Comparisons

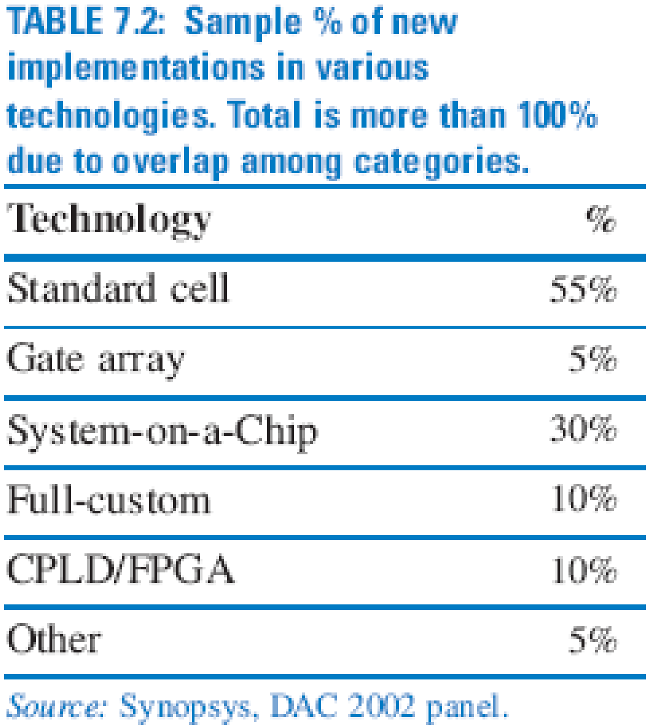
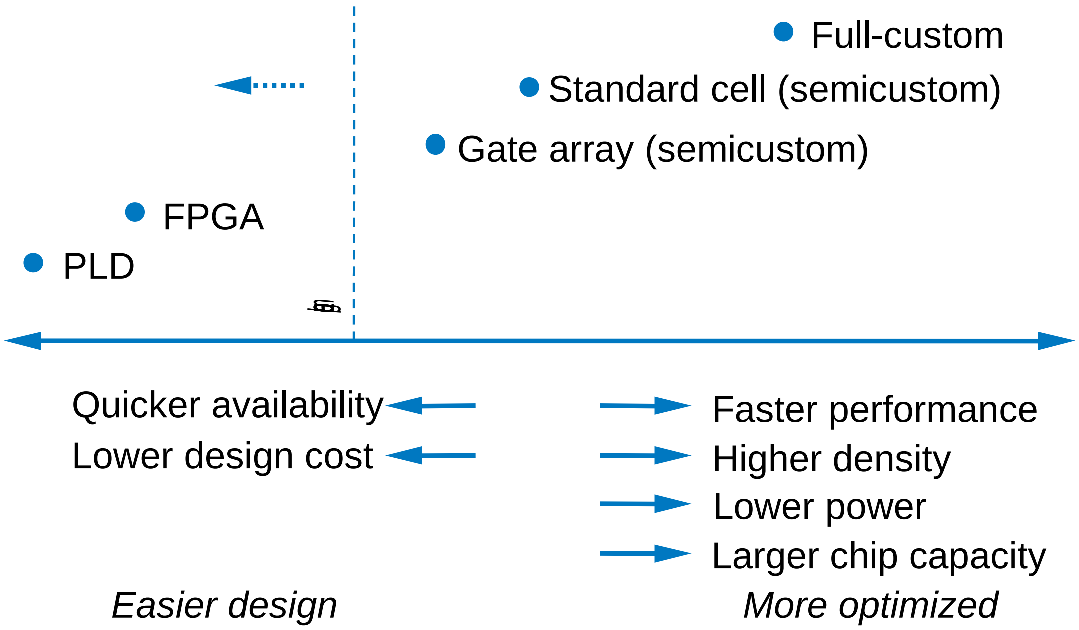


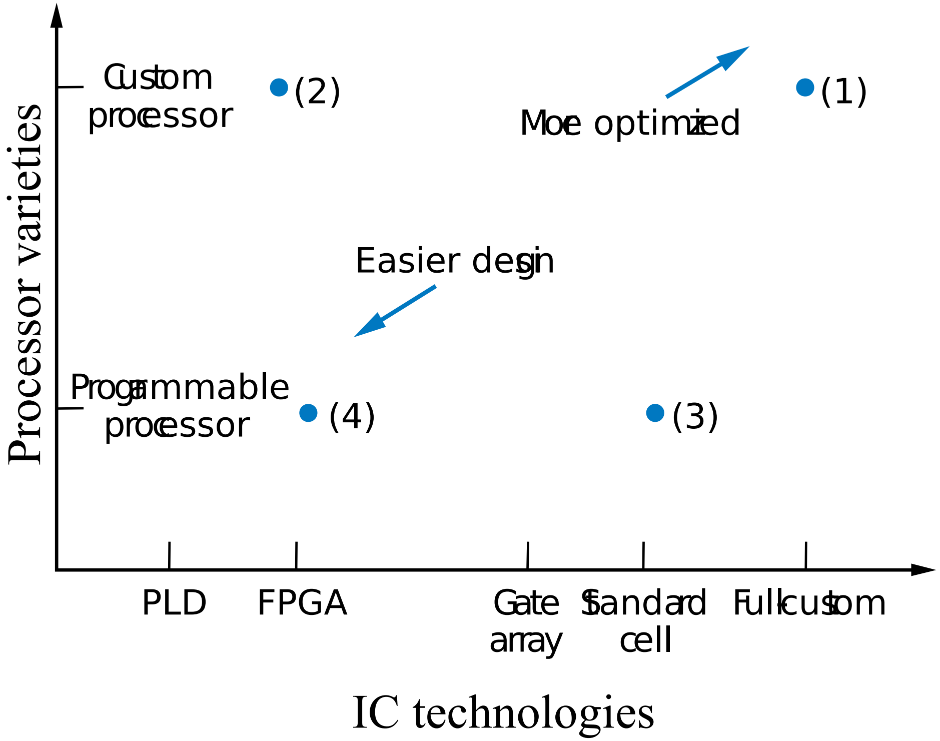
TABLE 7.2: Sample % of new implementations in various technologies. Total is more than 100% due to overlap among categories.

Technology	%
Standard cell	55%
Gate array	5%
System-on-a-Chip	30%
Full-custom	10%
CPLD/FPGA	10%
Other	5%

Source: Synopsys, DAC 2002 panel.



Technology Comparisons



(1): Custom processor in full-custom IC

Highly optimized

(2): Custom processor in FPGA

Parallelized circuit, slower IC technology but programmable

(3): Programmable processor in standard cell IC

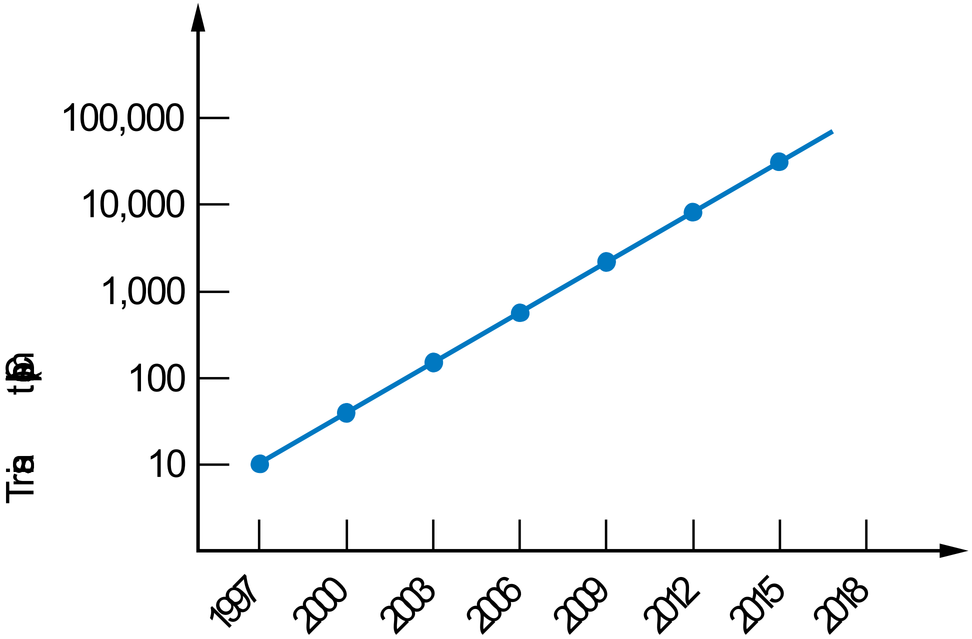
Program runs (mostly) sequentially on moderate-costing IC

(4): Programmable processor in FPGA

Not only can processor be programmed, but FPGA can be programmed to implement multiple processors/coprocessors

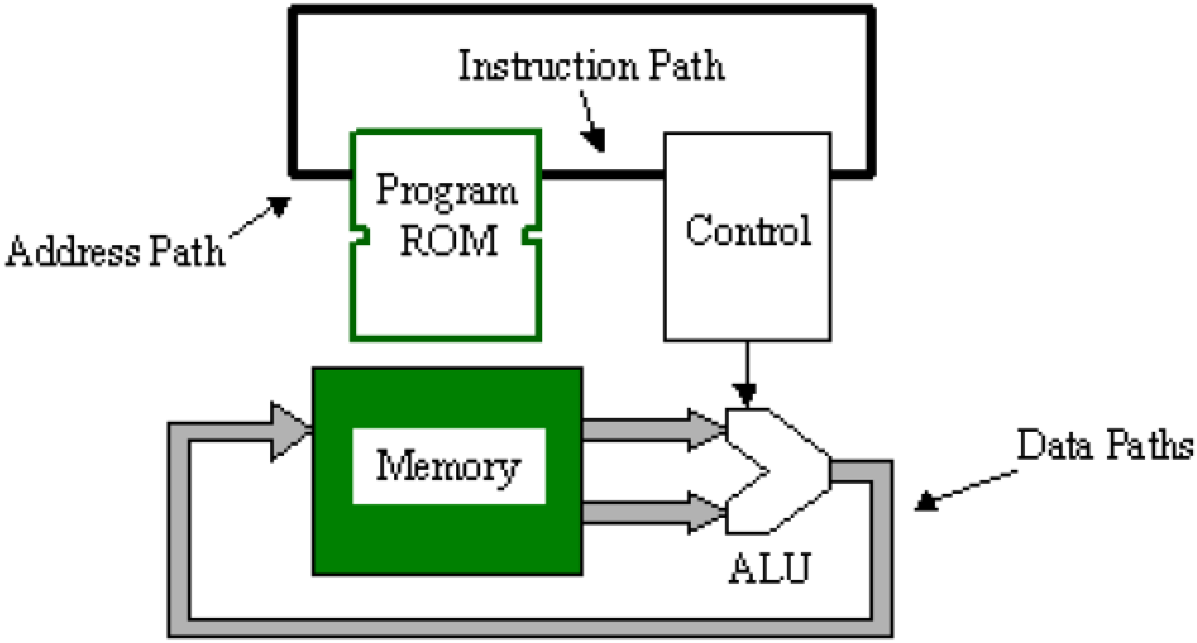
Key Trend in Implementation Technologies

- Transistors per IC doubling every 18 months for past three decades
  - Known as "Moore's Law"
  - Tremendous implications – applications infeasible at one time due to outrageous processing requirements become feasible a few years later
  - Can Moore's Law continue?

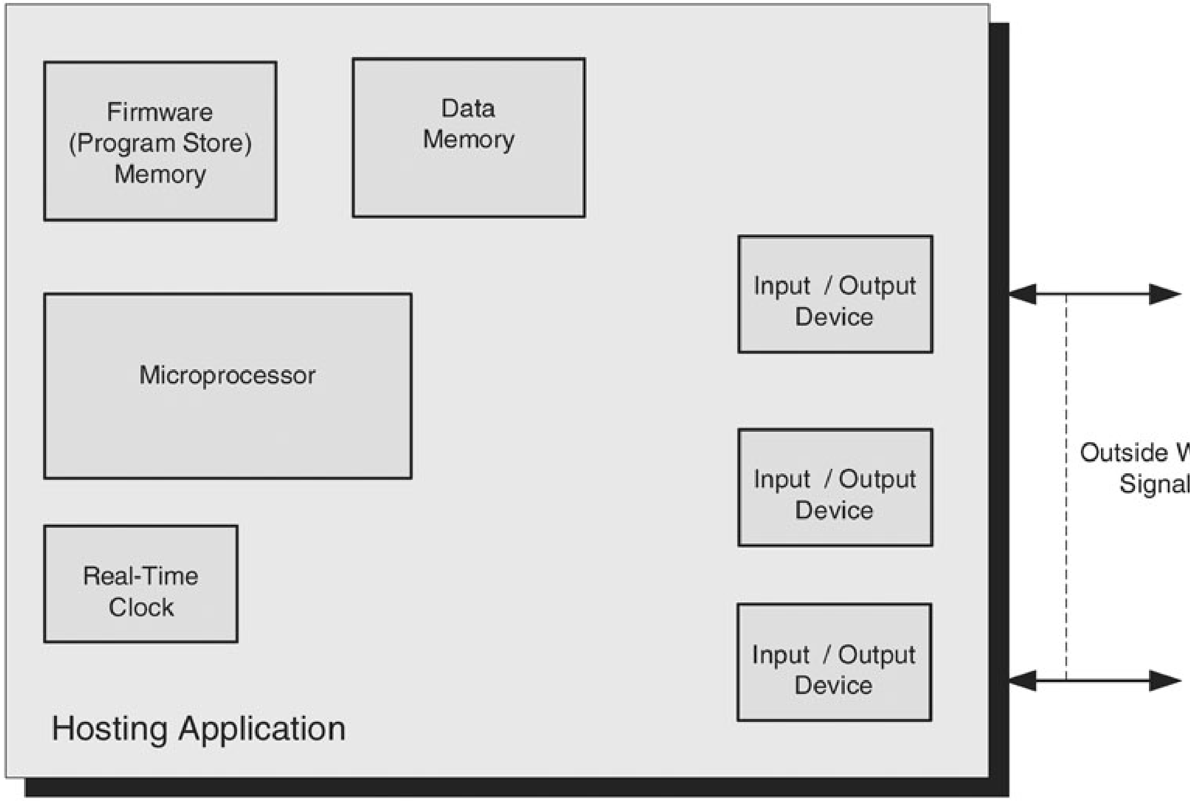


Procesor i pamięć wbudowane w FPGA

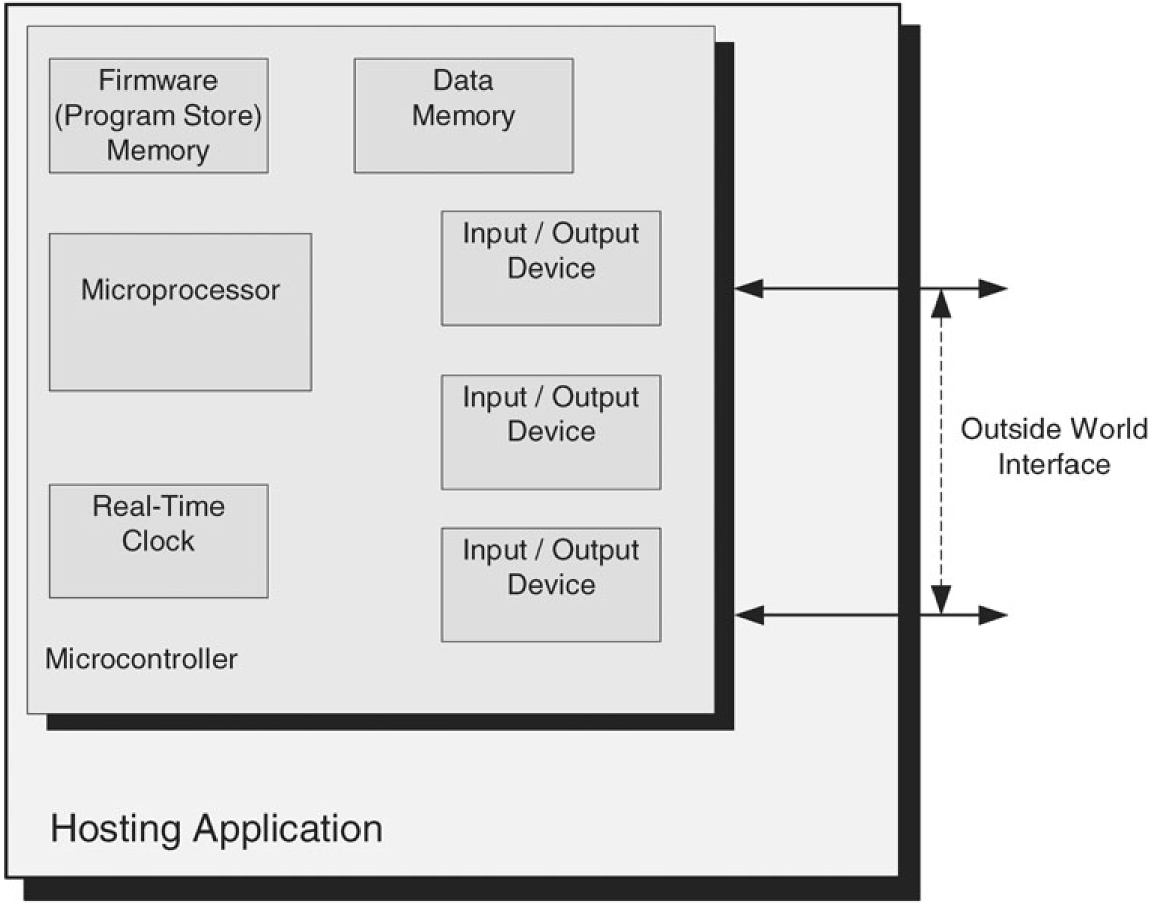
- W strukturze FPGA można skonfigurować
  - Procesor
    - np. open source picoBlaze (Verilog, VHDL)
  - Pamięć programu (ROM)
  - Pamięć danych
  - Interfejsy
  - Układy we/wy



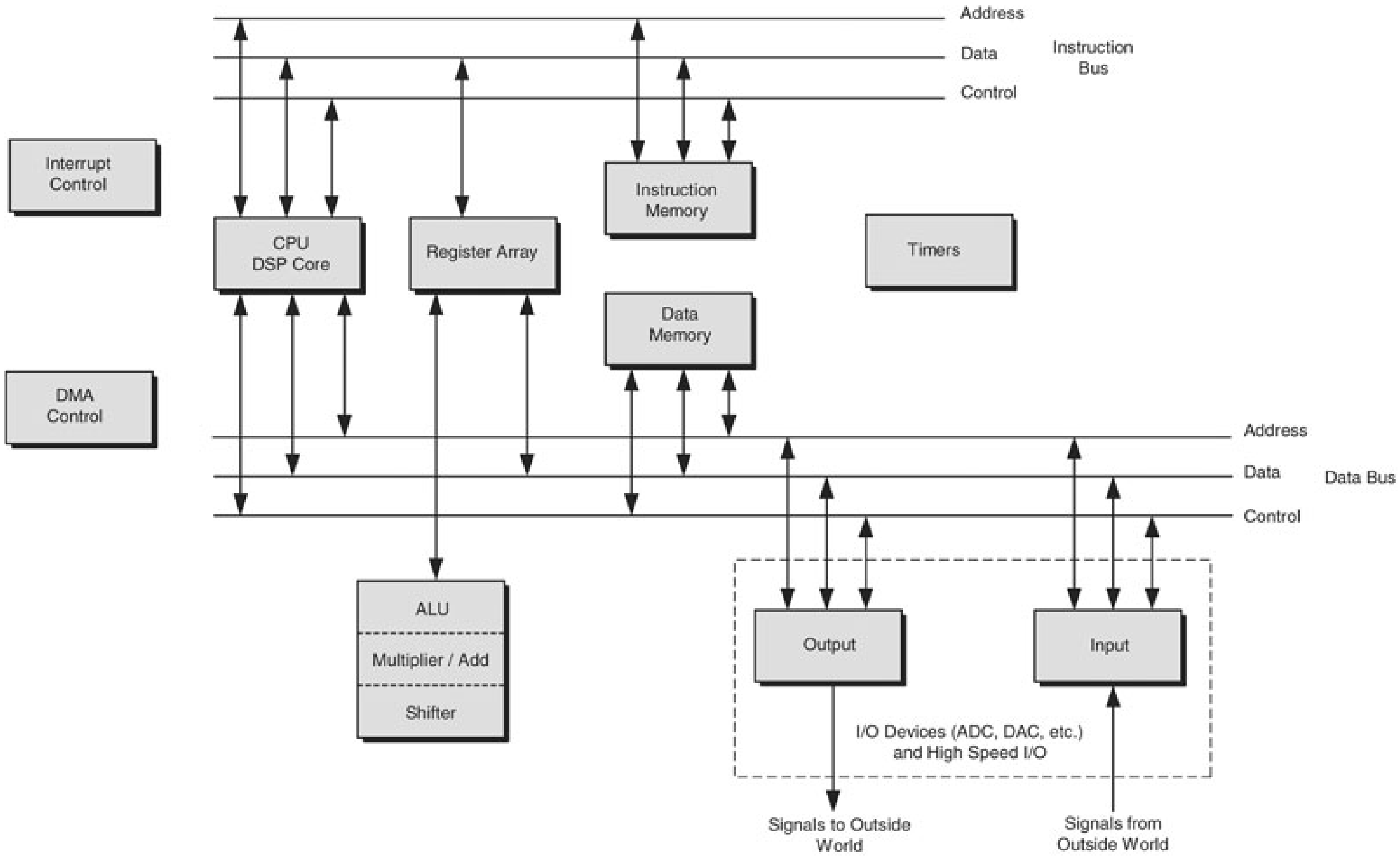
uP based system



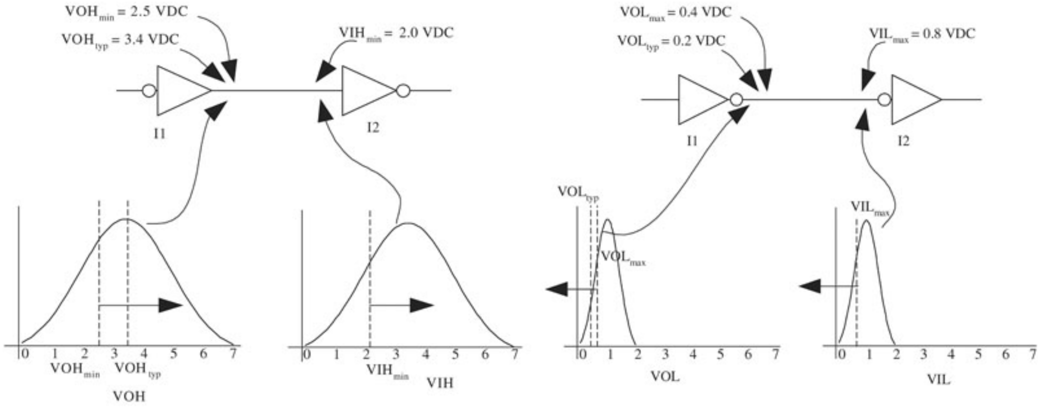
uC based system

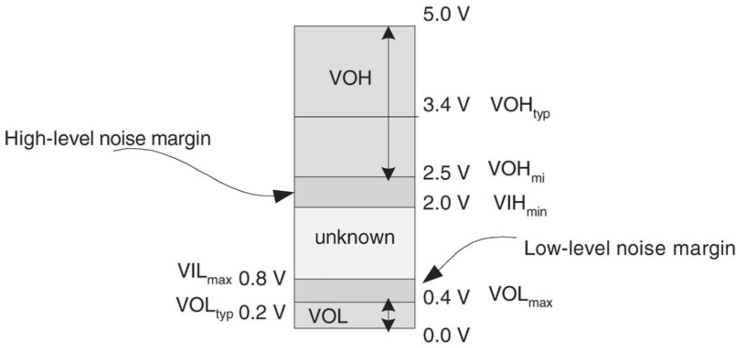


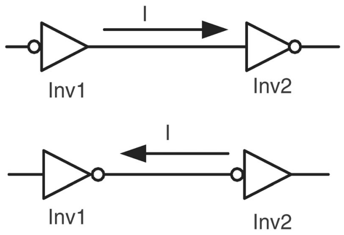
DSP

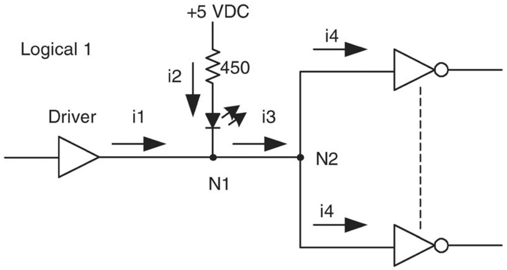

Address
Data
Control
Instruction Bus
Interrupt Control
CPU DSP Core
Register Array
Instruction Memory
Timers
Data Memory
DMA Control
Address
Data
Control
Data Bus
ALU
Multiplier / Add
Shifter
Output
Input
I/O Devices (ADC, DAC, etc.) and High Speed I/O
Signals to Outside World
Signals from Outside World

Poziomy sygnałów logicznych

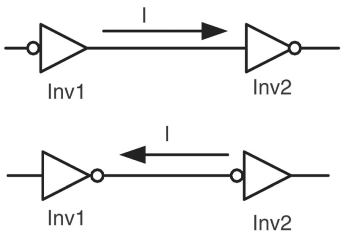


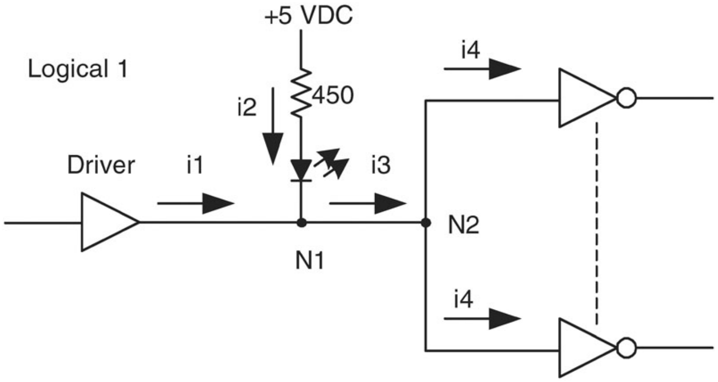


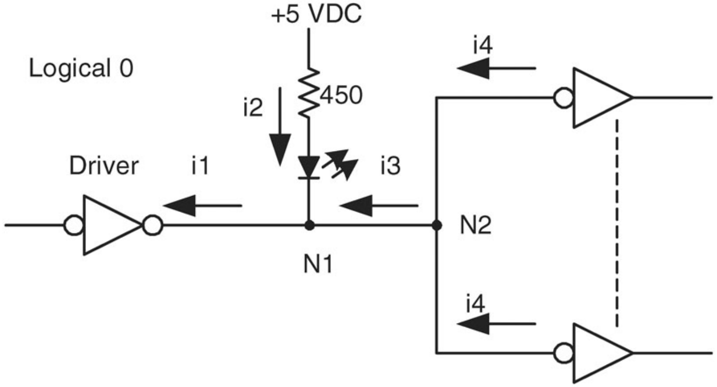




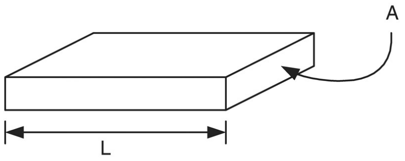
Poziomy sygnałów logicznych

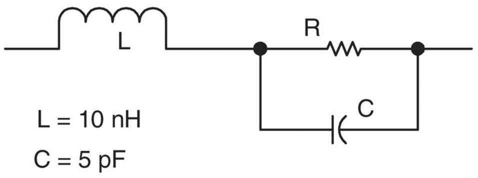


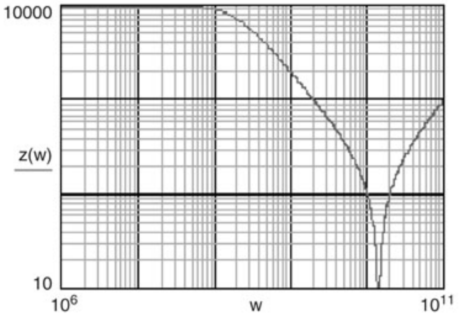


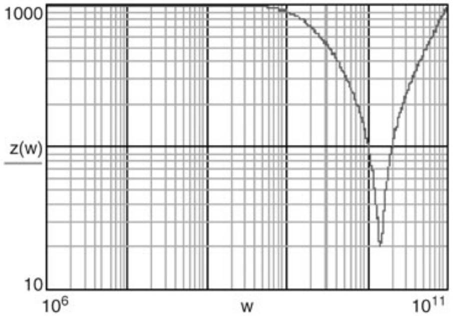


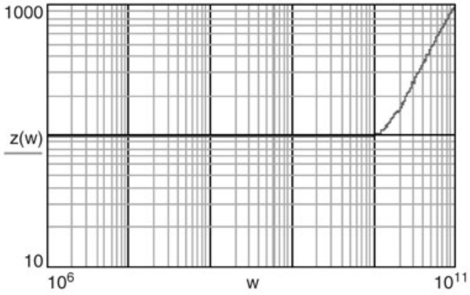
Model i charakterystyki opornika z uwzględnieniem elementów pasożytniczych



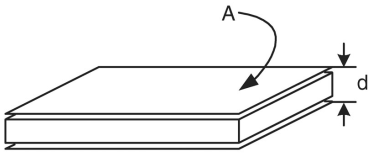


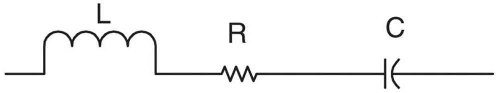




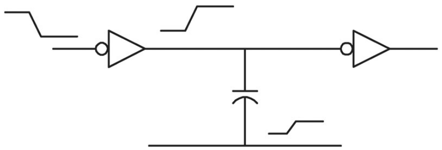


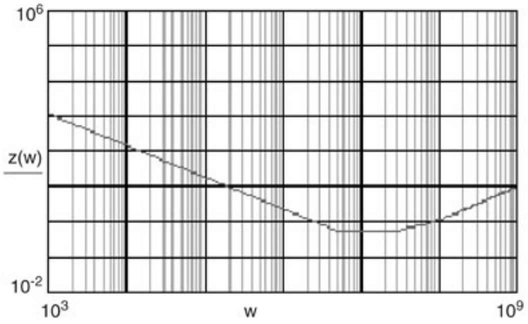
Model i charakterystyki kondensatora z uwzględnieniem elementów pasożytniczych

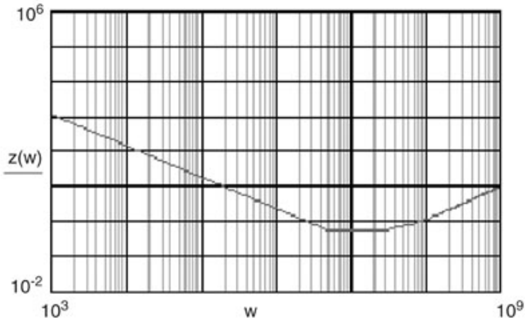


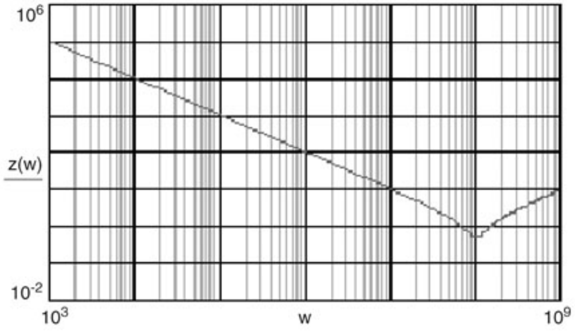


L = 10 nH
R = 0.5 ohm

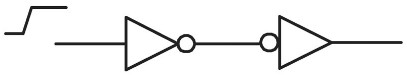


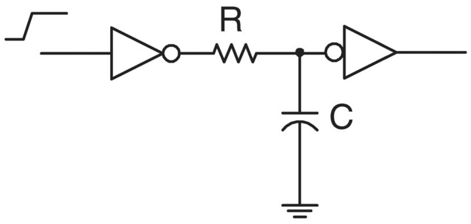


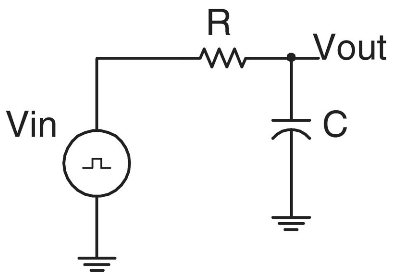


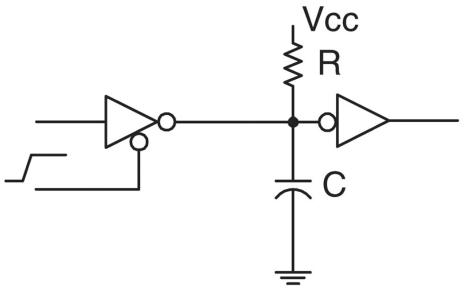


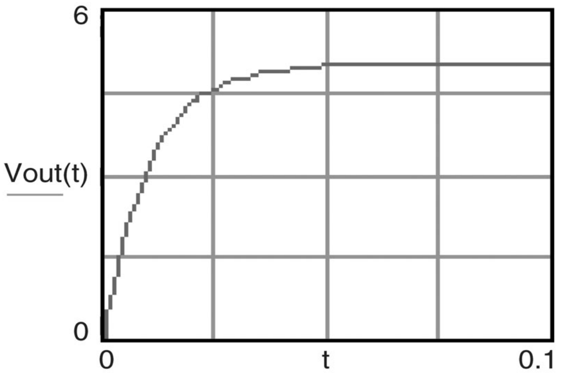
Połączenia układów: modele z elementami pasożytniczymi



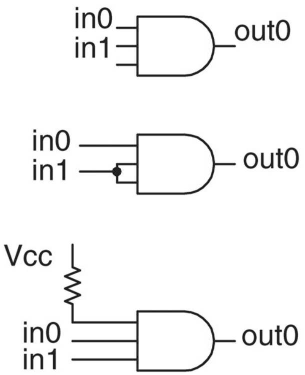


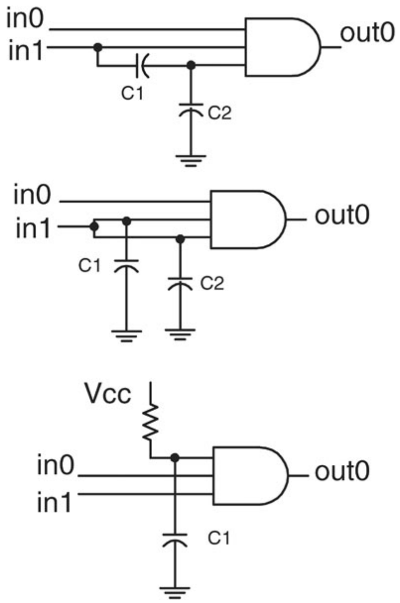


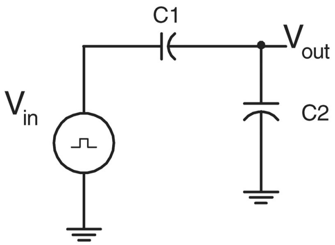




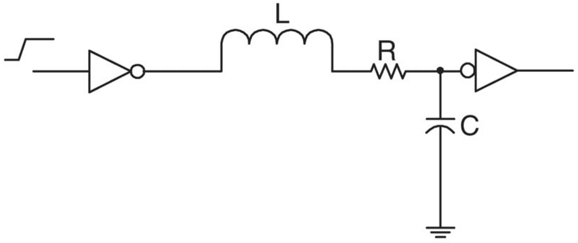
Niewykorzystane wejścia bramek

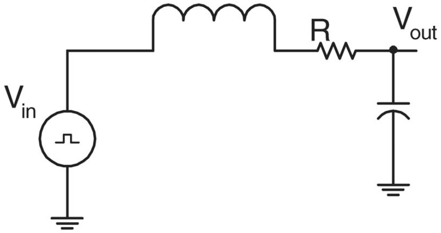


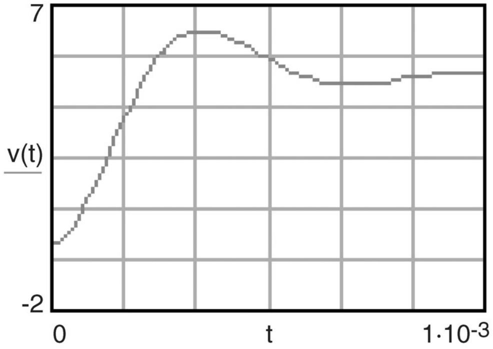




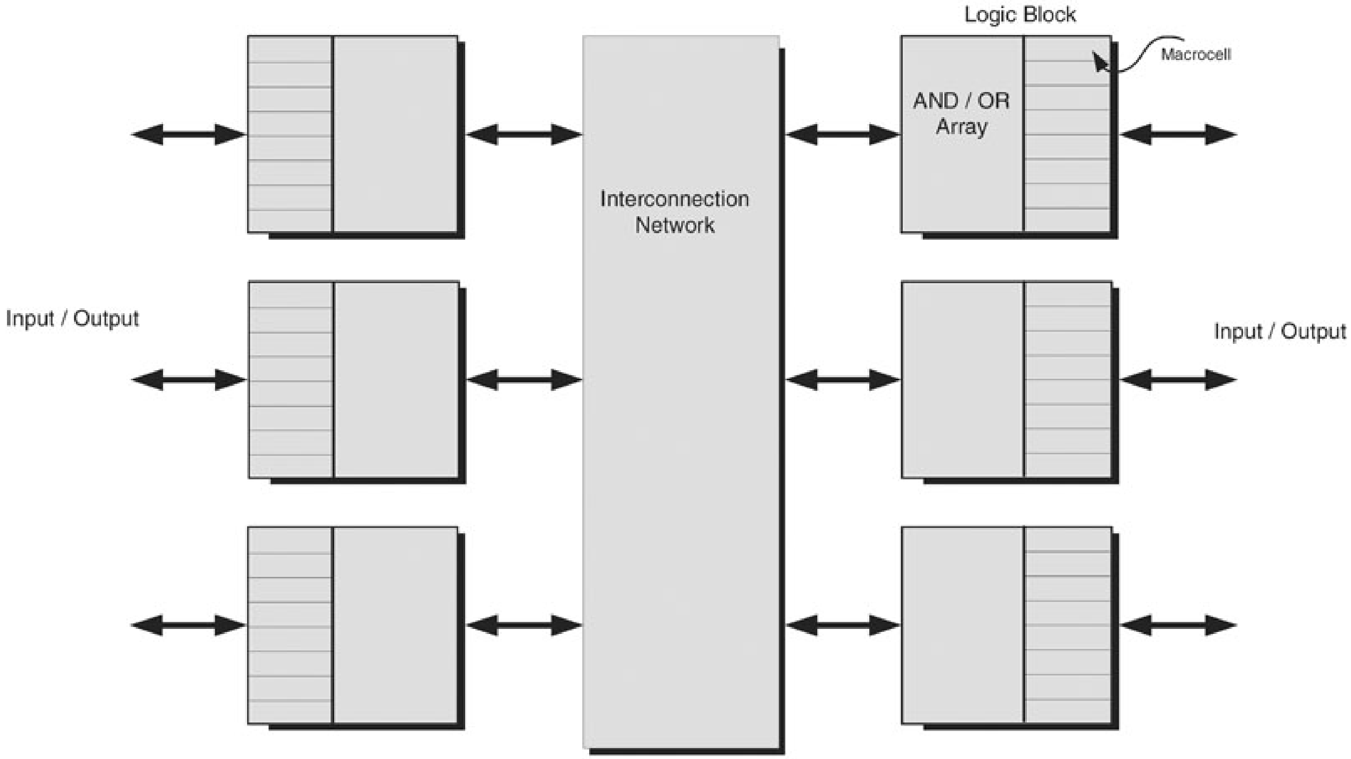
Połączenia układów: modele 2 rzędu z elementami pasożytniczymi



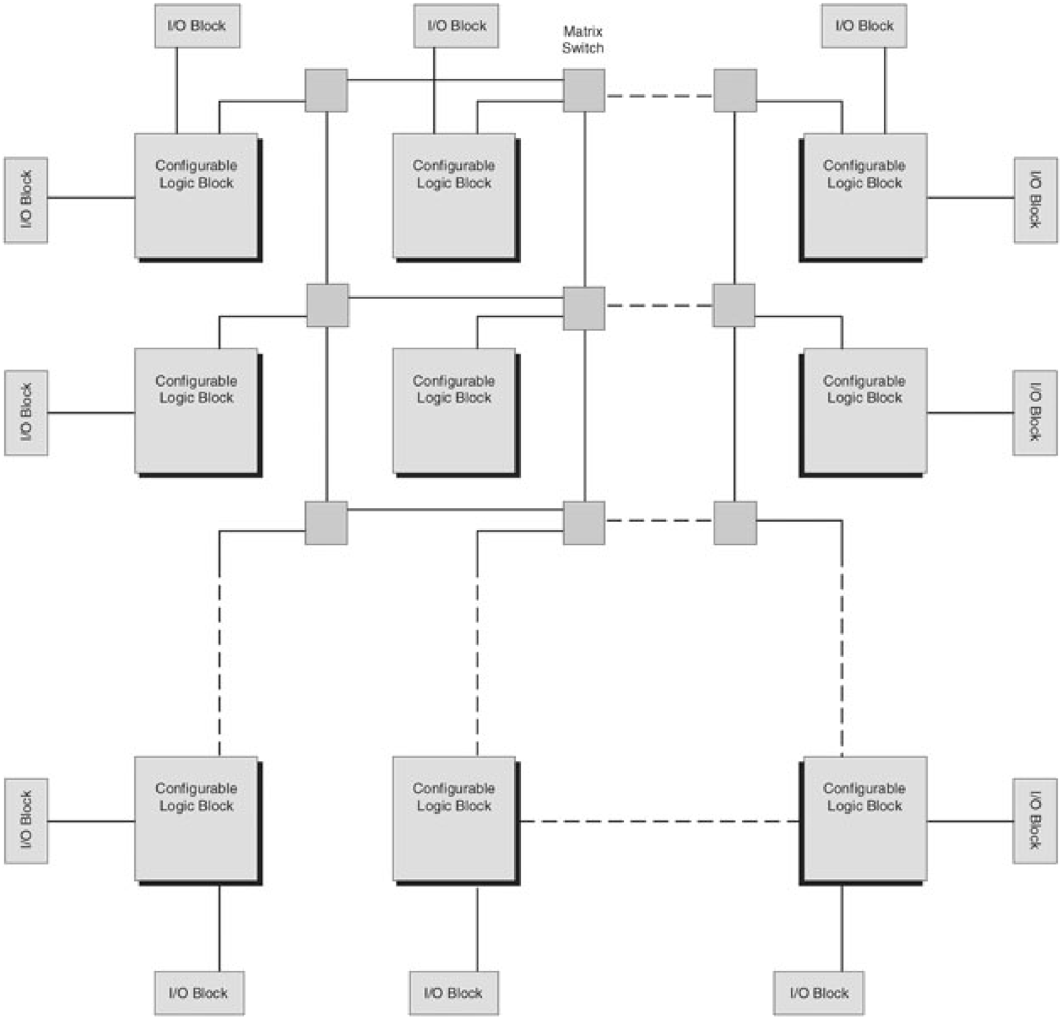




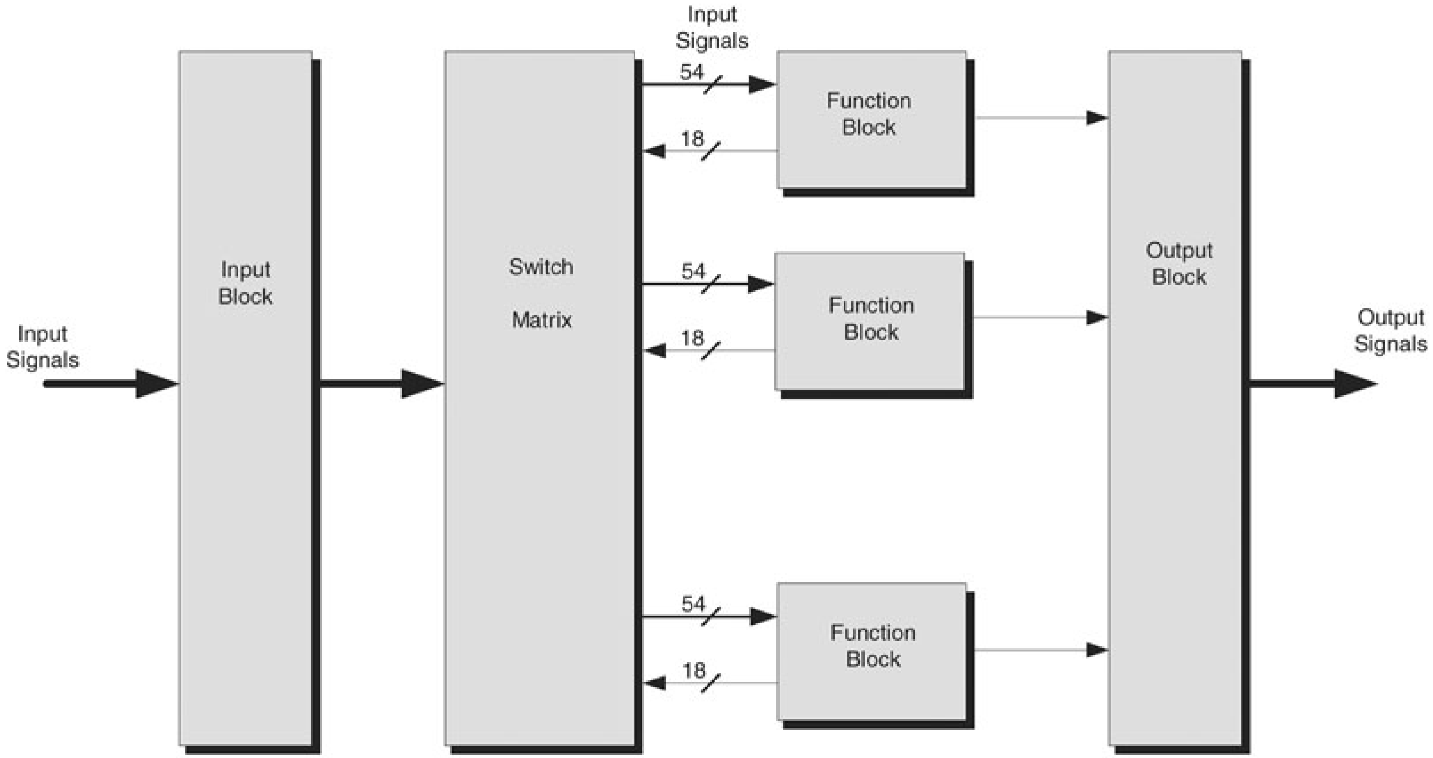
CPLD


Logic Block
Macrocell
AND / OR
Array
Interconnection
Network
Input / Output
Input / Output

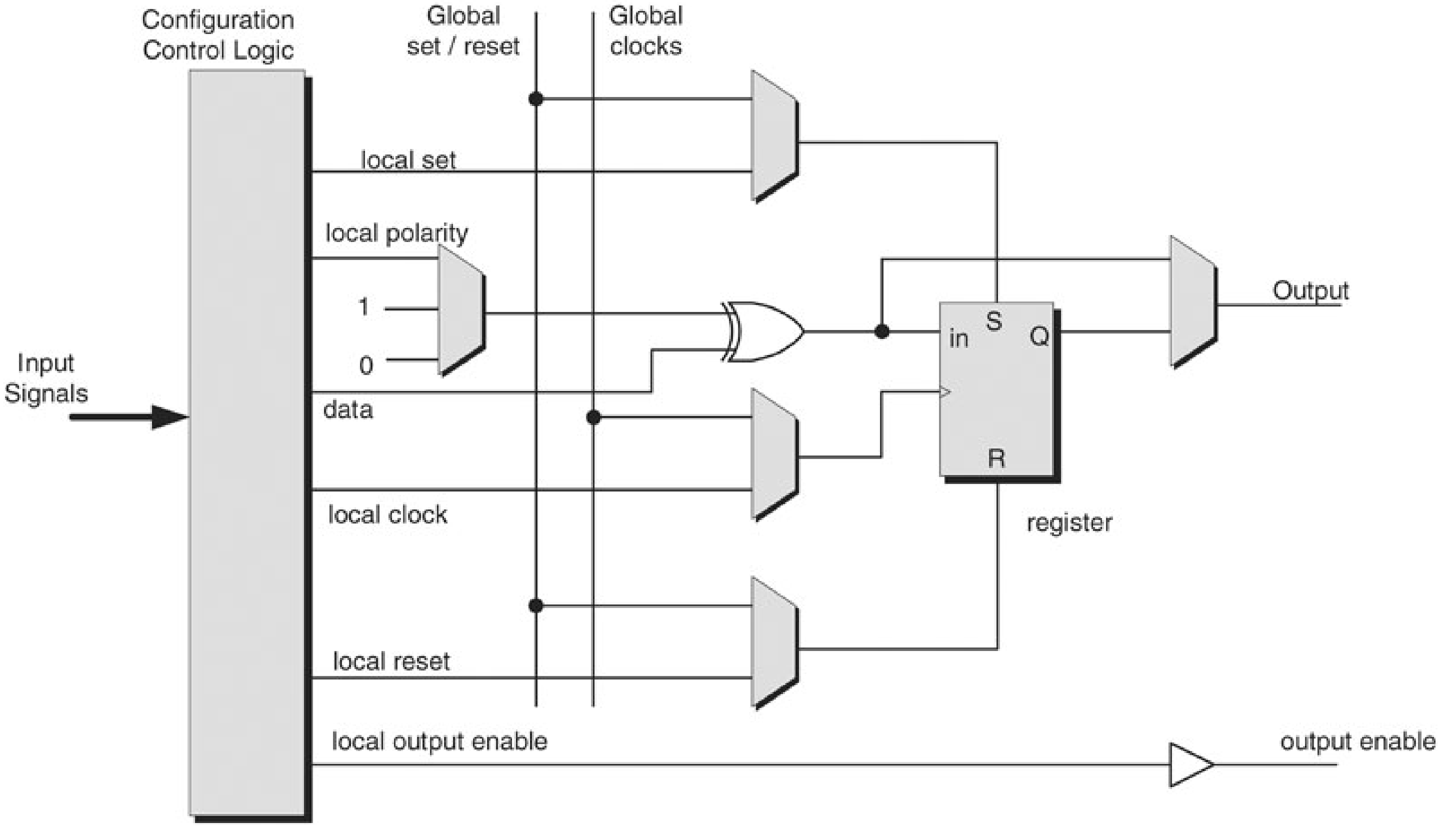
FPGA


I/O Block
I/O Block
Matrix Switch
I/O Block
I/O Block
Configurable Logic Block
Configurable Logic Block
Configurable Logic Block
I/O Block
I/O Block
Configurable Logic Block
Configurable Logic Block
Configurable Logic Block
I/O Block
I/O Block
Configurable Logic Block
Configurable Logic Block
Configurable Logic Block
I/O Block
I/O Block
I/O Block
I/O Block

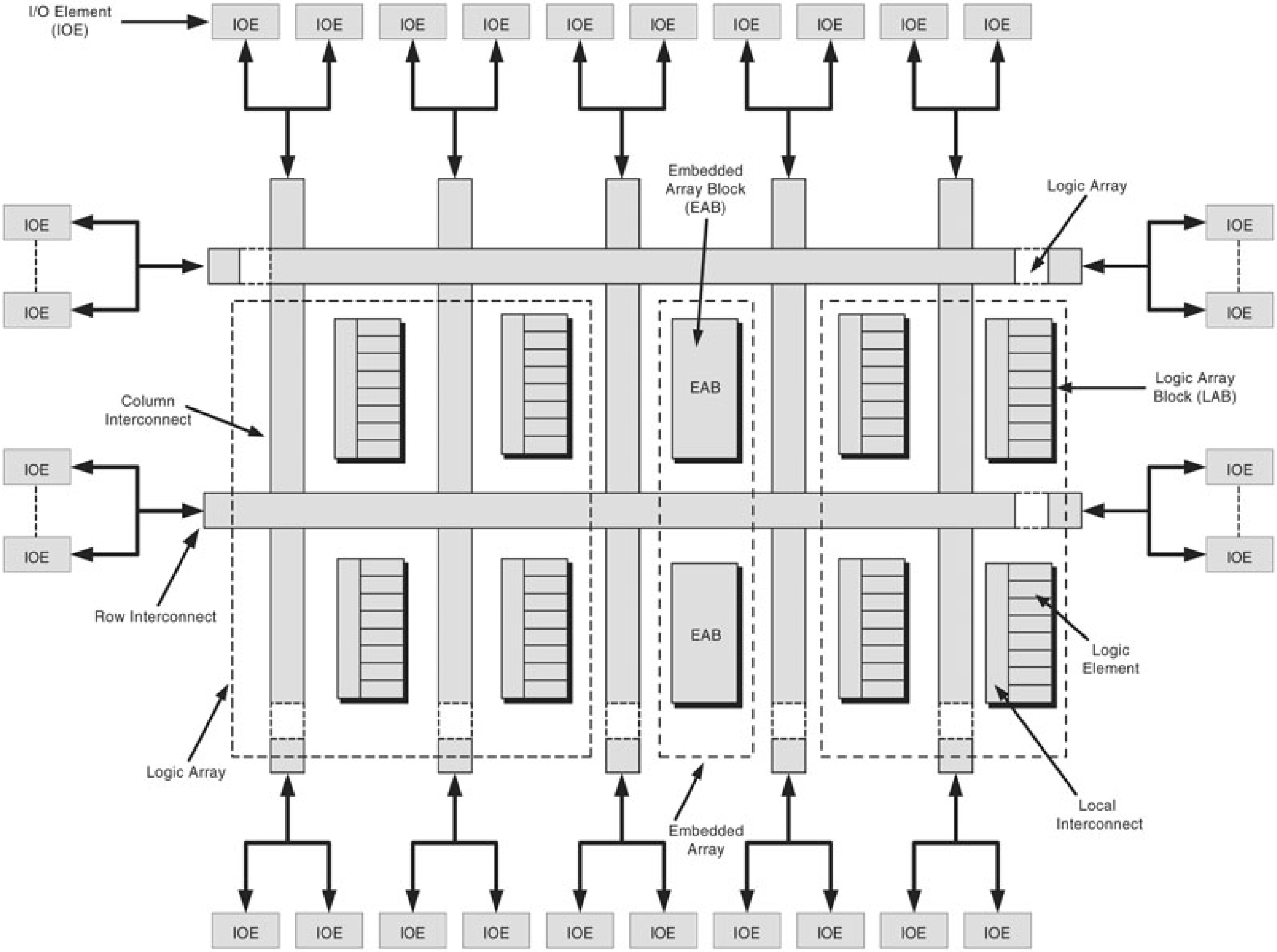
Xilinx CPLD


Input
Signals
Input
Block
Switch
Matrix
Input
Signals
54
18
Function
Block
54
18
Function
Block
54
18
Function
Block
Output
Block
Output
Signals

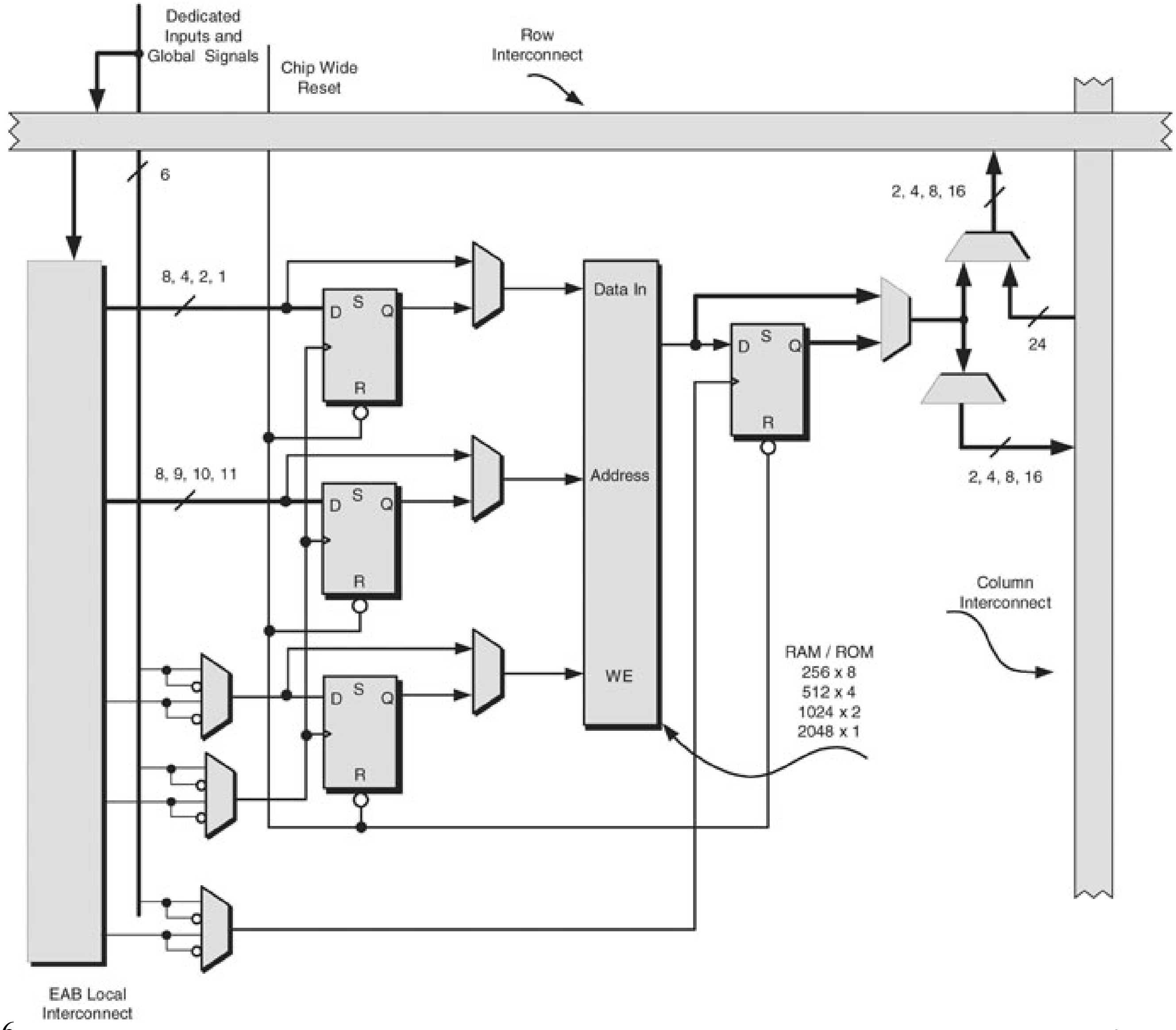
Xilinx CPLD


Configuration Control Logic
Global set / reset
Global clocks
local set
local polarity
1
0
Input Signals
data
local clock
local reset
local output enable
in
S
Q
R
register
Output
output enable

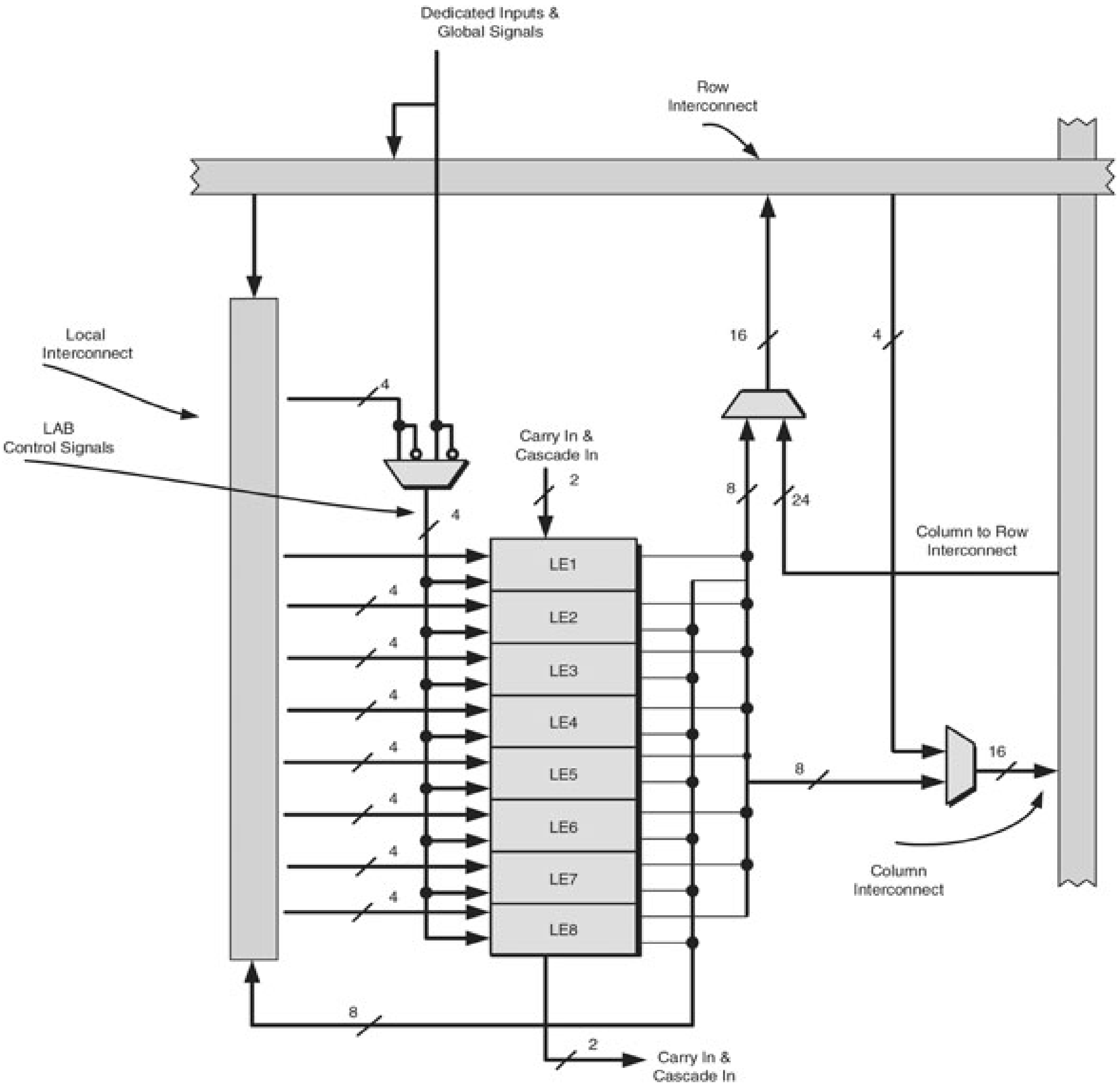
Altera FPGA


I/O Element (IOE)
IOE
Embedded Array Block (EAB)
Logic Array
Column Interconnect
EAB
Logic Array Block (LAB)
Row Interconnect
Logic Element
Logic Array
Embedded Array
Local Interconnect

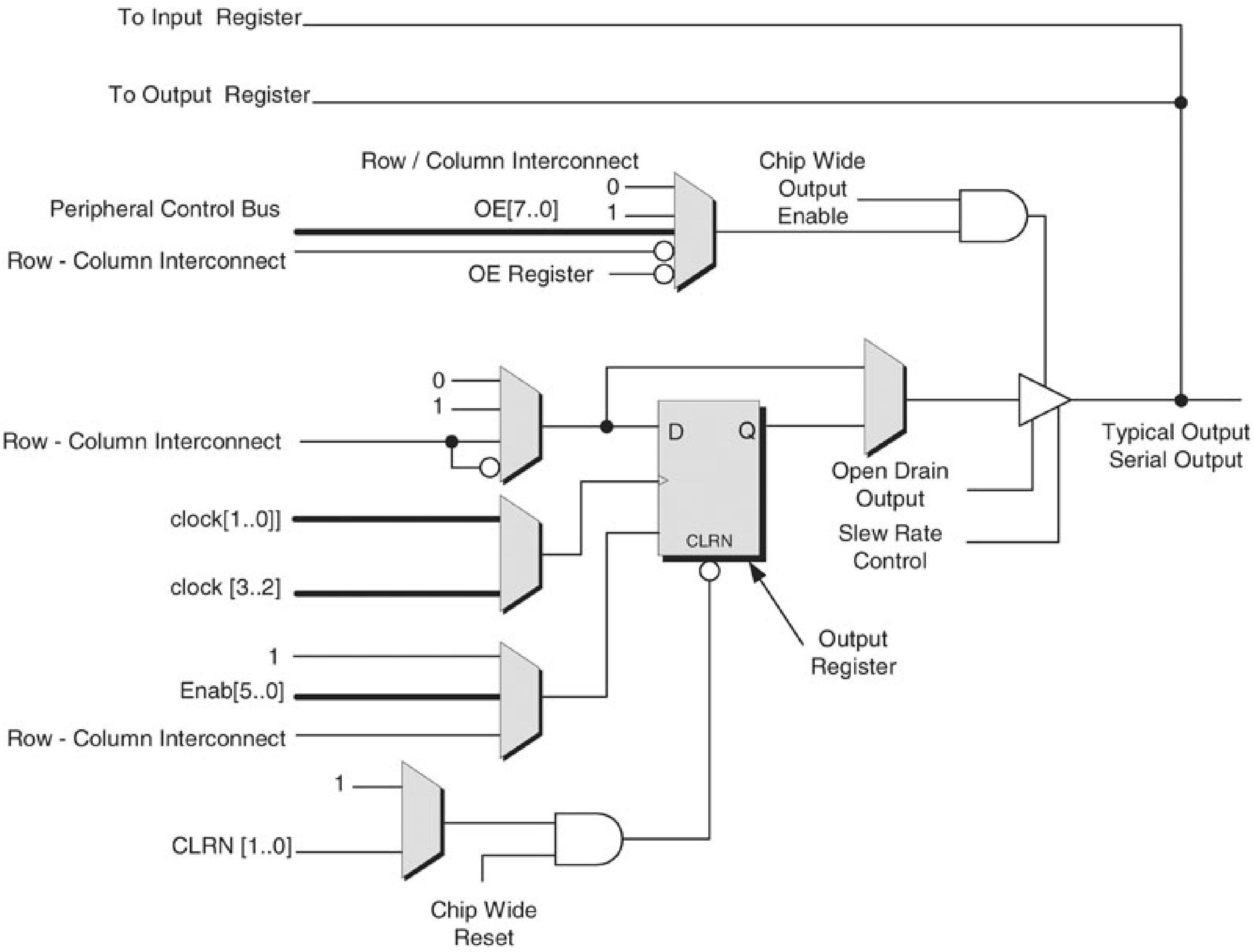
Altera FPGA


Dedicated Inputs and Global Signals
Chip Wide Reset
Row Interconnect
6
2, 4, 8, 16
8, 4, 2, 1
D
S
Q
R
Data In
24
8, 9, 10, 11
Address
2, 4, 8, 16
Column Interconnect
WE
RAM / ROM
256 x 8
512 x 4
1024 x 2
2048 x 1
EAB Local Interconnect

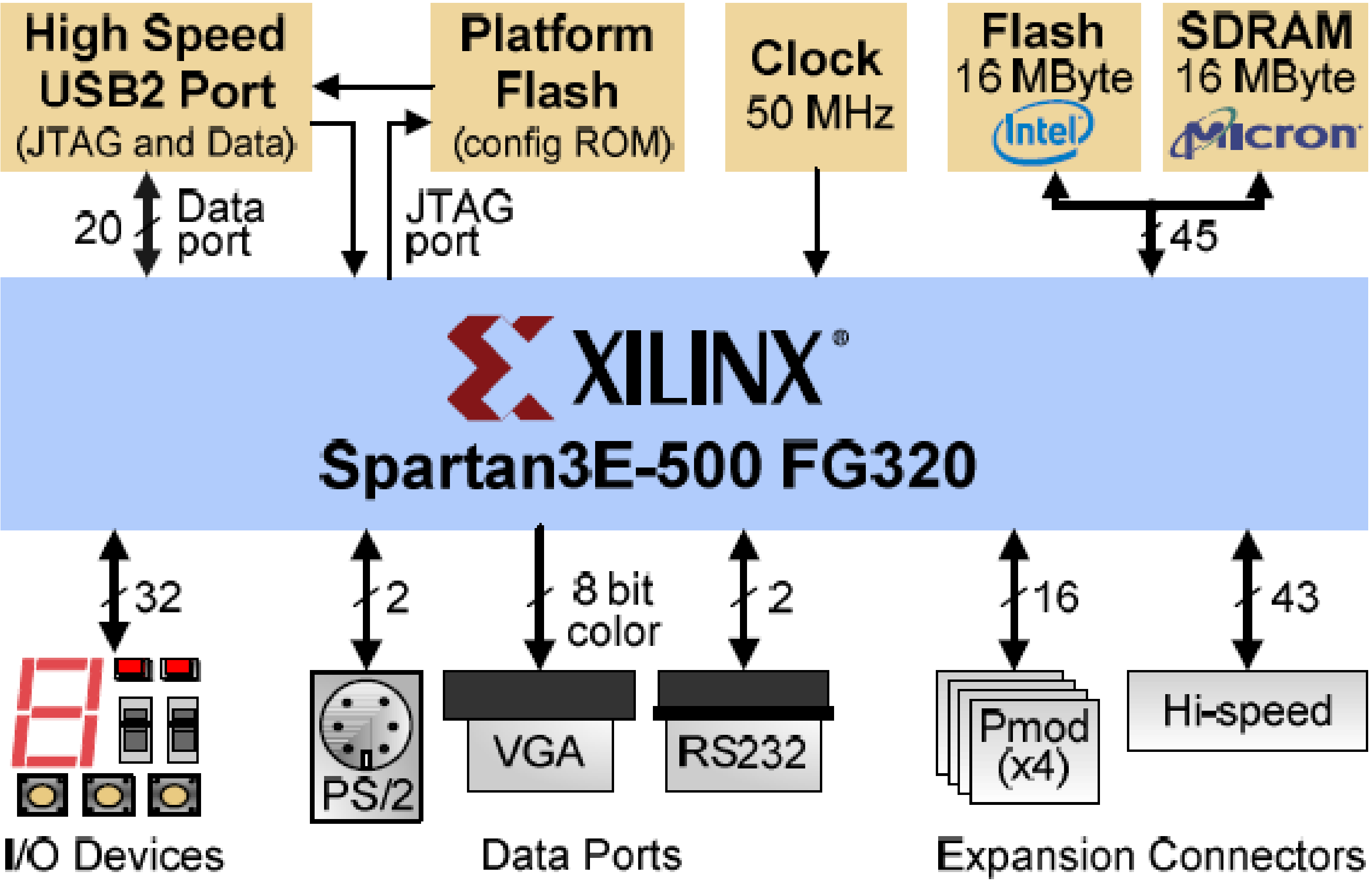
Altera FPGA


Dedicated Inputs &
Global Signals
Row
Interconnect
Local
Interconnect
LAB
Control Signals
Carry In &
Cascade In
2
4
4
16
4
8
24
Column to Row
Interconnect
LE1
LE2
LE3
LE4
LE5
LE6
LE7
LE8
4
4
4
4
4
4
4
8
16
Column
Interconnect
8
2
Carry In &
Cascade In

Altera FPGA


To Input Register
To Output Register
Row / Column Interconnect
Peripheral Control Bus
OE[7..0]
0
1
Row - Column Interconnect
OE Register
Chip Wide Output Enable
0
1
Row - Column Interconnect
D
Q
CLRN
clock[1..0]]
clock [3..2]
Open Drain Output
Slew Rate Control
Typical Output Serial Output
1
Enab[5..0]
Row - Column Interconnect
Output Register
1
CLRN [1..0]
Chip Wide Reset

Xilinx FPGA na płycie Nexys2



Xilinx FPGA na płycie Nexys2

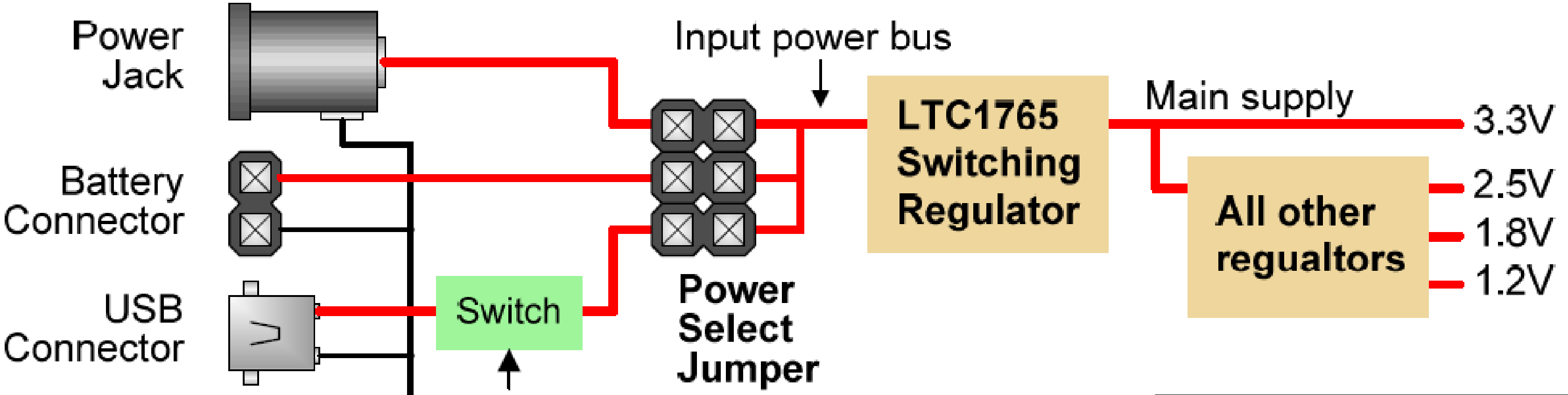
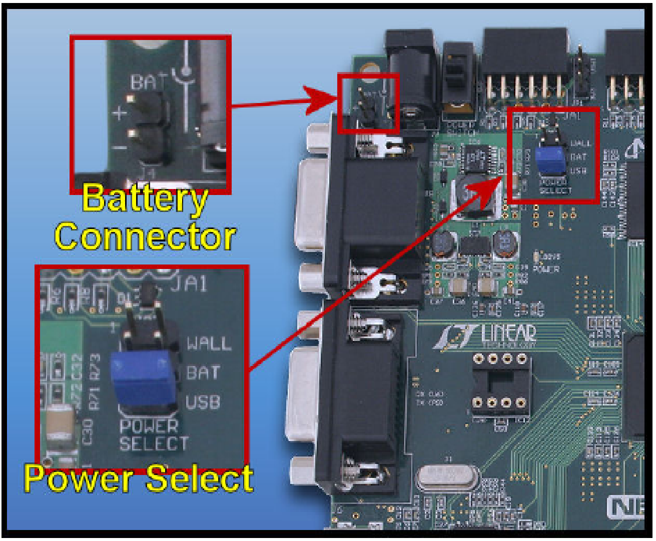
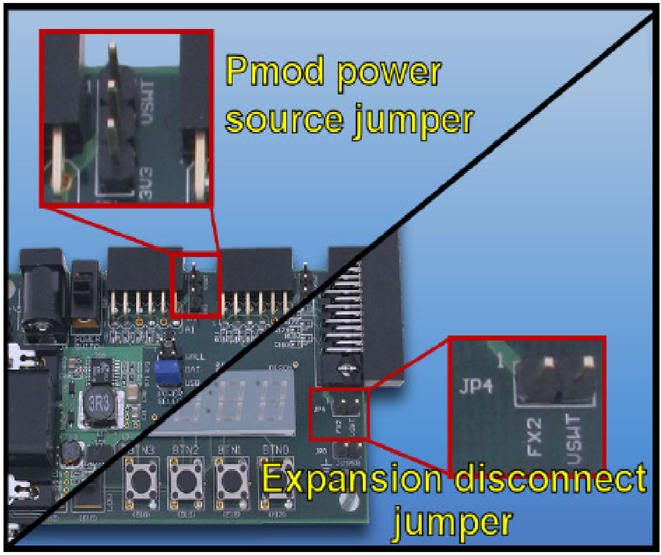
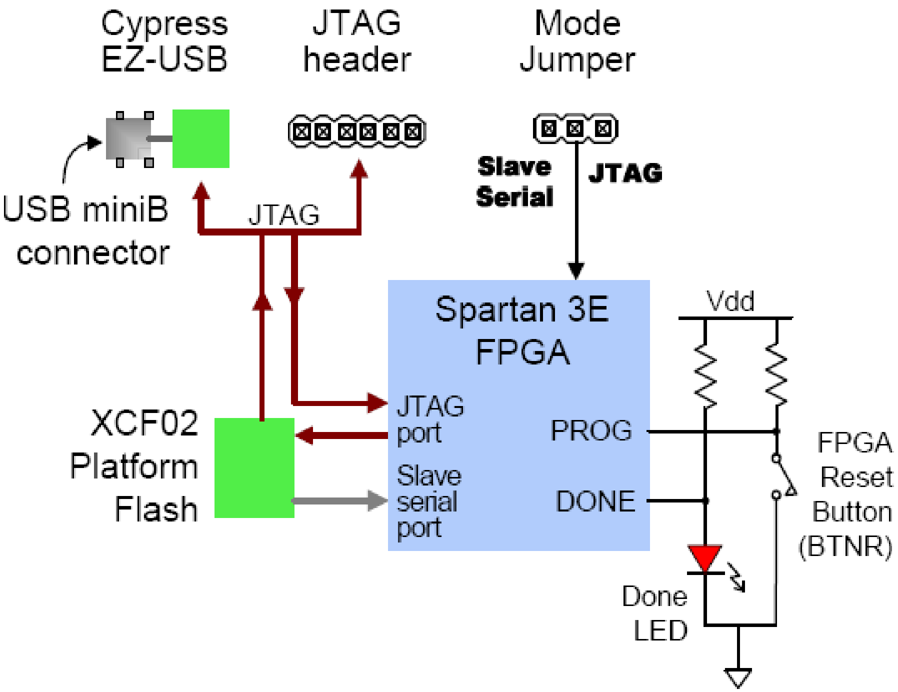


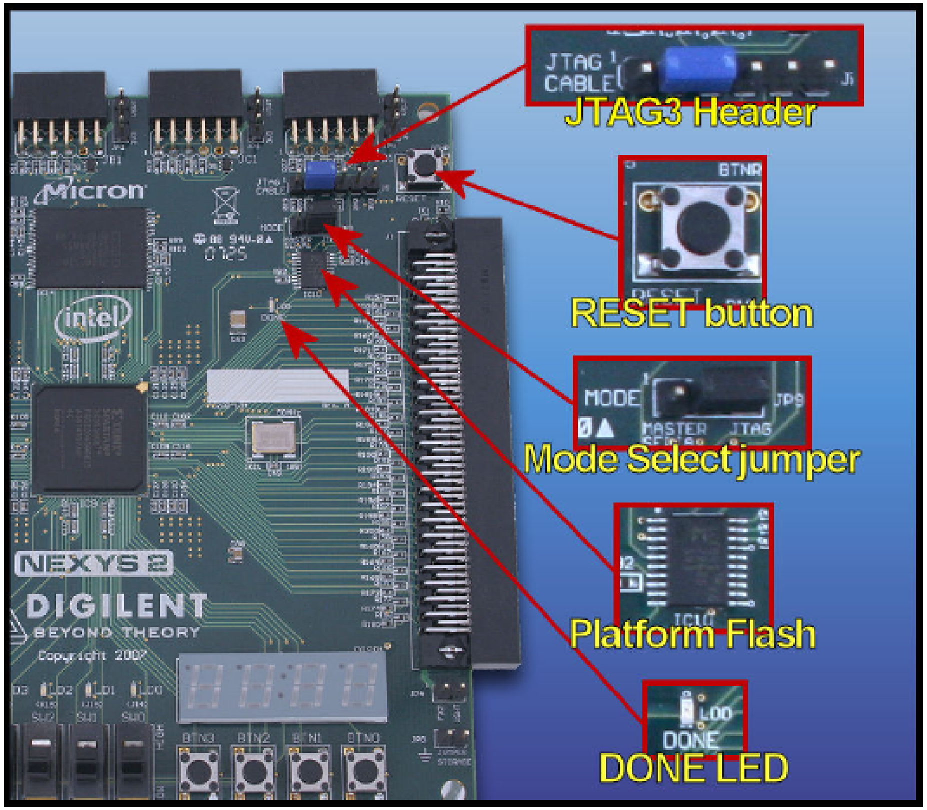
Table 1: Nexys2 Power Supplies		
Supply	Device	Amps (max/typ
3.3V main	IC6: LTC1765	3A/100mA
2.5V FPGA	IC7: LTC3417	1.4A/50mA
1.2V FPGA	IC7: LTC3417	1.4A/200mA
1.8V SRAM	IC8: LTC1844	150mA/90mA
3.3V USB	IC5: LTC1844	150mA/60mA



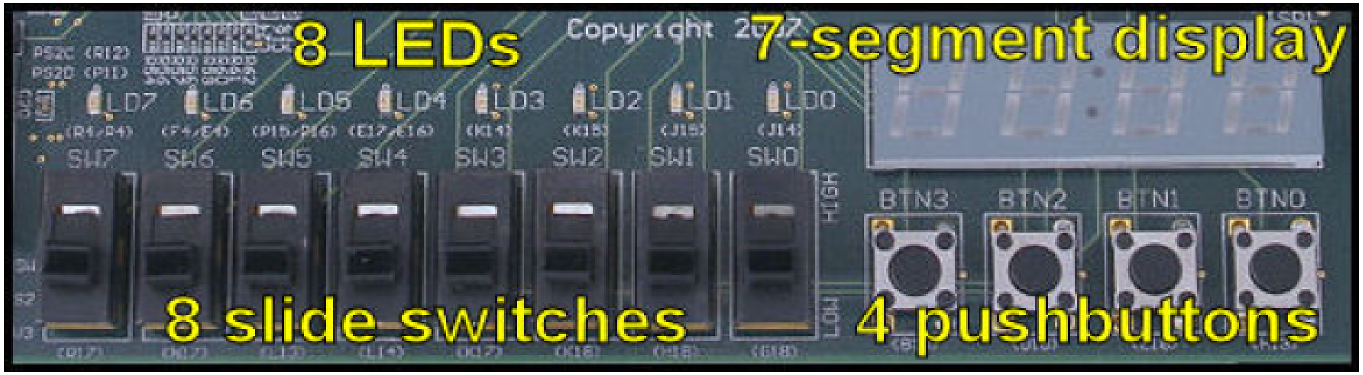


Xilinx FPGA na płycie Nexys2

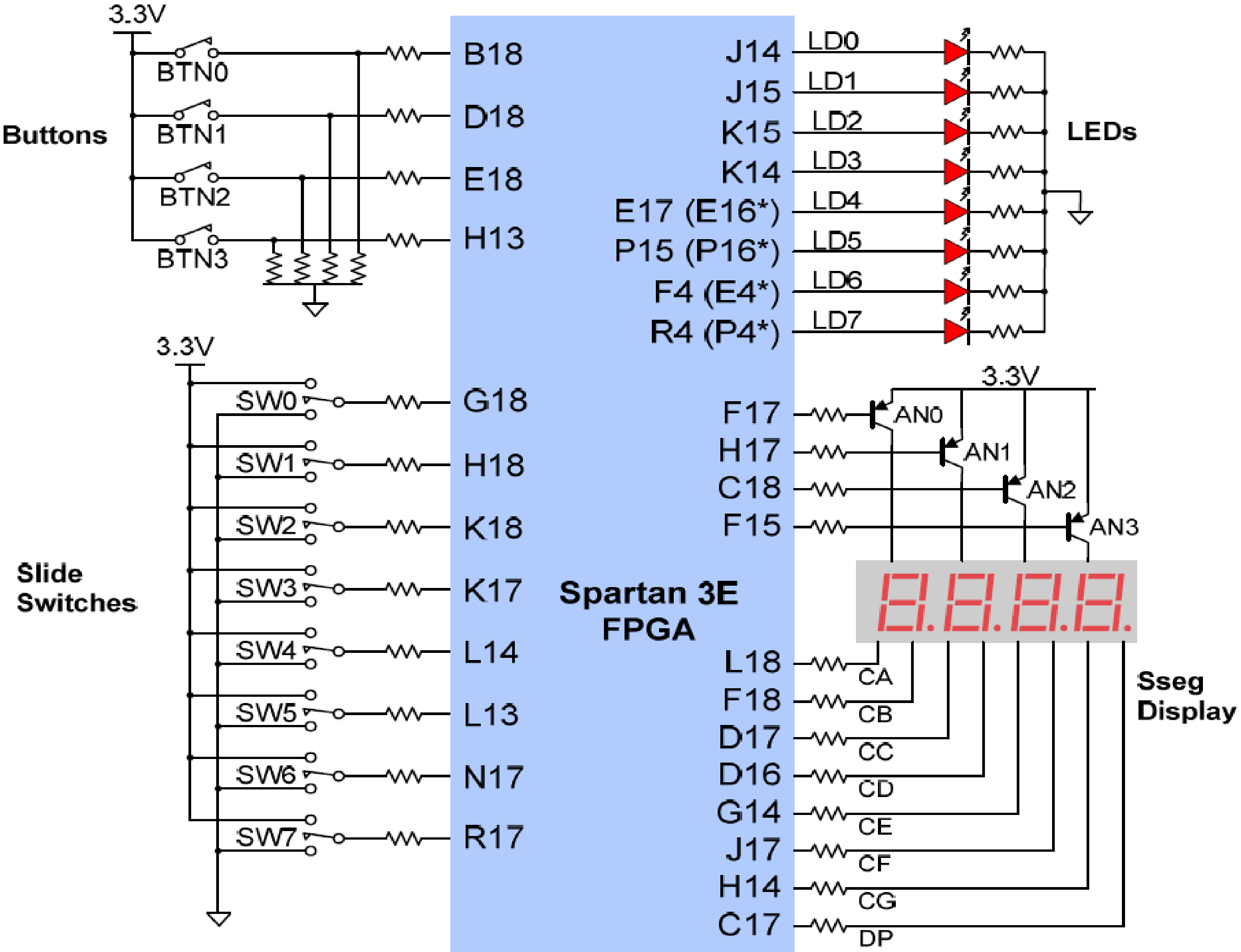




Xilinx FPGA na płycie Nexys2



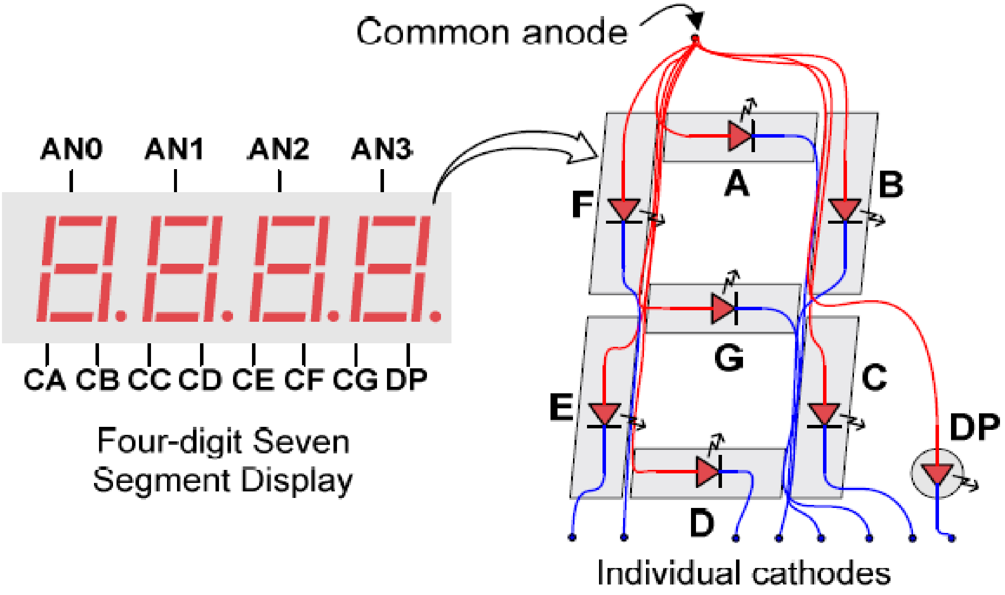
Xilinx FPGA na płycie Nexys2

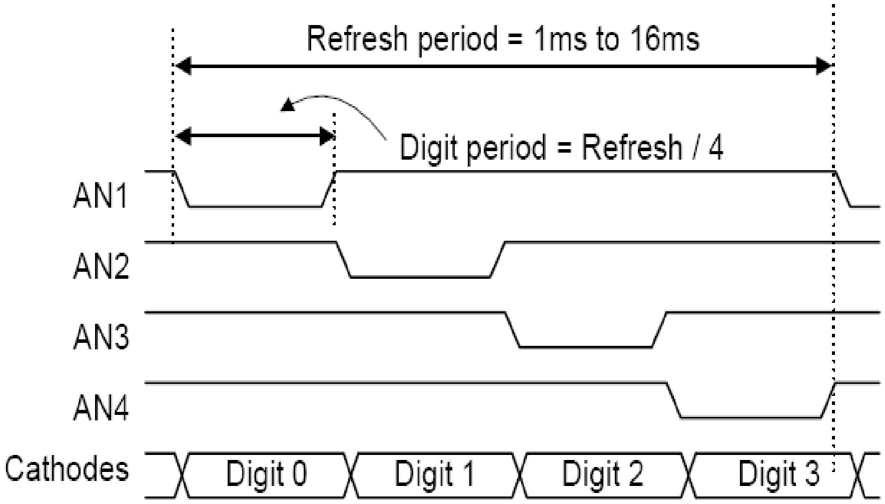


Digital D
Copyright © 2006
Frank Vahid

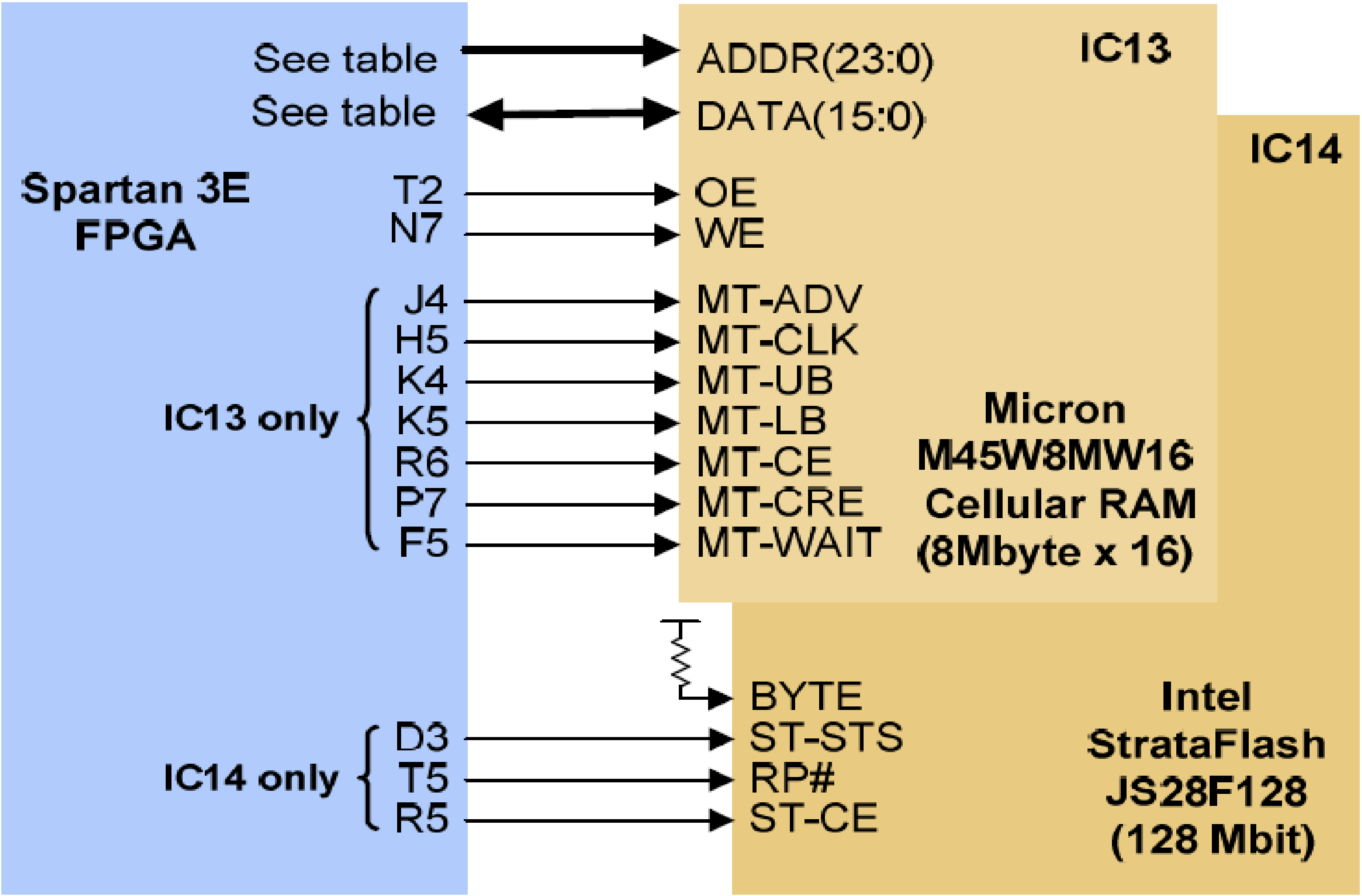
Embedded Systems, © 2008, James K. Peckol

Xilinx FPGA na płycie Nexys2

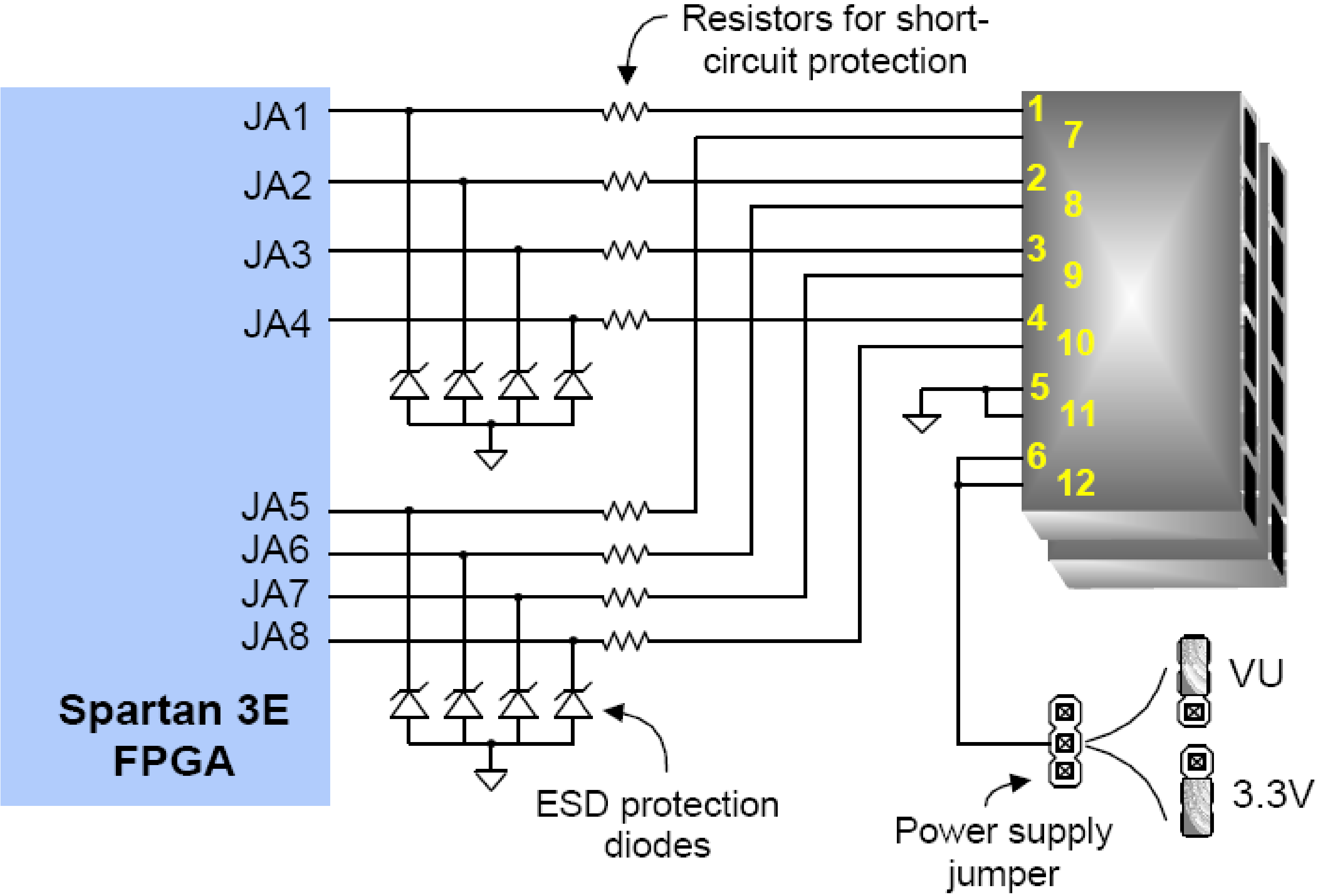




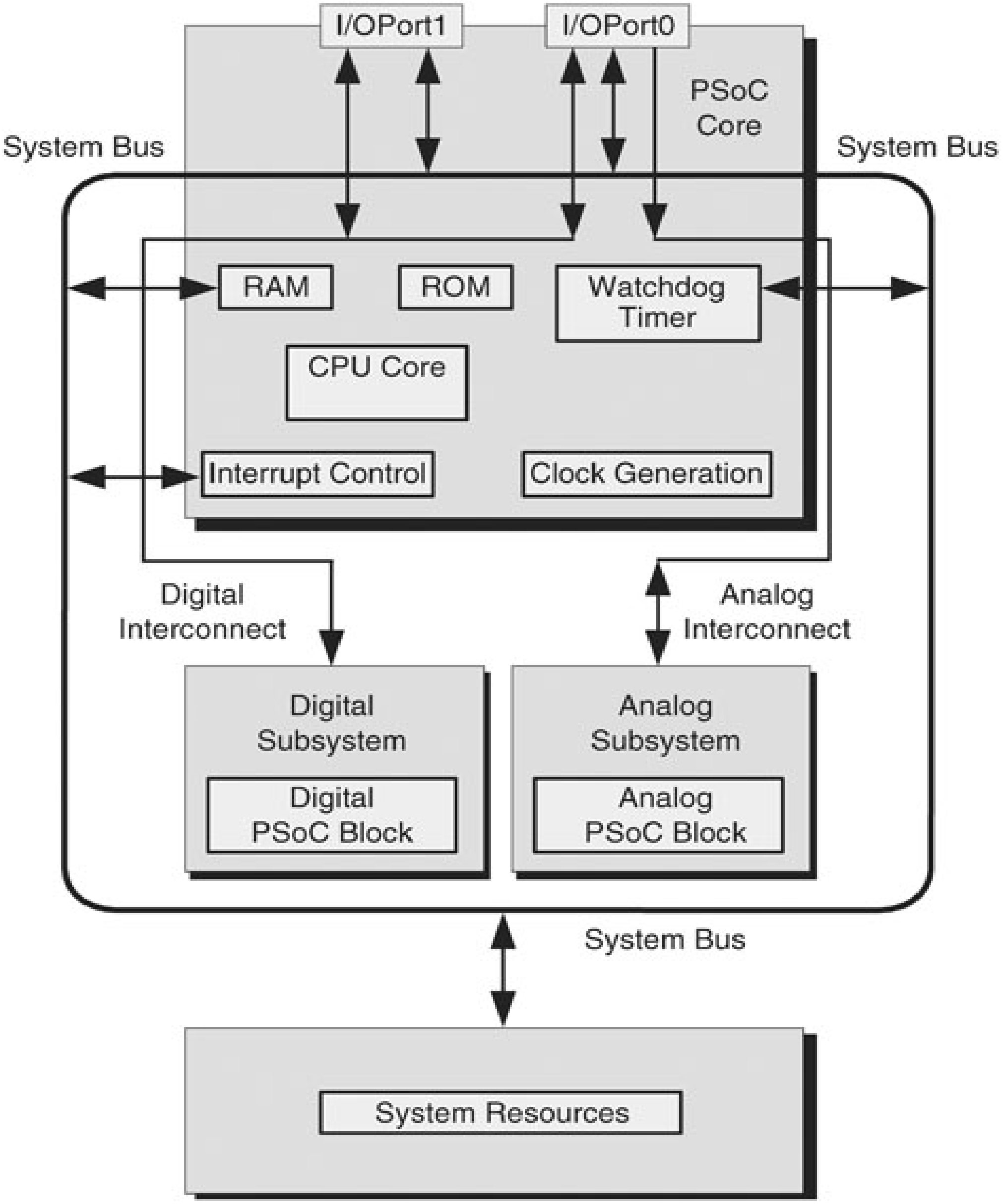
Xilinx FPGA na płycie Nexys2



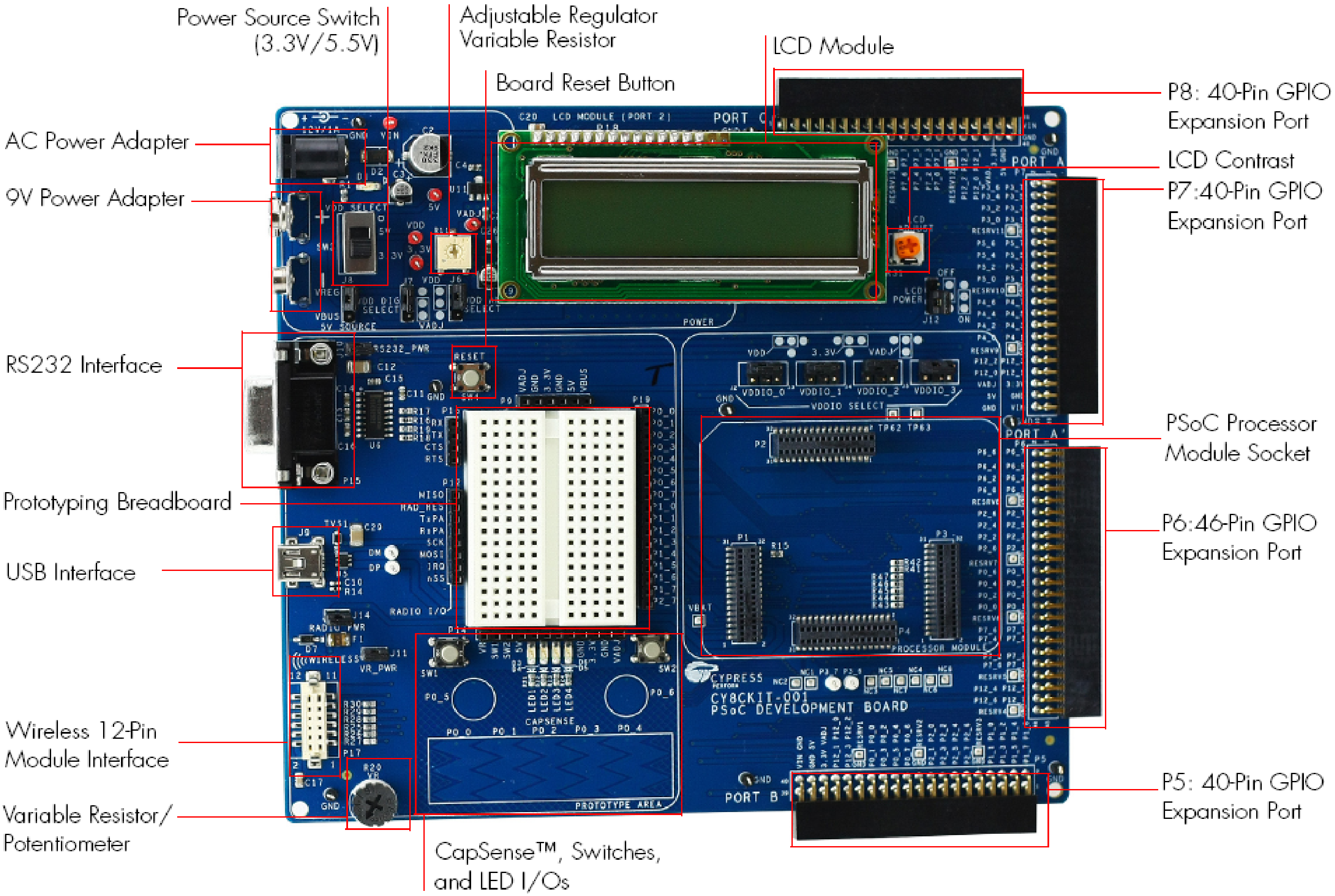
Xilinx FPGA na płycie Nexys2



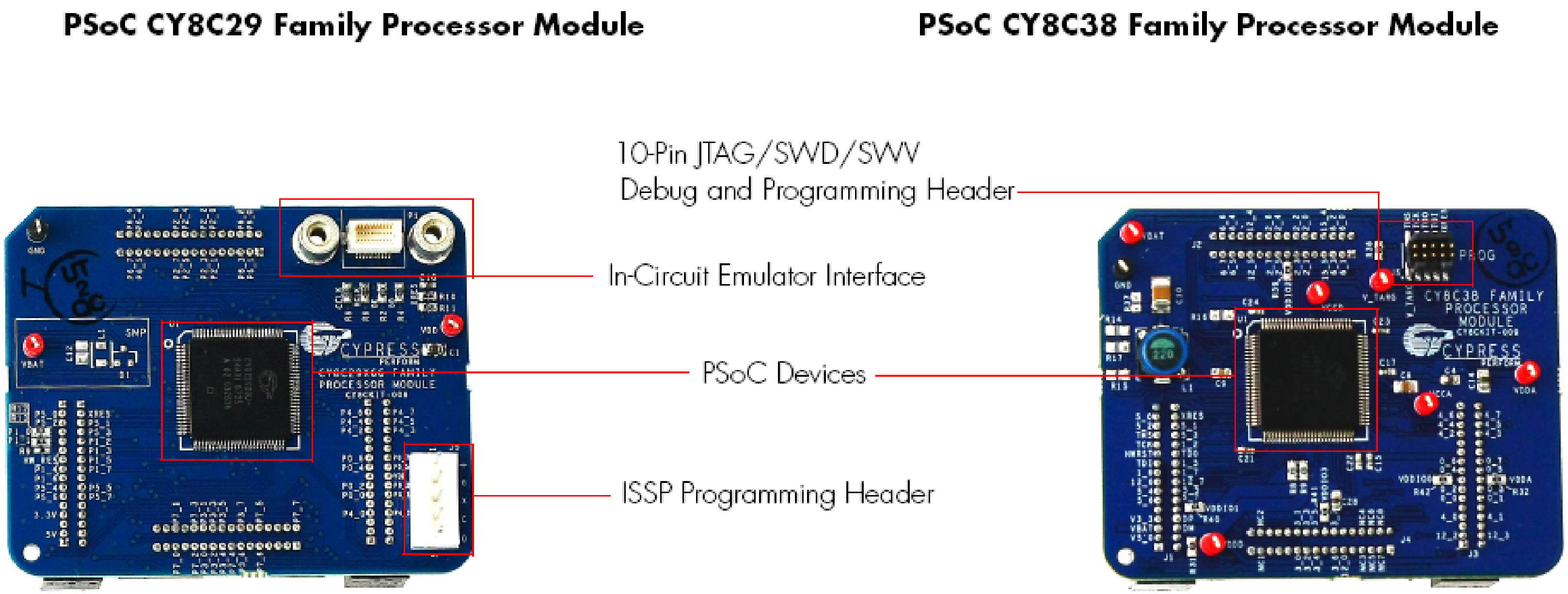
Cypress PSoC



Cypress PSoC - zestaw uruchomieniowy


Power Source Switch (3.3V/5.5V)
Adjustable Regulator Variable Resistor
Board Reset Button
LCD Module
P8: 40-Pin GPIO Expansion Port
LCD Contrast
P7:40-Pin GPIO Expansion Port
AC Power Adapter
9V Power Adapter
RS232 Interface
Prototyping Breadboard
USB Interface
PSoC Processor Module Socket
P6:46-Pin GPIO Expansion Port
Wireless 12-Pin Module Interface
Variable Resistor/ Potentiometer
CapSense™, Switches, and LED I/Os
P5: 40-Pin GPIO Expansion Port
CYPRESS
CY8CKIT-001
PSoC DEVELOPMENT BOARD

Cypress PSoC - moduły procesorów



PSoC CY8C29 Family Processor Module

PSoC CY8C38 Family Processor Module

10-Pin JTAG/SWD/SWV Debug and Programming Header

In-Circuit Emulator Interface

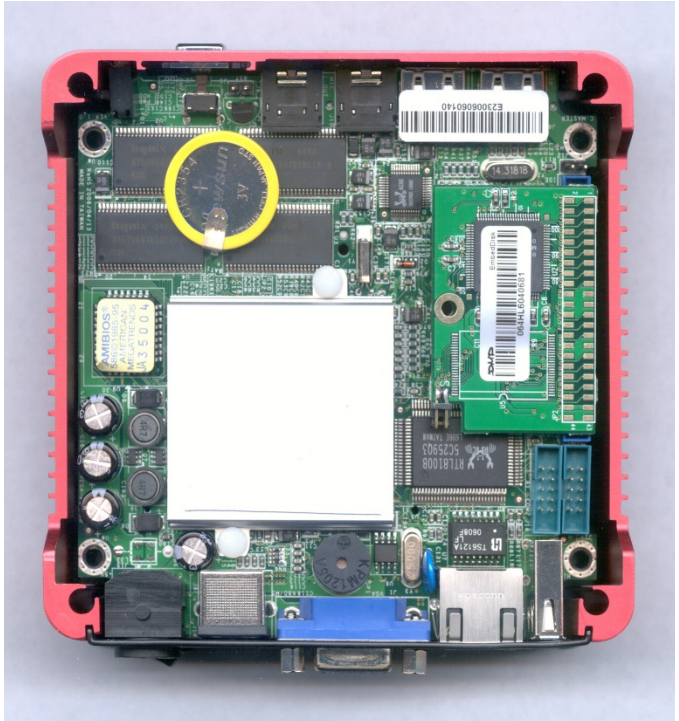
PSoC Devices

ISSP Programming Header

SoC, PSoC, & SoPC Design

- Design a single chip solution
- Recent option for Embedded Systems
- System on a Chip (SoC) – one chip replaces a embedded system board
- System on a Programmable Chip (SoPC) is an SoC that uses a large FPGA
- Intellectual Property (IP) cores are used to provide processors and common hardware for SoPC designs. IP cores are drop in design elements purchased from another vendor

eBox 2300- SoC Embedded PC



PSoC Design

- Programmable System on a Chip (PSoC)
- A small low-cost microcontroller with on chip memory, some programmable logic, and mixed signal hardware (A/D)
- Tools provided to configure hardware and then write software in assembly or C
- Also has a new Labview type development tool
- Limited to a just few K of memory – no OS
- PSoC is a trademark of Cypress
- PSoC II coming soon – 32-bit ARM based?

SoPC Design

- Use a large Field Programmable Gate Array (FPGA) with a processor IP core
- In a Soft IP core logic elements from the FPGA are used to build the processor.
- In a Hard IP core a full custom VLSI layout for the processor is placed in the FPGA
- Soft cores more flexible, but have slower clock rates

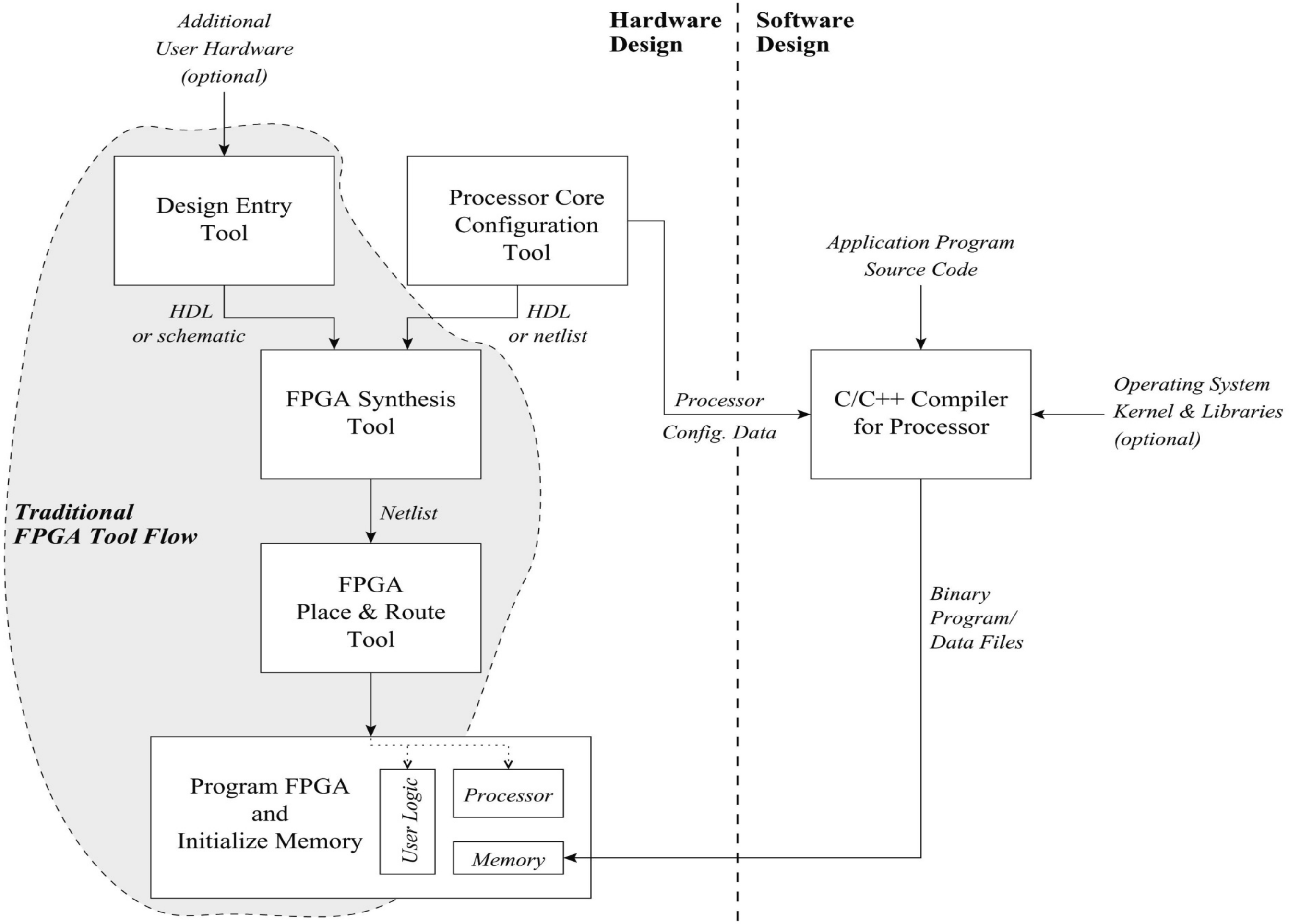
Processor Cores for SoPC

- Soft Processor Cores for FPGAs
  - NIOS II - Altera
  - Microblaze, Picoblaze, 8051 - Xilinx
- Hard Processor Cores for FPGAs
  - ARM Altera
  - PowerPC Xilinx
  - S5 Tensilica Xtensa - Stretch
- Largest FPGAs can support several processors on one chip
- Cost as low as $.35 per processor

Software for SoPC Design

- Traditional FPGA tools (VHDL & Verilog)
  - FPGA also used to build additional hardware
- Processor Configuration Tool (Soft Cores)
- C/C++ Compiler for Processor Program Code
- Tools to debug code and load memories
- Optional OS support for Processor
- Board Support Package (BSP) provides OS I/O device drivers for a specific board design

FPGA-based SOPC CAD Tool Flow



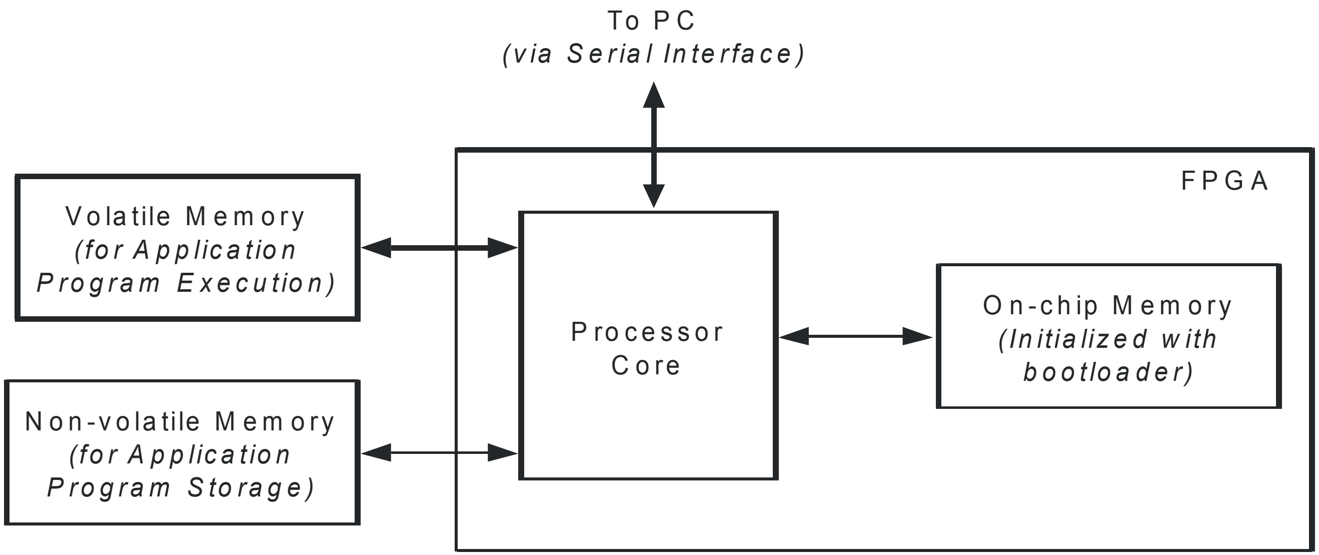
FPGA-based SOPC

FEATURES OF COMMERCIAL SOFT PROCESSOR CORES

Feature	**Nios 3.1**	**MicroBlaze 3.2**
Datapath	16 or 32 bits	32 bits
Pipeline Stages	5	3
Frequency	up to 150 MHz	up to 150 MHz
Gate Count	26,000–40,000	30,000–40,000
Register File	up to 512 (window size: 32)	32 general purpose and 32 special purpose
Instruction Word	16 bits	32 bits
Instruction Cache	Optional	Optional
Hardware Multiplier	Optional	Optional

This clock speed is not achievable on all devices. Some devices limit the maximum frequency to as low as 50 MHz.

SOPC Memory Organization

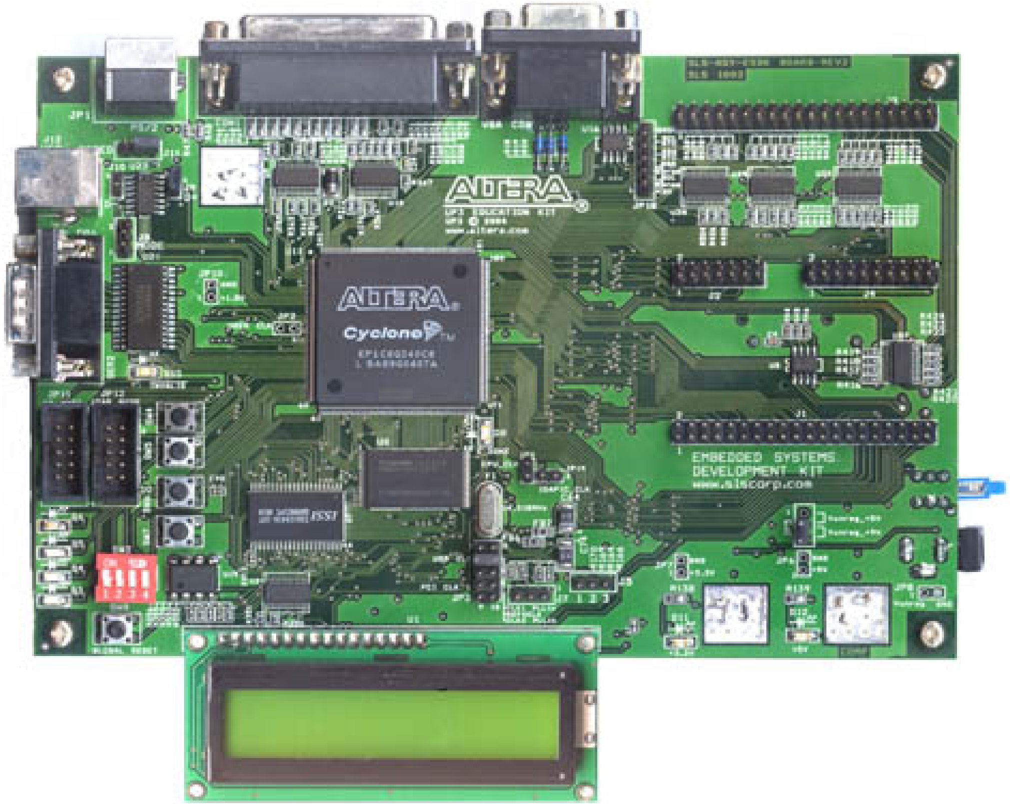


Non-volatile (Flash) Memory is used both to boot processor software at power on and to configure the FPGA hardware

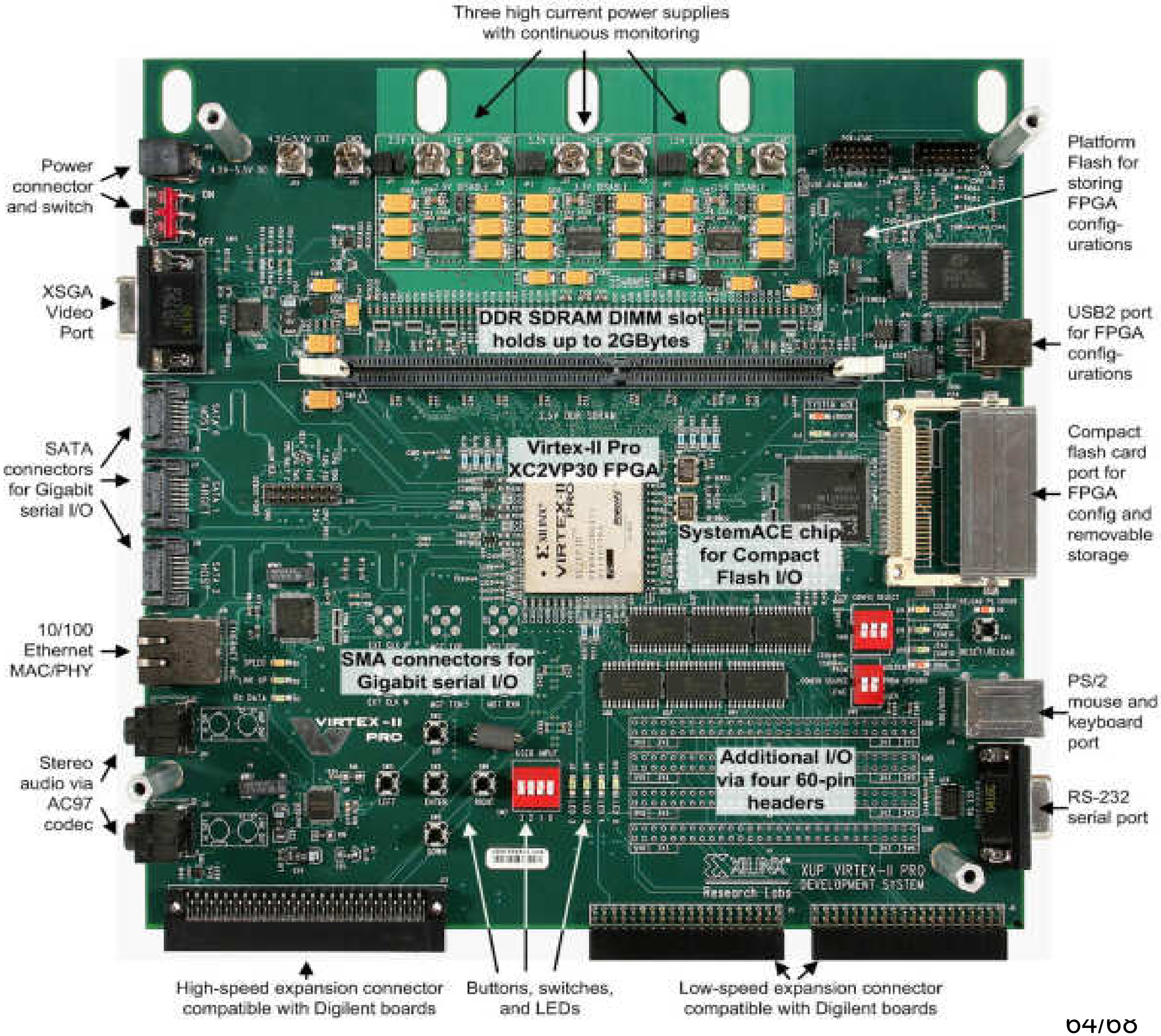
Altera SoPC

Altera's SoPC UP 3 board contains a 200,000 gate FPGA with Flash and SRAM memory.

It can run a soft Nios II RISC processor IP core



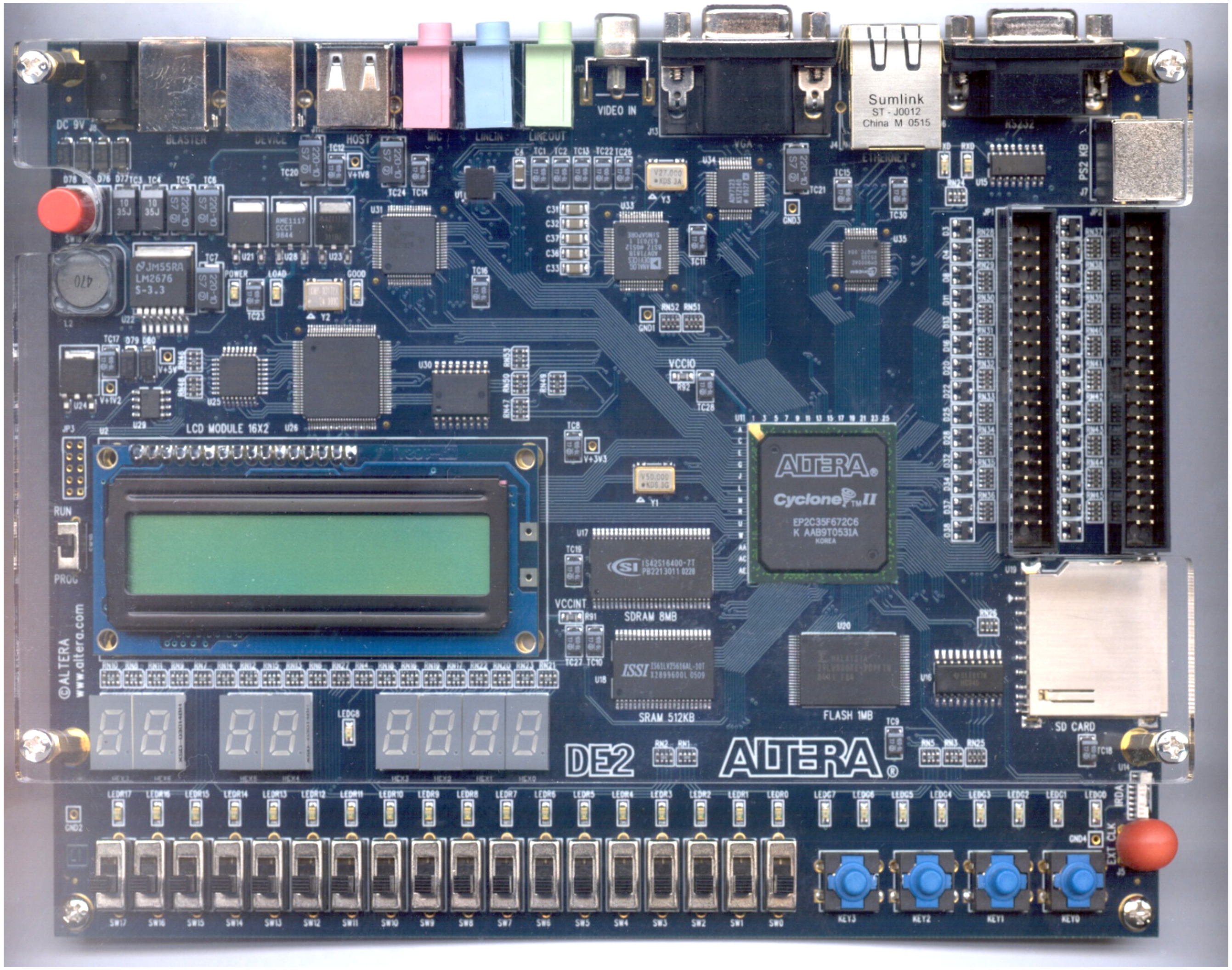
Xilinx SoPC FPGA Board


Three high current power supplies with continuous monitoring
Power connector and switch
XSGA Video Port
SATA connectors for Gigabit serial I/O
10/100 Ethernet MAC/PHY
Stereo audio via AC97 codec
DDR SDRAM DIMM slot holds up to 2GBytes
Virtex-II Pro XC2VP30 FPGA
SystemACE chip for Compact Flash I/O
SMA connectors for Gigabit serial I/O
Additional I/O via four 60-pin headers
Platform Flash for storing FPGA config-urations
USB2 port for FPGA config-urations
Compact flash card port for FPGA config and removable storage
PS/2 mouse and keyboard port
RS-232 serial port
High-speed expansion connector compatible with Digilent boards
Buttons, switches, and LEDs
Low-speed expansion connector compatible with Digilent boards
XILINX XUP VIRTEX-II PRO DEVELOPMENT SYSTEM
Research Labs

Xilinx SoPC FPGA Features

- Xilinx Virtex-2 Pro FPGA with 3M Logic Gates, 136 18-bit multipliers for DSP applications, 2,448Kb of internal SRAM, and two PowerPC Microprocessors
- DDR SDRAM DIMM that can accept up to 2Gbytes of RAM
- 10/100 Ethernet port & USB 2.0 port
- Non-volatile Flash memory & Compact Flash card slot
- VGA Video port & PC Audio Codec
- Serial ATA, PS/2, & RS-232 ports
- High and Low Speed I/O expansion connectors with a large collection of available expansion boards
- Low cost: Board is $299 for Universities
- Xilinx SOPC Development tools are free for Universities

Newer Altera DE -2 board – A bit more I/O and larger FPGA with hardware multipliers

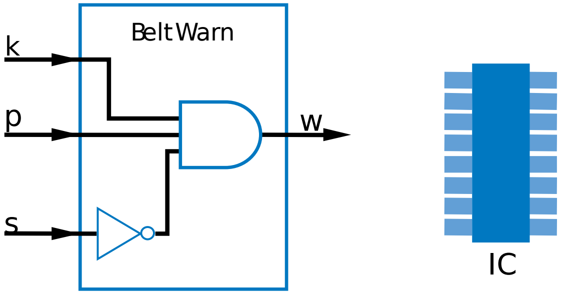


Conclusions

- SoPC is at the leading edge of electronic systems design
- Opportunities exist for innovative design approaches using processors, memory, and programmable FPGA logic
- New approaches and new tools are needed to explore and develop designs

Chapter Summary

- Many ways to get from design to physical implementation
  - Manufactured IC technologies
    - Full-custom IC
      - Decide on every transistor and wire
    - Semi-custom IC
      - Transistor details pre-designed
      - Standard cell: Place cells and wire them
      - Gate array: Just wire existing gates
  - FPGAs
    - Fully programmable
  - Other technologies
    - Logic ICs, PLD, SoC, PSoC, SoPC



(a) Digital circuit design

(b) Physical implementation