Ministry of Electronics
and Information
Technology

Certificate Course in Accelerator Design using HLS Programming

PG Program in Embedded and SoC Design

National Institute of Electronics and Information Technology

NIELIT OVERVIEW

- An Autonomous Scientific Society under the Ministry of Electronics and Information Technology (MeitY).
- Engaged both in Formal & Non-Formal Education in the area of Information, Electronics and Communications Technology (IECT).
- Actively engaged in Capacity Building and Skill Development in the areas of IECT.
- One of the National Examination Body and Accreditation Body for IT, Electronics & Digital Literacy Courses.

FACULTY MEMBERS

- Ishant Kumar Bajpai, Scientist D
- Chandralakha R Pillai, RS(VLSI)
- SS-Sanjana S, Intern(VLSI)

TIME TABLE

Days	Morning Session	Afternoon Session
10/01/2024	Introduction to the course and HLS Toolchain Configuration-IKB	Lab Session-CP/SS
11/01/2024	Data Types & Conditional Statements-IKB	Lab Session-CP/SS
12/01/2024	Combinational loop unrolling-IKB	Lab Session-CP/SS
13/01/2024	**Saturday**	
14/01/2024	**Sunday**	
15/01/2024	**Pongal Holiday**	
16/01/2024	Flip flops and Single Cycle Design Flow-IKB	Lab Session-CP/SS
17/01/2024	State Machines & Utilities Design-IKB	Lab Session-CP/SS
18/01/2024	Multi-Cycle Design-IKB	Lab Session-CP/SS
19/01/2024	Interface Synthesis-IKB	Lab Session-CP/SS
20/1/2024	**Saturday**	
21/1/2024	**Sunday**	

TIME TABLE

22/01/2024	Arrays & Pointers in HLS-IKB	Lab Session-CP/SS
23/01/2024	AXI protocol & Memory Mapped implementation in HLS-IKB	Lab Session-CP/SS
24/01/2024	Introduction to Image Processing, Image Conversion, and Histogram equalization-IKB	Lab Session-CP/SS
25/01/2024	Image Filtering & Edge Detection-IKB	Lab Session-CP/SS
26/01/2024	**Republic Day Holiday**	
27/01/2024	**Saturday**	
28/01/2024	**Sunday**	
29/01/2024	Image Thresholding & Noise-IKB	Lab Session-CP/SS
30/01/2024	Image Thresholding on FPGA-IKB	Lab Session-CP/SS
31/01/2024	Examples of Scaling-IKB	Lab Session-CP/SS
01/02/2024	ML Algorithm implementation on FPGA-IKB	Lab Session-CP/SS
02/02/2024	Distributed Arithmetic-IKB	Lab Session-CP/SS

TIME TABLE

05/02/2024	Mapping of Signal Processing on FPGA-IKB	Lab Session-CP/SS
06/02/2024	Mapping of Signal Processing on FPGA-IKB	Lab Session-CP/SS
07/02/2024	Functional Pipelineing-IKB	Lab Session-CP/SS
08/02/2024	Case Study 1	Case Study 1
09/02/2024	Case Study 2	Case Study 2
10/02/2024	**Saturday**	
11/02/2024	**Sunday**	
12/02/2024	Mini Project 1	Mini Project 1
13/02/2024	Mini Project 2	Mini Project 2
14/02/2024	Mini Project 3	Mini Project 3
15/02/2024	Revision	Revision

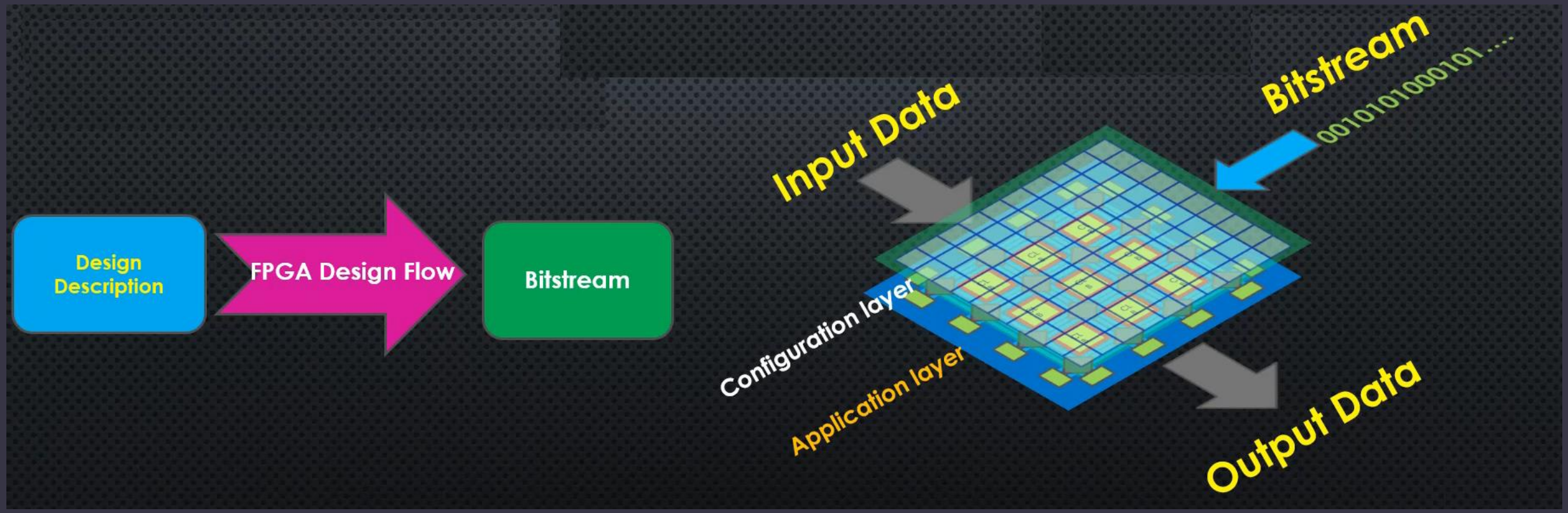
HLS PROGRAMMING

Lecture - 1

INTRODUCTION

- Preliminaries
  - Basic of the C/C++ Programming
  - Basic of the execution model of a software program on a CPU
  - Specific knowledge of the FPGA is an added advantage

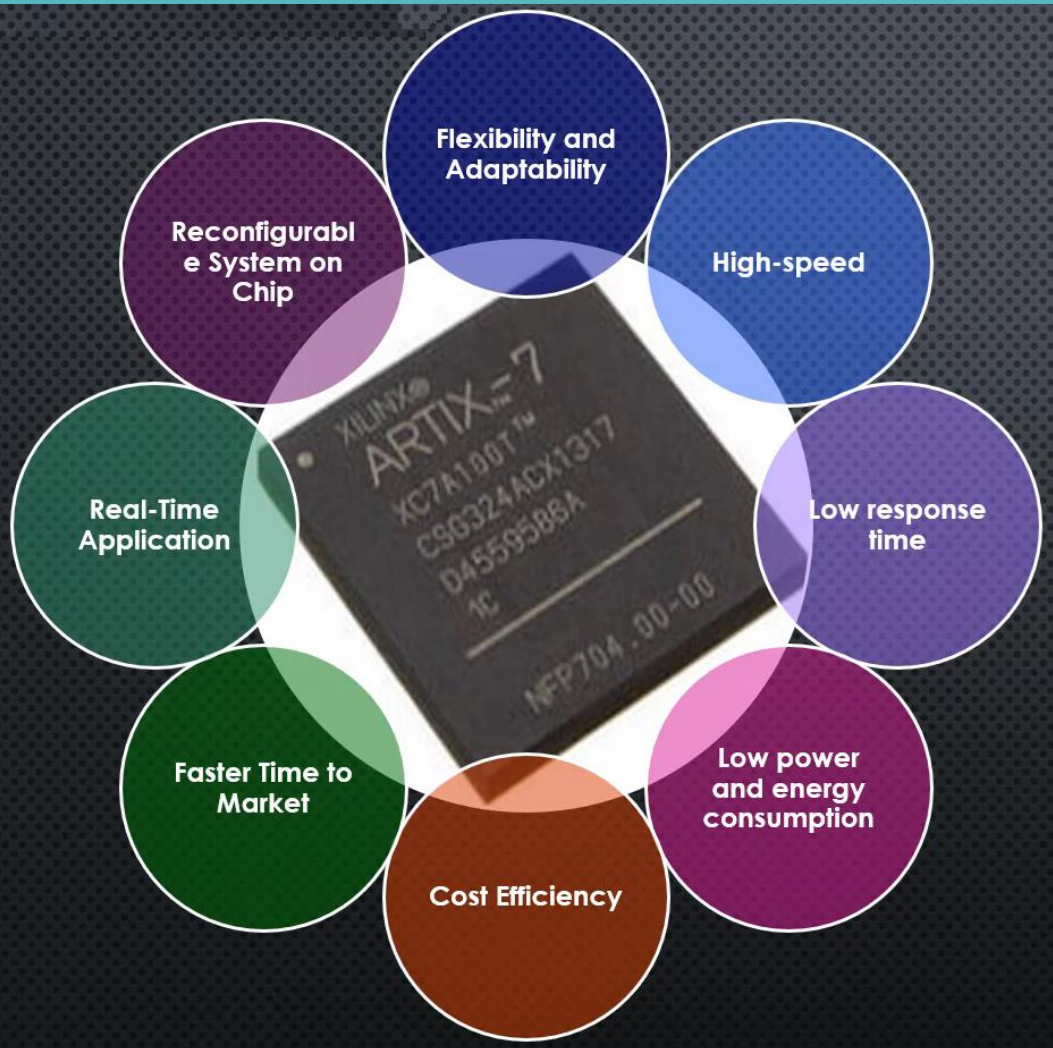
ROLE OF THE HIGH-LEVEL SYNTHESIS IN CONFIGURATION OF THE FPGA



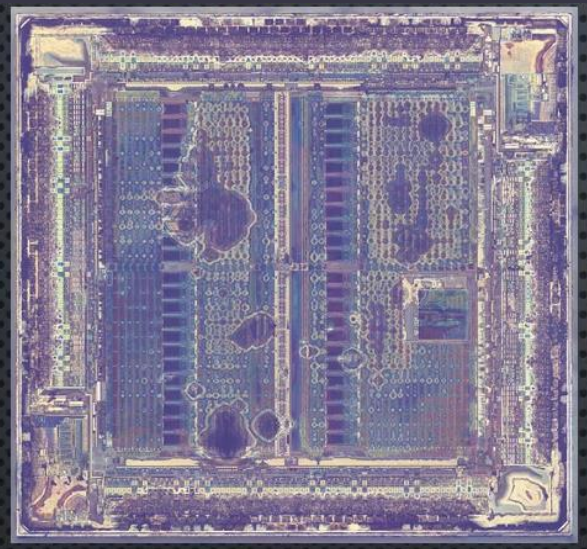
ASSIGNMENT Q- 1

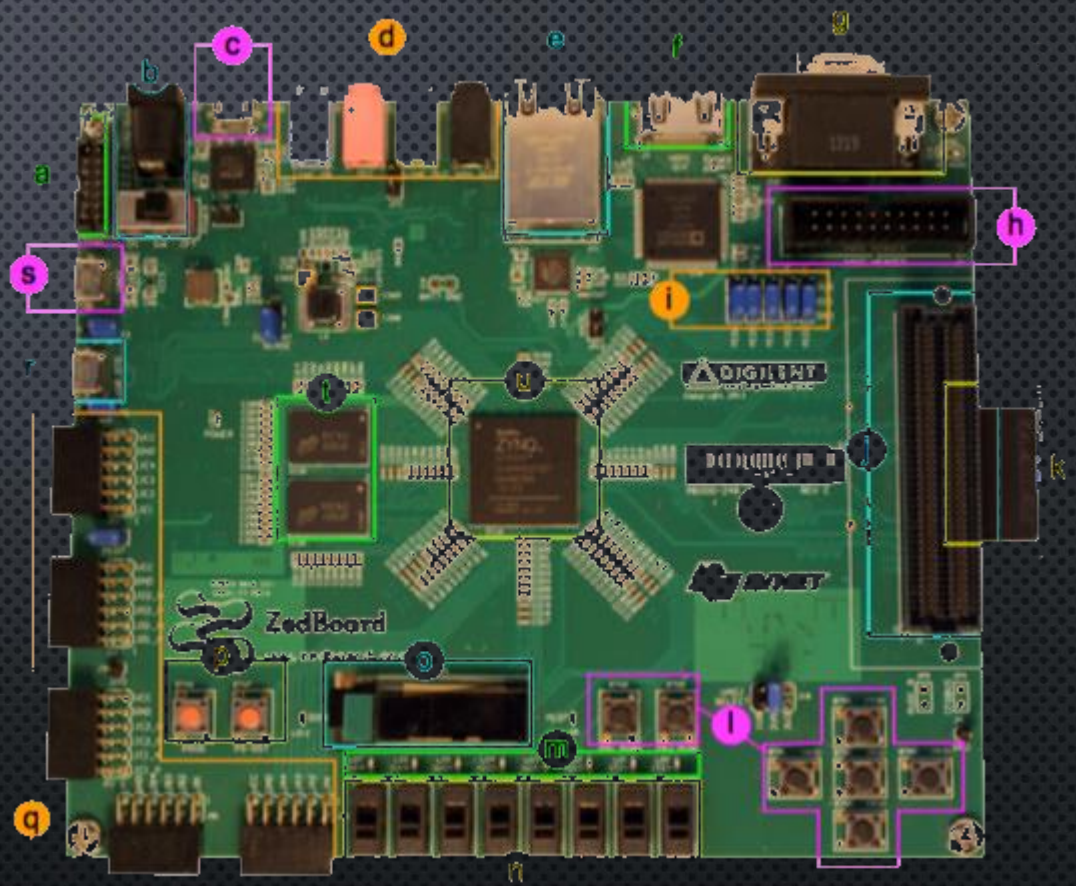
- Explain the structure of a function in C/C++
- What are the different ways to describe a call-by-value and call-by-reference argument in a C/C++?

FPGA FEATURES



FPGA HISTORY





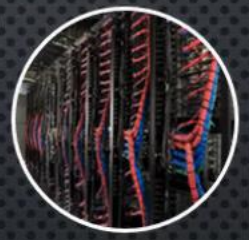
FPGA APPLICATIONS

Industry

- Aerospace & Defense
- Automotive
- Medical

Communications

- 5G Wireless
- Wired Communications
- Wireless Communications



Data Center

- Database and Data Analytics
- Financial Technology
- High Performance Computing

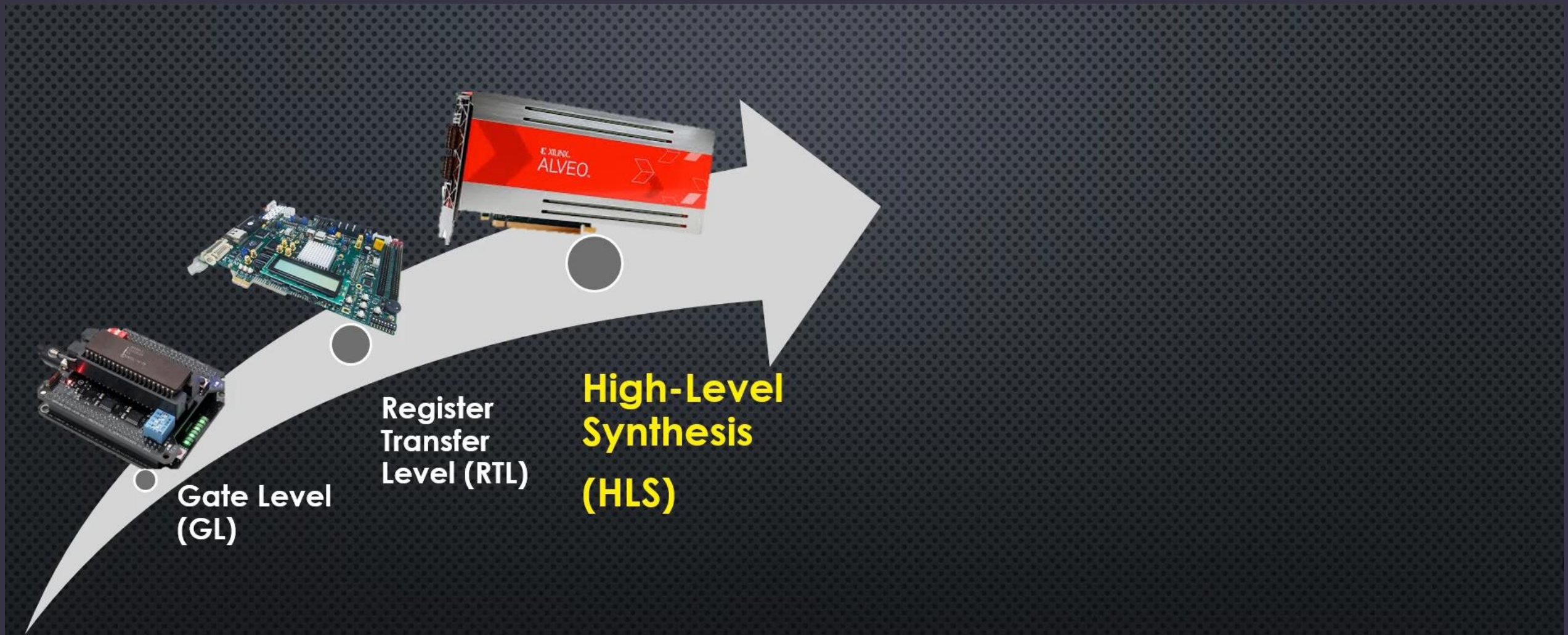
AI

- Deep Learning
- Inference
- Neural Network

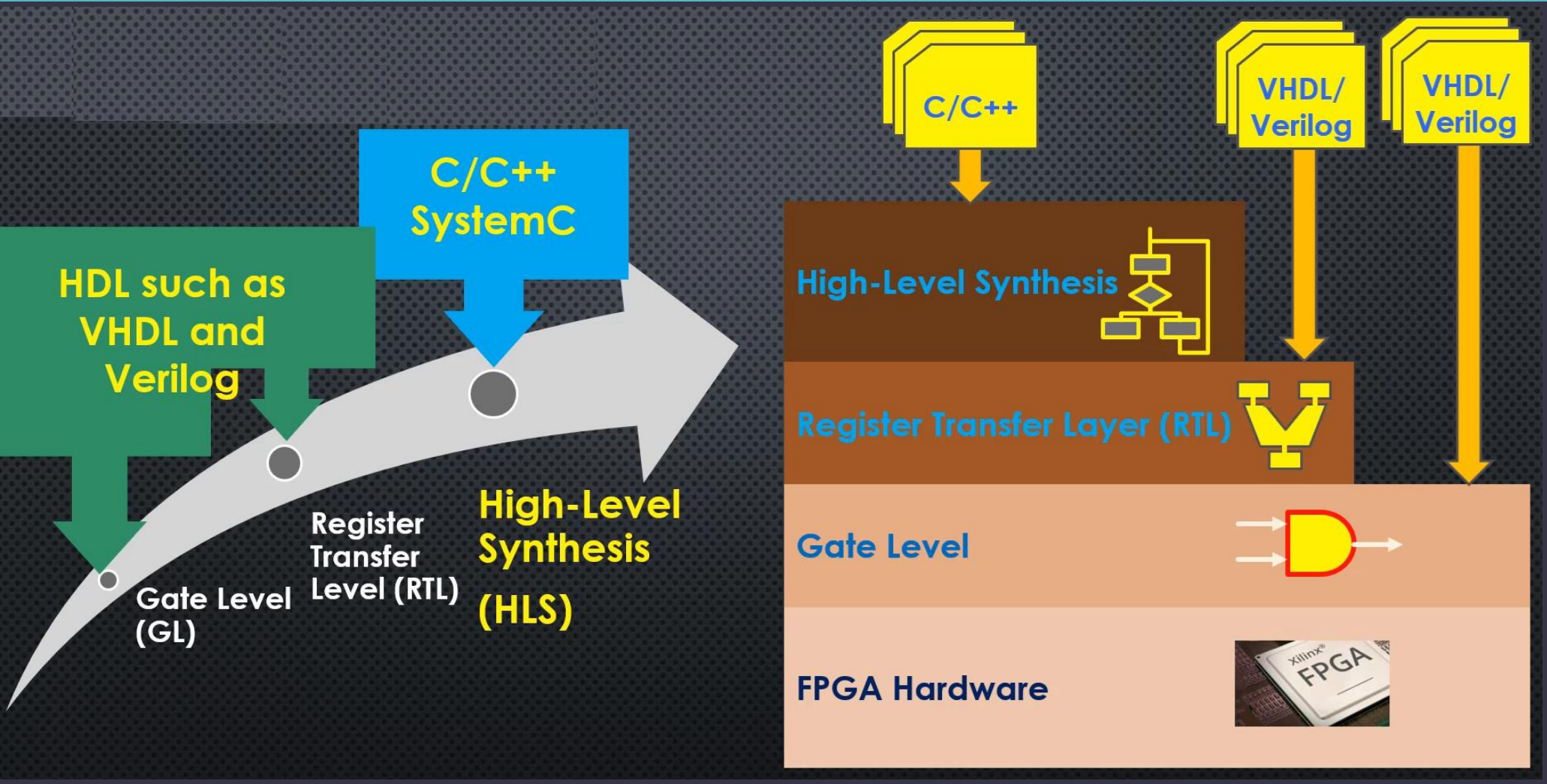
ASSIGNMENT Q- 2

- Make a List of Three Features of FPGA Technology
- List some of the possible applications for FPGAs by connecting to the Xilinx Website

FPGA CONCEPT: DESIGN APPROACH



DESIGN LANGUAGES



HLS

HLS is pretty hot these days, and it is a topic which is definitely here to stay.

HLS is a set of

- coding styles,
- optimisation techniques
- synthesis tools

HLS IN PRACTICE

Nvidia in designing Tegra embedded GPU

Google in building VP9 encoder/decoder

Qualcomm in image processing

AMD in designing microprocessors

ST Micro in implementing H.265

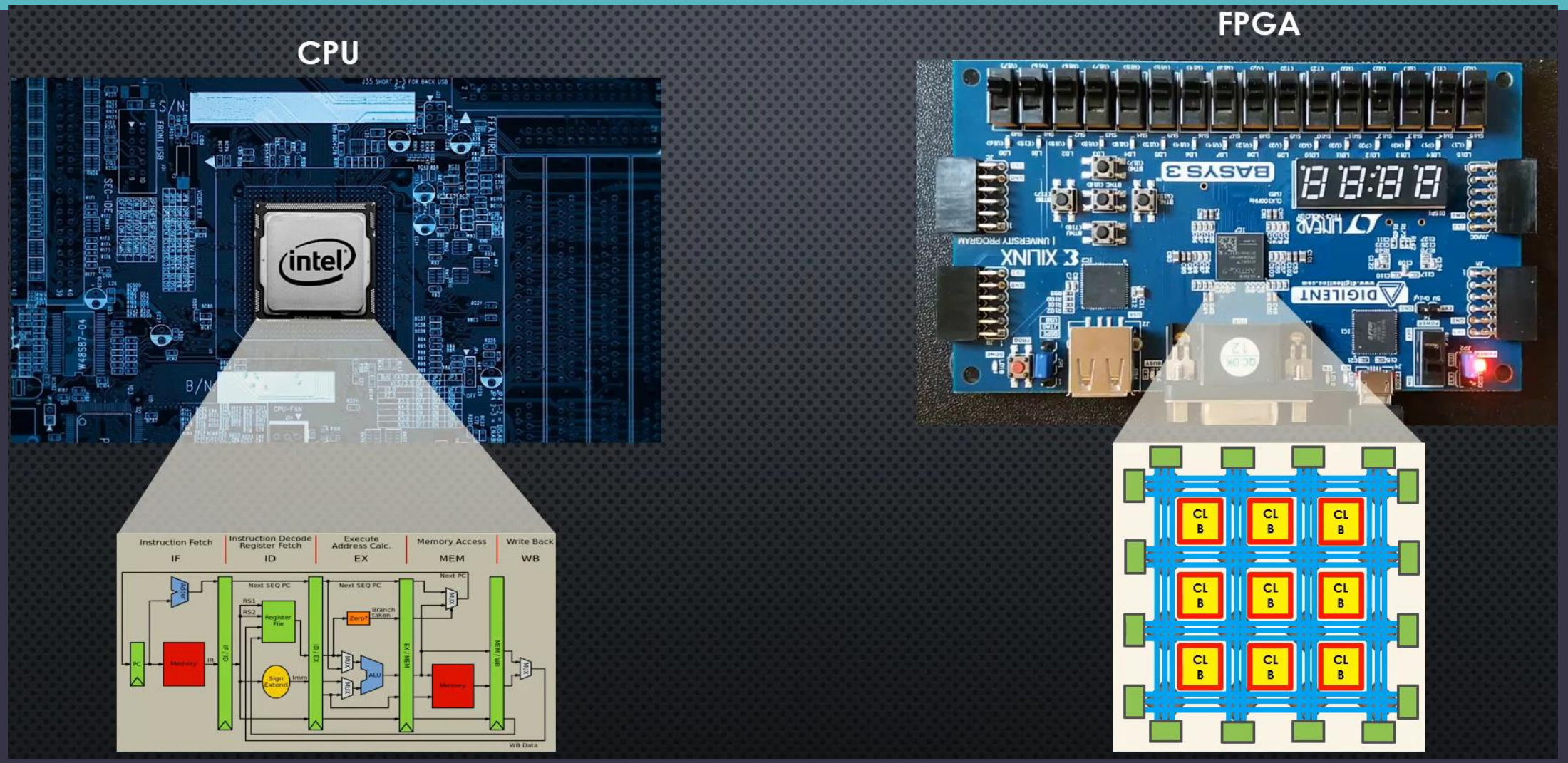
ASSIGNMENT Q- 3

- Which Design level is more suitable for implementing a machine-learning algorithm on an FPGA and why?
- Gate Level based on VHDL Language
- RTL based on Verilog Language
- HLS based on C/C++

FPGA PLATFORM VS. CPU PLATFORM

FPGA Concepts

FPGA VS. CPU


CPU
FPGA
intel
BASYS 3
XILINX
DIGILENT
Instruction Fetch
IF
Instruction Decode
Register Fetch
ID
Execute
Address Calc.
EX
Memory Access
MEM
Write Back
WB
CL
B

FPGA VS CPU

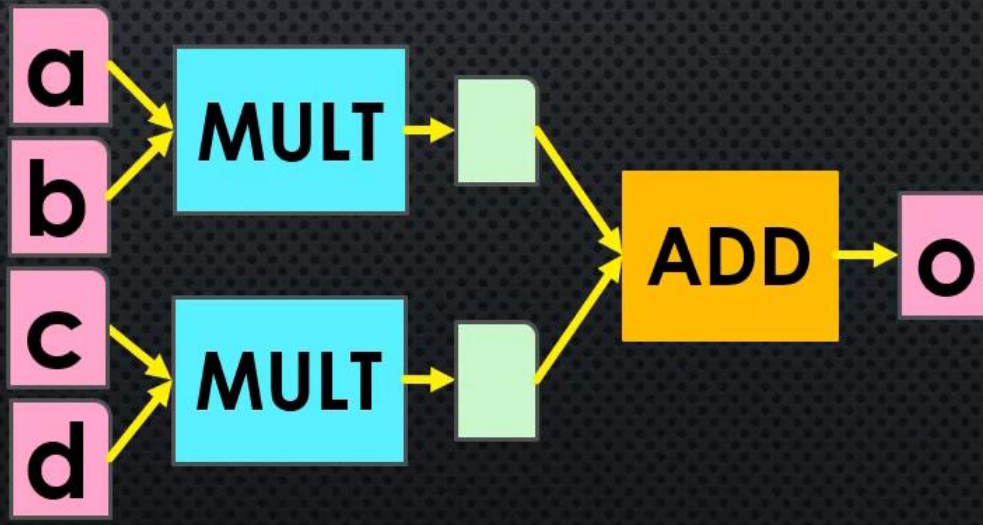
```
int func(int a, int b, int c , int d) {
    int  o = a*c+b*d;
    return o;
}
```

CPU Compiler

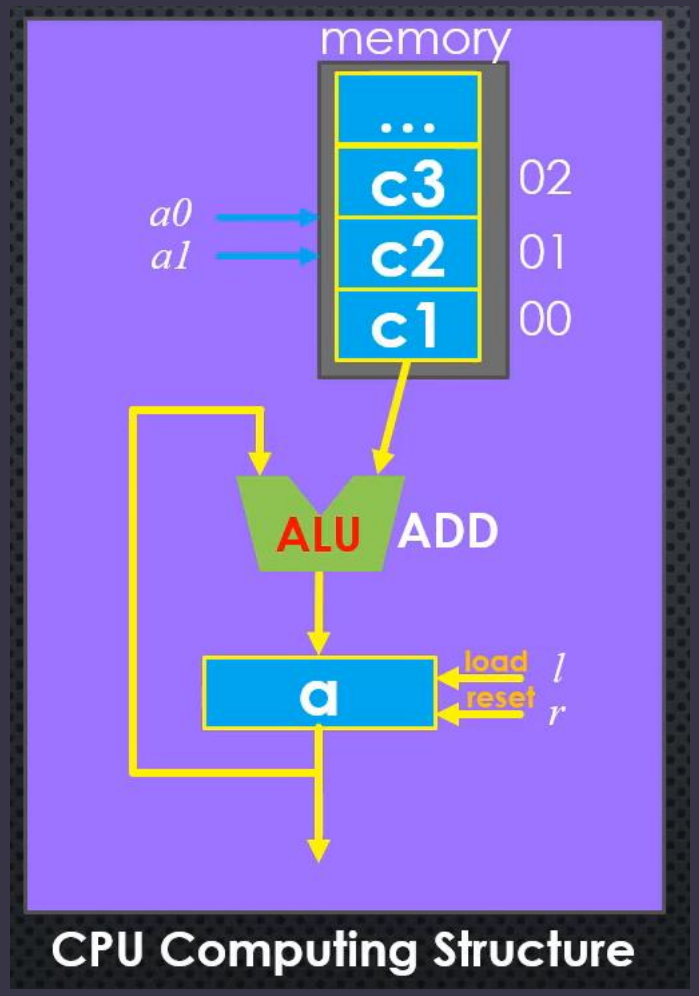
```
...
mov DWORD PTR -20[rbp], edi
mov DWORD PTR -24[rbp], esi
mov DWORD PTR -28[rbp], edx
...
imul eax, DWORD PTR -28[rbp]
...
imul eax, DWORD PTR -32[rbp]
add eax, edx
...
mov eax, DWORD PTR -4[rbp]
.cfi_def_cfa 7, 8
ret
...
```

```
int func(int a, int b, int c , int d) {
    int  o = a*c+b*d;
    return o;
}
```

FPGA Synthesizer



EXAMPLE CPU




```
a = 0;
a = a+c1;
a = a+c2;
a = a+c3;
```

C Code

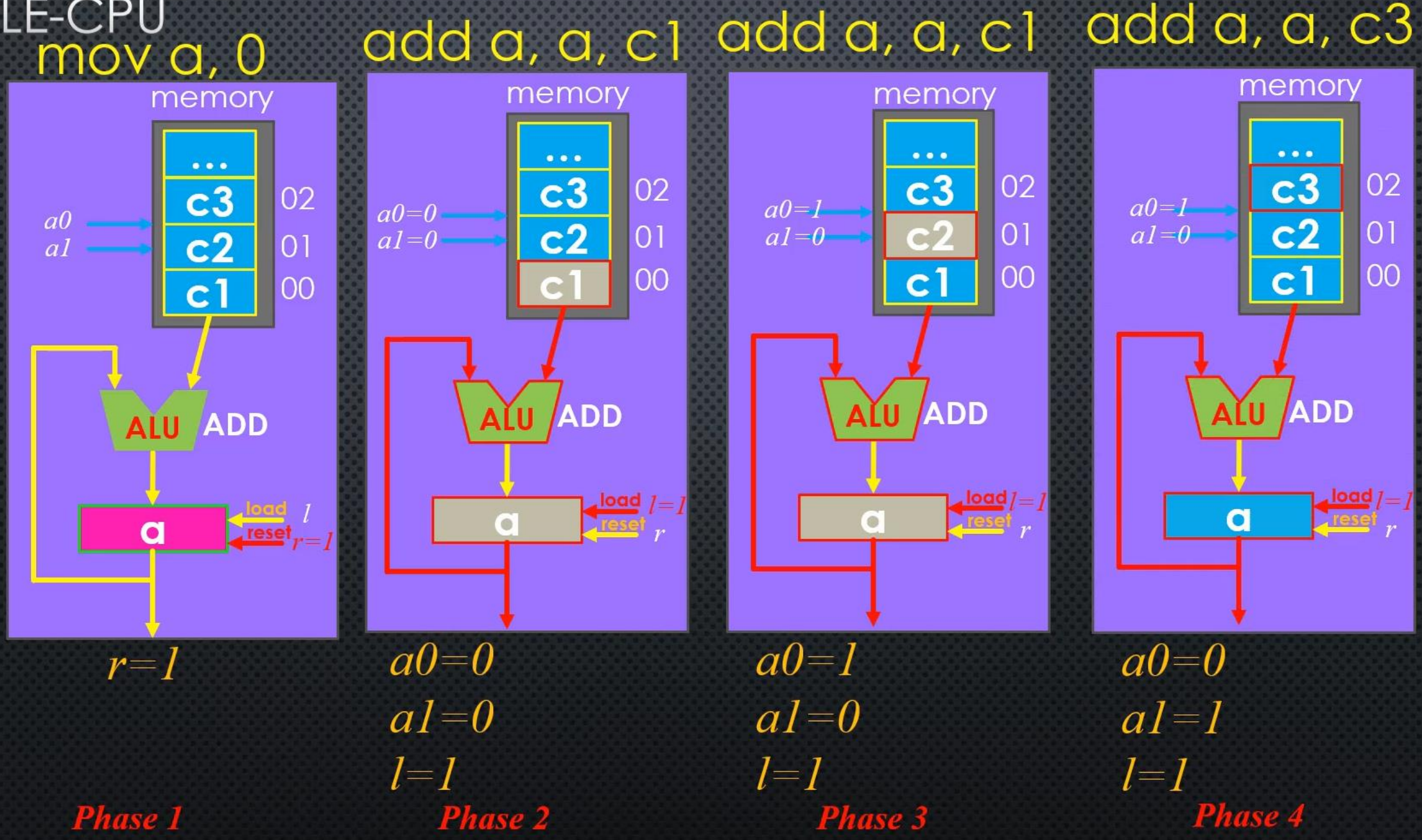
```
mov a, 0
add a, a, c1
add a, a, c2
add a, a, c3
```

Assembly

r=1, l=0, a0=0, a1=0
r=0, l=1, a0=0, a1=0
r=0, l=1, a0=0, a1=1
r=0, l=1, a0=1, a1=0

Machine Code

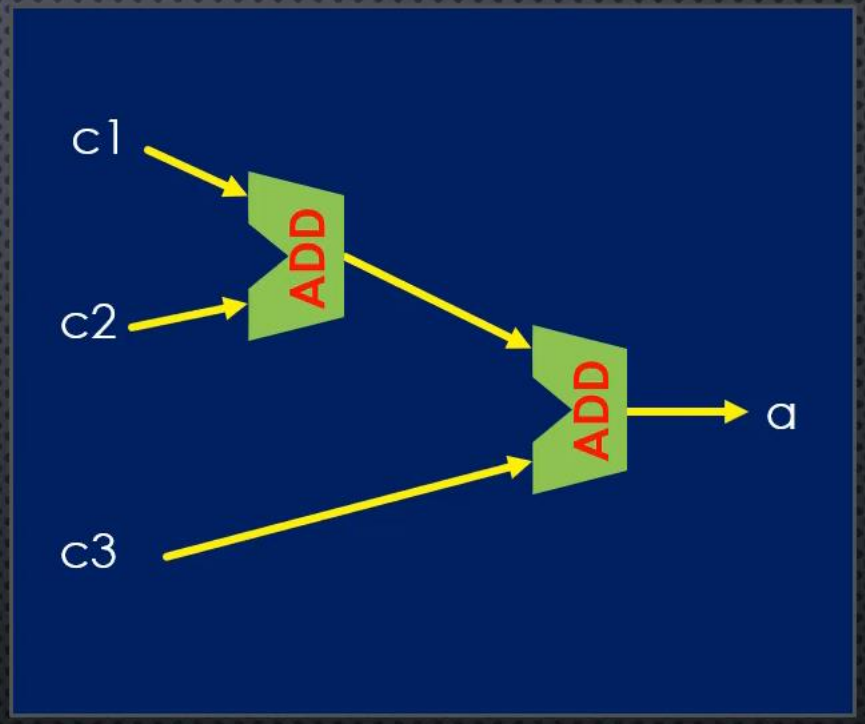
EXAMPLE-CPU


mov a, 0
add a, a, c1
add a, a, c1
add a, a, c3
memory
...
c3
c2
c1
02
01
00
a0
a1
a0=0
a1=0
a0=1
a1=0
ALU
ADD
a
load
reset
l
r
l=1
r=1
r=1
a0=0
a1=0
l=1
a0=1
a1=0
l=1
a0=0
a1=1
l=1
Phase 1
Phase 2
Phase 3
Phase 4

EXAMPLE CPU

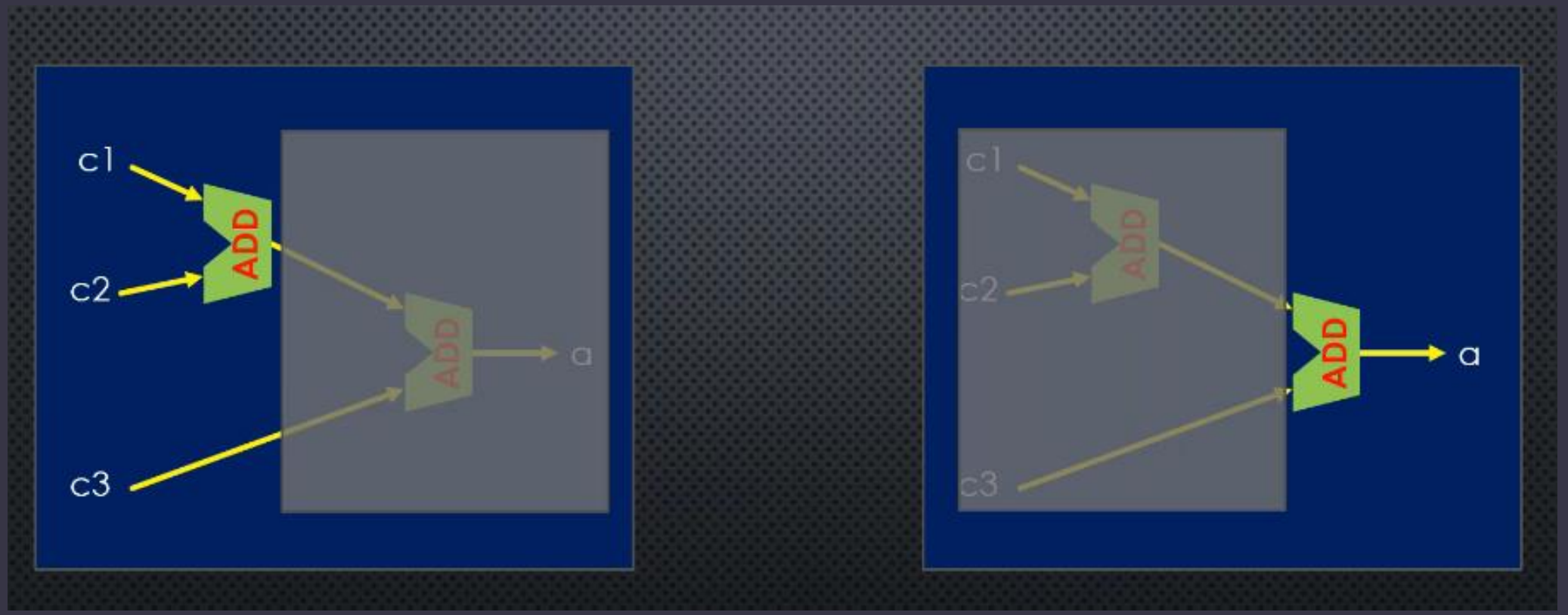
```
a = 0;
a = a+c1;
a = a+c2;
a = a+c3;
```

Code

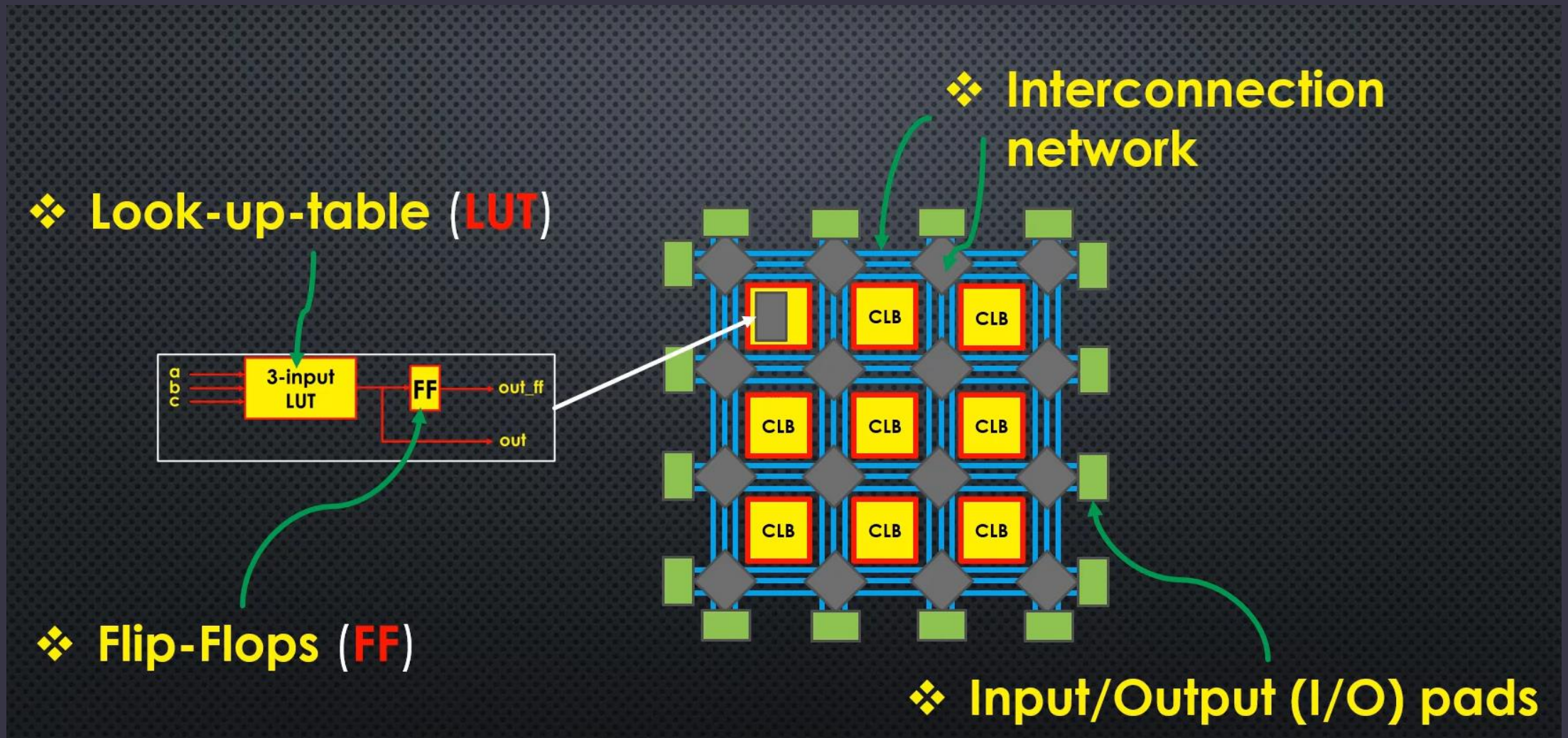


FPGA Computing Structure

EXAMPLE CPU



NEXT STEP


❖ Look-up-table (LUT)
❖ Interconnection network
a
b
c
3-input LUT
FF
out_ff
out
CLB
❖ Flip-Flops (FF)
❖ Input/Output (I/O) pads

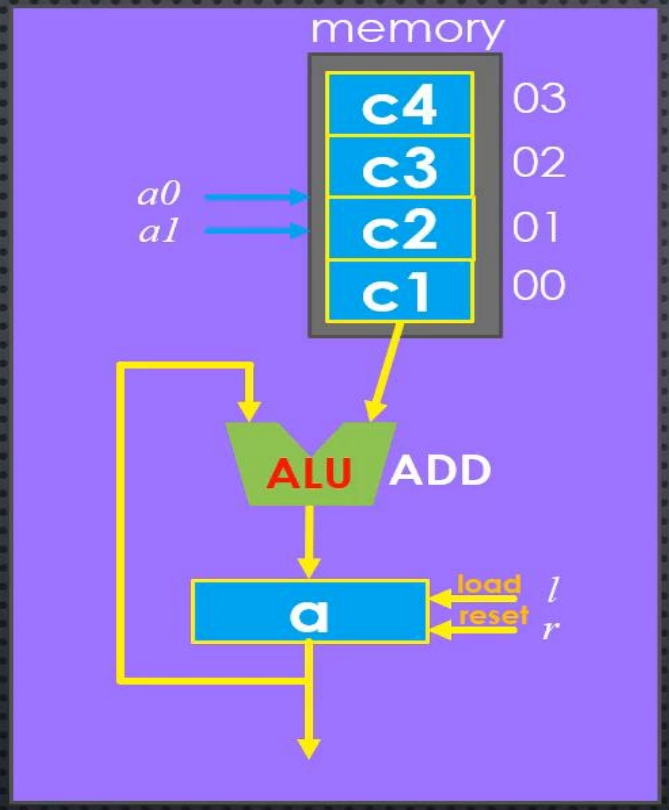
FPGA VS CPU - SUMMARY

- Whereas CPUs typically execute instructions sequentially, FPGAs run then in parallel
- Instruction dependency is important in FPGA to exploit parallelism.

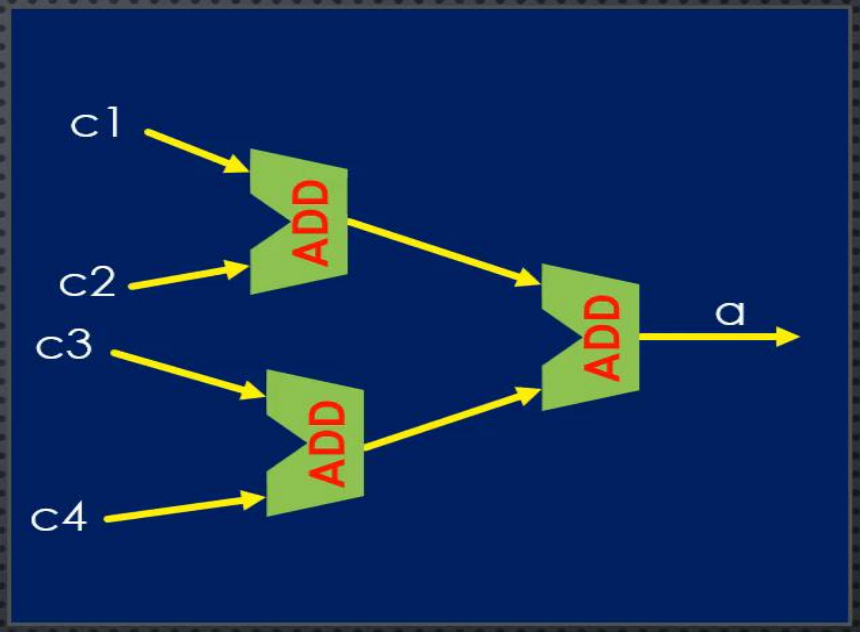
ASSIGNMENT Q-4

```
a = 0;
a = a+c1;
a = a+c2;
a = a+c3;
a = a+c4;
```

Code



CPU Computing Structure



FPGA Computing Structure

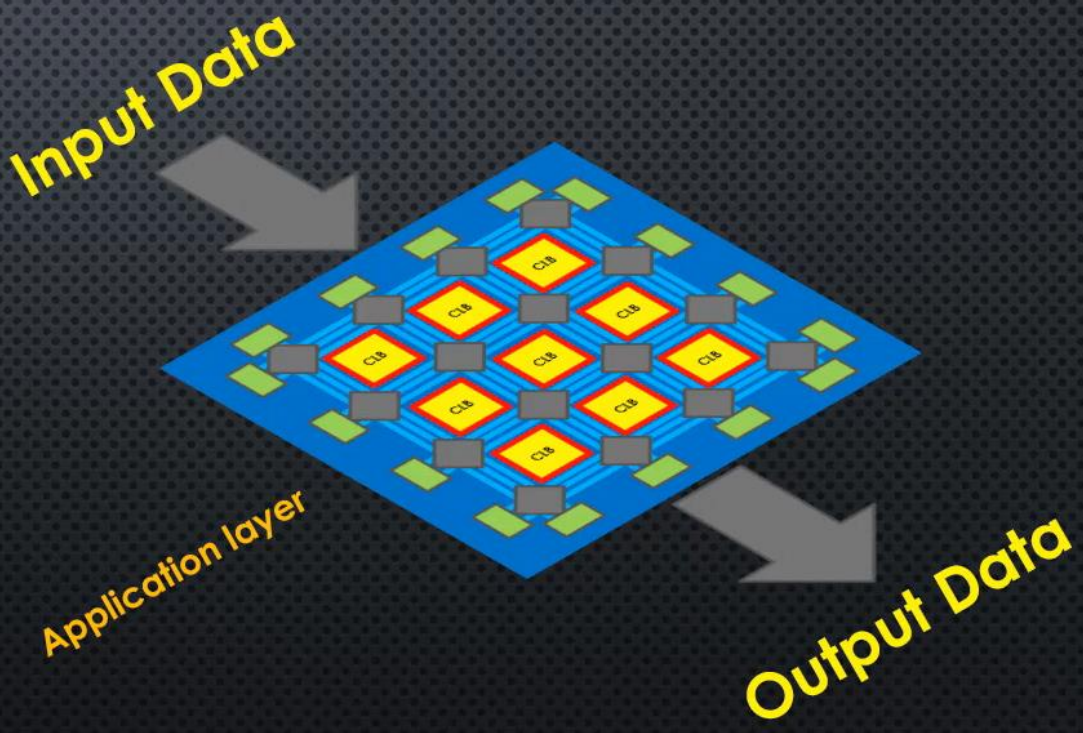
Show the execution steps of the code on CPU and FPGA.

FPGA BASICS

FPGA Concepts

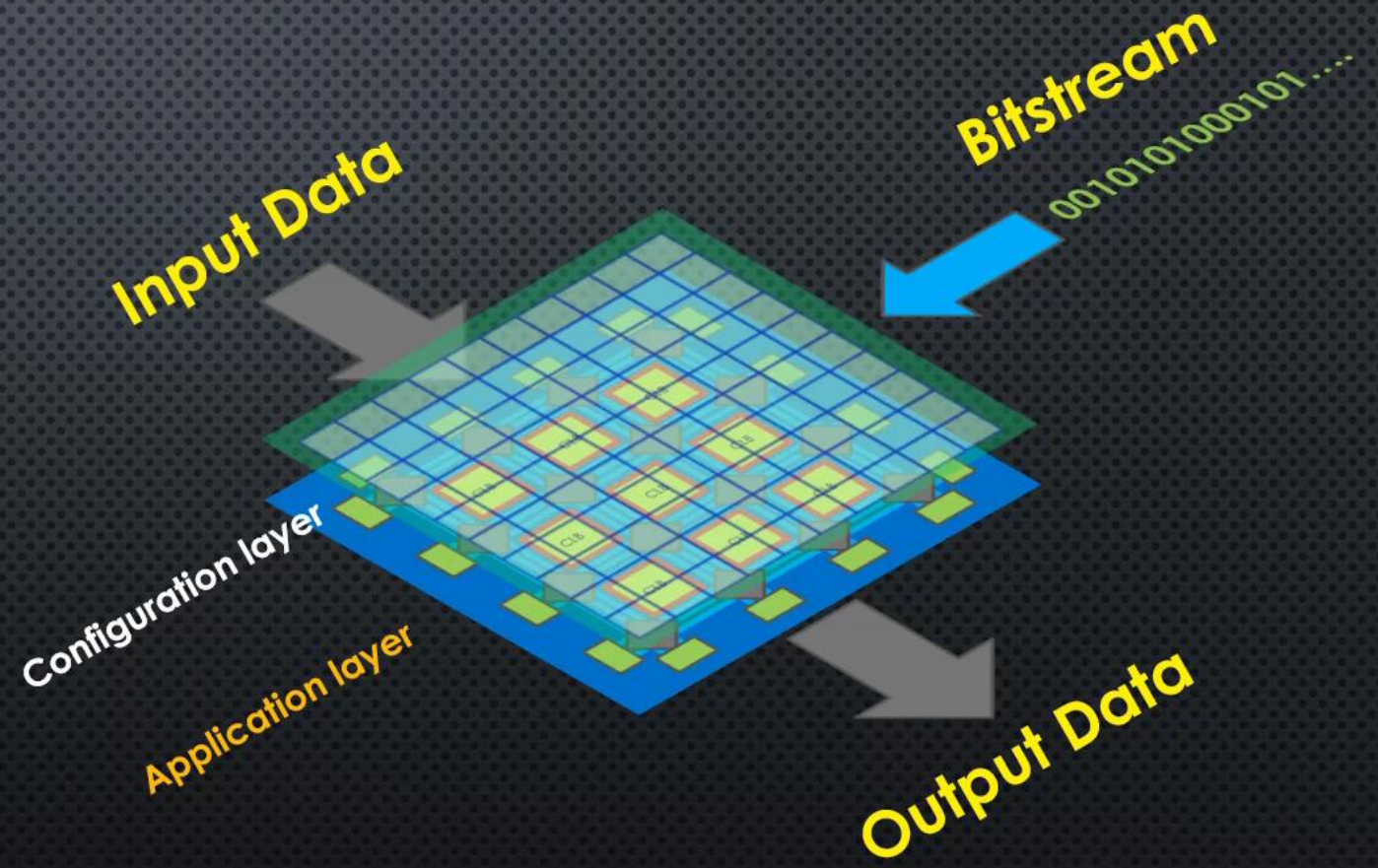
FPGA STRUCTURE

- Application Layer
- Configuration layer

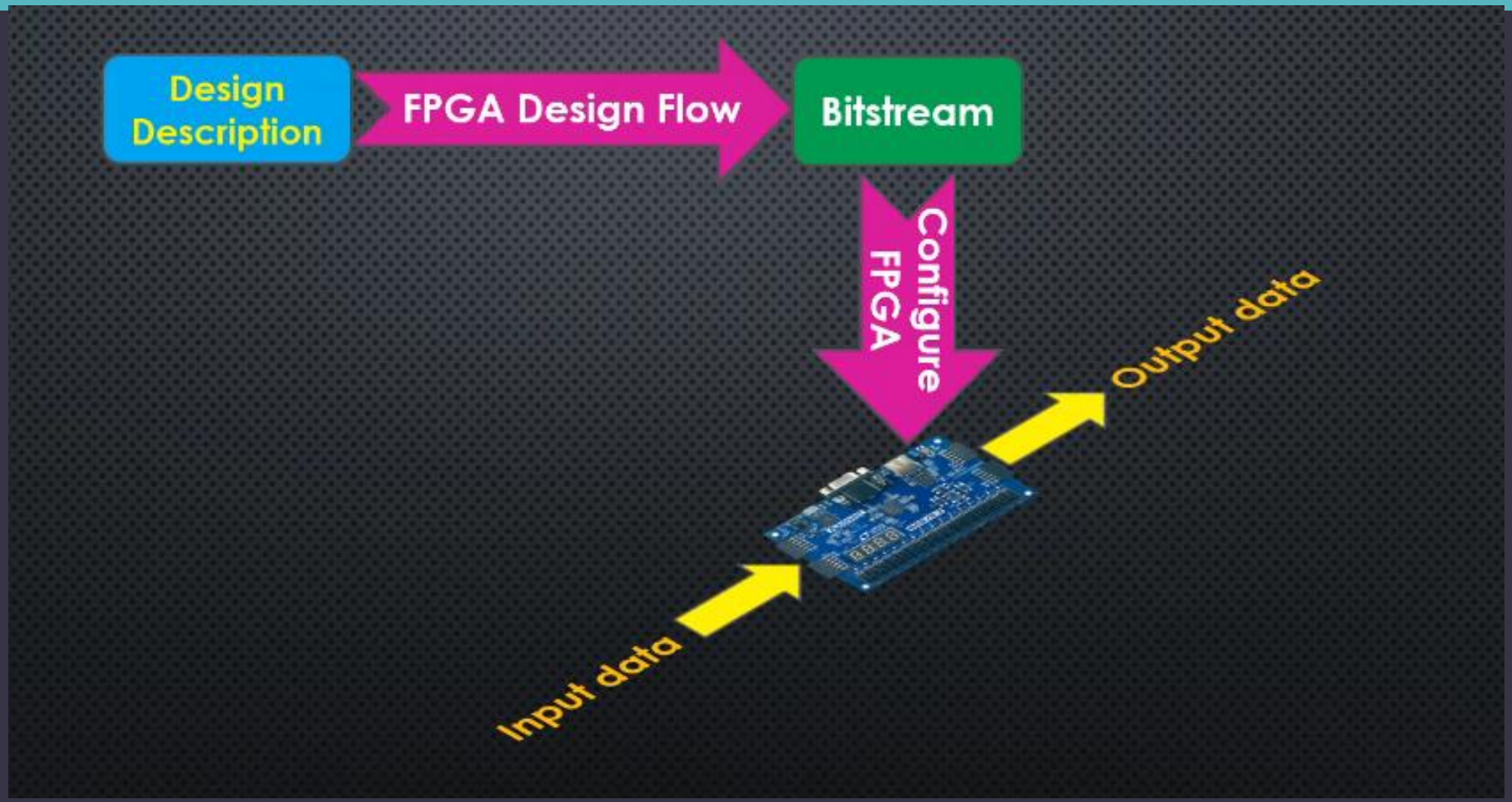


FPGA STRUCTURE

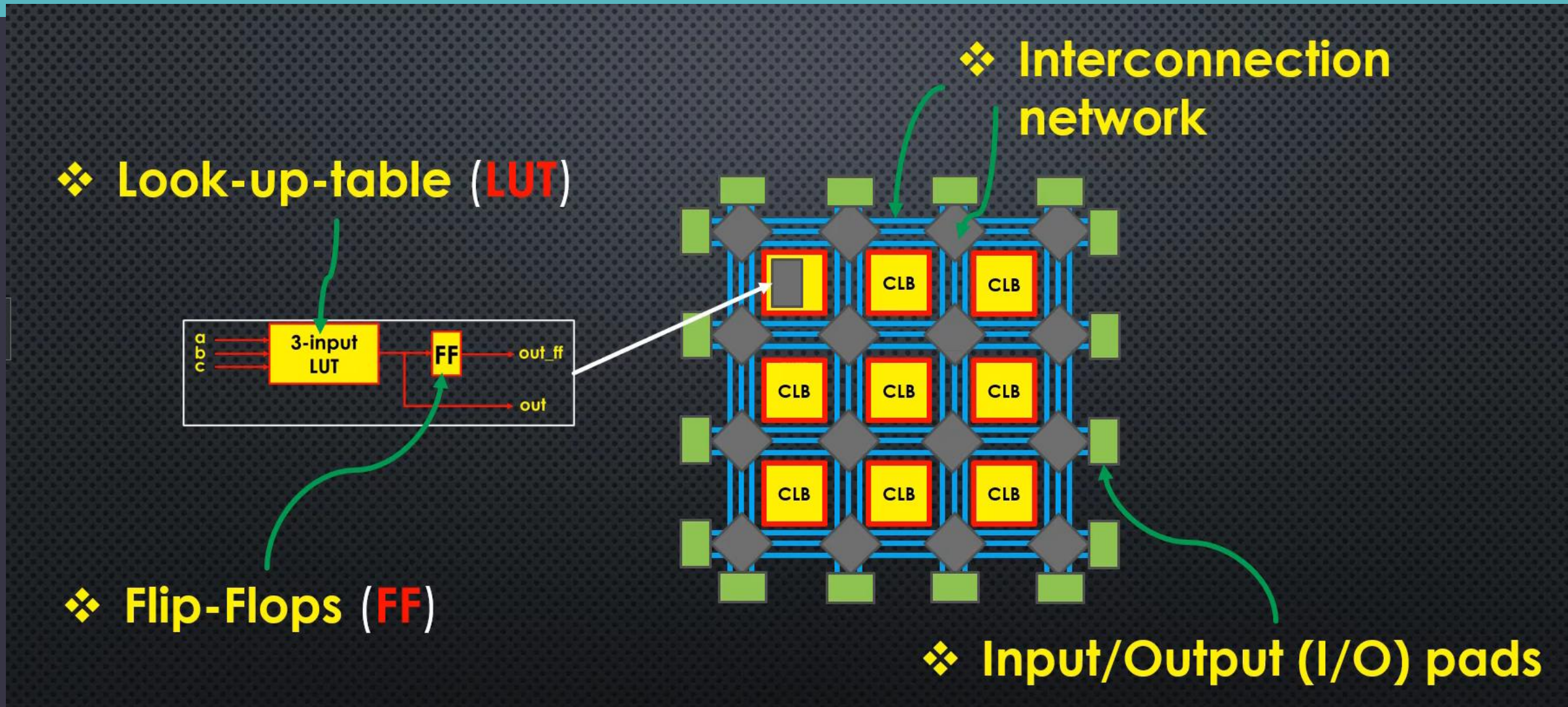
- Application Layer
- Configuration layer



FPGA DESIGN FLOW

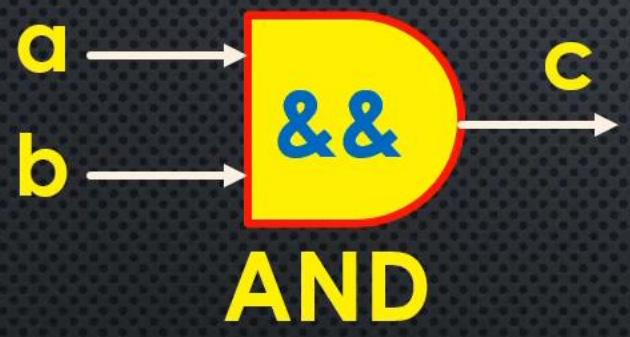


FPGA BASIC ELEMENTS



LOGICAL OPERATORS

1 = True
0 = False

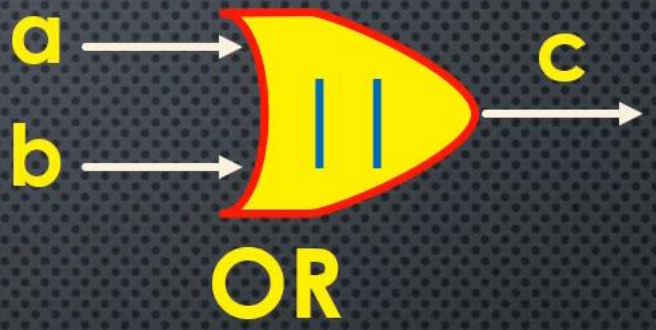


Truth Table

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

Truth Table

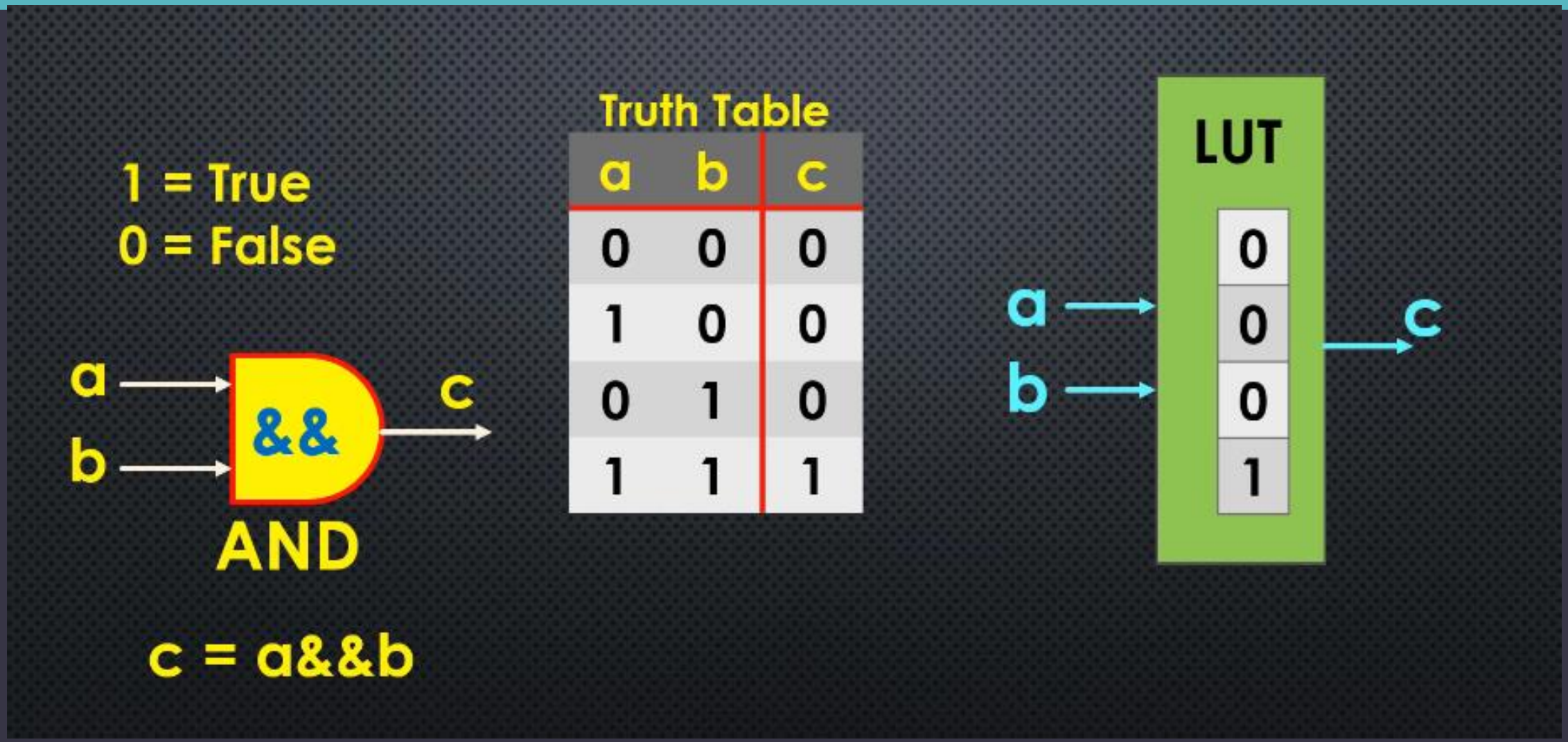
a	b
0	1
1	0



Truth Table

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

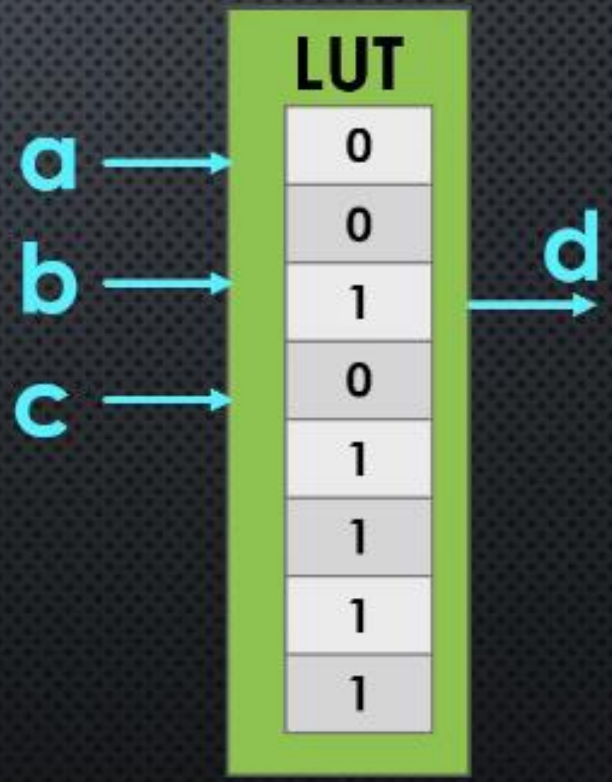
LUT


1 = True
0 = False
a
b
&&
c
AND
c = a&&b
Truth Table
a b c
0 0 0
1 0 0
0 1 0
1 1 1
LUT
0
0
0
1
a
b
c

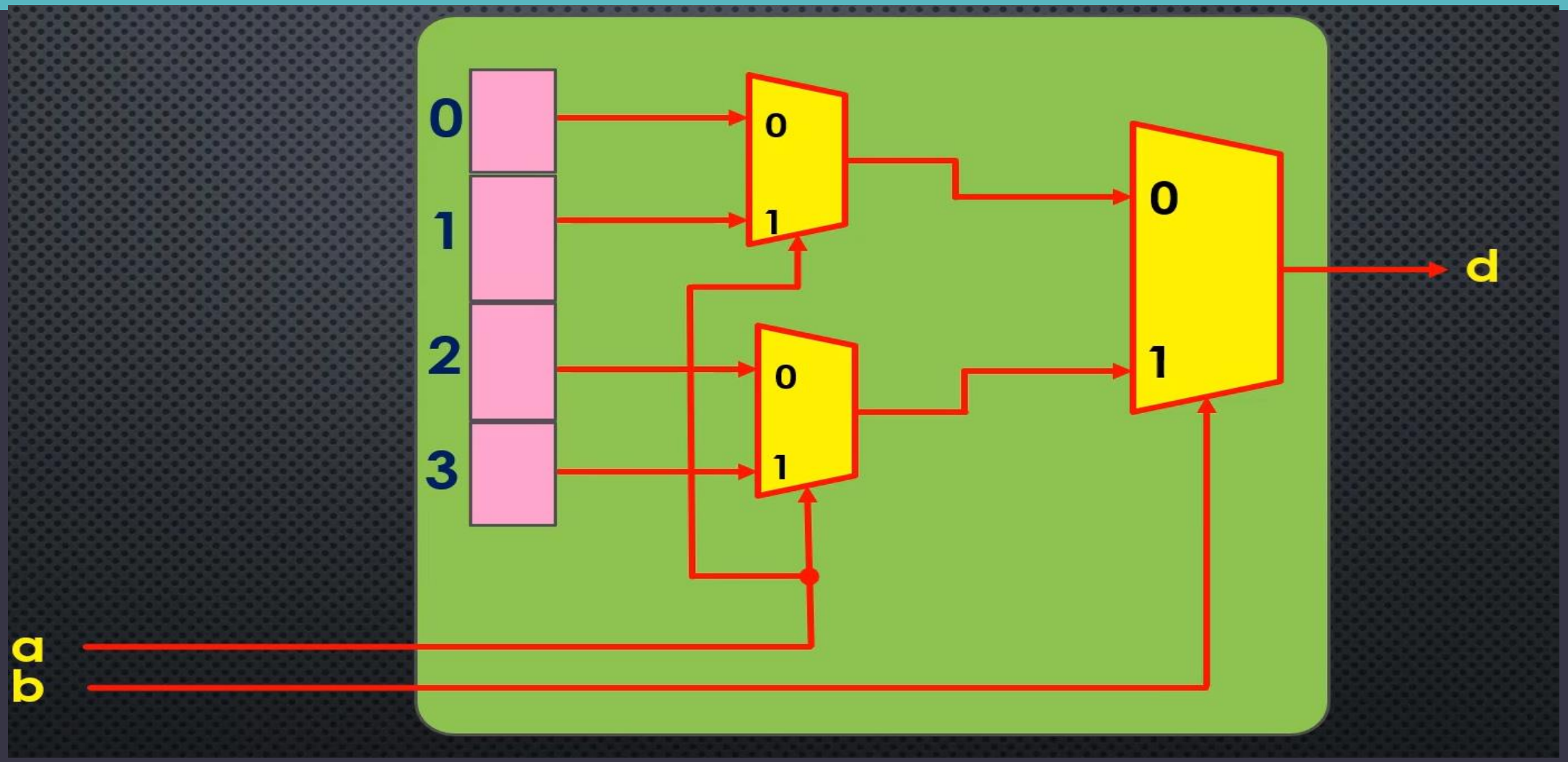
LUT - EXAMPLE

d = (!a && b) || c;

a	b	c	d
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	1



NEXT STEP



SUMMARY – FPGA BASICS

- A traditional FPGA consists of LUTs, FFs, I/O pins and finally a network connecting these elements
- All these components are reconfigurable.

ASSIGNMENT Q-5

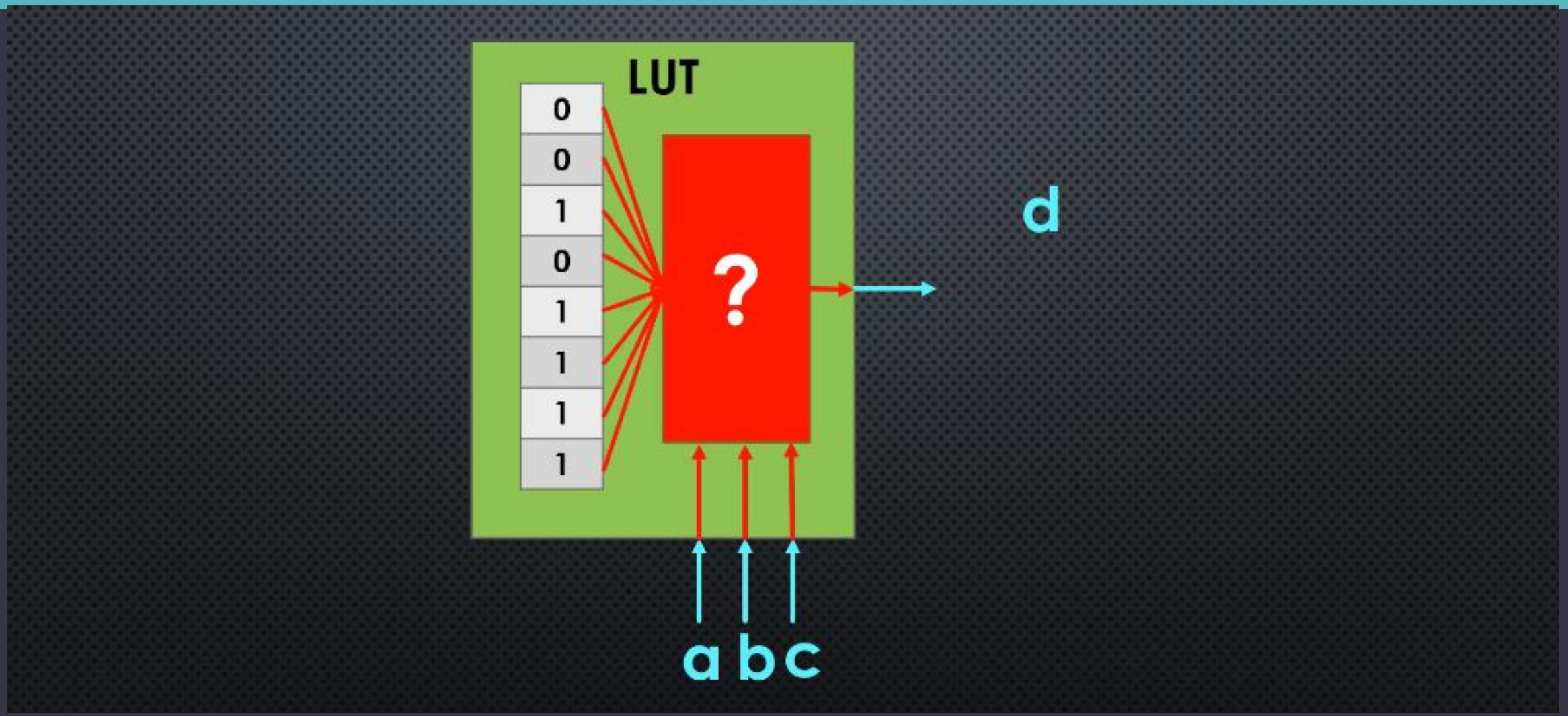
What is the LUT content for the following logic expression

```
d = (a && b) || (!c&&b);
```

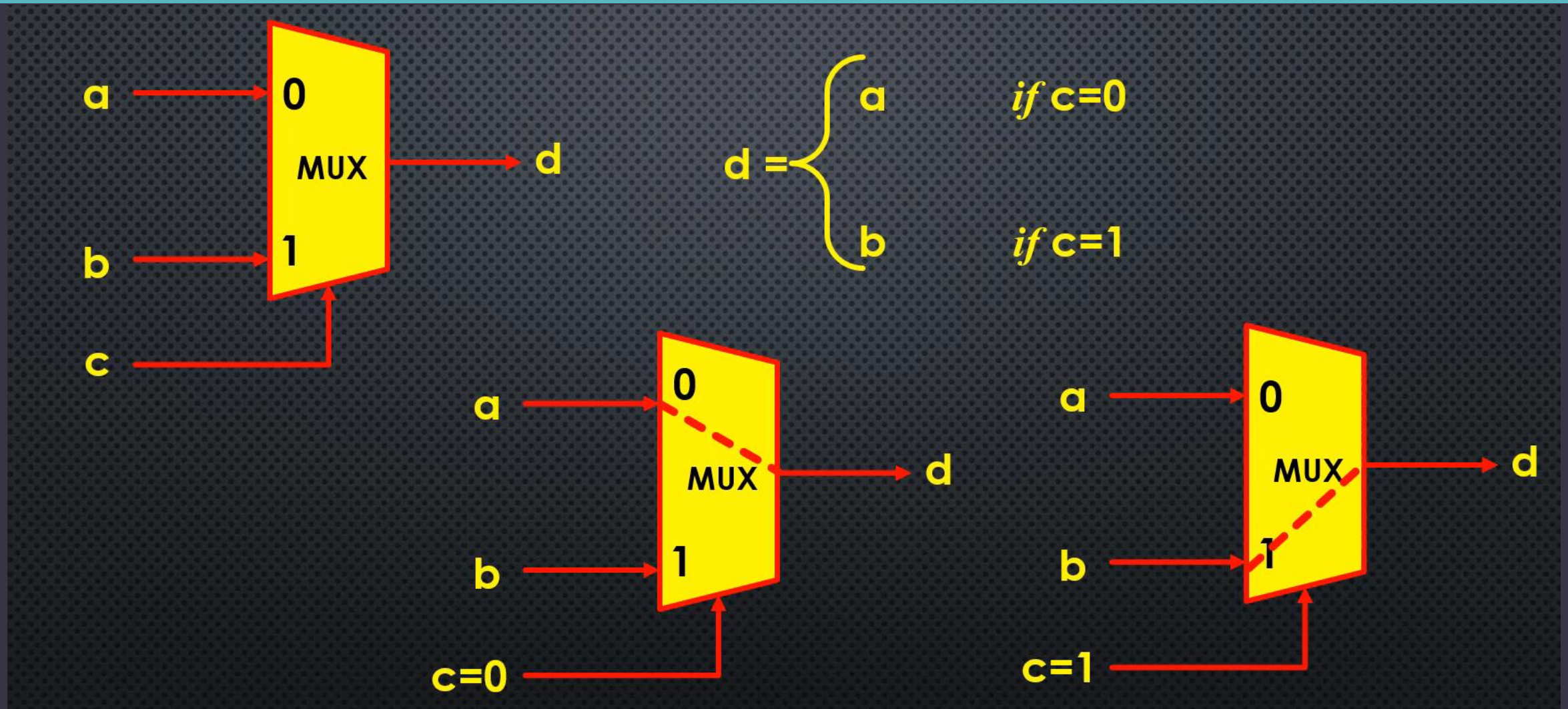
LOOKUP TABLE (LUT)

FPGA Concepts

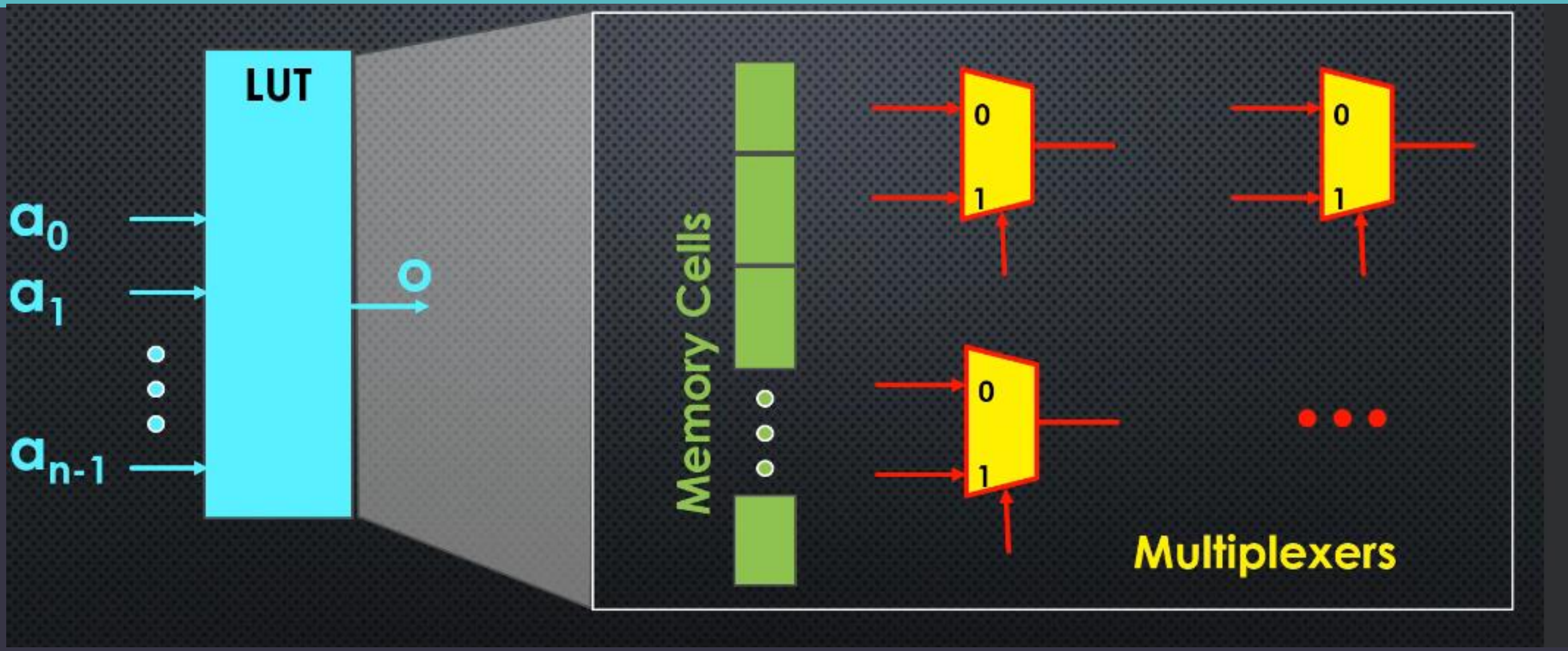
HOW TO SELECT THE OUTPUT?



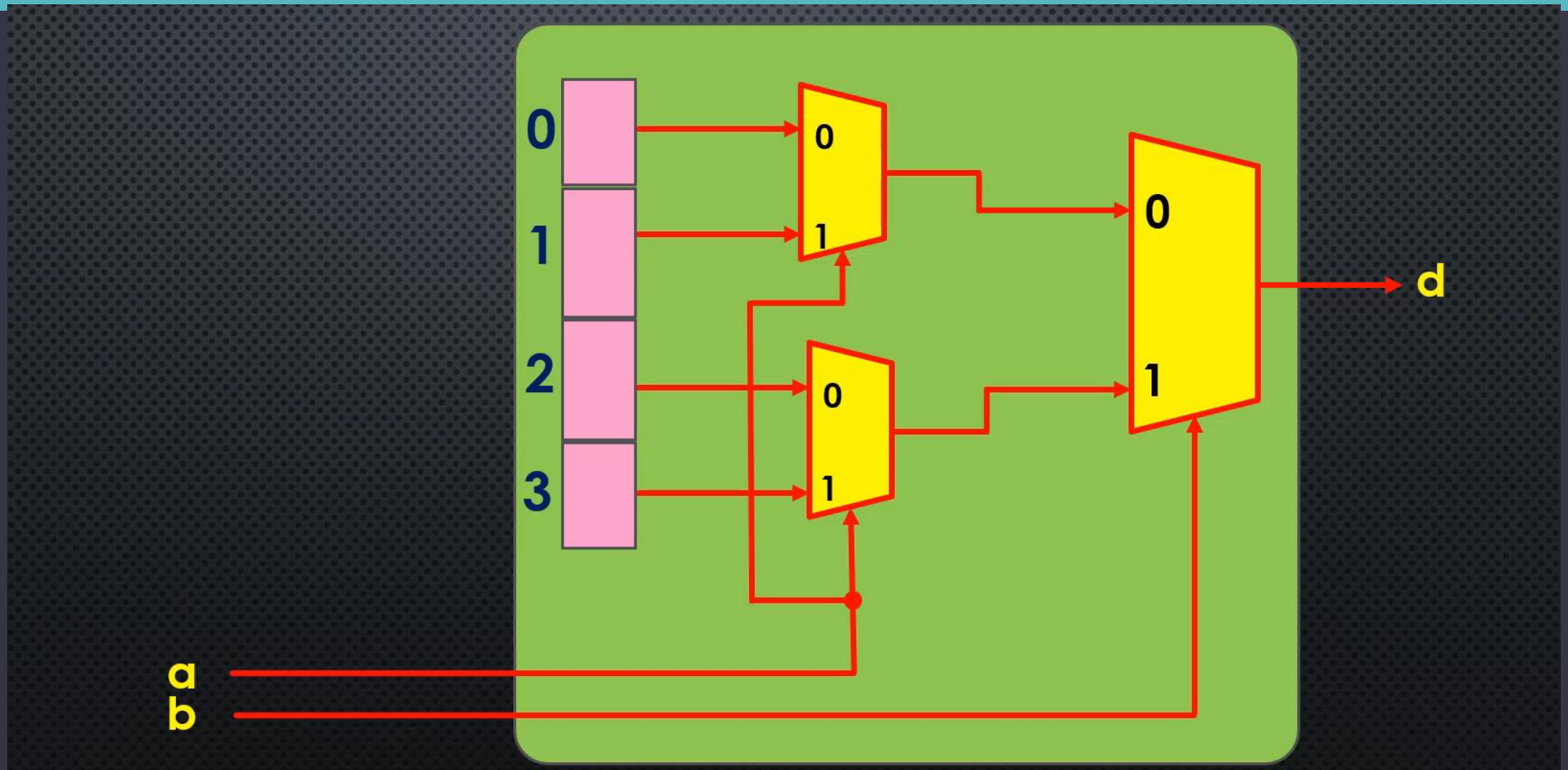
MULTIPLEXER (MUX)



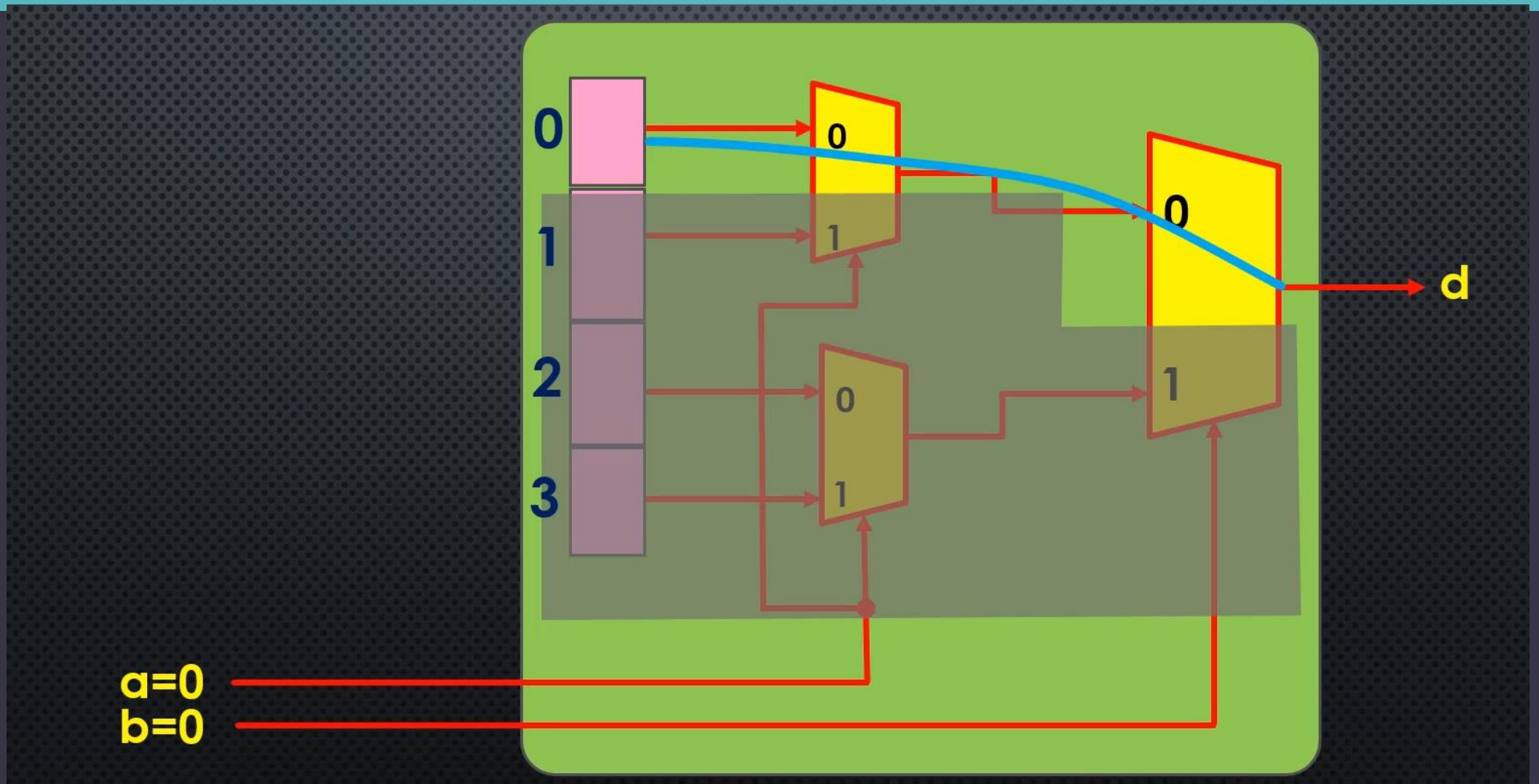
LUT STRUCTURE



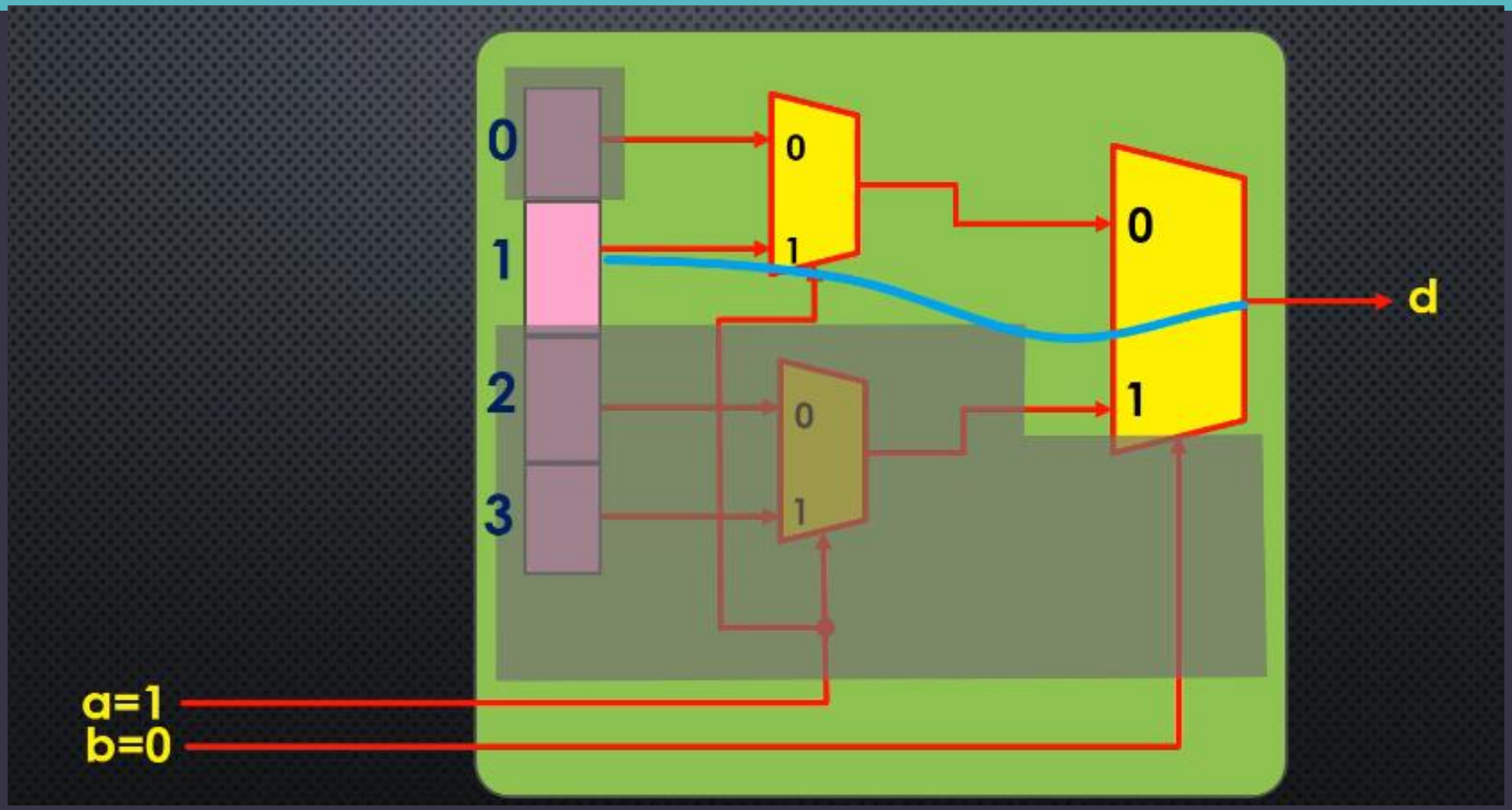
2-INPUT LUT



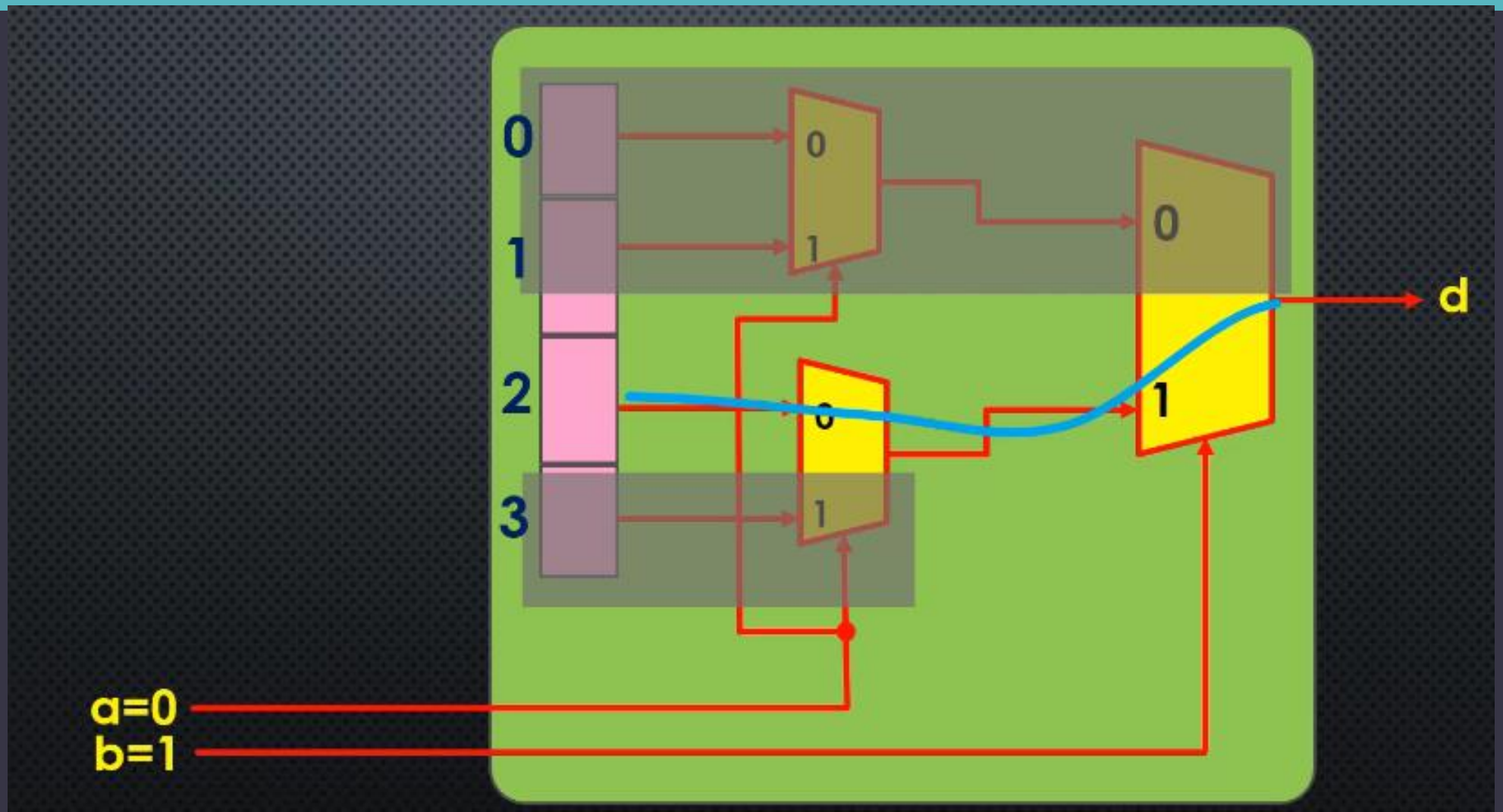
2-INPUT LUT



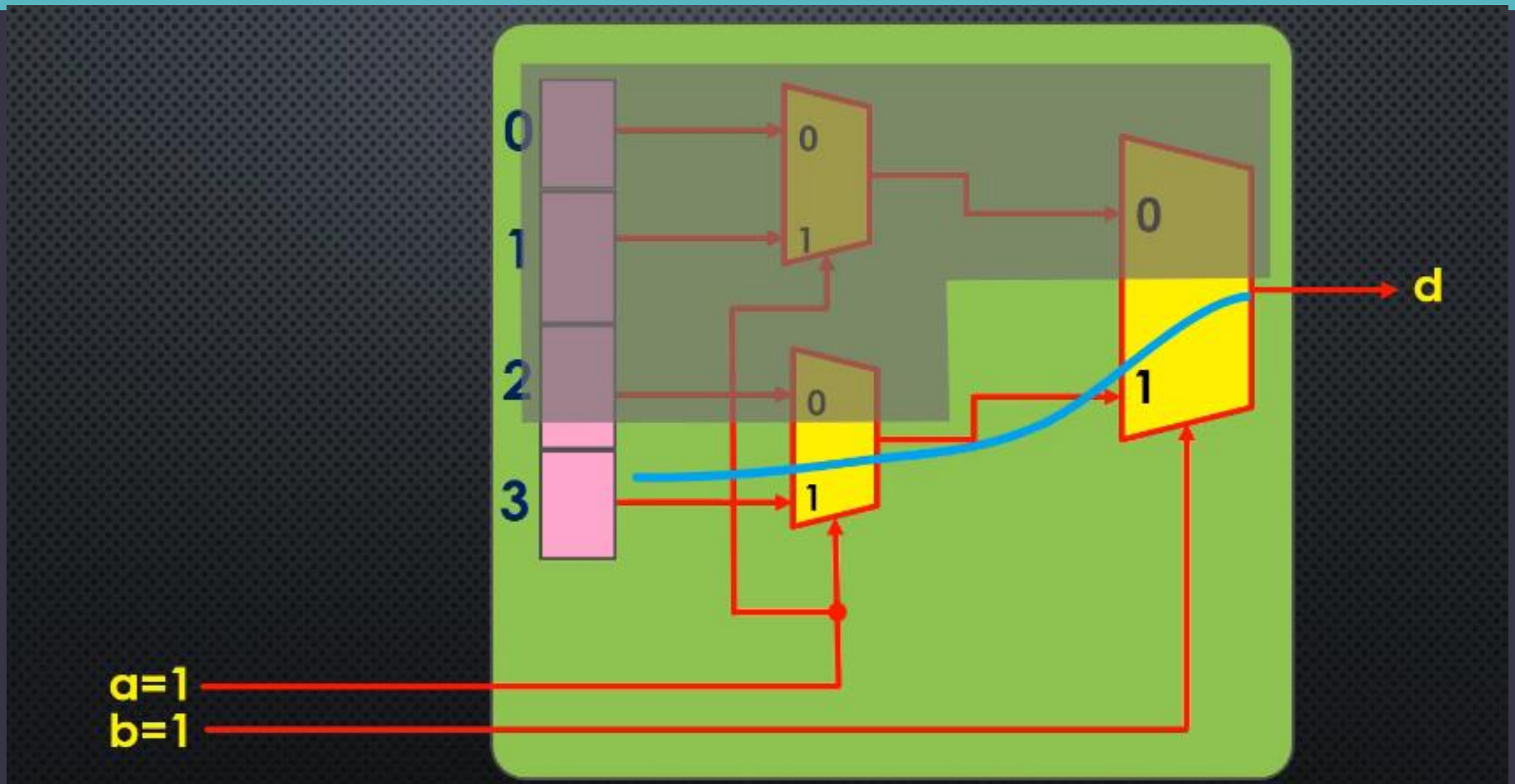
2-INPUT LUT



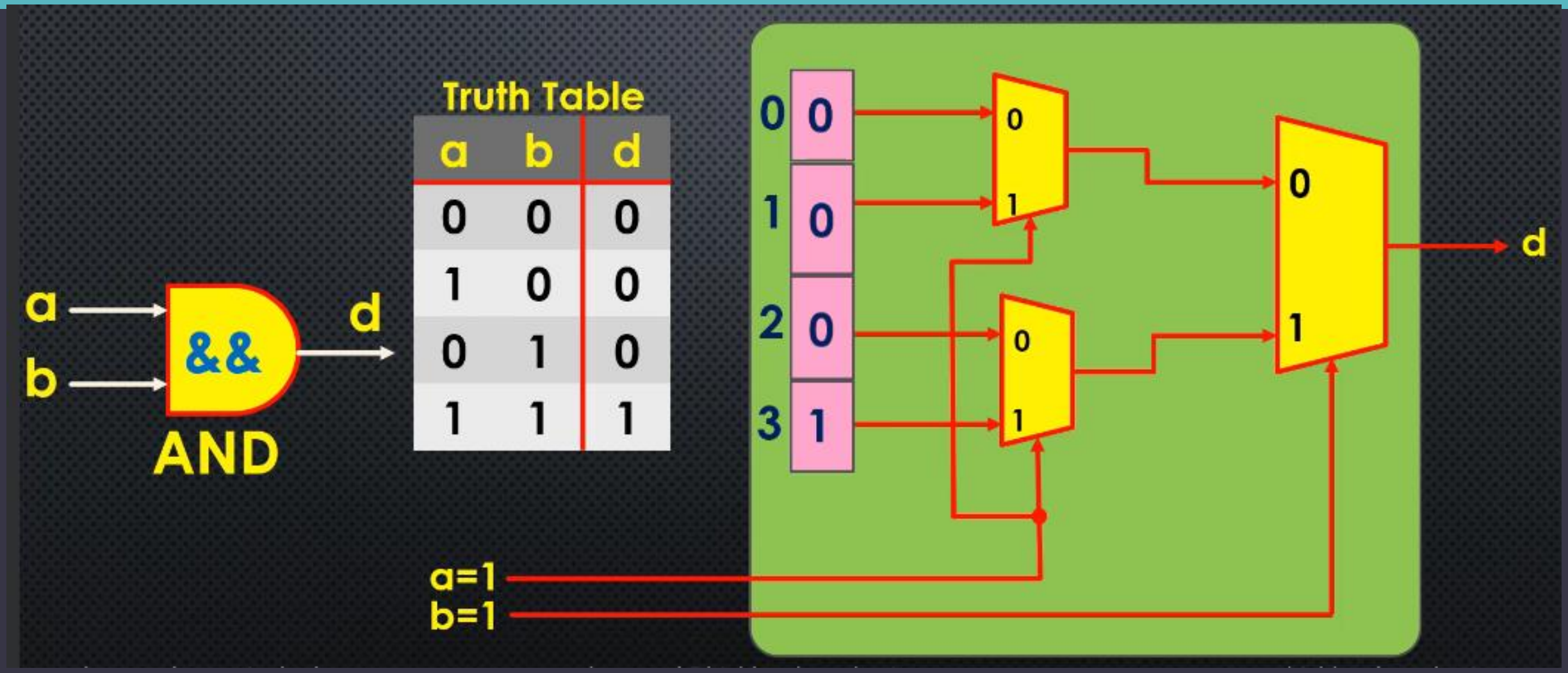
2-INPUT LUT



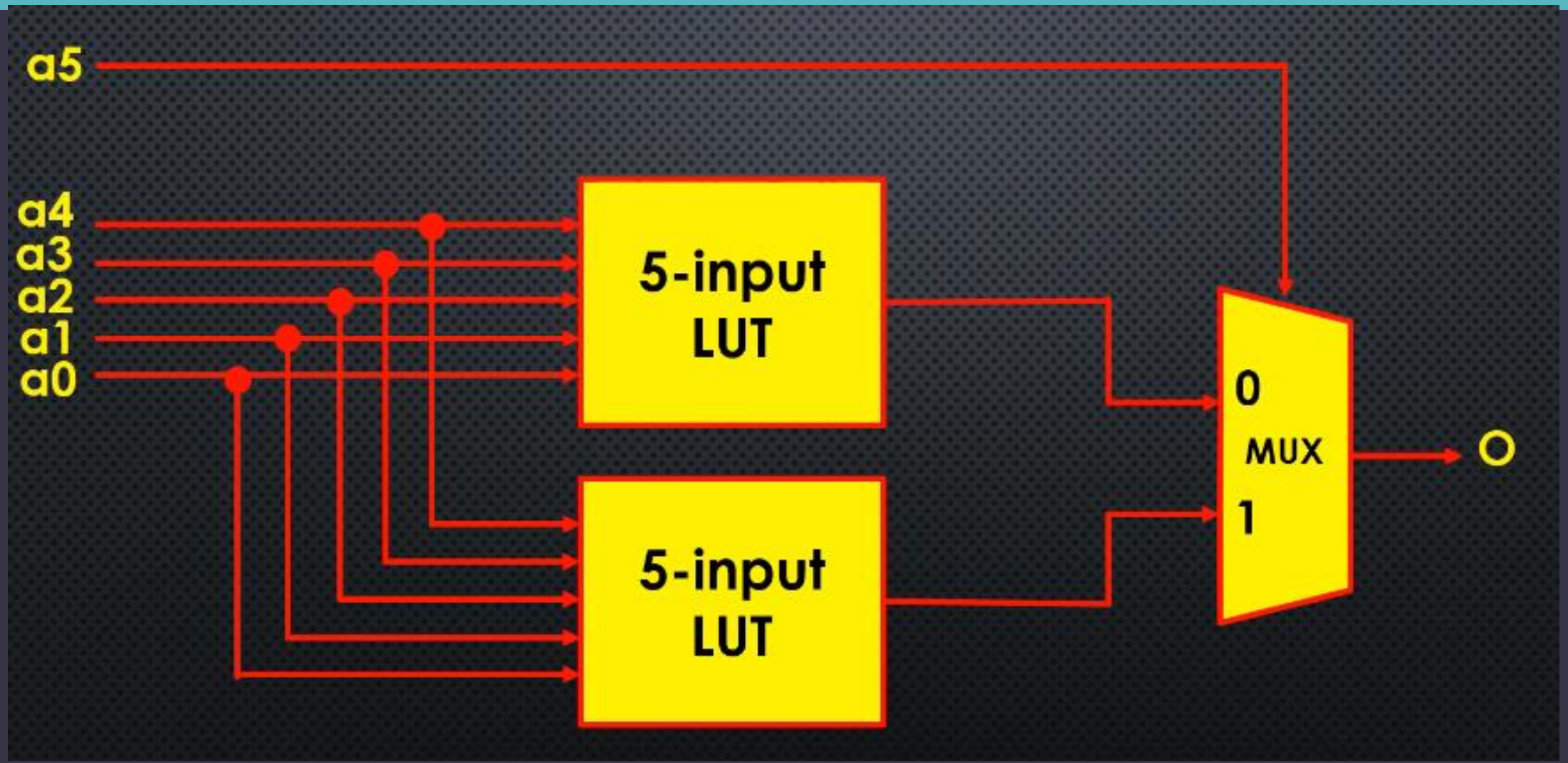
2-INPUT LUT



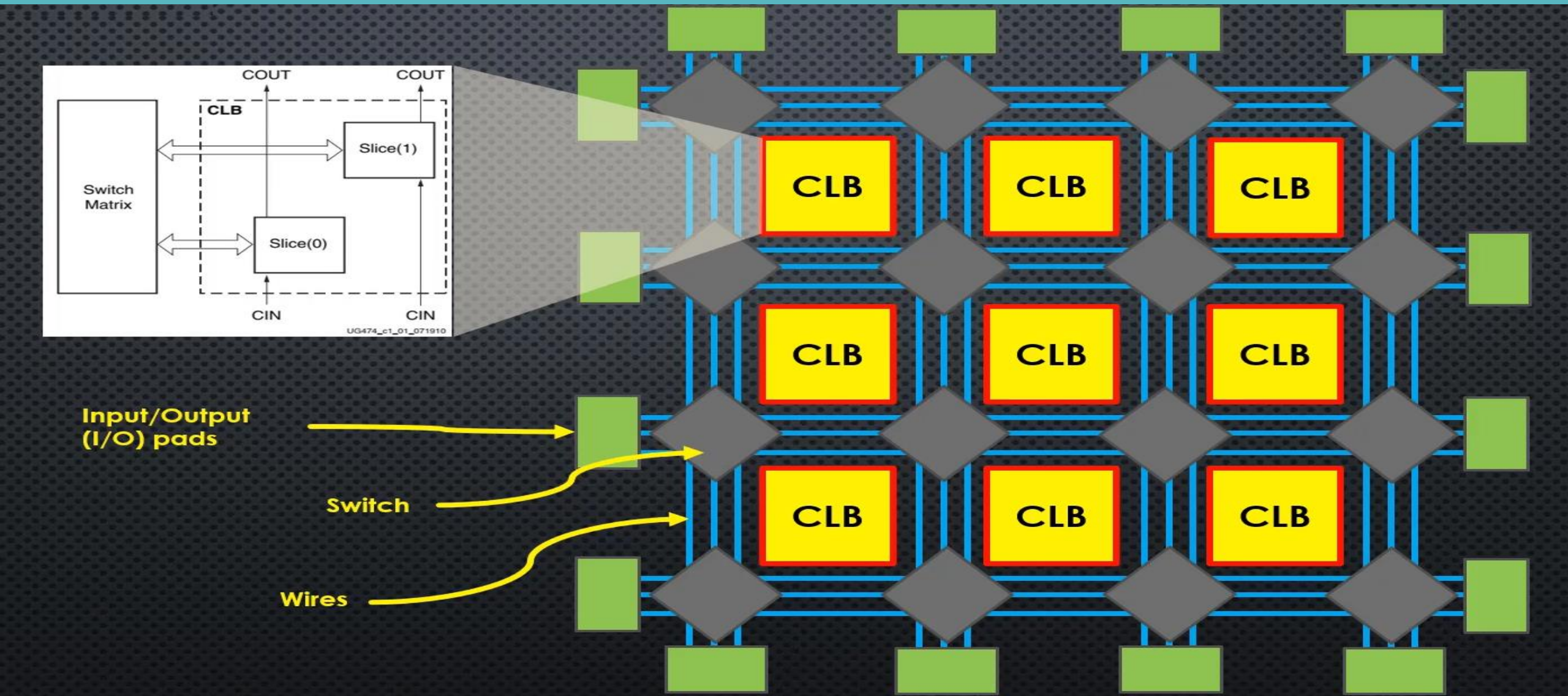
2-INPUT LUT


a
b
&&
d
AND
Truth Table
a b d
0 0 0
1 0 0
0 1 0
1 1 1
0 0
1 0
2 0
3 1
0
1
0
1
0
1
d
a=1
b=1

LUT - EXAMPLE



NEXT


COUT
COUT
CLB
Slice(1)
Switch Matrix
Slice(0)
CIN
CIN
UG474_c1_01_071910
CLB
Input/Output (I/O) pads
Switch
Wires

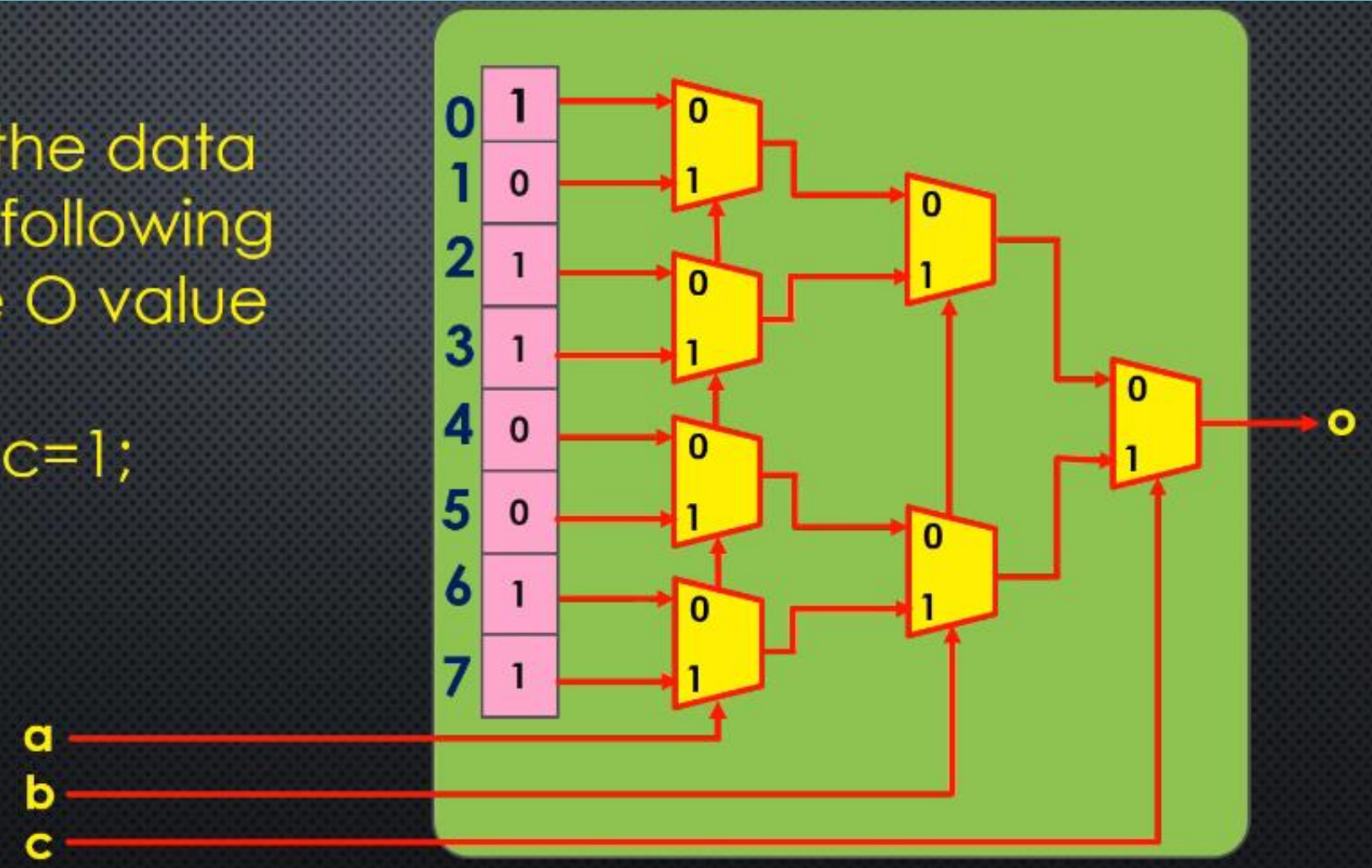
SUMMARY - LUT

The multiplexer circuit concept is used to connect LUT's reconfigurable memory cells to the output

ASSIGNMENT Q-6

Determine the data path in the following LUT and the O value

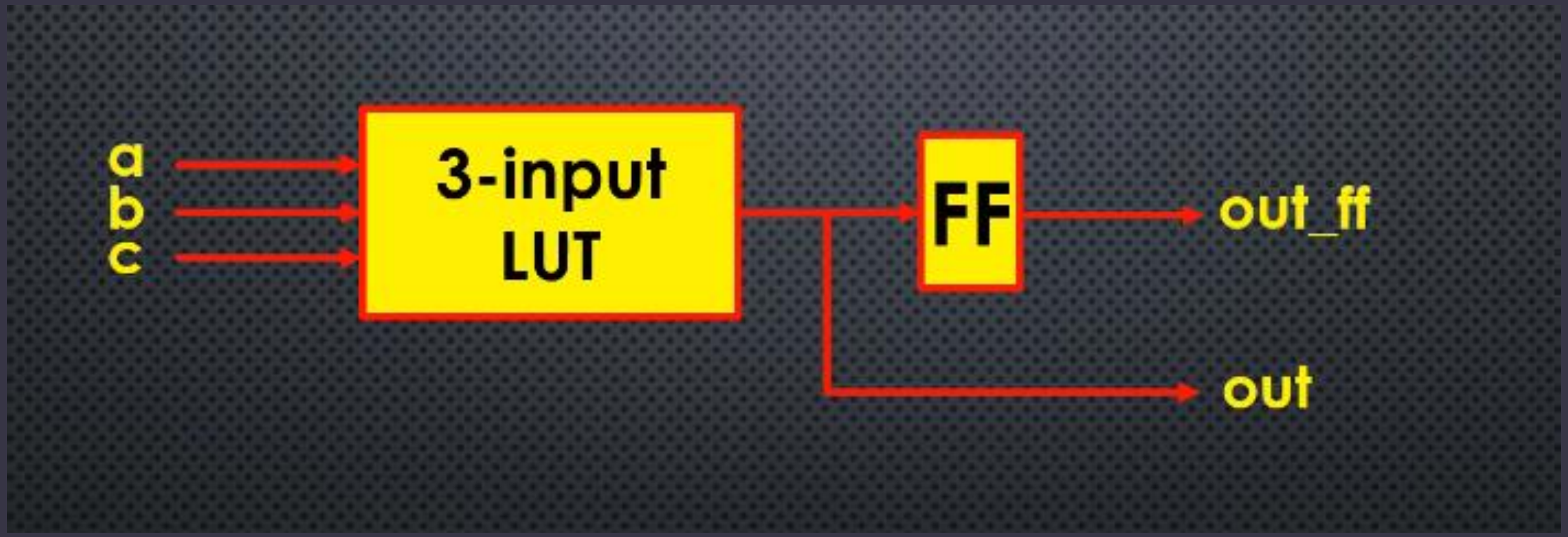
if a=0, b=1, c=1;



FLIP FLOP AND OTHER ELEMENTS

FPGA Concepts

FLIP FLOPS(FF)



SLICES

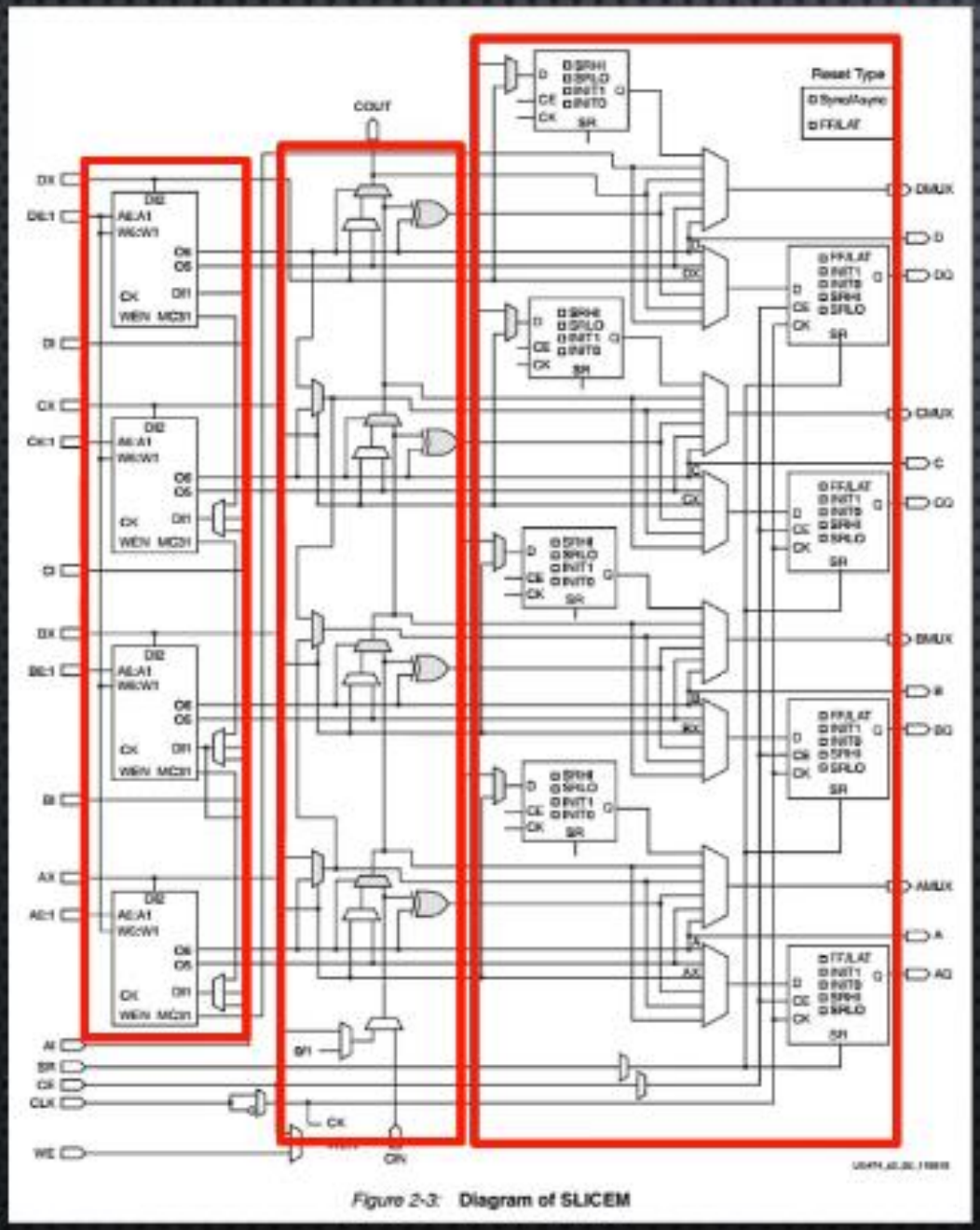
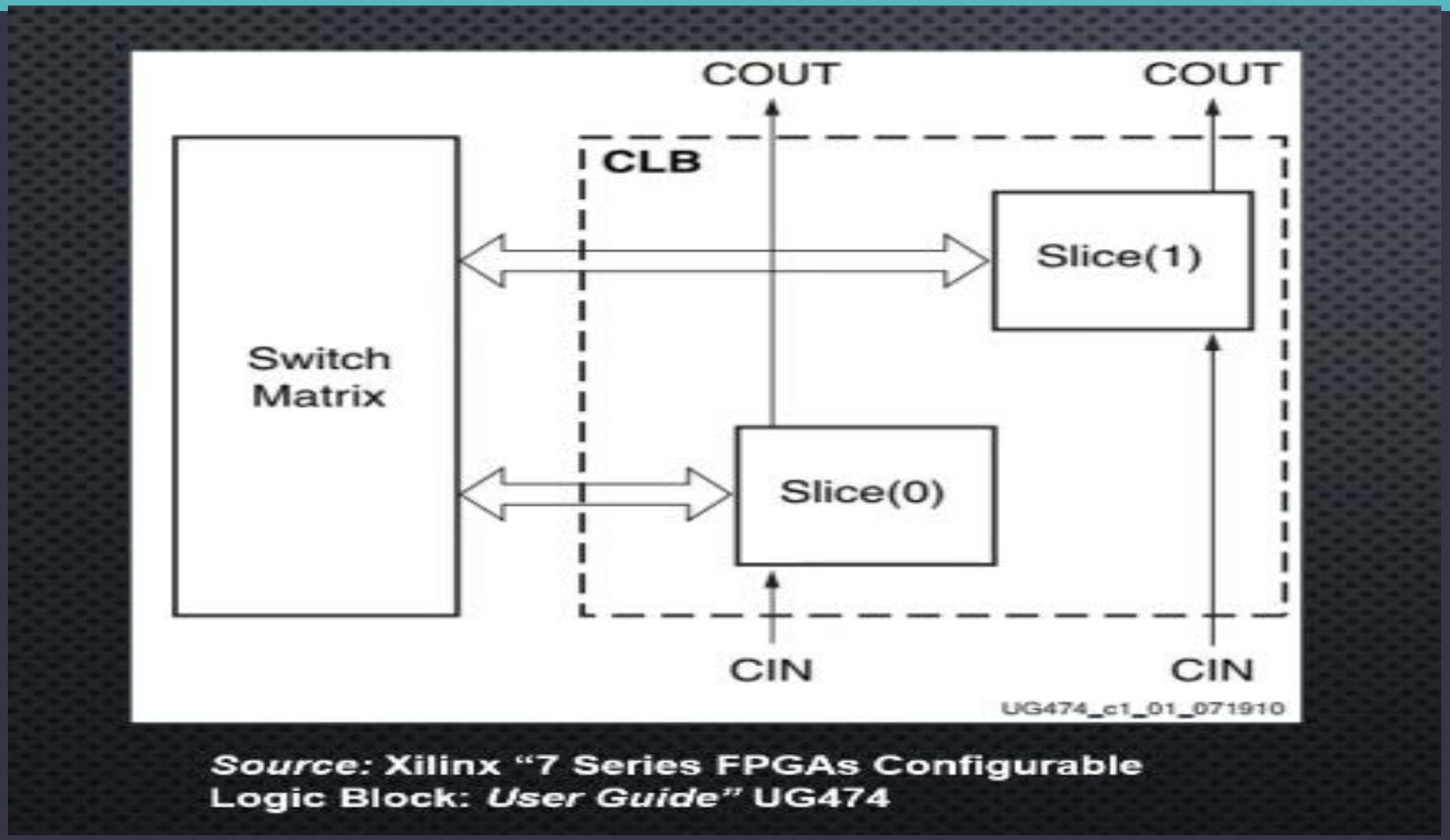


Figure 2-3: Diagram of SLICEM

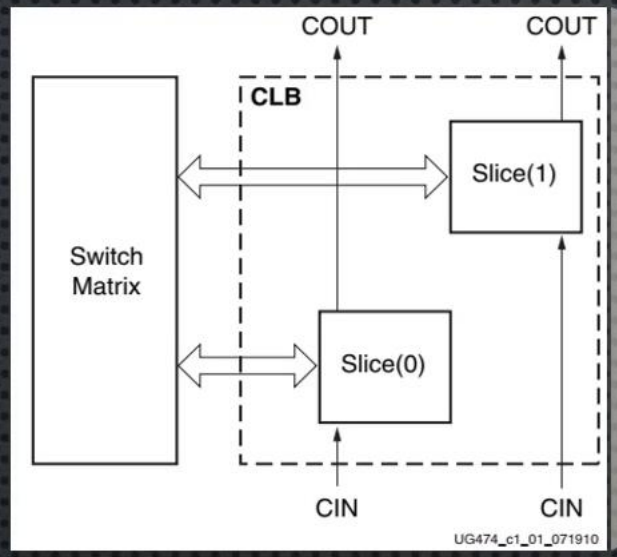
Source: Xilinx "7 Series FPGAs Configurable Logic Block: *User Guide*" UG474

CONFIGURATION LOGIC BLOCK



Source: Xilinx "7 Series FPGAs Configurable Logic Block: *User Guide*" UG474

FPGA ARCHITECTURE



CLB CLB CLB
CLB CLB CLB
CLB CLB CLB

Input/Output (I/O) pads

Switch

Wires

COMPUTATIONAL AND DATA STORAGE BLOCKS

These additional elements include

- Block RAM (BRAM) to act as a single or multiple memory to save large amount of data
- Phase-locked loops (PLLs) for driving the FPGA fabric at different clock rates
- High-speed serial transceivers
- Off-chip memory controllers
- Multiply-accumulate blocks (DSP)
- Embedded multi-core CPU

SUMMARY

- An FPGA has computational, storage and additional elements
- Combinational elements can be used for logical and arithmetic operations
- The storage elements can be used as registers or blocks of memories
- Additional elements can be used for arithmetic operations, serial communication or other functions

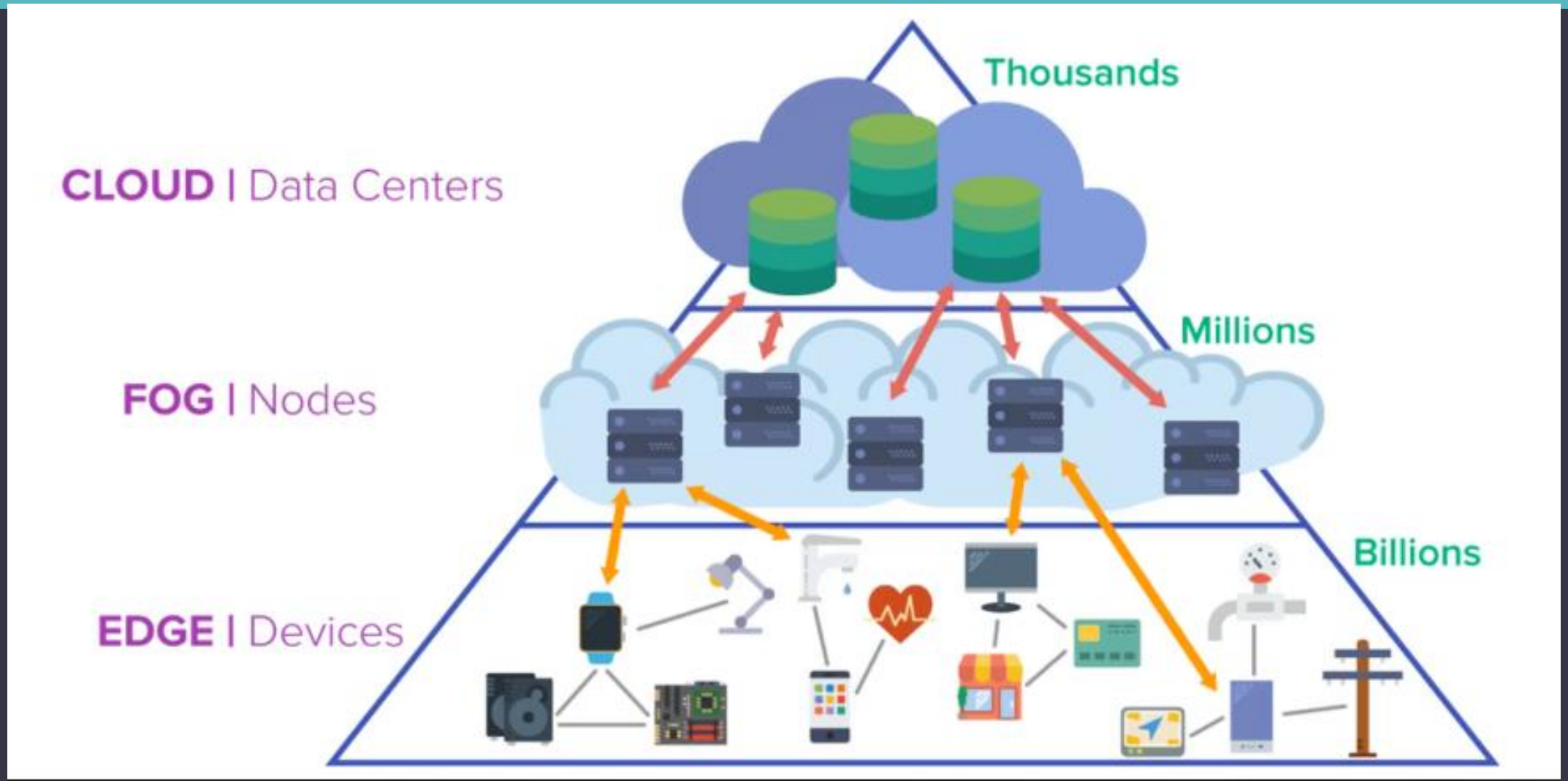
ASSIGNMENT Q-7

1) What are the two main memory resources in an FPGA?
2) Name two elements that can implement arithmetic operators?

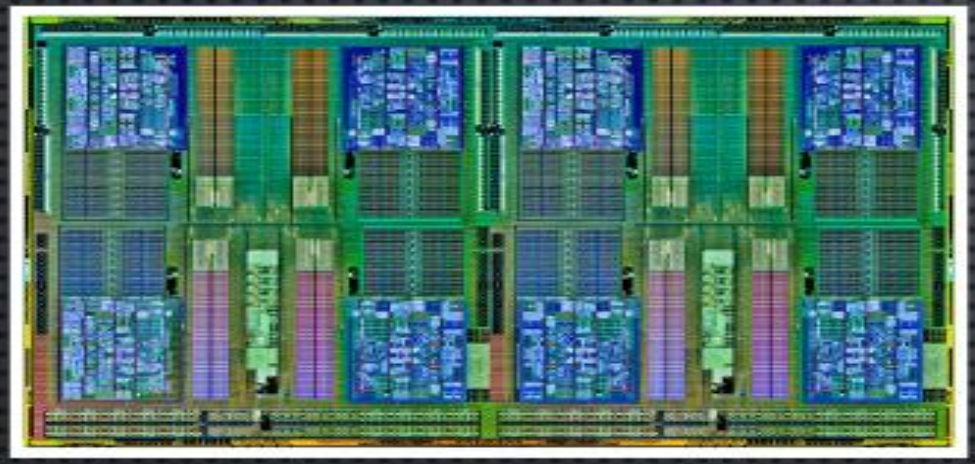
WHY HLS

FPGA Concepts

APPLICATION AREAS



COMPUTING PLATFORMS



multi-core CPU

many core GPU

COMPUTING PLATFORMS - ISSUES

High power and energy consumption

They are not so efficient

Not easy to modify

FPGA-BASED COMPUTING

Video and Imaging

XILINX

XILINX
NGCODEC VYUSYNC
Adaptive Bit Rate Video Transcoding Solution

MEGH COMPUTING
Megh Computing - Video Analytics Solution

DEEPOLY
Ultra High Performance Image Processing Solution

Video and Image Processing
BOSTON
Servers | Storage | Solutions
Boston Stream +

Industrial

XILINX

Computational Storage

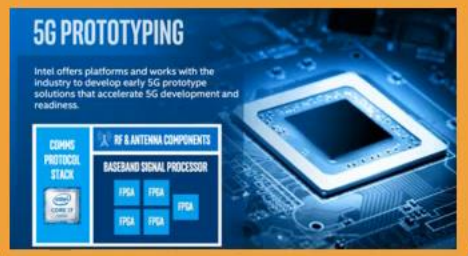
bigstream
Hyper-Acceleration Software

EIDETICOM
Database Acceleration

Networking and Security
SILEX INSIGHT
77x TLS Handshake Hardware Accelerator for

Networking and Security
ZoTech
AMI for Hardware-accelerated RSA Operations

XILINX



5G Wireless

Financial Technology

XILINX
Binomial Options Model

NTT Disruption
Accelerated Market Risk

ALGO-LOGIC
Pre-Trade Risk Check

Financial Computing
MAXELER
Standard Initial Margin Model (SIMM) calculation

XILINX

High Performance Computing

illumina
DRAGEN Complete Ultra-rapid analysis of Next Generation Sequencing (NGS)
Acceleration vs. CPU: 90X

byte LAKE
CFD Kernels optimized for Xilinx Alveo
Acceleration vs CPU: up to 4x

High Performance Computing
opensource.com
Bellman-Ford algorithm for solving single-source

High Performance Computing
NECST laboratory
ProFAX: Hardware Acceleration of a Protein

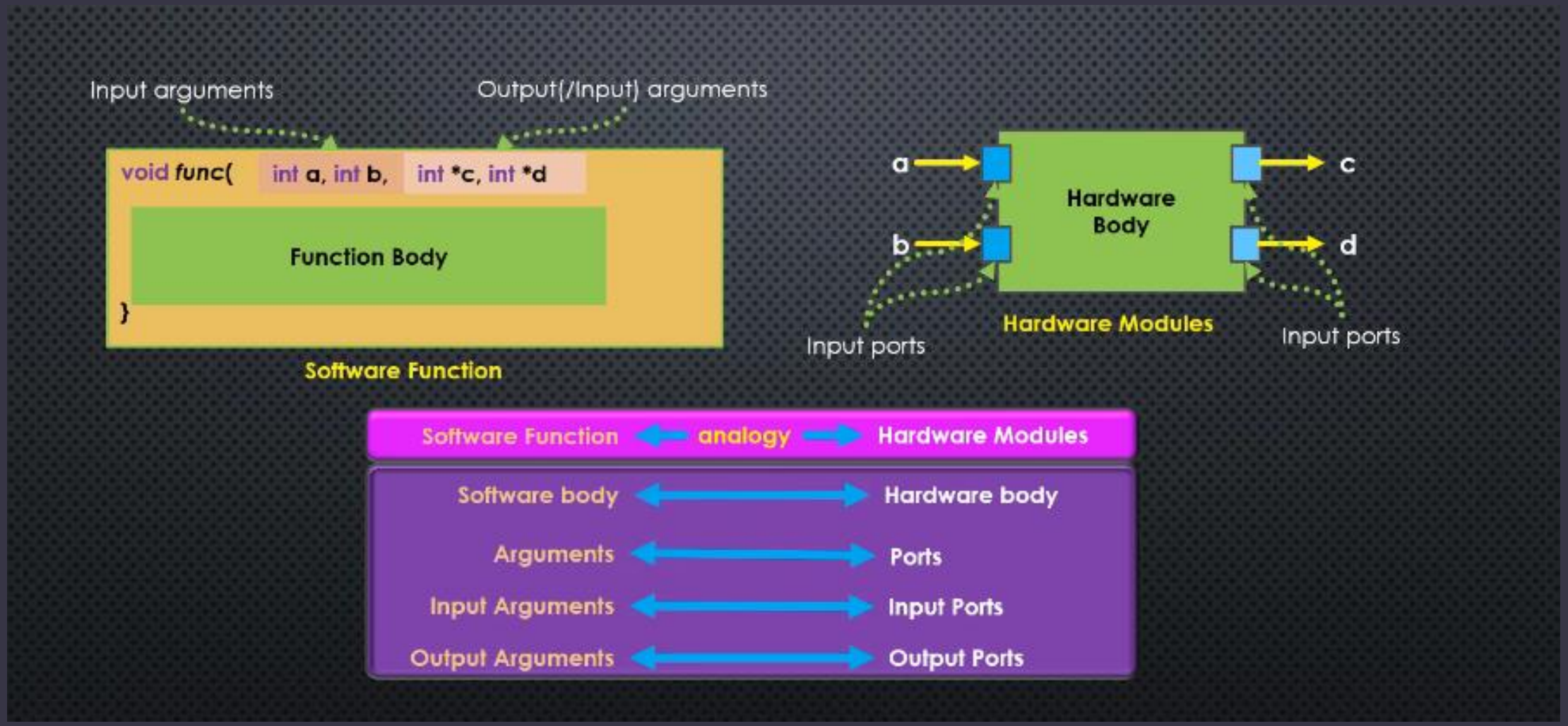
High Performance Computing
MAXELER
Maxeler Conjugate Gradient IP

XILINX

Automotive

Using FPGAs Mission-Critic

NEXT


Input arguments
Output(/Input) arguments
void func(int a, int b, int *c, int *d
Function Body
}
Software Function
a
b
Hardware Body
c
d
Hardware Modules
Input ports
Input ports
Software Function
analogy
Hardware Modules
Software body
Hardware body
Arguments
Ports
Input Arguments
Input Ports
Output Arguments
Output Ports

SUMMARY - HLS

Compared to the CPUs and GPUs, FPGAs can provide

- lower energy consumption
- higher performance
- and shorter time to market

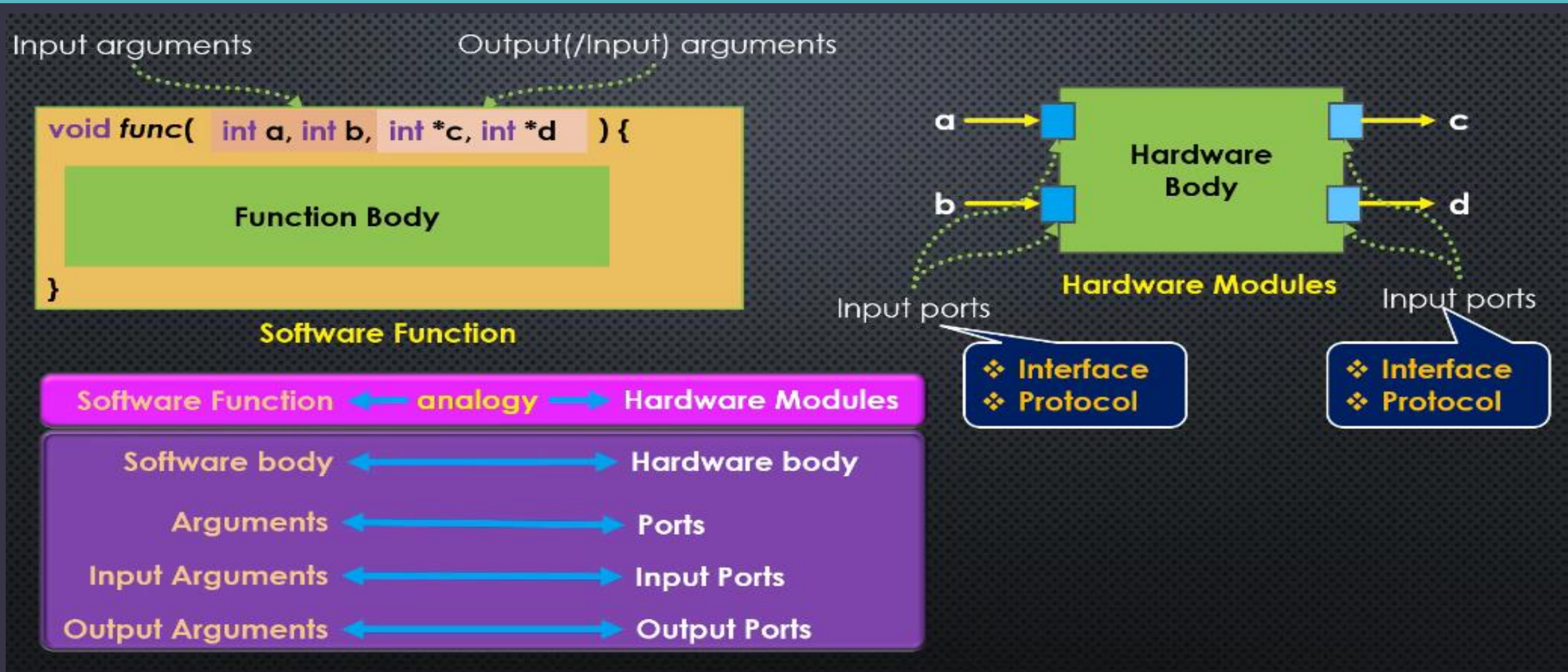
ASSIGNMENT Q-8

By connecting to the Xilinx website, find a couple of companies that use FPGAs in their products.

HW/SW ANALOGY

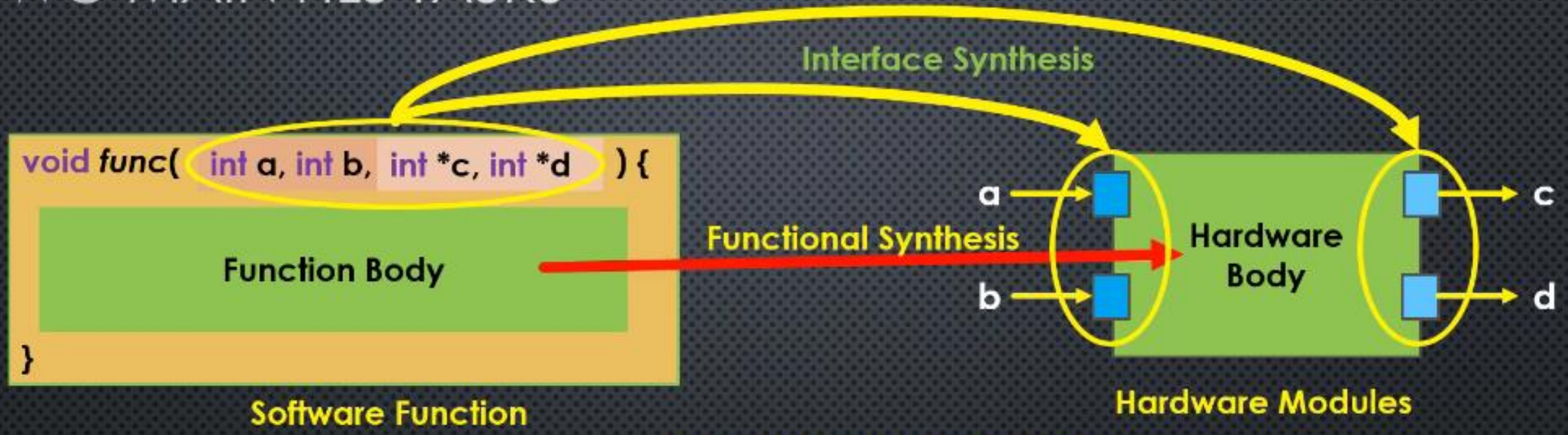
FPGA Concepts

SOFTWARE/HARDWARE ANALOGY



SOFTWARE/HARDWARE ANALOGY - **DIFFERENCES**

TWO MAIN HLS TASKS



1- **Function Synthesis**

Parallelism

2- **Interface Synthesis**

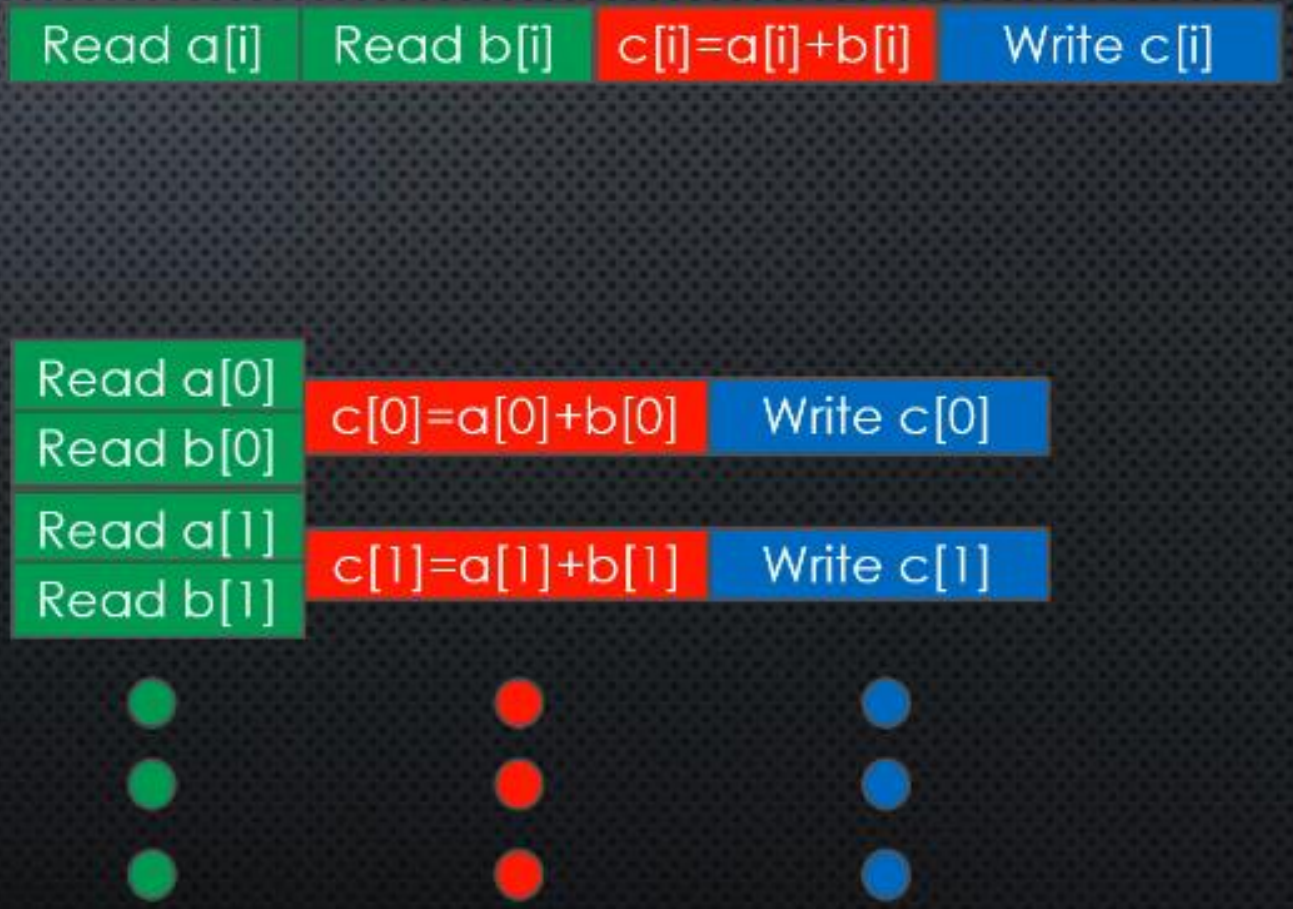
Bandwidth and Throughput

HOW USERS CAN HELP

```
void add_vec(int a[N], int b[N], int c[N]) {

  for (int i = 0; i < N; i++) {

    c[i] = a[i] + b[i];
  }

}
```

HOW USERS CAN HELP

```
void add_vec(int a[N], int b[N], int c[N]) {
#pragma HLS m_axi port=a
#pragma HLS m_axi port=b
#pragma HLS m_axi port=c

  for (int i = 0; i < N; i++) {
  #pragma HLS UNROLL
     c[i] = a[i] + b[i];
  }

}
```



SUMMARY

- There are clear analogy between a software C function and a hardware module
- Compiler directives should be added to the C-code to add hardware-related information and guide the synthesis tool to implement the code efficiently

ASSIGNMENT Q-9

What are the roles of two groups of pragmas added to an HLS description of an algorithm?

VIRTUAL LAB ACCESS

Demo

Any Question...

Thank you