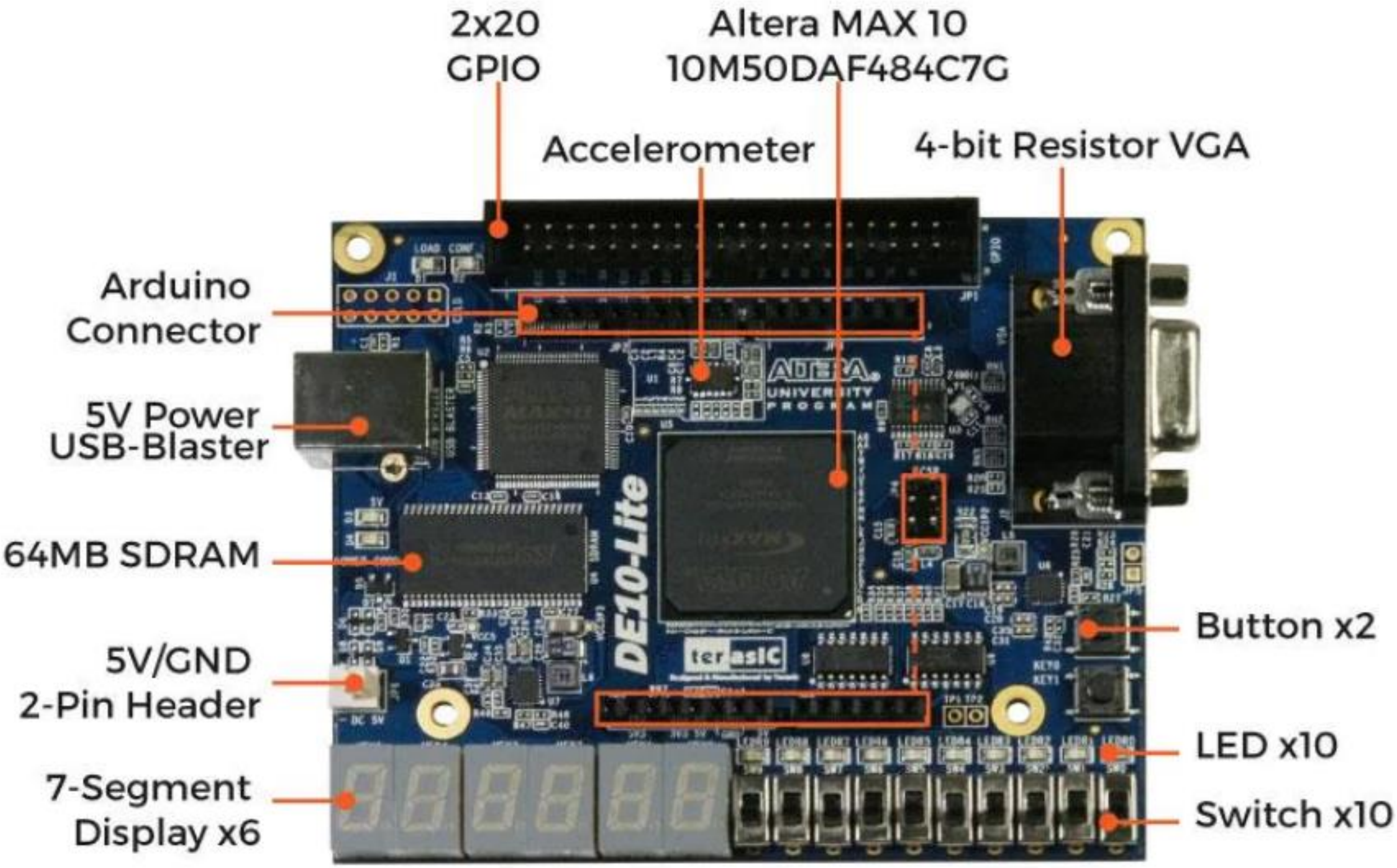
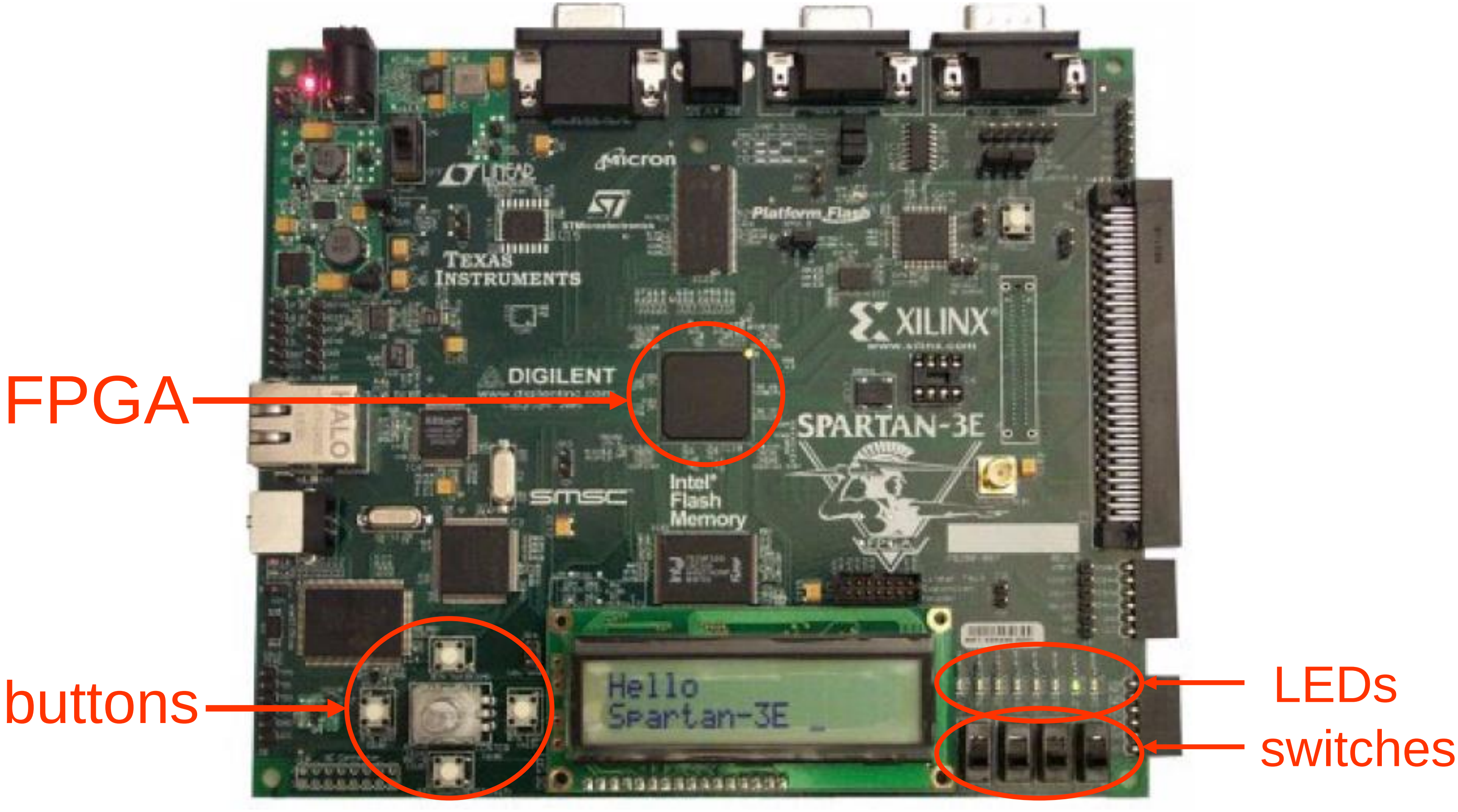
FPGA - Field Programmable Gate Array

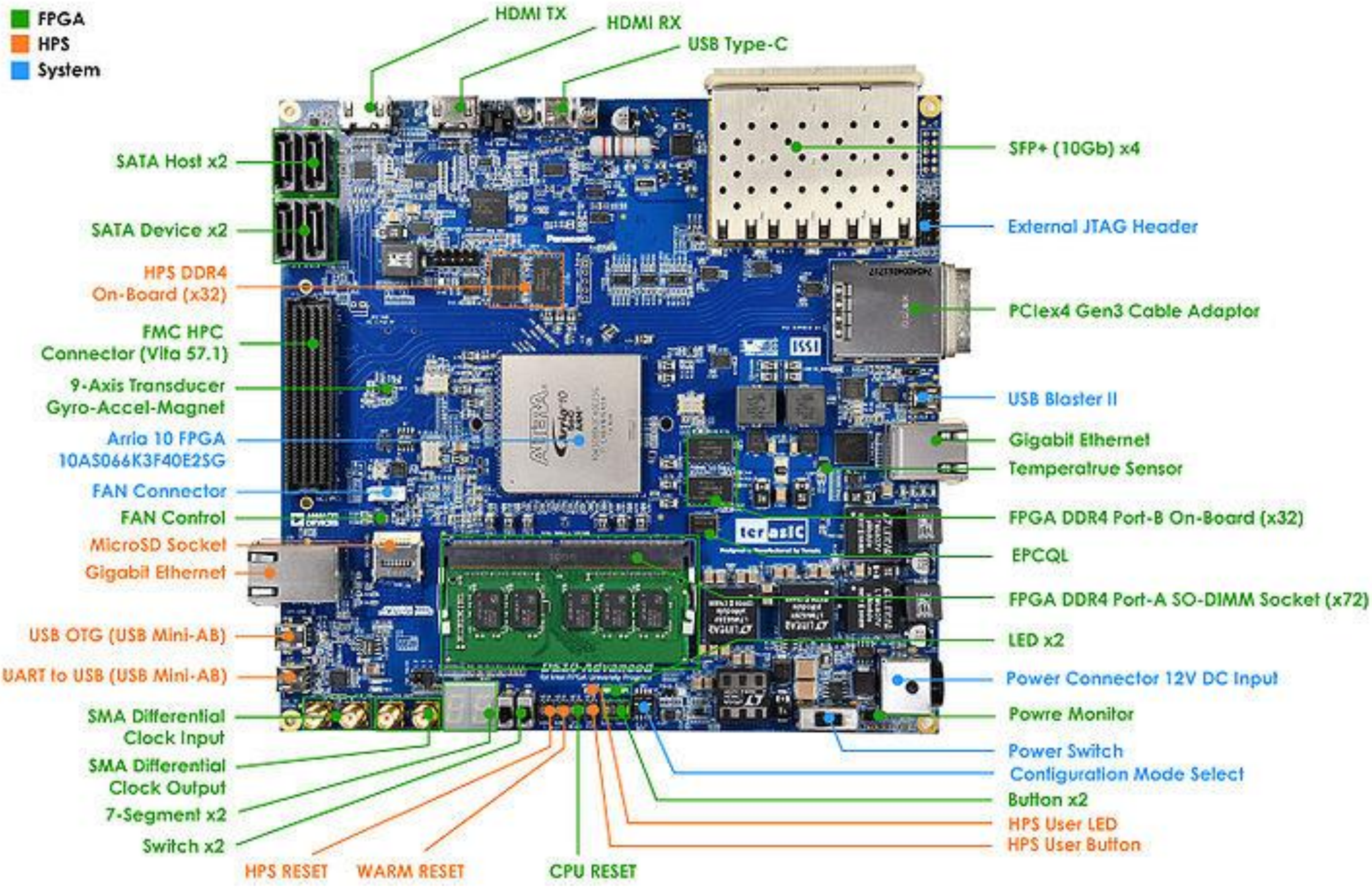
What Are FPGAs

- ***Field-Programmable*** Gate Array
- Can be configured to act like any circuit – More later!
- [illegible]s, b[illegible] [illegible]erat


2x20
GPIO
Altera MAX 10
10M50DAF484C7G
Accelerometer
4-bit Resistor VGA
Arduino
Connector
5V Power
USB-Blaster
64MB SDRAM
5V/GND
2-Pin Header
7-Segment
Display x6
Button x2
LED x10
Switch x10
DE10-Lite
terasIC
ALTERA
UNIVERSITY PROGRAM

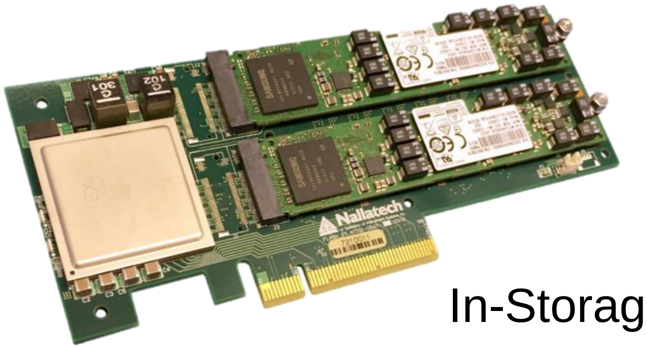
Xilinx Spartan-3E Starter Kit



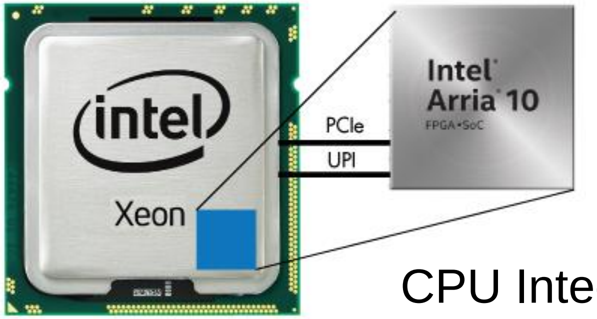

🟩 FPGA
🟧 HPS
🟦 System
HDMI TX
HDMI RX
USB Type-C
SFP+ (10Gb) x4
External JTAG Header
PCIex4 Gen3 Cable Adaptor
USB Blaster II
Gigabit Ethernet
Temperatrue Sensor
FPGA DDR4 Port-B On-Board (x32)
EPCQL
FPGA DDR4 Port-A SO-DIMM Socket (x72)
LED x2
Power Connector 12V DC Input
Powre Monitor
Power Switch
Configuration Mode Select
Button x2
HPS User LED
HPS User Button
SATA Host x2
SATA Device x2
HPS DDR4
On-Board (x32)
FMC HPC
Connector (Vita 57.1)
9-Axis Transducer
Gyro-Accel-Magnet
Arria 10 FPGA
10AS066K3F40E2SG
FAN Connector
FAN Control
MicroSD Socket
Gigabit Ethernet
USB OTG (USB Mini-AB)
UART to USB (USB Mini-AB)
SMA Differential
Clock Input
SMA Differential
Clock Output
7-Segment x2
Switch x2
HPS RESET
WARM RESET
CPU RESET

FPGAs Come In Many Forms

PCIe-Attached



In-Storage



CPU Integrated

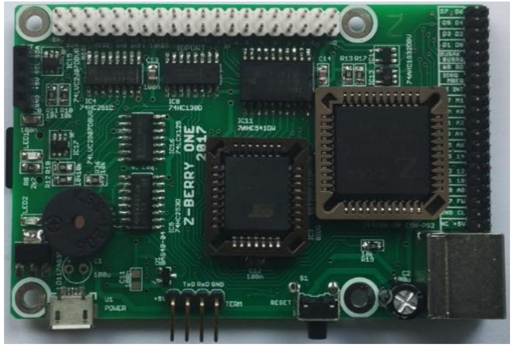
In-Network

How Is It Different From CPU/GPUs

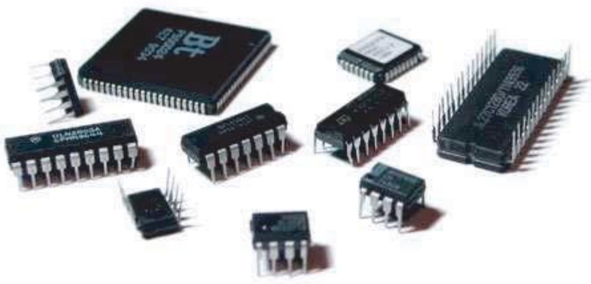
- GPU – The other major accelerator
- CPU/GPU hardware is fixed
  - “General purpose”
  - we write programs (sequence of instructions) for them
- FPGA hardware is not fixed
  - “Special purpose”
  - Hardware can be whatever we want
  - Will our hardware require/support software? Maybe!
- Optimized hardware is very efficient
  - GPU-level performance**
  - 10x power efficiency (300 W vs 30 W)

Analogy

CPU/GPU comes with fixed circuits

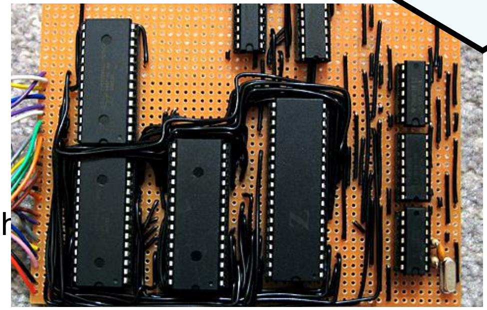


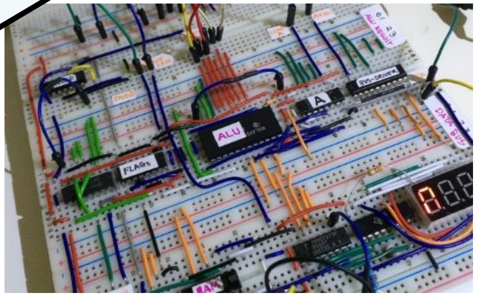
FPGA gives you a big bag of components



To build whatever

Could be a CPU/GPU



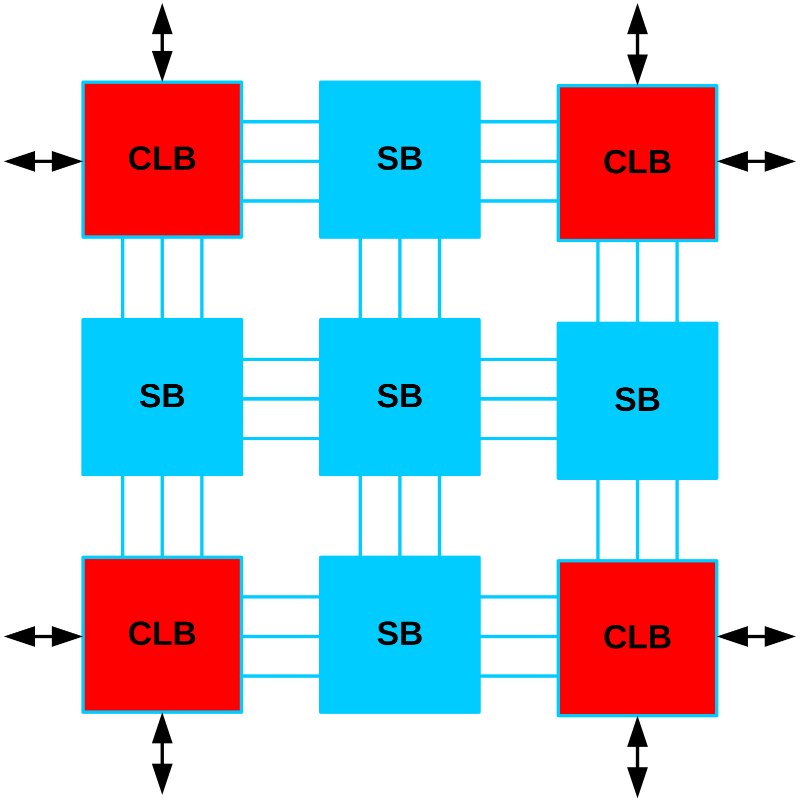


“The Z-Berry”
“Experimental Investigations on Radiation Characteristics of IC Ch
benryves.com “Z80 Computer”
Shadi Soundation: Homebrew 4 bit CPU

FPGA Principles

- A Field-Programmable Gate Array (FPGA) is an integrated circuit that can be configured by the user to emulate any digital circuit as long as there are enough resources
- An FPGA can be seen as an array of Configurable Logic Blocks (CLBs) connected through programmable interconnect (Switch Boxes)

FPGA structure

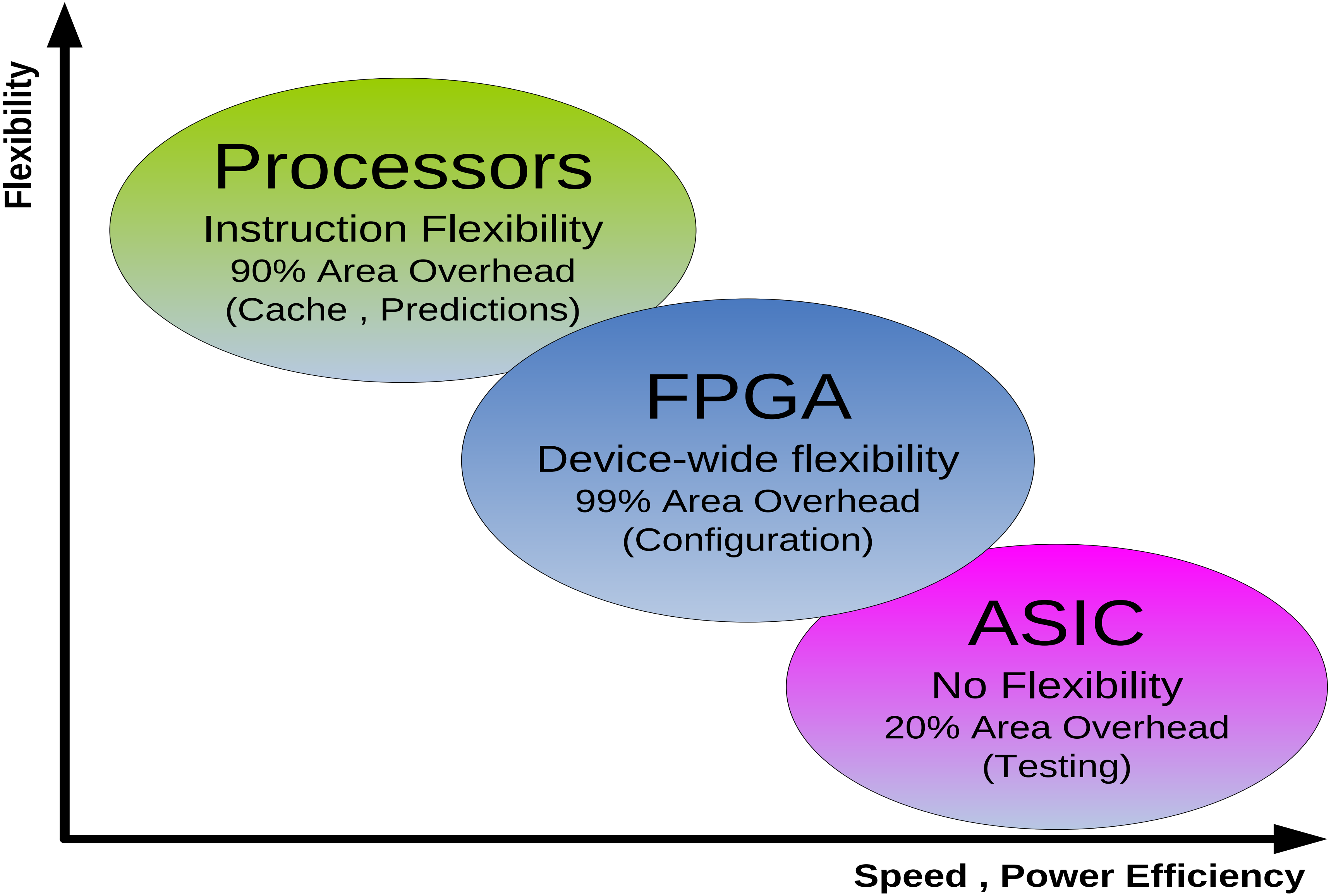


🟥 Configurable Logic Blocks

🟦 Interconnection Network

⬛ I/O Signals (Pins)

Comparison



Processors

Instruction Flexibility

90% Area Overhead

(Cache , Predictions)

FPGA

Device-wide flexibility

99% Area Overhead

(Configuration)

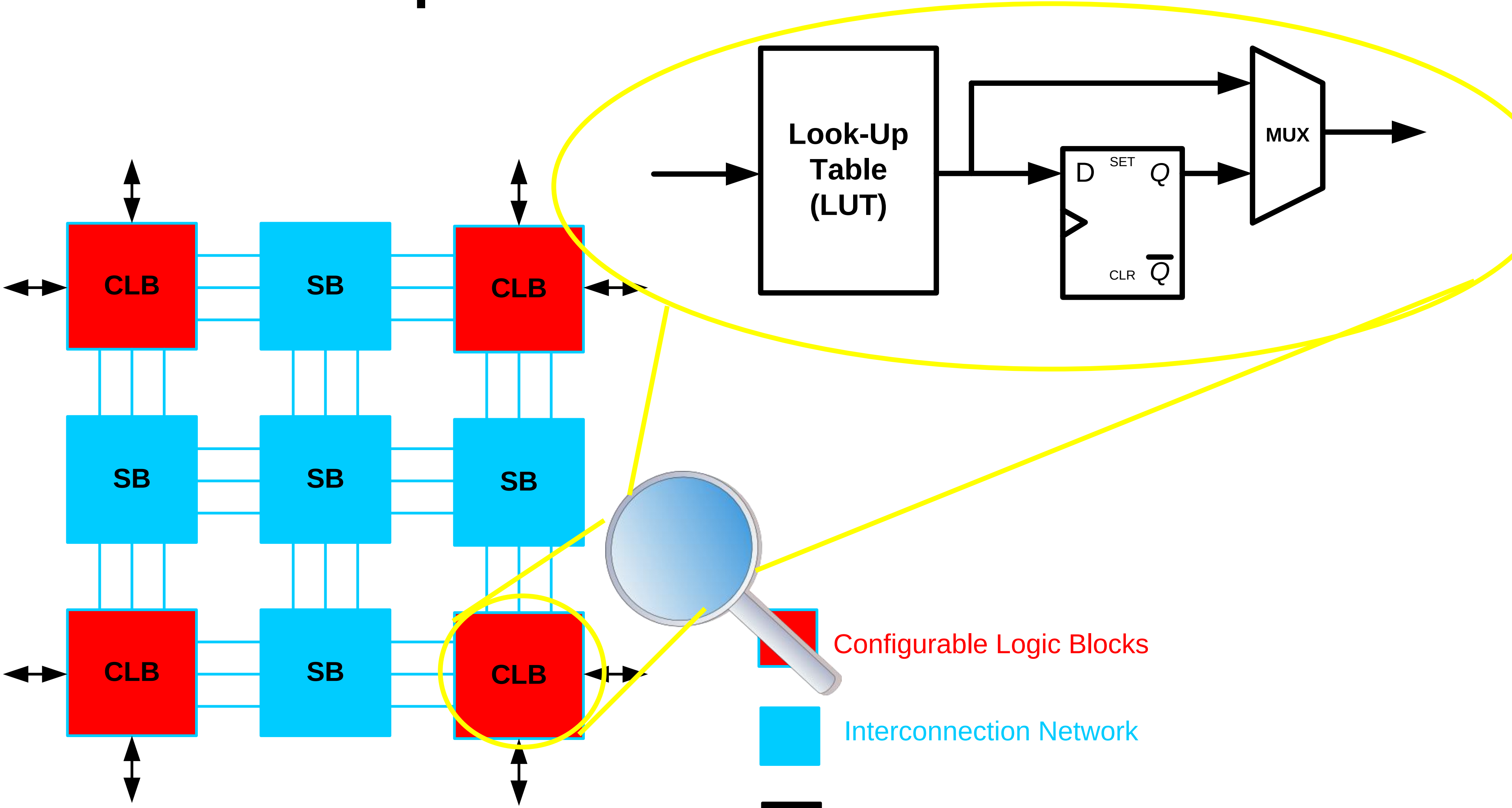
ASIC

No Flexibility

20% Area Overhead

(Testing)

Simplified CLB Structure

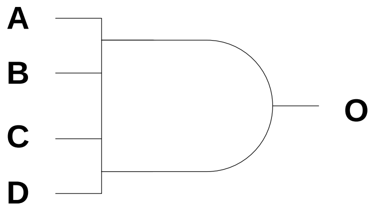


🟥 Configurable Logic Blocks

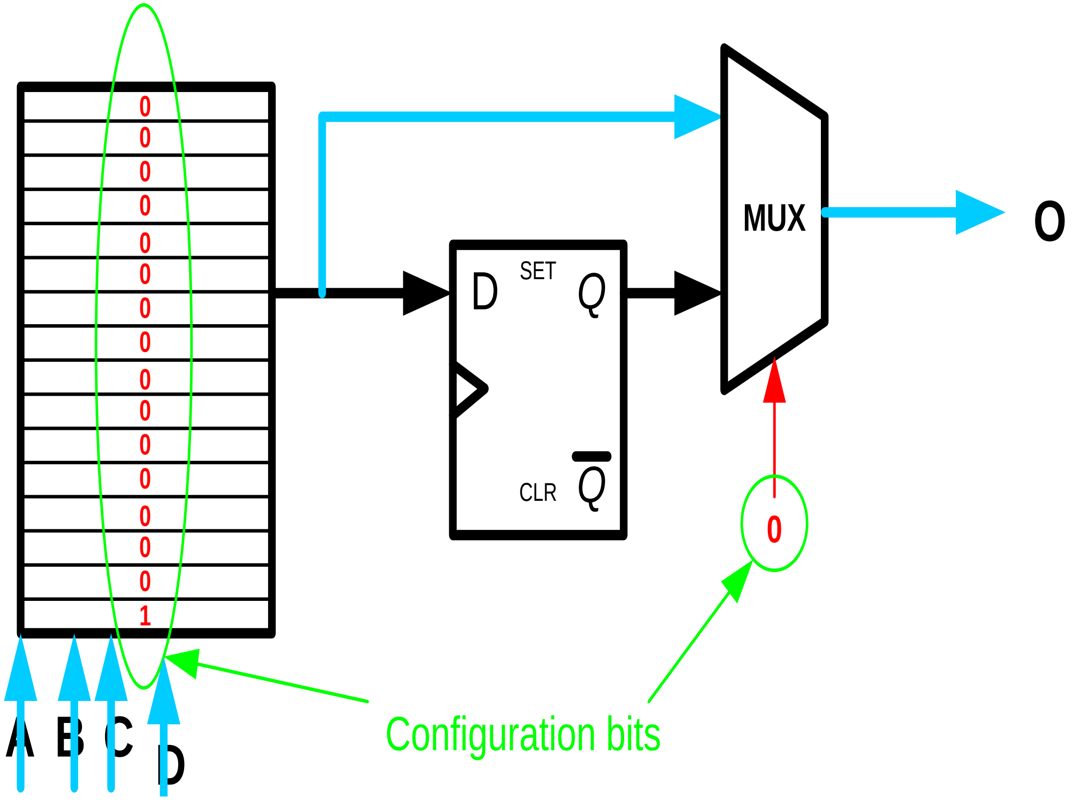
🟦 Interconnection Network

⬛ I/O Signals (Pins)

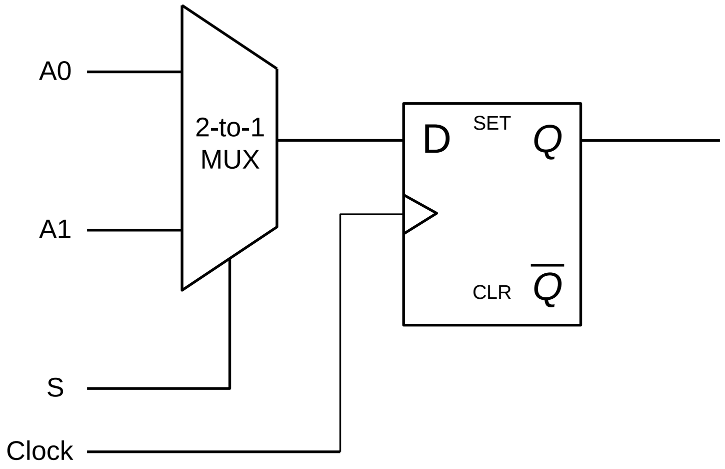
Example: 4-input AND gate



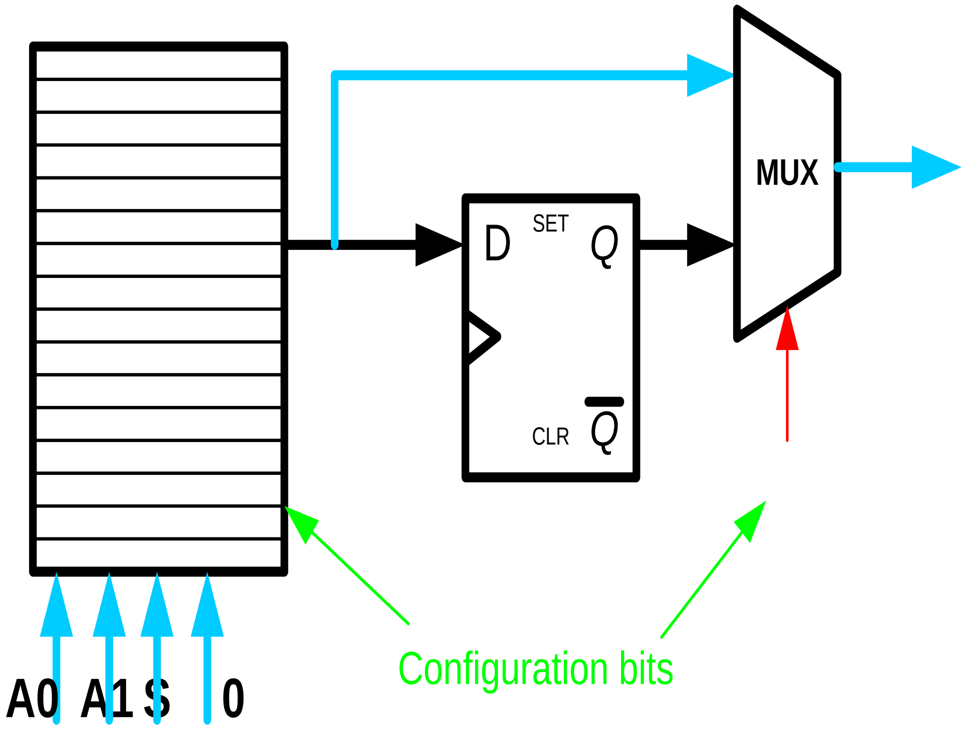
A	B	C	D	O
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1



Example 2: Find the configuration bits for the following circuit

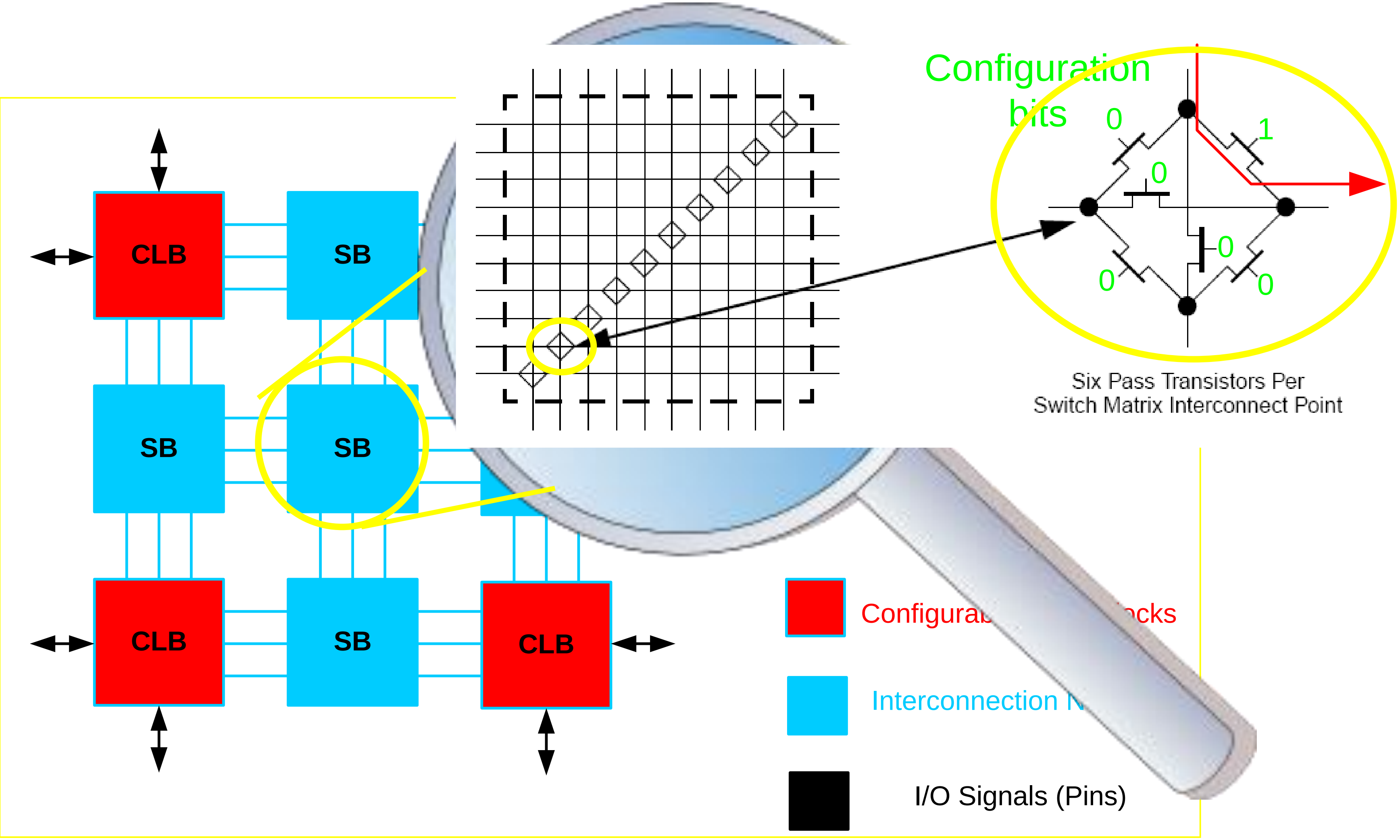


A0	A1	S	
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	



Configuration bits

Interconnection Network



Configuration bits

Six Pass Transistors Per Switch Matrix Interconnect Point

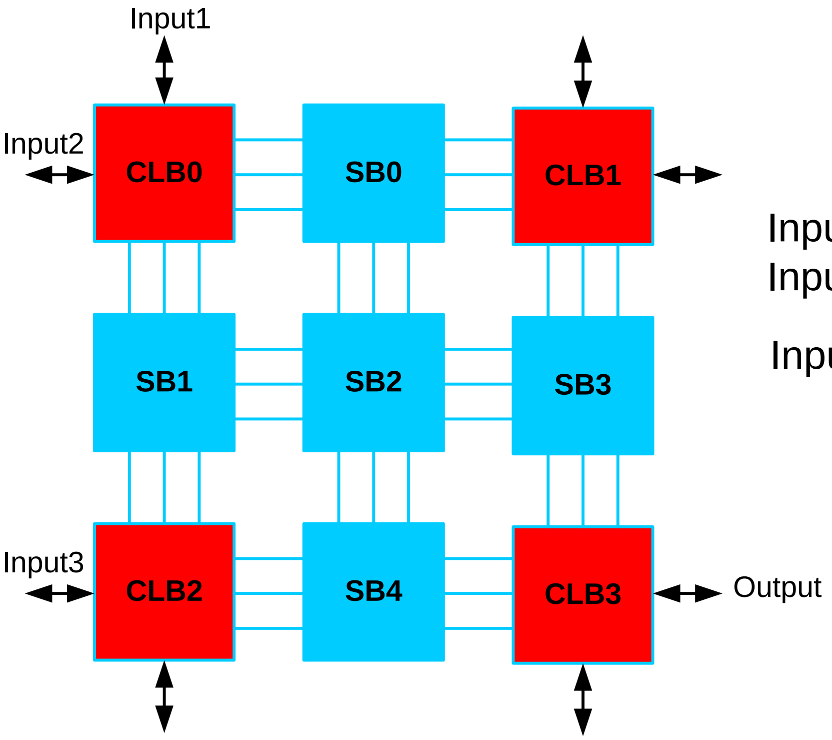
🟥 Configura cks

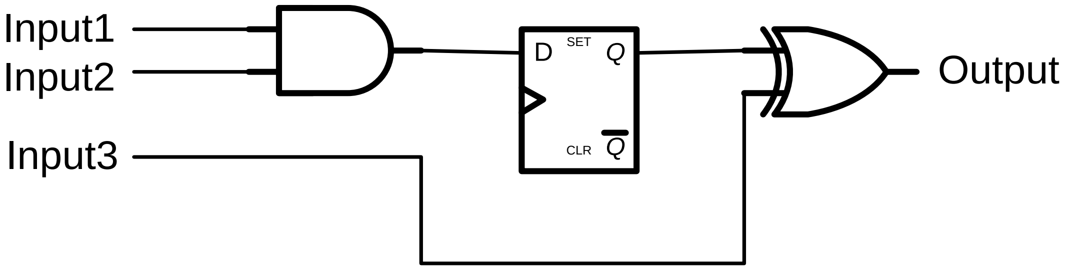
🟦 Interconnection N

⬛ I/O Signals (Pins)

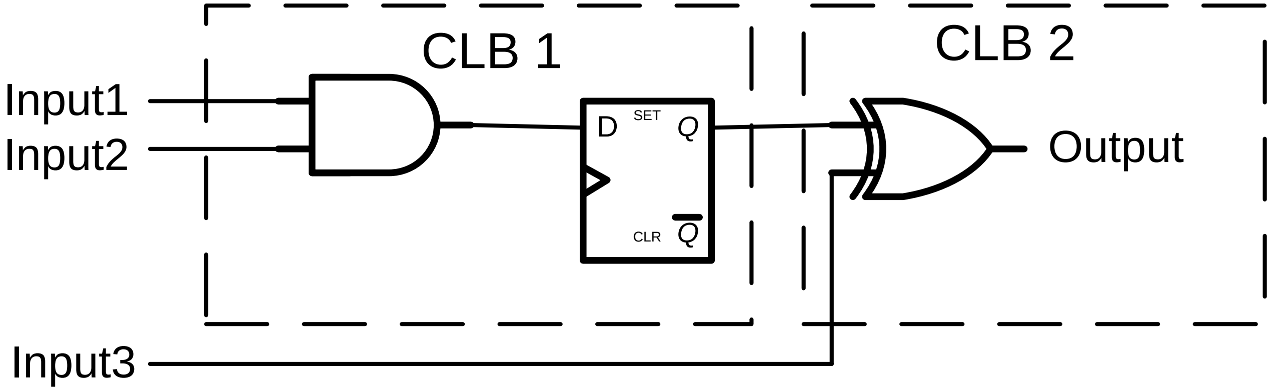
Example 3

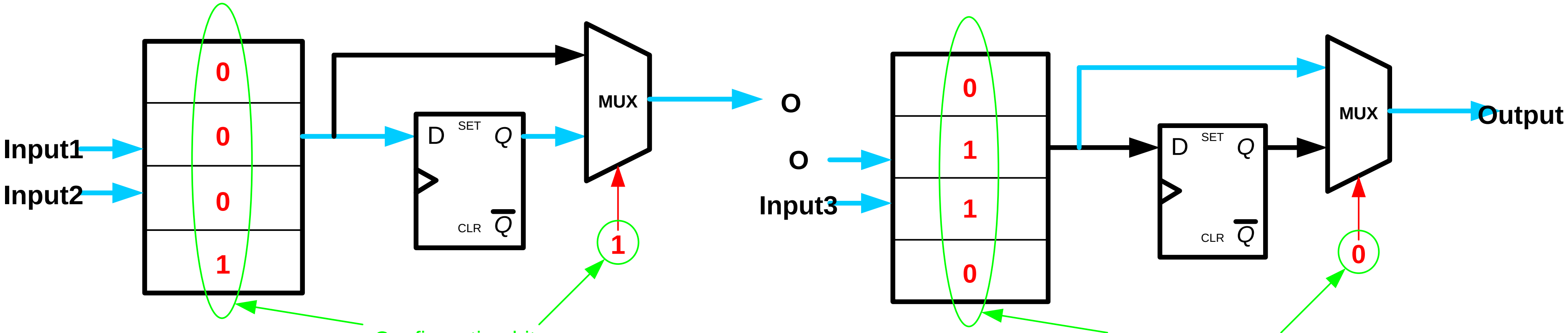
- Determine the configuration bits for the following circuit implementation in a 2x2 FPGA, with I/O constraints as shown in the following figure. Assume 2-input LUTs in each CLB.





CLBs required

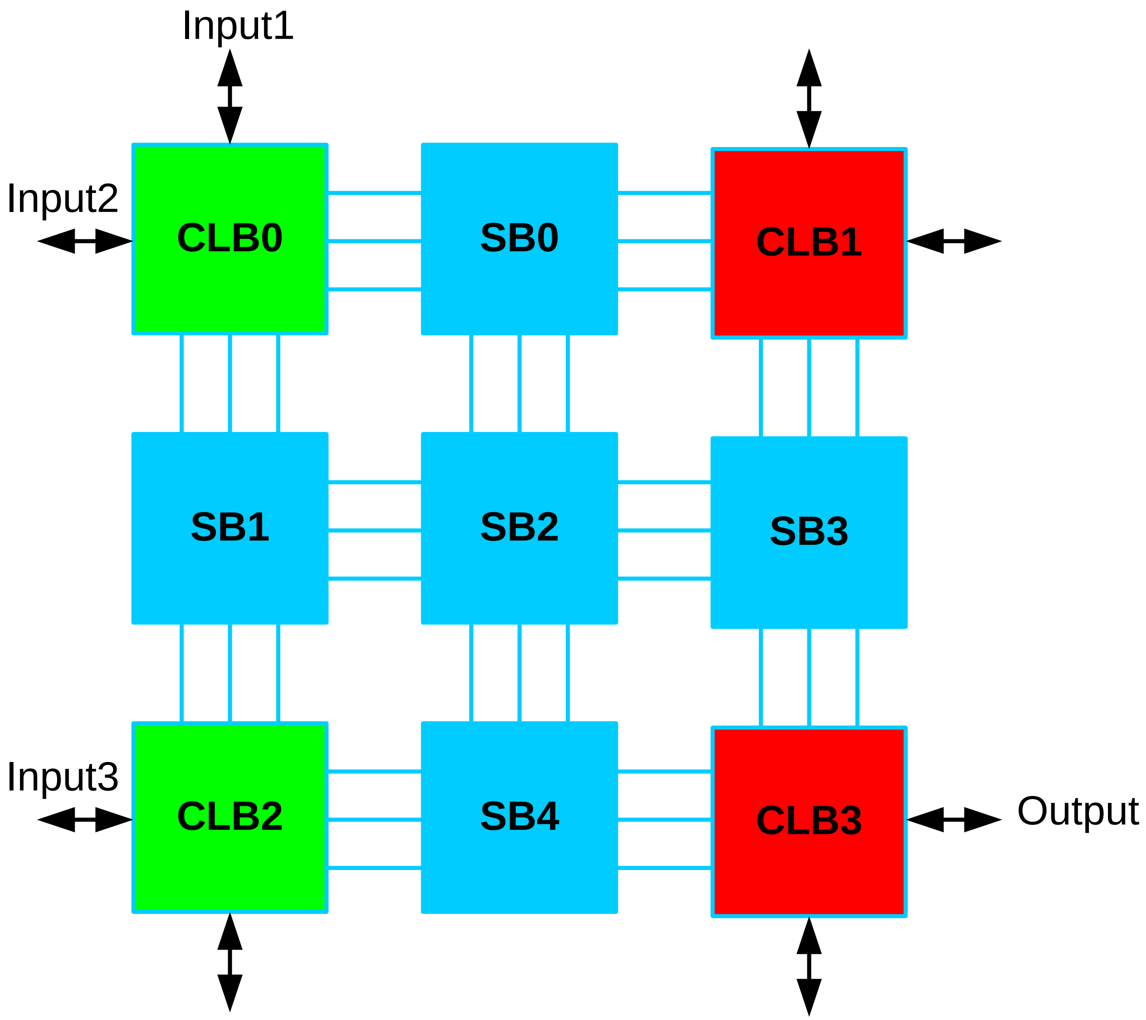




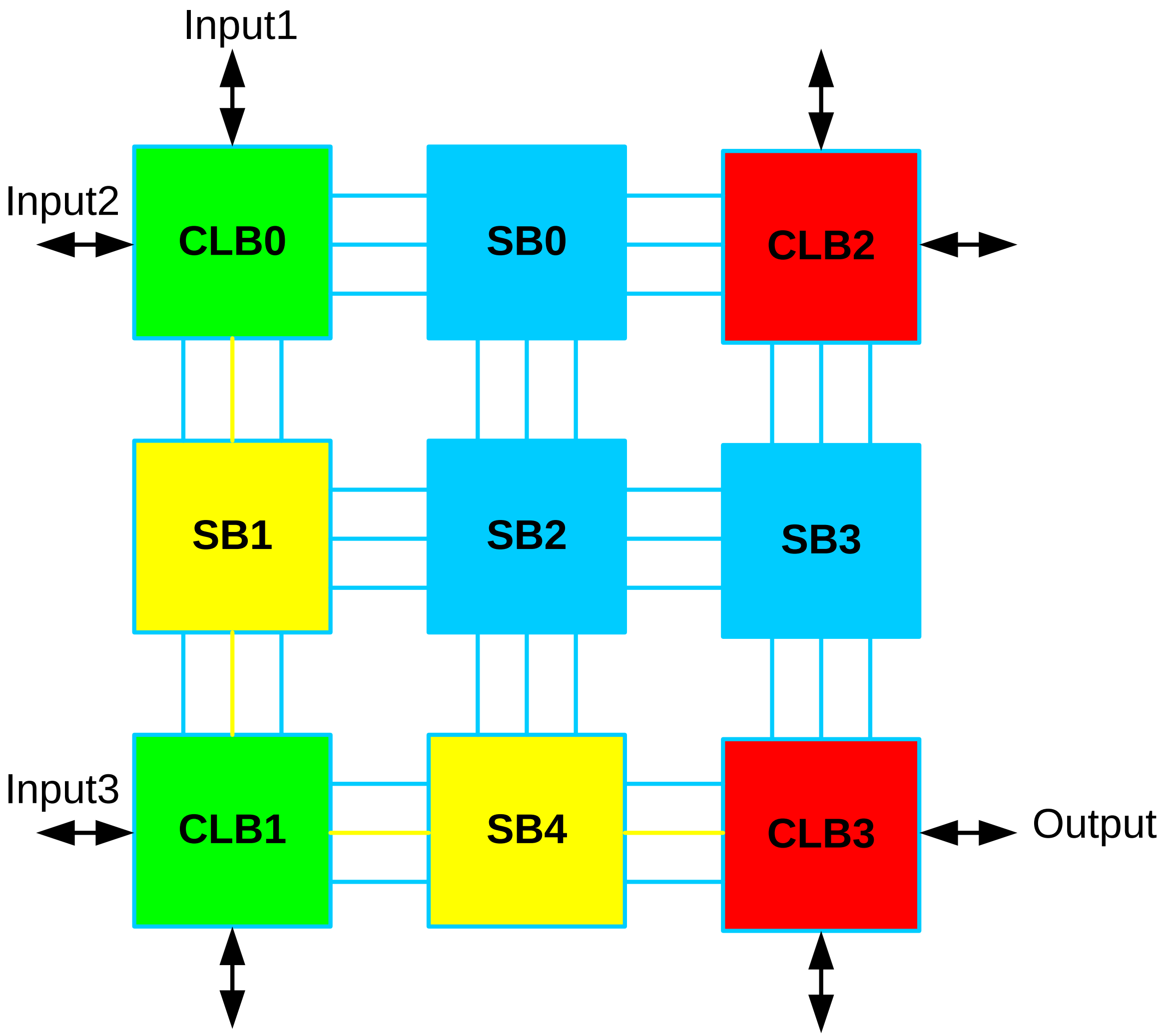
Configuration bits

Configuration bits

Placement: Select CLBs

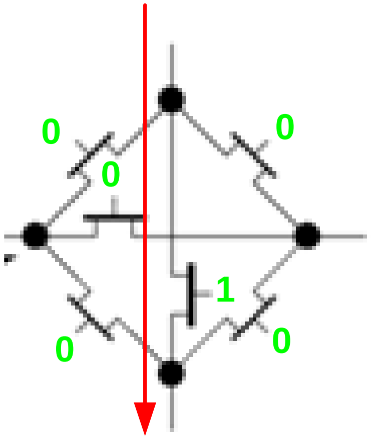


Routing: Select path



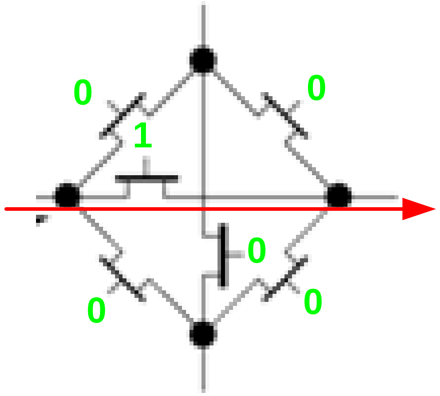
SB1

Configuration bits



SB4

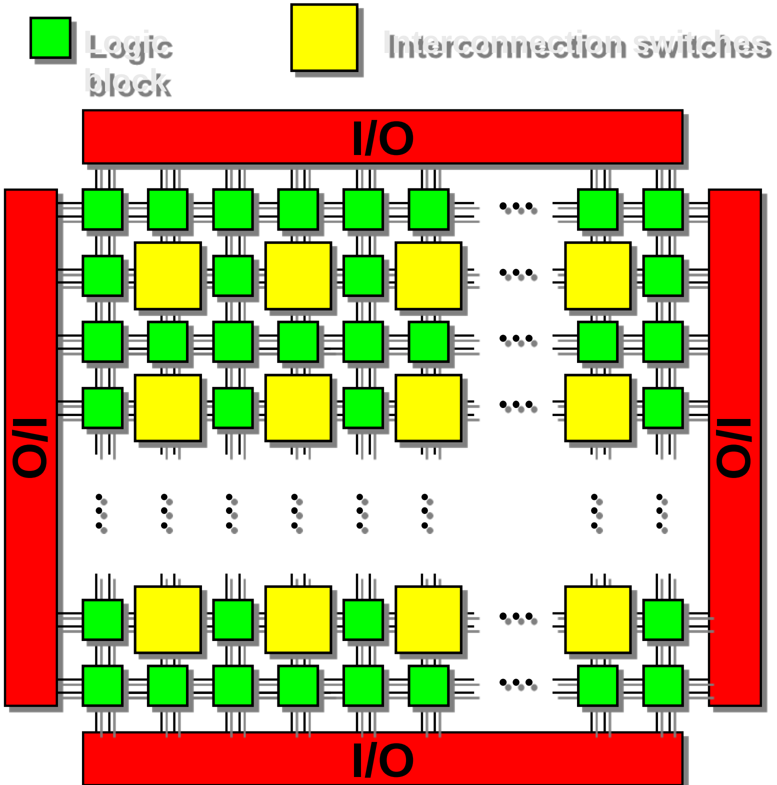
Configuration bits

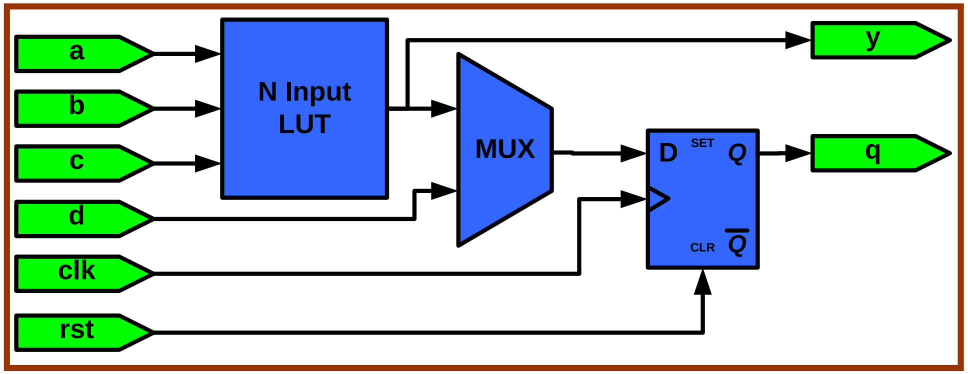


Configuration Bitstream

- The configuration bitstream must include ALL CLBs and SBs, even unused ones
- CLB0: 00011
- CLB1: 01100
- CLB2: XXXXX
- CLB3: XXXXX
- SB0: 000000
- SB1: 000010
- SB2: 000000
- SB3: 000000
- SB4: 000001

- Programmable logic blocks (Logic Element “LE”)
  Implement combinatorial and sequential logic. Based on LUT and DFF.
- Programmable I/O blocks
  Configurable I/Os for external connections supports various voltages and tri-states.
- Programmable interconnect
  Wires to connect inputs , outputs and logic blocks.
  - clocks
  - short distance local connections
  - long distance connections across chip

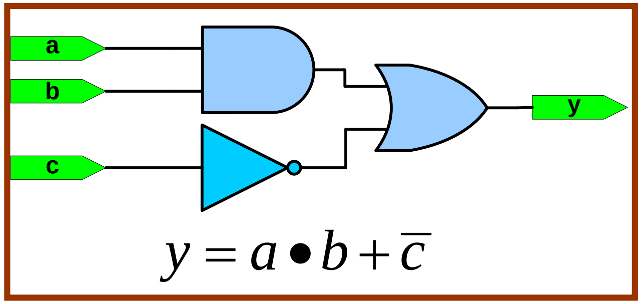




Example 4:Configuring LUT

- LUT is a RAM with data width of 1bit.
- The contents are programmed at power up

Required Function

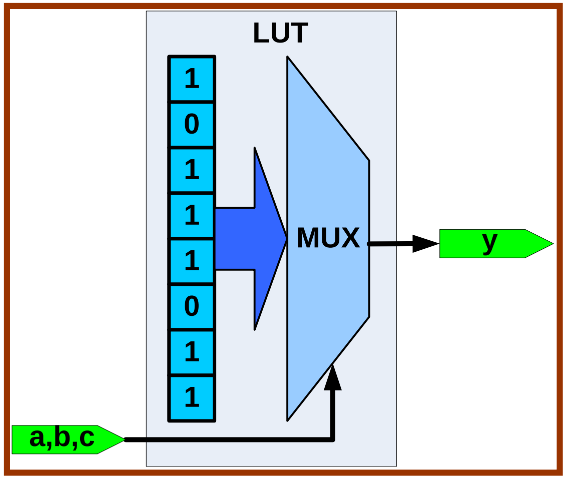


$$y = a \bullet b + \overline{c}$$

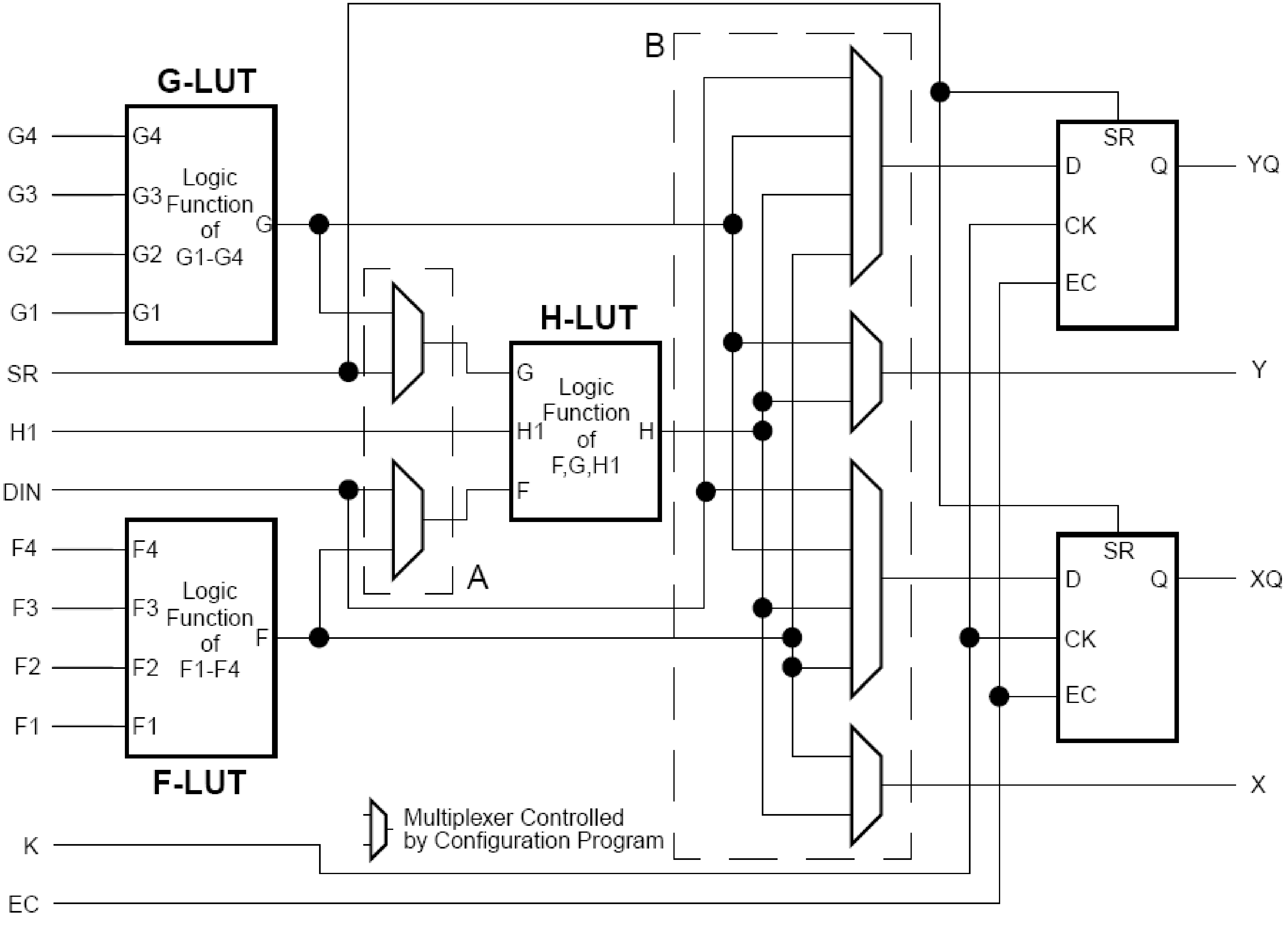
Truth Table

a	b	c	y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

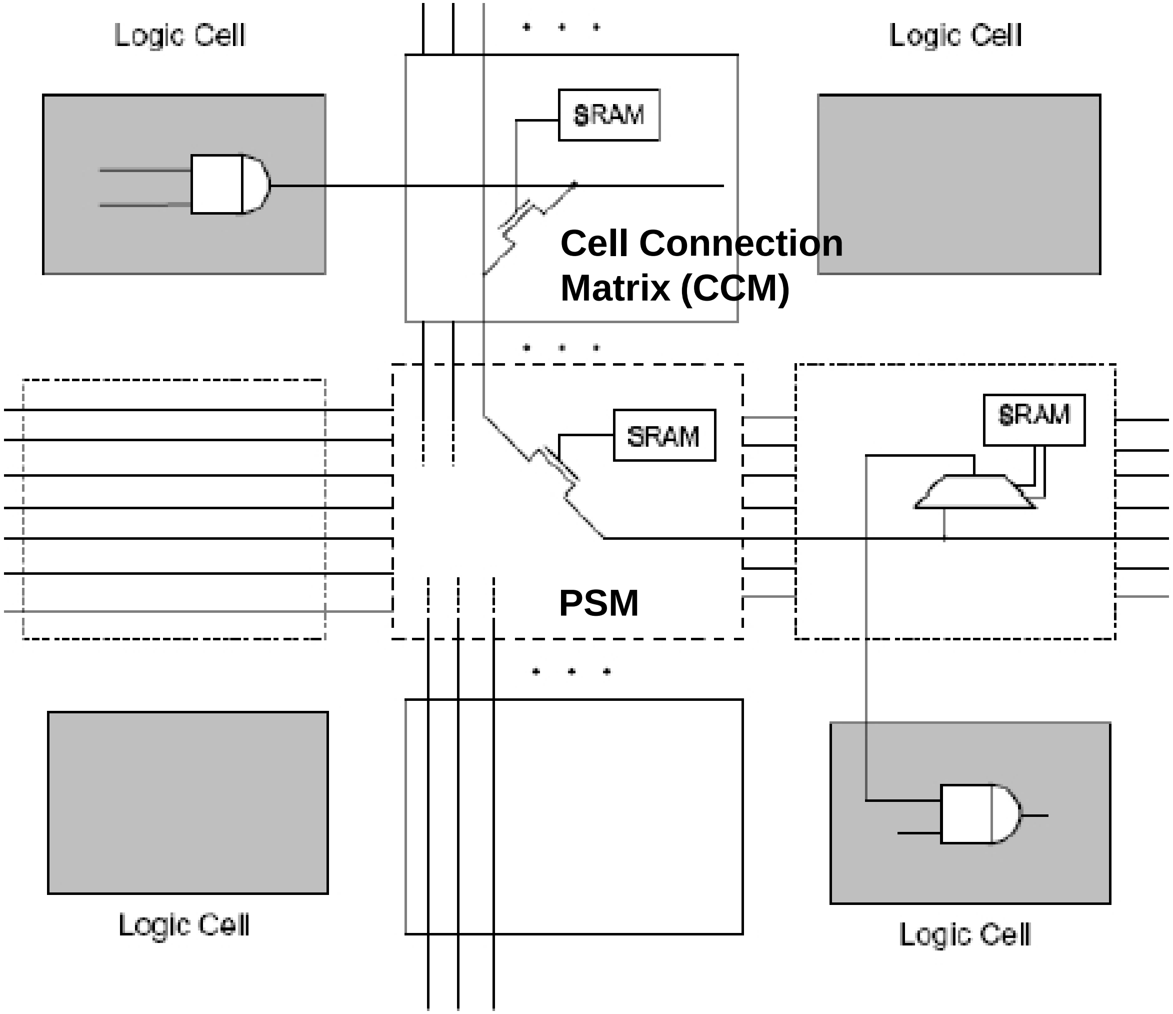
Programmed LUT



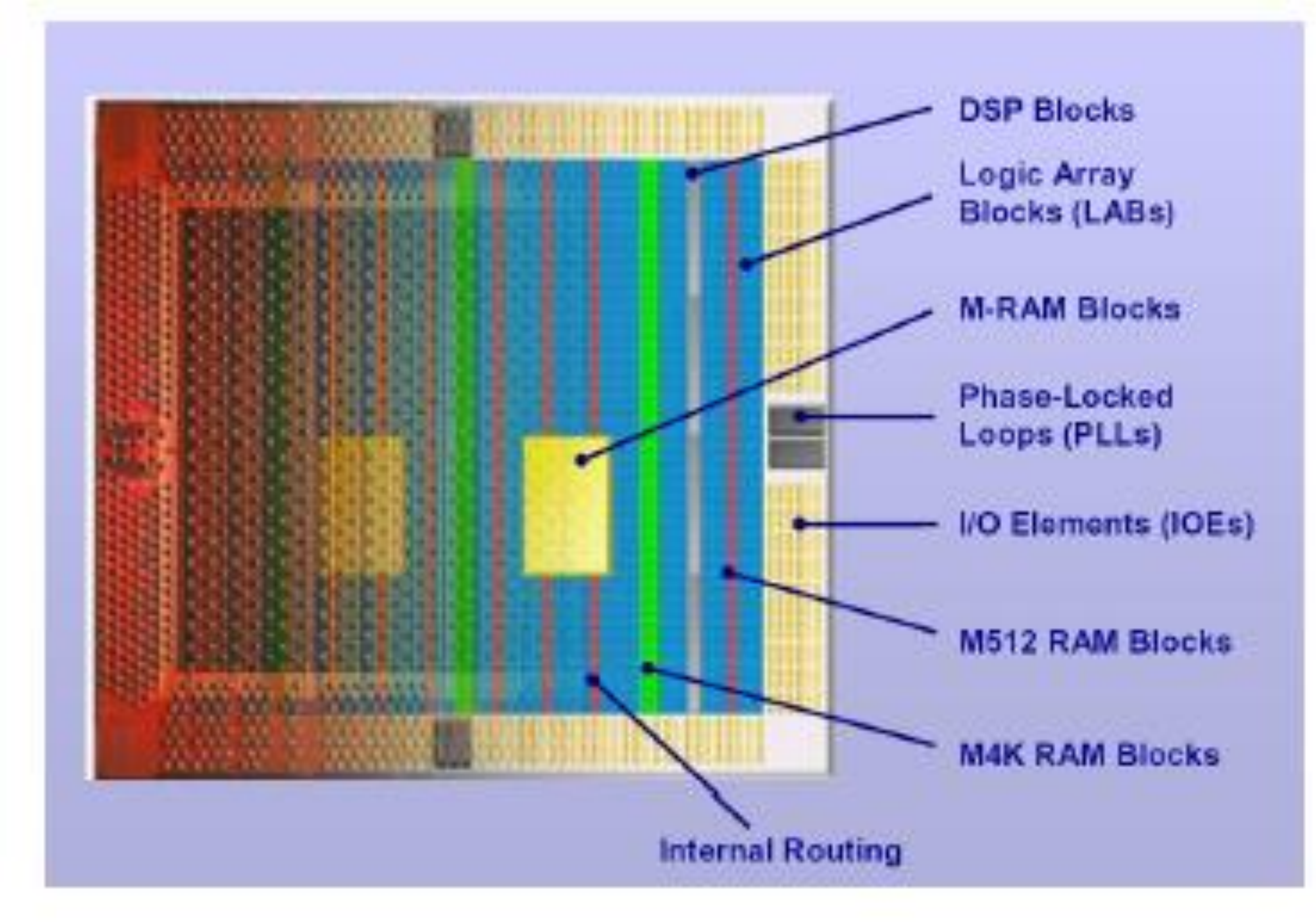
Realistic FPGA CLB: Xilinx



SRAM-type FPGA Interconnect Architecture (contd)



Components of Modern FPGAs



FPGA EDA Tools

- Must provide a design environment based on digital design concepts and components (gates, flip-flops, MUXs, etc.)
- Must hide the complexities of placement, routing and bitstream generation from the user. Manual placement, routing and bitstream generation is infeasible for practical FPGA array sizes and circuit complexities.

FPGA EDA Tools

- High level Description of Logic Design
  - Hardware Description Language (Textual)

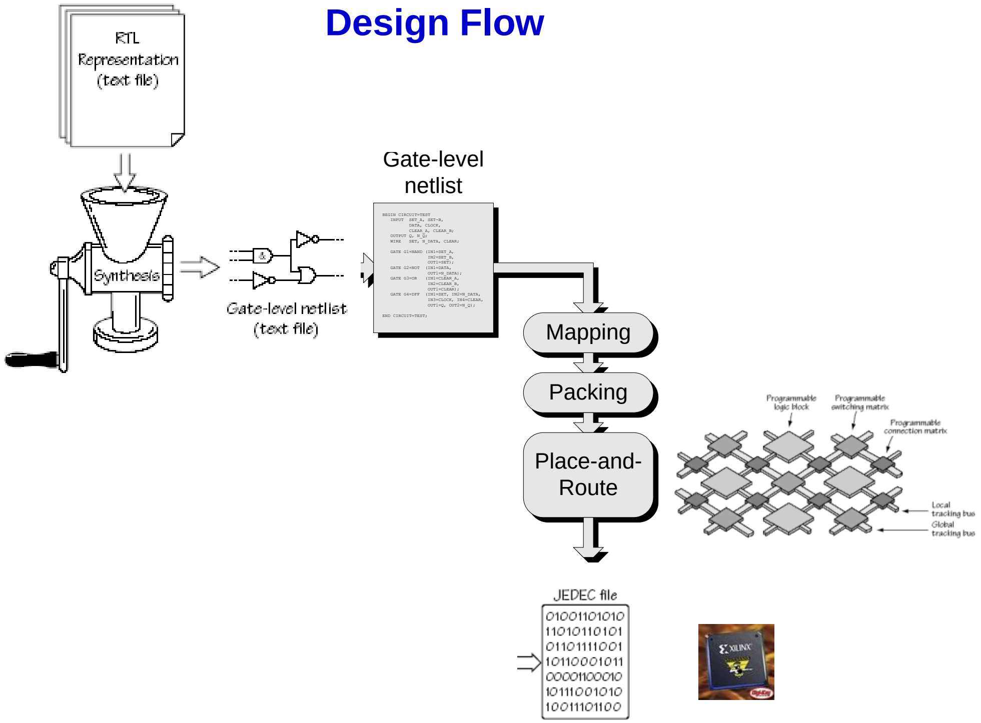
- Compile (Synthesise) into Netlist.
- Boolean Logic Gates.

- Target FPGA Fabric
  - Mapping
  - Routing
- Bit File for FPGA

- Commercial CAE Tools (Complex & Expensive)

- Logic Simulation

Design Flow



```
BEGIN CIRCUIT=TEST
   INPUT  SET_A, SET-B,
          DATA, CLOCK,
          CLEAR_A, CLEAR_B;
   OUTPUT Q, N_Q;
   WIRE   SET, N_DATA, CLEAR;

   GATE G1=NAND (IN1=SET_A,
                 IN2=SET_B,
                 OUT1=SET);
   GATE G2=NOT  (IN1=DATA,
                 OUT1=N_DATA);
   GATE G3=OR   (IN1=CLEAR_A,
                 IN2=CLEAR_B,
                 OUT1=CLEAR);
   GATE G4=DFF  (IN1=SET, IN2=N_DATA,
                 IN3=CLOCK, IN4=CLEAR,
                 OUT1=Q, OUT2=N_Q);

END CIRCUIT=TEST;
```

```
01001101010
11010110101
01101111001
10110001011
00001100010
10111001010
10011101100
```

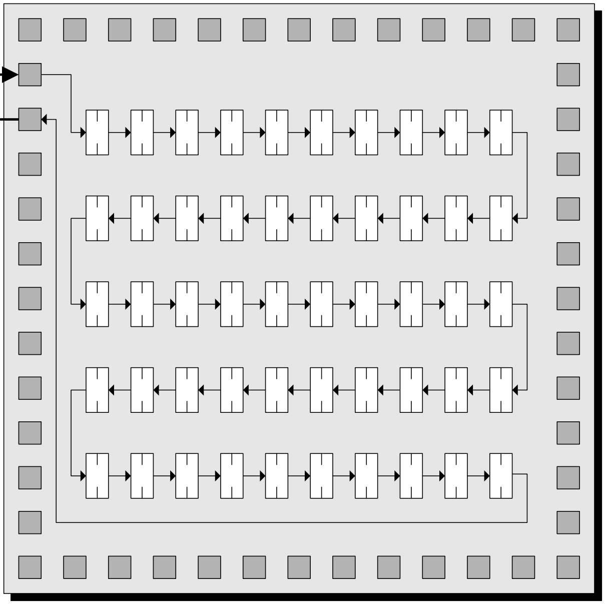
Configuring an FPGA

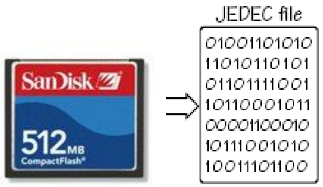
- Millions of SRAM cells holding LUTs and Interconnect Routing
- Volatile Memory. Loses configuration when board power is turned off.
- Keep Bit Pattern describing the SRAM cells in non-Volatile Memory e.g. ROM or Digital Camera card
- Configuration takes ~ secs

JTAG Port

Configuration data in

Configuration data out



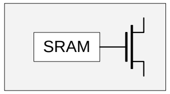


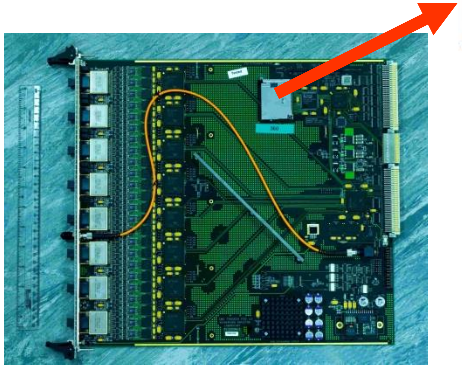
01001101010
11010110101
01101111001
10110001011
00001100010
10111001010
10011101100

Programming Bit File

▨ = I/O pin/pad

▯ = SRAM cell

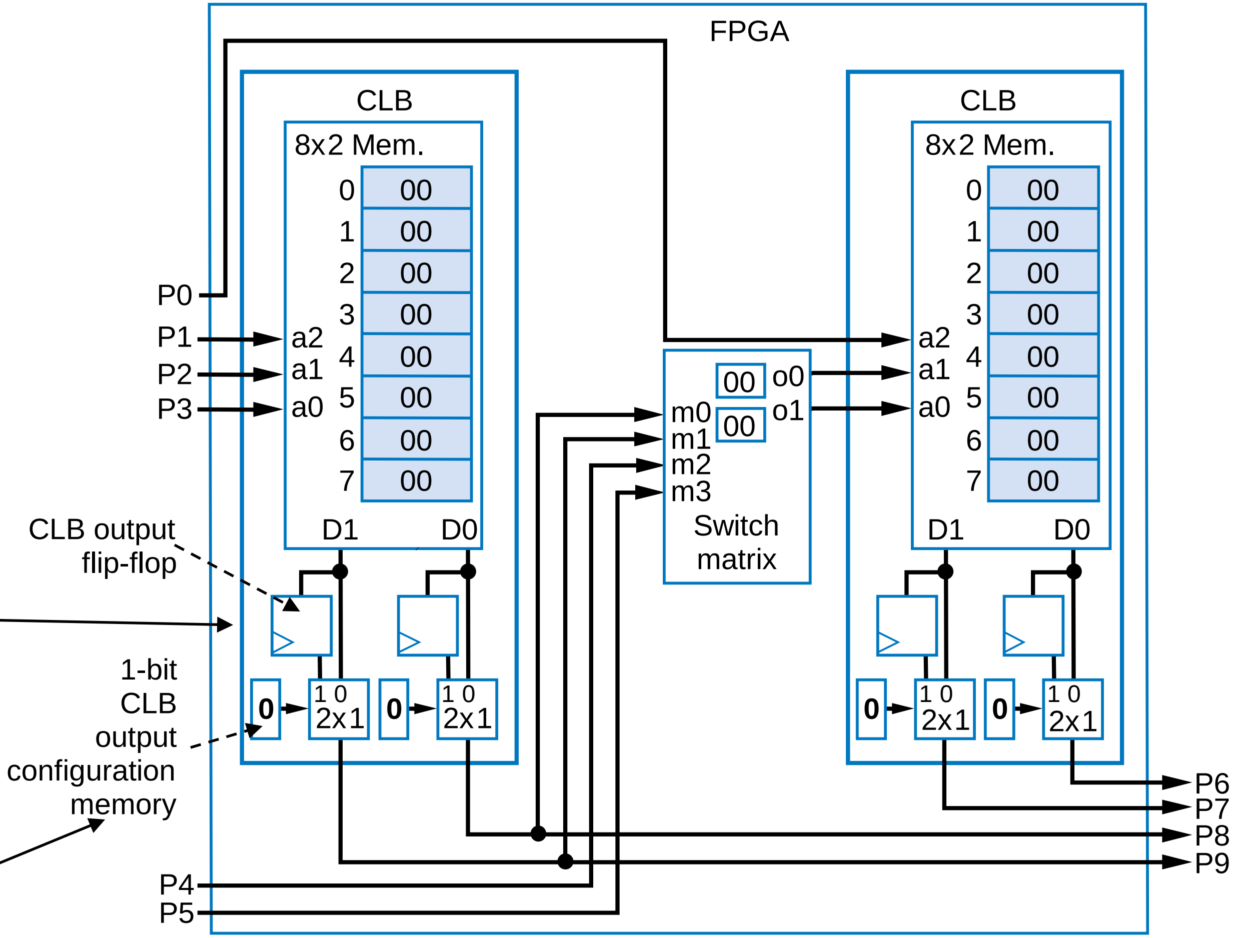




JTAG Testing

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

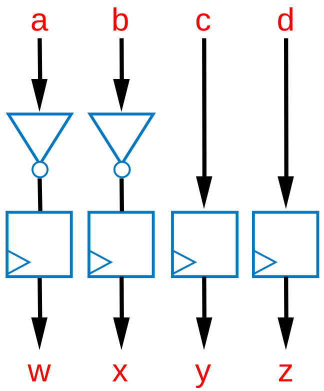


CLB output flip-flop

1-bit CLB output configuration memory

a

FPGA Internals: Sequential Circuit Example using CLBs



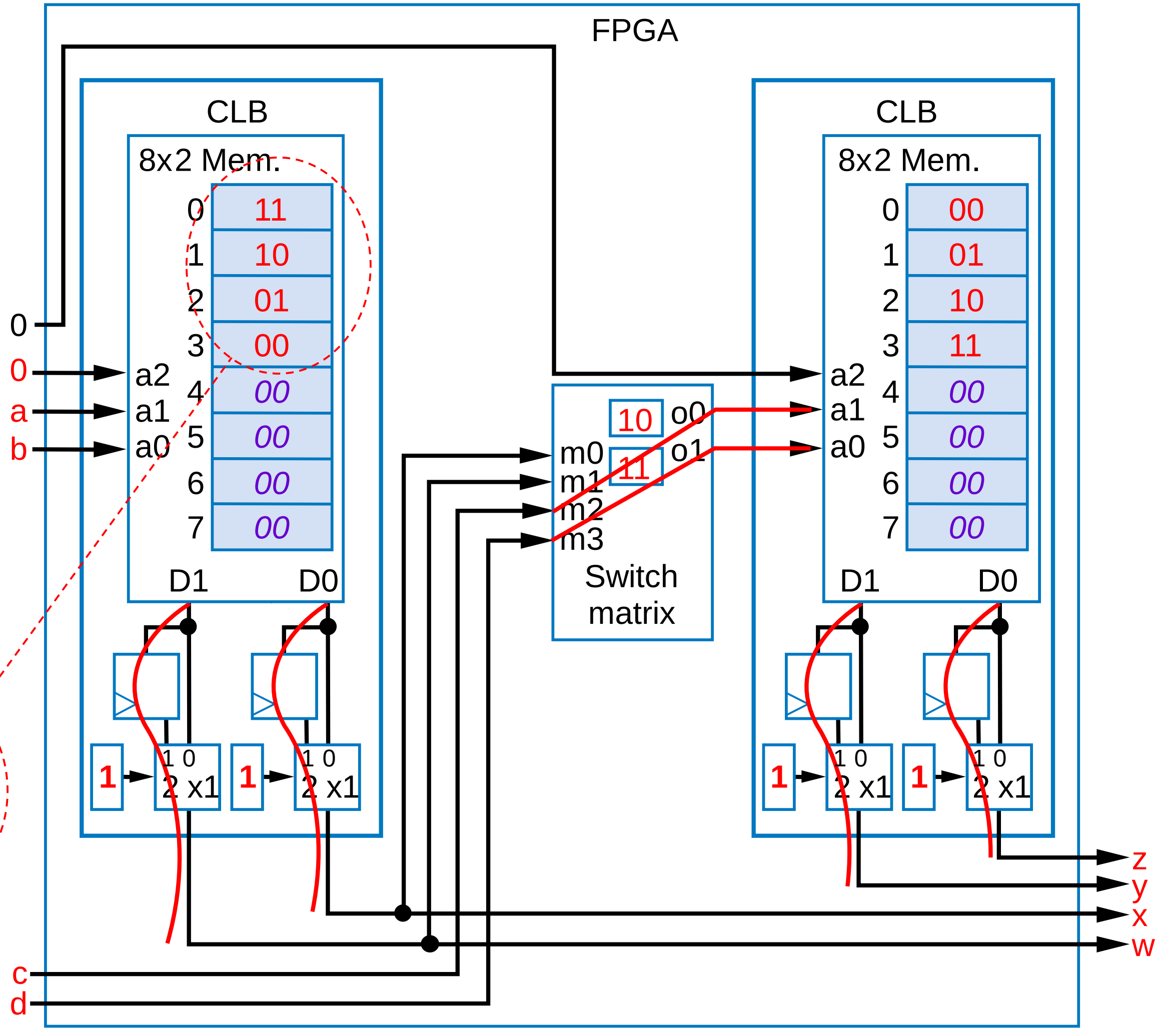
(**a**)

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**)

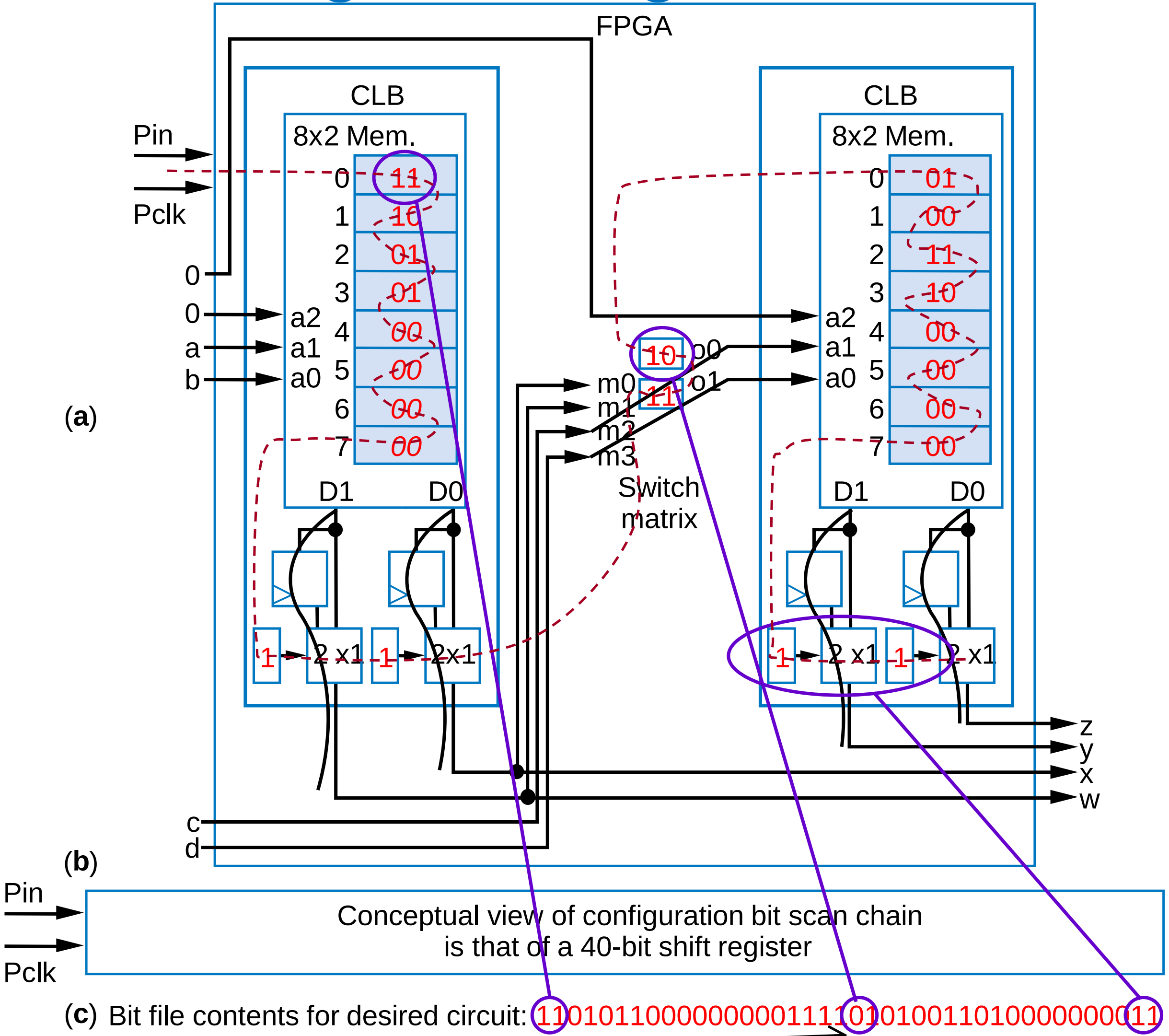


(**c**)

a

FPGA Internals: 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



(c) Bit file contents for desired circuit: 1101011000000000111101010011010000000011

a

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.

Summary

- Programmable Logic Devices
  - Basics
  - Evolution
- Field Programmable Gate Arrays (FPGAs)
  - Architecture

- Design Flow
  - Hardware Description Languages
  - Design Tools