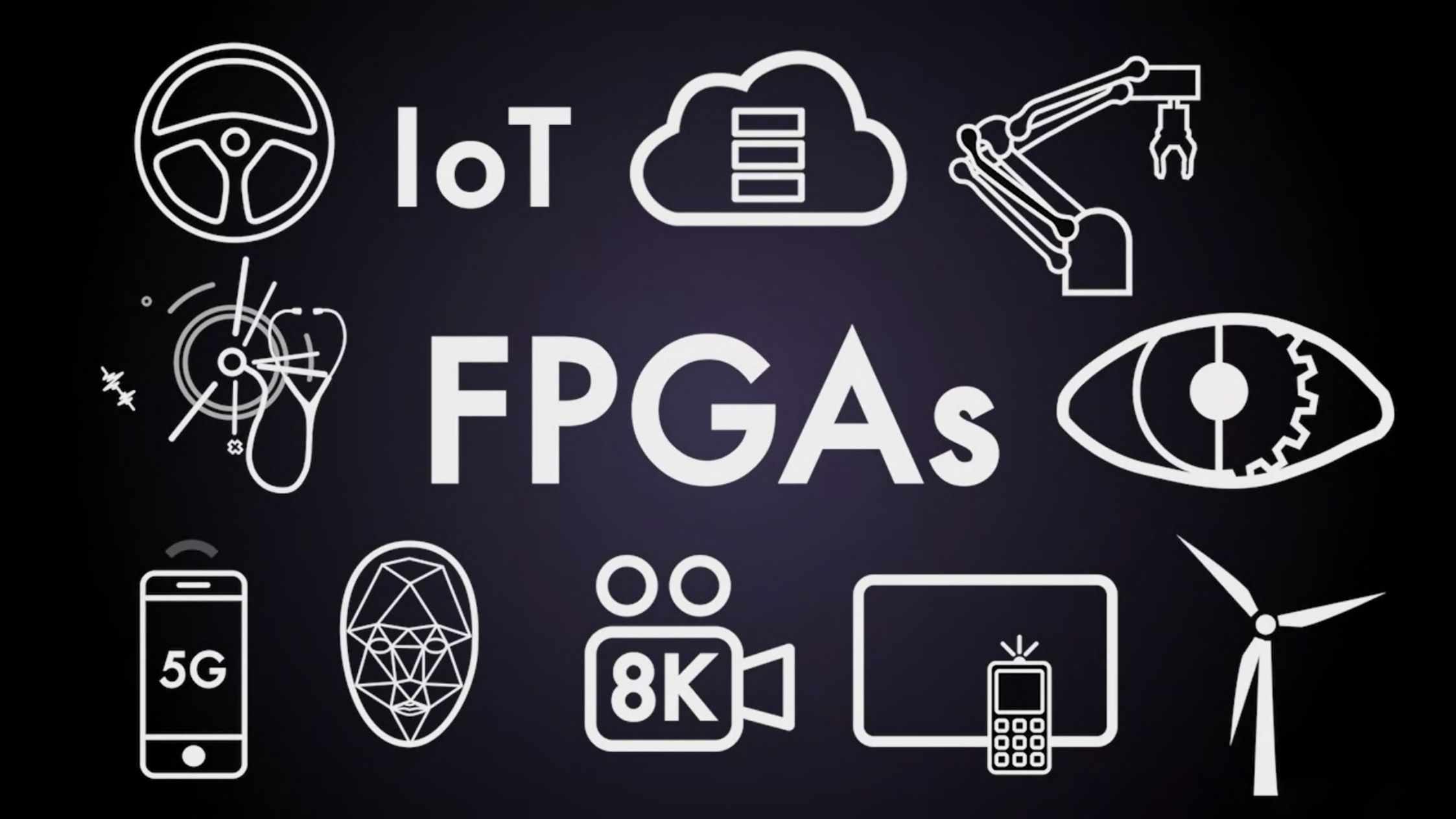

IoT
FPGAs
5G
8K


PLD Revenue
$6B
$7.3B
$8.6B
$10B
2017
2018
2019
2020

FPGA Design

Provides design with ultimate flexibility

Parts provide almost any digital function


SoC
Quick development
Sophisticated products
Relatively low cost
1:29 / 1:52


Be Boulder.
1:48 / 1:52

University of Colorado **Boulder**

1:50 / 1:52

What is an FPGA?

How FPGA technology was developed

How to select best architecture for a given application?

How to use state of the art software tools for FPGA development

Solve critical digital design problems using FPGAs


Scherr, Timothy Lee
Senior Instructor
Assoc Faculty Director
Electrical, Computer and Energy Engineering
University of Colorado Boulder
2:04 / 6:24

Quartus® Prime
Design Suite

INTEL QSYS PRO SYSTEM INTEGRATION TOOL

TIMEQUEST TIMING ANALYZER

MODELSIM-INTEL FPGA EDITION SOFTWARE

DSP BUILDER FOR INTEL FPGAS

POWER ANALYZER

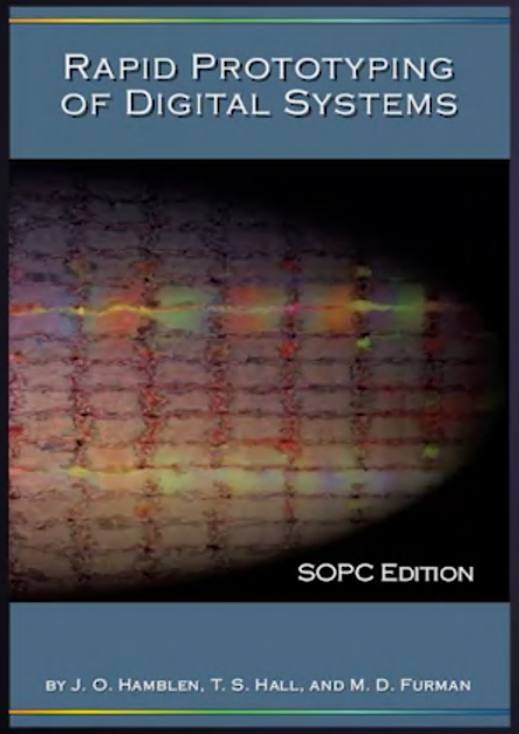
Syllabus

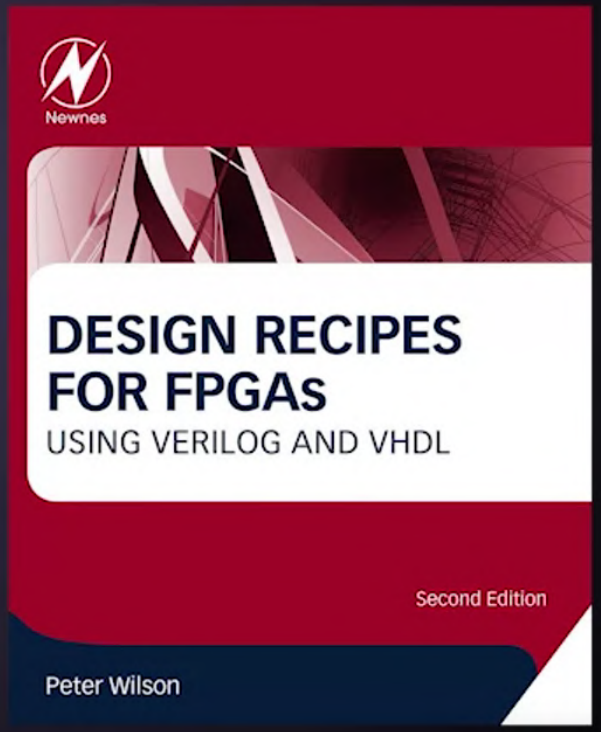
https://www.altera.com – Altera FPGA supplier site
https://www.altera.com/events/northamerica/altera-soc-developers-forum/overview.tablet.html - SoC Developers Forum
https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/tt/tt_my_first_fpga.pdf - First FPGA design tutorial
http://www.microsemi.com/products/fpga-soc/fpga-and-soc Microsemi FGPA supplier site
http://www.latticesemi.com/en/Products.aspx - Lattice Semiconductor FPGA supplier site
http://www.xilinx.com – Xilinx FGPA supplier Site
http://www.cypress.com/products/programmable-system-chip-psoc - Cypress PSoC supplier site
http://www.quicklogic.com – Quicklogic programmable logic supplier site
http://www.achronix.com/?url=sourcetech411 – Achornix FPGA supplier site
http://www.atmel.com/products/other/field_programmable_gate_array/default.aspx - Atmel FPGA supplier site
http://www.e2v-us.com – e2V site, military FPGAs

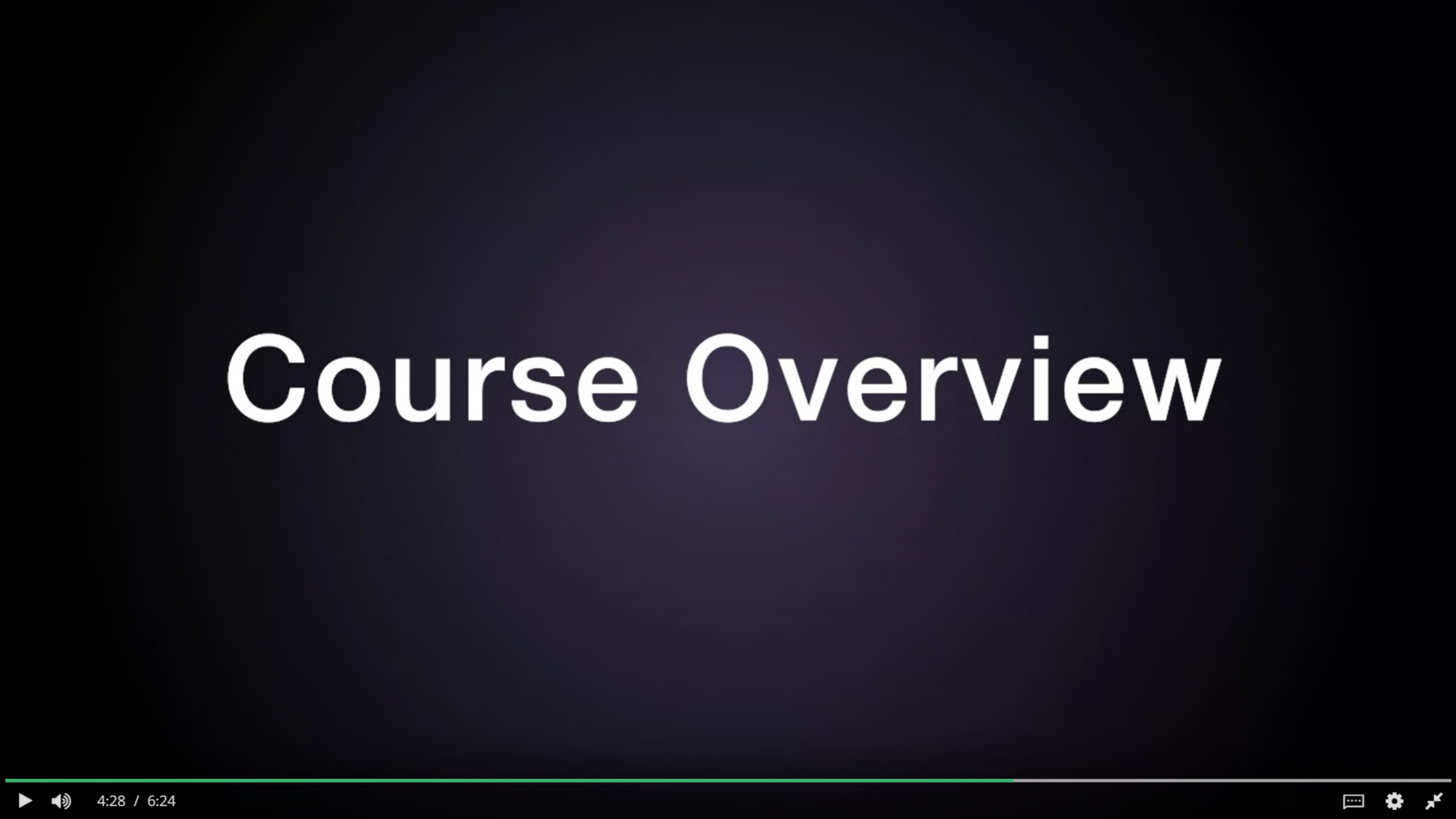
Magazines: EDN . Electronic Design . EETimes . Portable Design . Circuit Cellar Ink . Popular Science . Dr. Dobbs Journal . IEEE
Magazines: Embedded Systems Design . Embedded Systems Design Europe . Embedded Computing Design . Military Embedded Systems
http://www.eetimes.com/programmable-logic-designline.asp - Programmable Logic Designline
http://www.eetimes.com/soc-designline.asp - SoC Designline
http://www.eetimes.com/document.asp?doc_id=1274593 – how to design an FPGA from scratch
http://www.eetimes.com/author.asp?section_id=216&doc_id=1326502 – FPGAs for MCU designers

http://www.mathworks.com/solutions/fpga-design/ - FPGA design with Matlab
http://www.mathworks.com/products/?s_tid=gn_ps – MATLAB/Simulink for FPGA Design
http://www.altium.com/files/training/module5fpgadesign.pdf - Altium Tool for FPGA design
http://www.synopsys.com/tools/implementation/fpgaimplementation/Pages/default.aspx - Synposis Tools for FPGA Design (Synplicity)
http://www.mentor.com/products/fpga/ - Mentor tools for FPGA Design
https://www.cadence.com/rl/resources/white_papers/fpga_wp.pdf - Cadence FPGA design guidance article
https://www.aldec.com/en/solutions/fpga_design - Aldec Tools for FPGA design

Reference Texts






Course Overview
4:28 / 6:24

Introduction to FPGA Design for Embedded Systems

FPGA history and architecture

Hands on experience with design tools

Current FPGA technology

FPGA design and analysis

Learning Approaches

In-video quizes

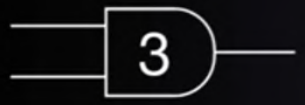
Exercises with industry tools

Discussion Forums

Assessments

History of Programmable Logic

CPLD, its architecture and performance



Introduce FPGA, contrast with CPLD,
and LUT makeup

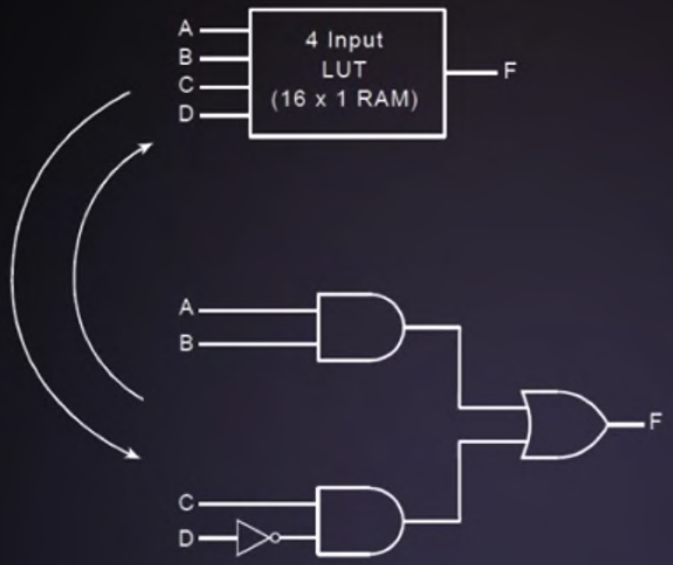
What is an FPGA?

A **field-programmable gate array (FPGA)** is an **integrated circuit** designed to be **configured by a customer** or a designer **after manufacturing** – hence "field-programmable".

The FPGA configuration is generally specified using a hardware description language (HDL), similar to that used for an application-specific integrated circuit (ASIC).

[1]

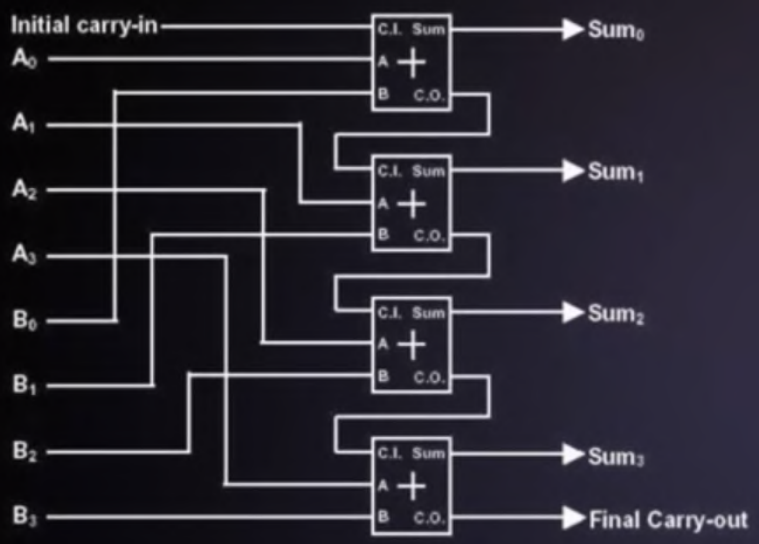
We will learn how FPGAs use LUTs to create logic

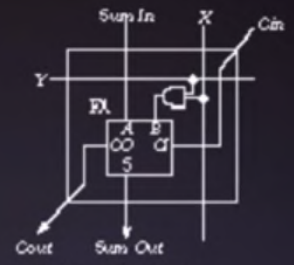


RAM Contents				
Address				Data
A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

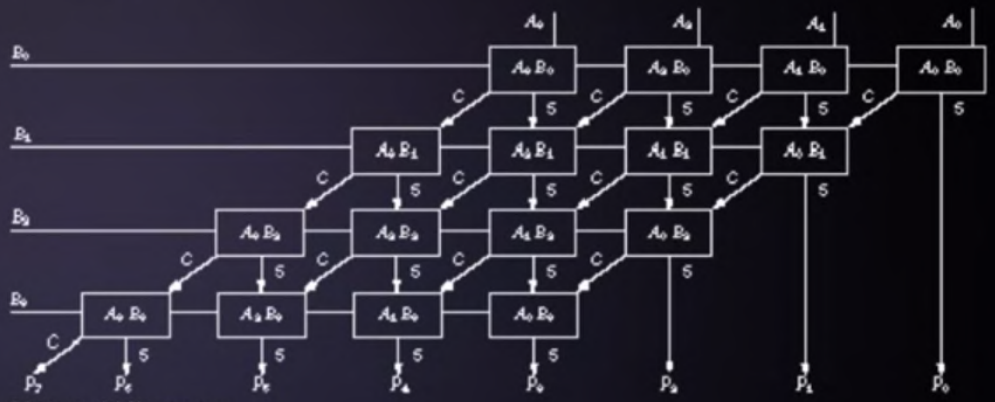
Figure 3.8 Using a look-up table (LUT) to model a gate network.

Logic design is different in FPGAs





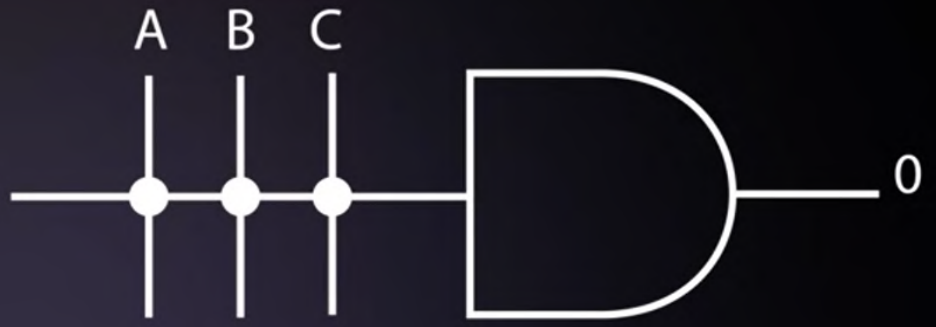
(a) Basic building block



(b) 4 × 4 multiplier structure


1956
PROM Invented
(Programmable Read only Memory)
1969
PROM made commercially available
1971
EPROM
(Eraseable Programmable
Read-only Memory)
1975
PLAs
(Programmable Logic Arrays)
1978
PALs
(Programmable Array Logic)

Conventional AND Gate



Programmable Logic Notation

Programmable Logic Notation

Hardwired

Programmable Link

Not Connected

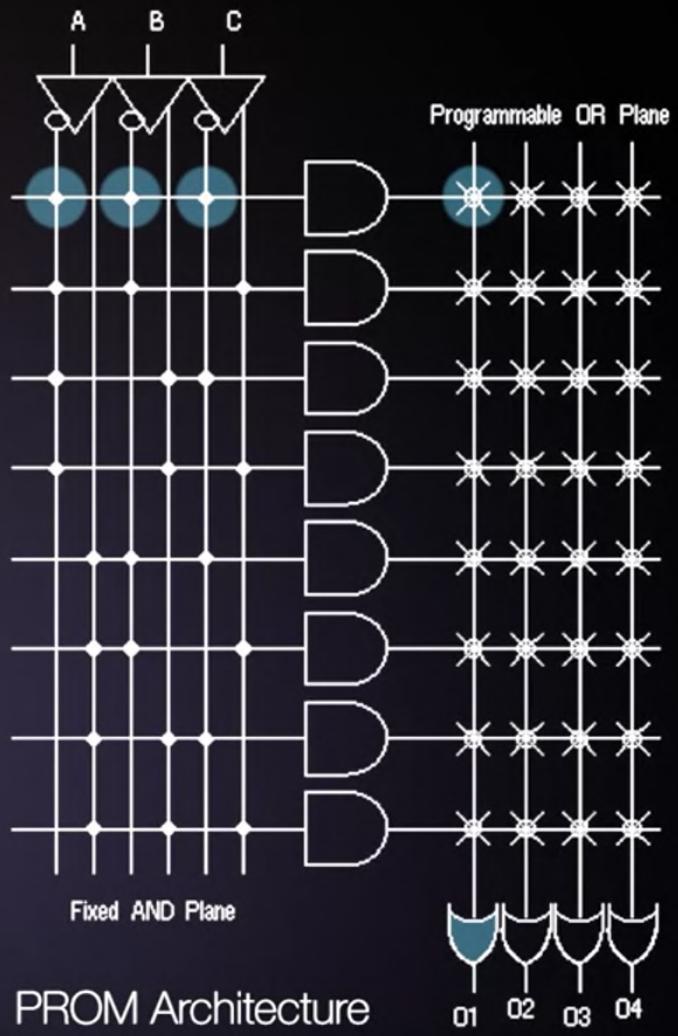
What is Programmable Logic?

- First "programmable logic" is a PROM
  - Has a fixed AND plane (product terms) (address decoding logic)
  - OR plane (Sum terms) programmable through the change of memory contents.

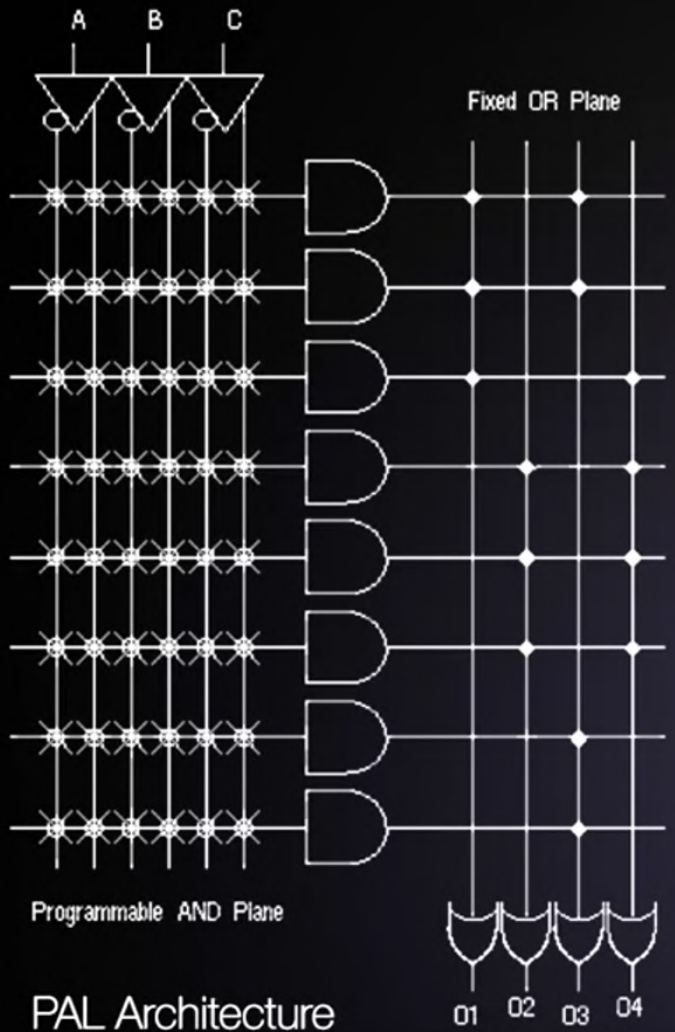
8x4 Prom

- Implement the logic function Output1 = ABC by storing in first column of the PROM 0b00000001.

- Then only when A, B, and C are all ones will output O1 be a one.

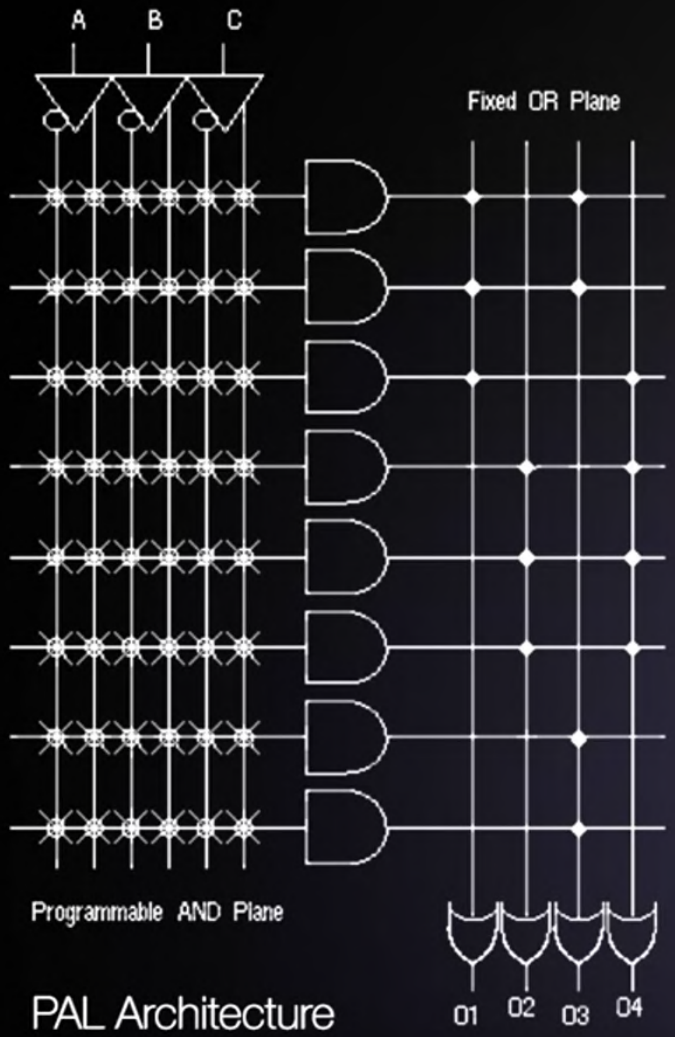


PROM Architecture [1]



[2] PAL Architecture

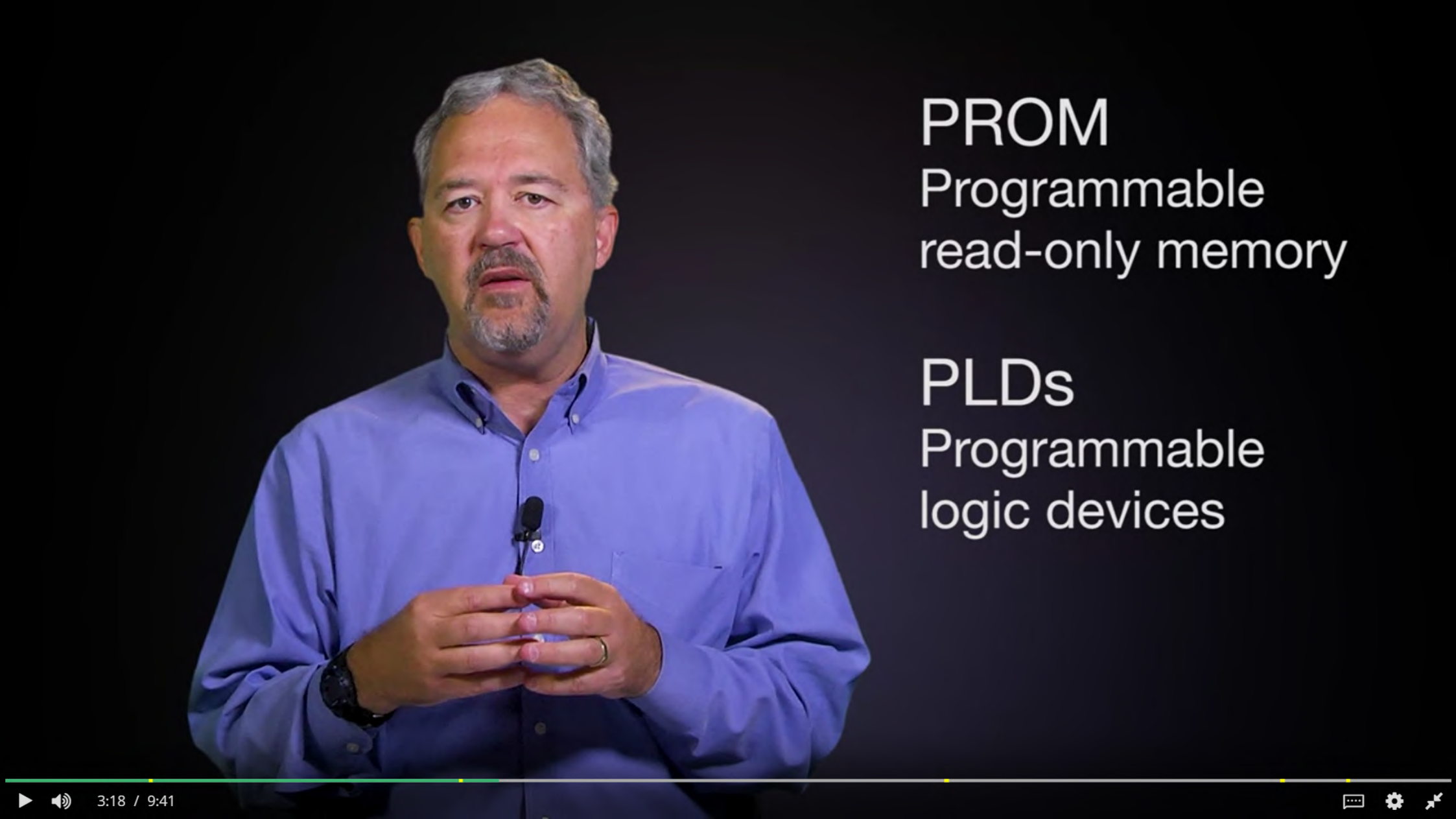
- Make the OR plane fixed and make the AND plane programmable.
- PAL devices are very popular and are still used in many designs, with common part numbers like 22V10 or 16R8.
- Further development lead to CPLDs, which were devices with multiple PALs in same package with registered outputs and an interconnecting programmable fabric.
- CPLD = Complex Programmable Logic Device

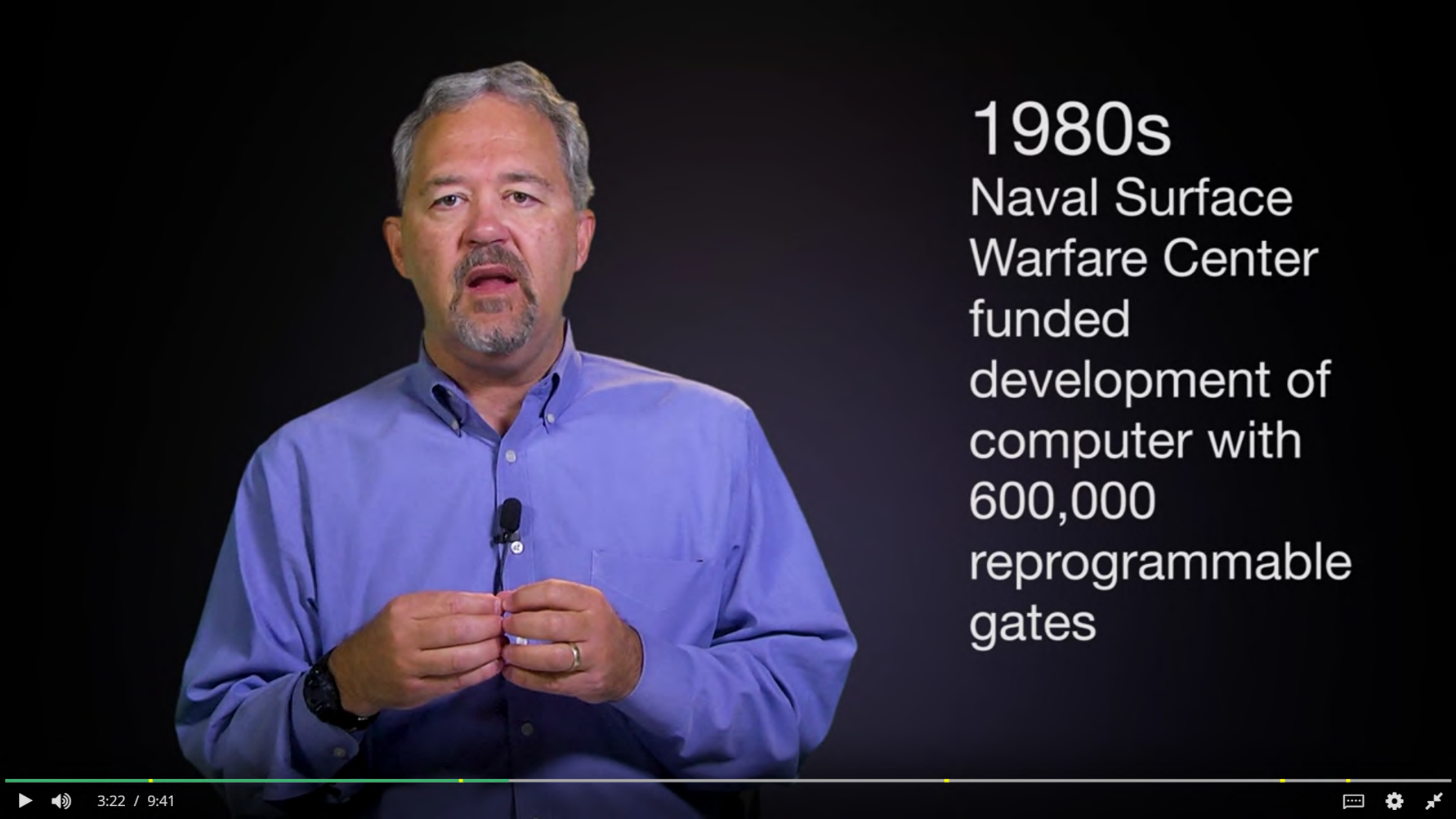


[2] PAL Architecture

For a history of the CPLD, see:

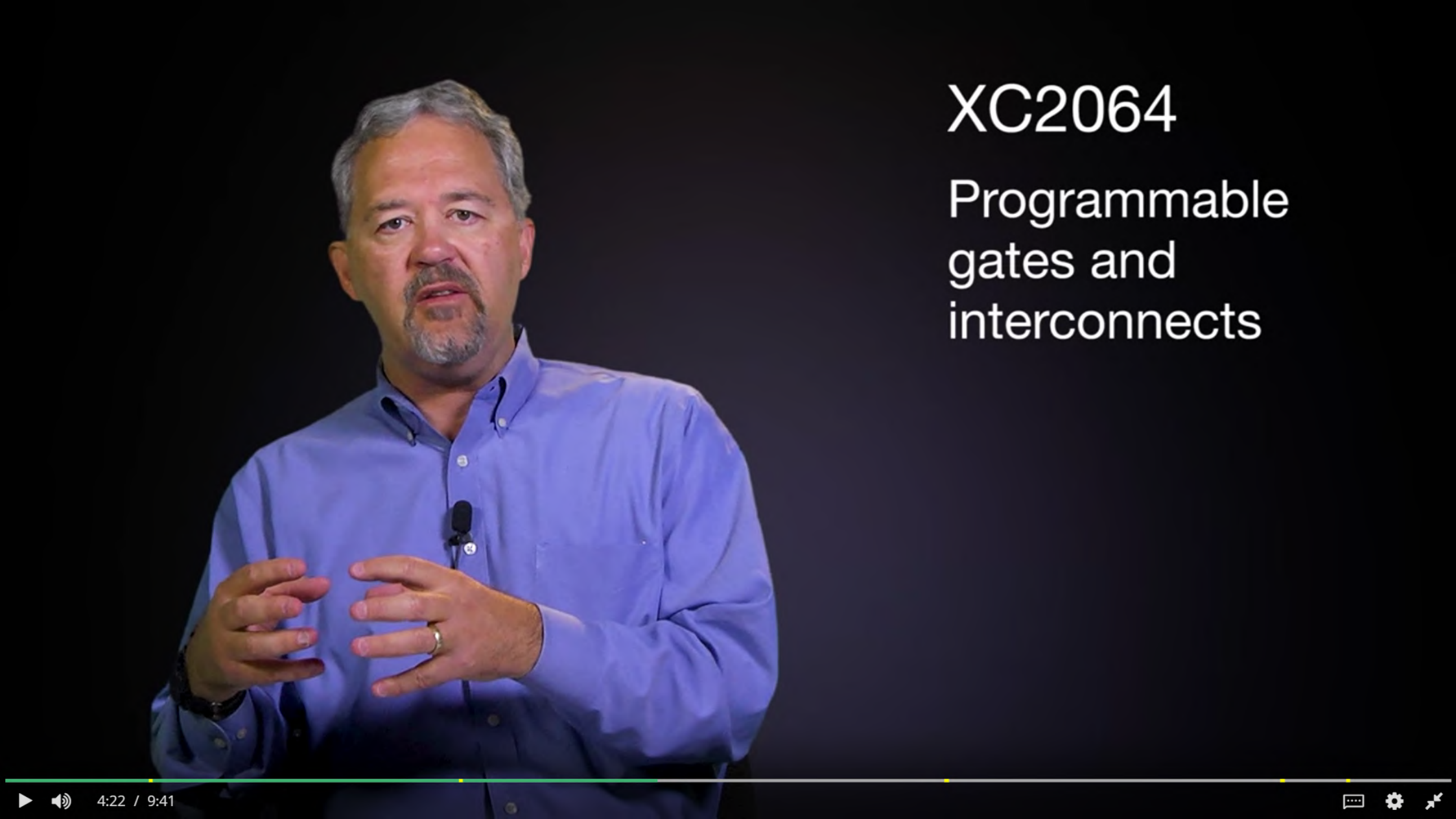
https://www.altera.com/solutions/technology/system-design/articles/2013/in-the-beginning.tablet.html

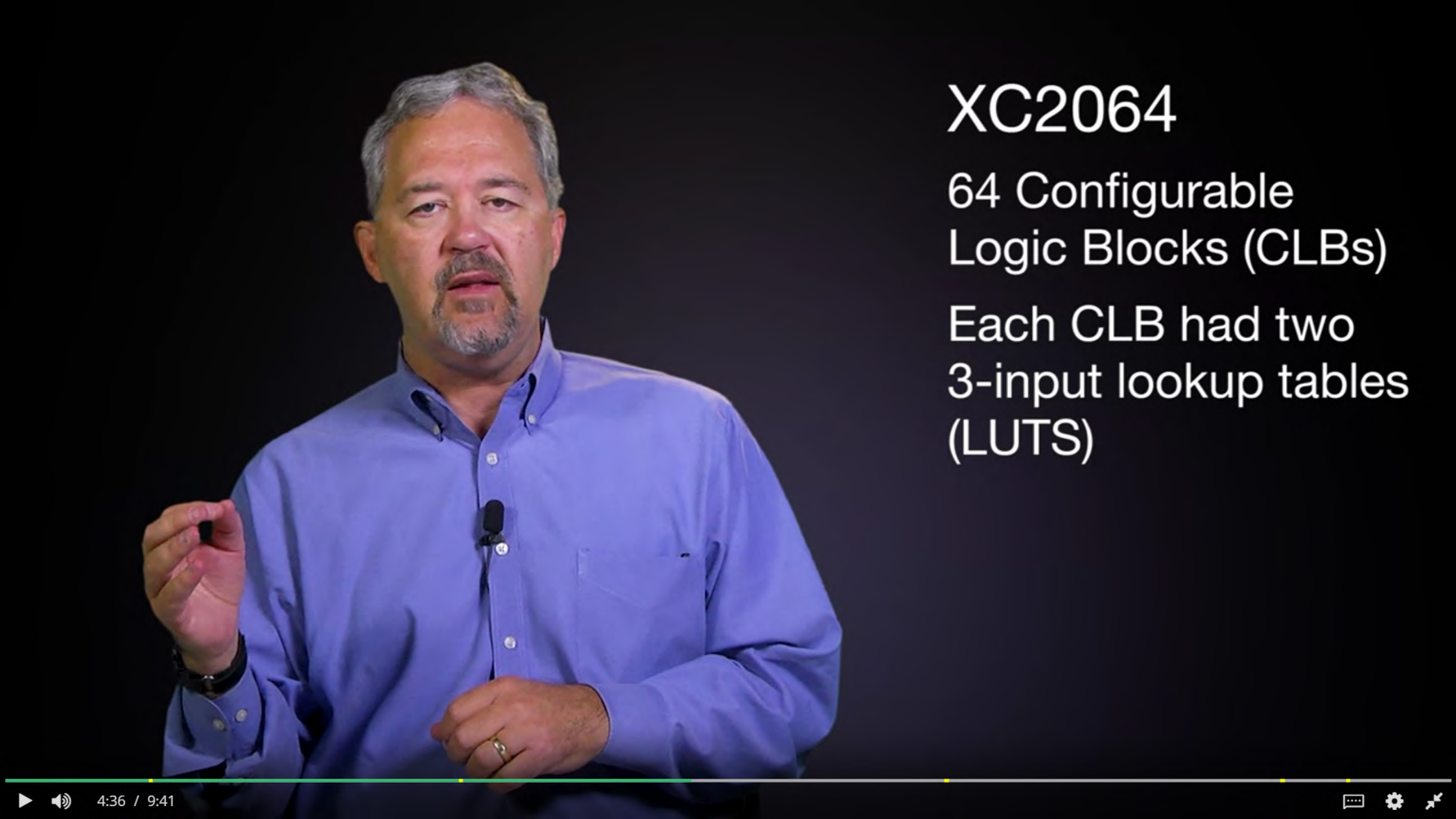

PROM
Programmable
read-only memory
PLDs
Programmable
logic devices
3:18 / 9:41

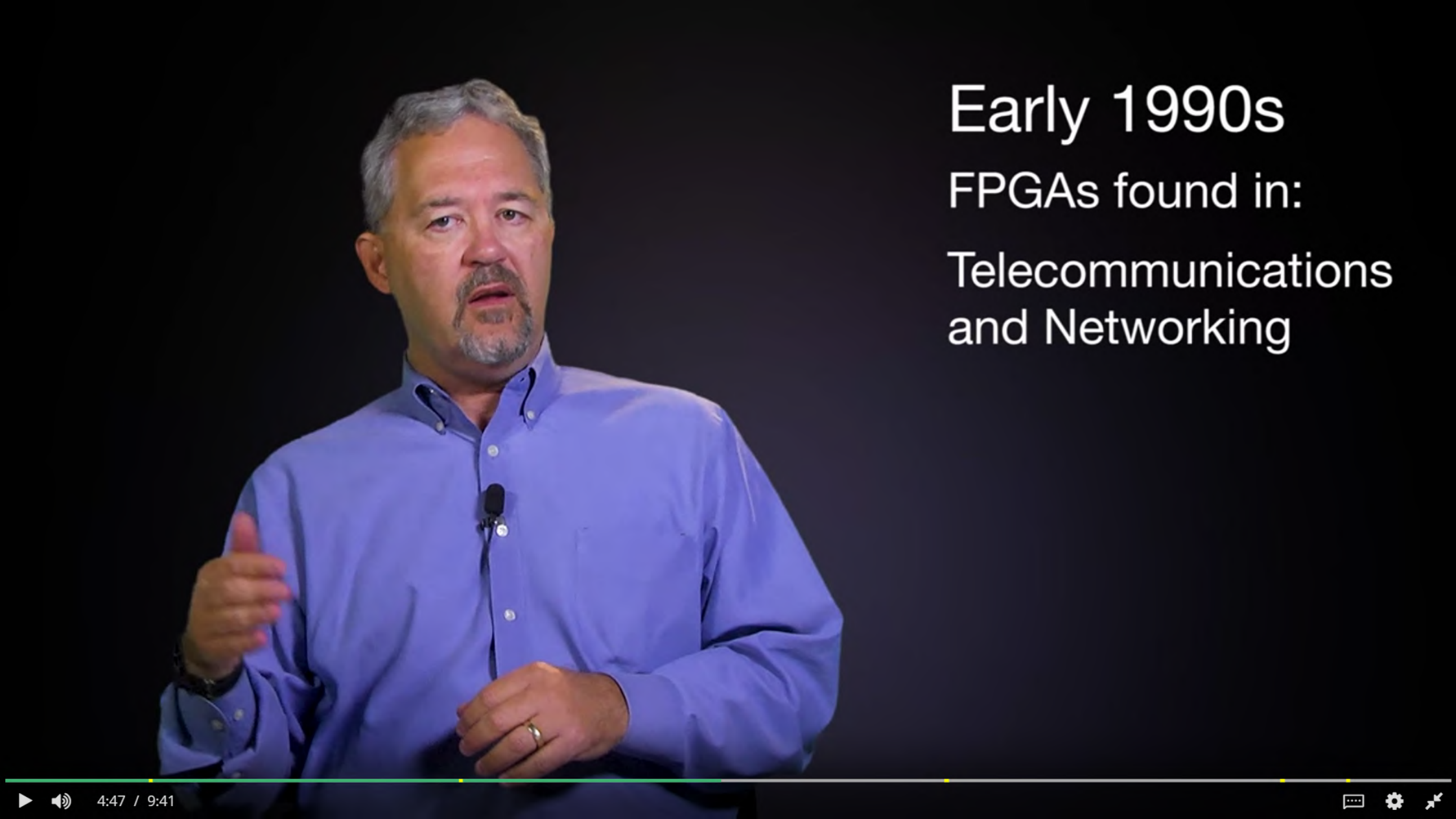

1980s
Naval Surface Warfare Center funded development of computer with 600,000 reprogrammable gates
3:22 / 9:41

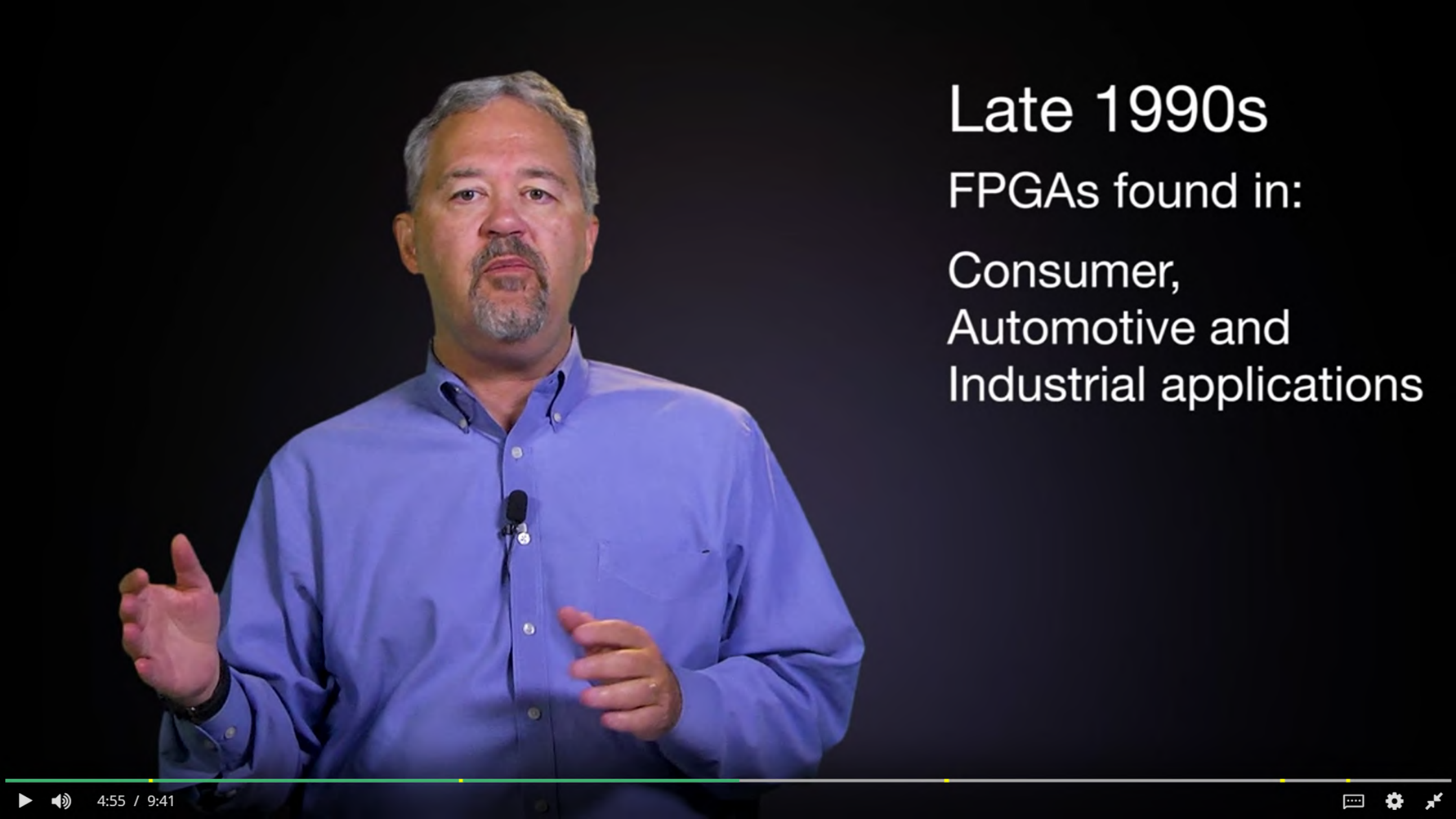

1983
Altera founded
1984
Delivered first reprogrammable logic device
The EP300, featuring a quartz window
3:37 / 9:41

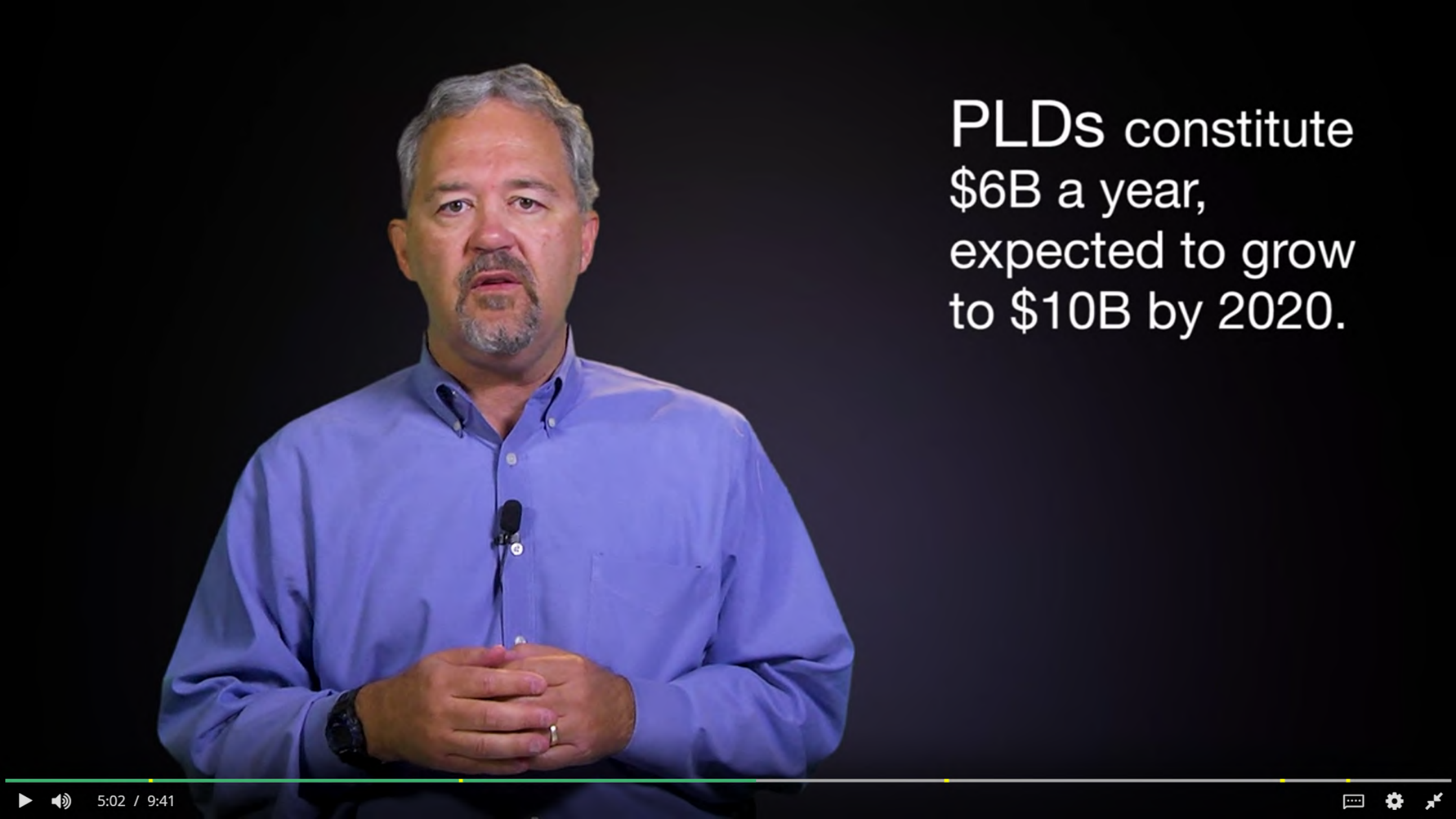

1985
First commercially
viable FPGA
XC2064
4:17 / 9:41

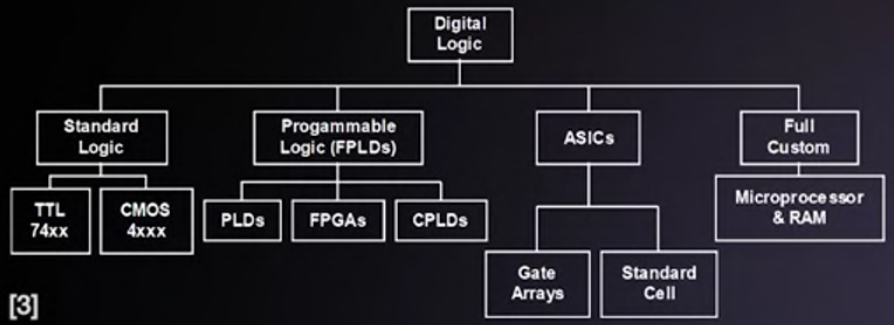

XC2064
Programmable
gates and
interconnects
4:22 / 9:41


XC2064
64 Configurable
Logic Blocks (CLBs)
Each CLB had two
3-input lookup tables
(LUTS)
4:36 / 9:41


Early 1990s
FPGAs found in:
Telecommunications
and Networking
4:47 / 9:41


Late 1990s
FPGAs found in:
Consumer,
Automotive and
Industrial applications
4:55 / 9:41


PLDs constitute
$6B a year,
expected to grow
to $10B by 2020.
5:02 / 9:41



[3]

- Standard Logic has much more breadth than depicted here.
- Includes additional TTL families (LVTTL, etc.)
- Additional CMOS families (HC, LVC, etc.)
- biCMOS (BCT, HCT, etc.)
- Differential Logic (ECL, LVDS, etc.)

Wire

Gate

Register/
Flip-flop

Application-Specific Standard Product (ASSP)

- Semiconductor device integrated circuit (IC) product that is dedicated to specific application market and sold to more than one user.

- marketed to multiple customers

- for a special application, but it is sold to any number of companies.

- generally offers the same performance characteristics and has the same die size as an ASIC.

A System on a Chip or System on Chip (SoC or SOC)

- an integrated circuit (IC) that integrates all components of a computer or other electronic system into a single chip.

- may contain digital, analog, mixed-signal, and often radio-frequency functions — all on a single chip substrate.

SOC

- SoCs are very common in the mobile electronics market because of their low power consumption.

- A typical application is in the area of embedded systems.

A System on a Chip or System on Chip (SoC or SOC)

- an integrated circuit (IC) that integrates more than one component type into a single chip along with a CPU.

- Typical component types are GPUs, communications interfaces, analog functions, and radios.

- If it includes programmable logic, then it is a Programmable SoC, or SoC FPGA.

A System on a Chip or System on Chip (SoC or SOC)

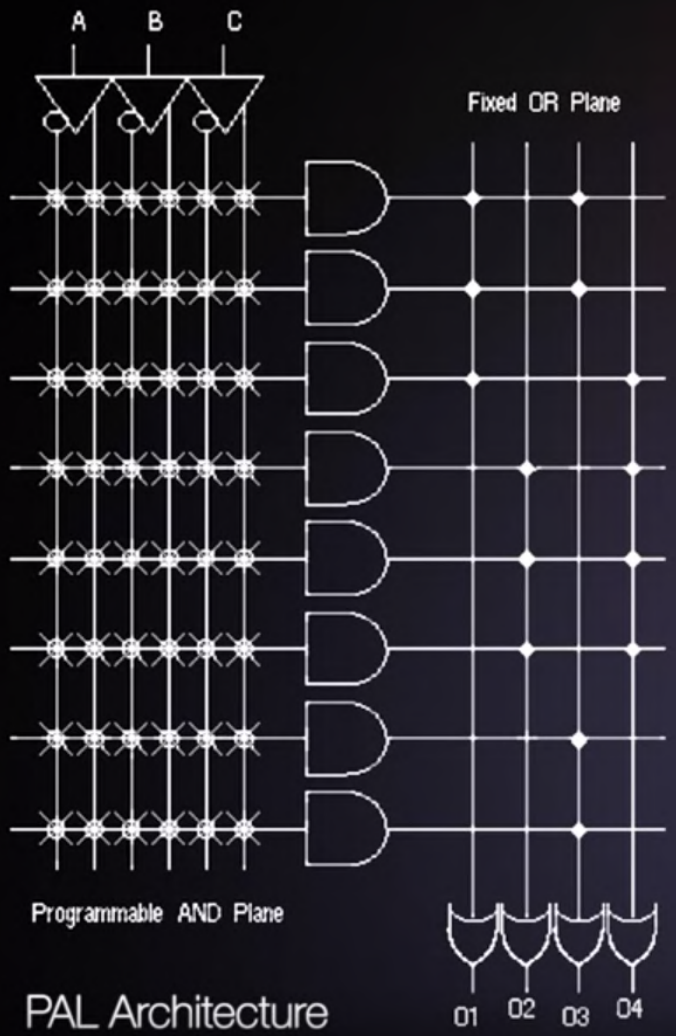
- The higher integration of an SoC provides lower cost, smaller size, and lower power than alternatives.

Programmable Logic Devices (PLDs) include:

- Simple PLDs like PROMs and PALs
- Complex PLDs (CPLDs)
- FPGAs
- SoC FPGAs

FPGAs < PLDs < Logical Devices

- FPGAs compete with ASICs and ASSPs, successfully displacing them in many applications

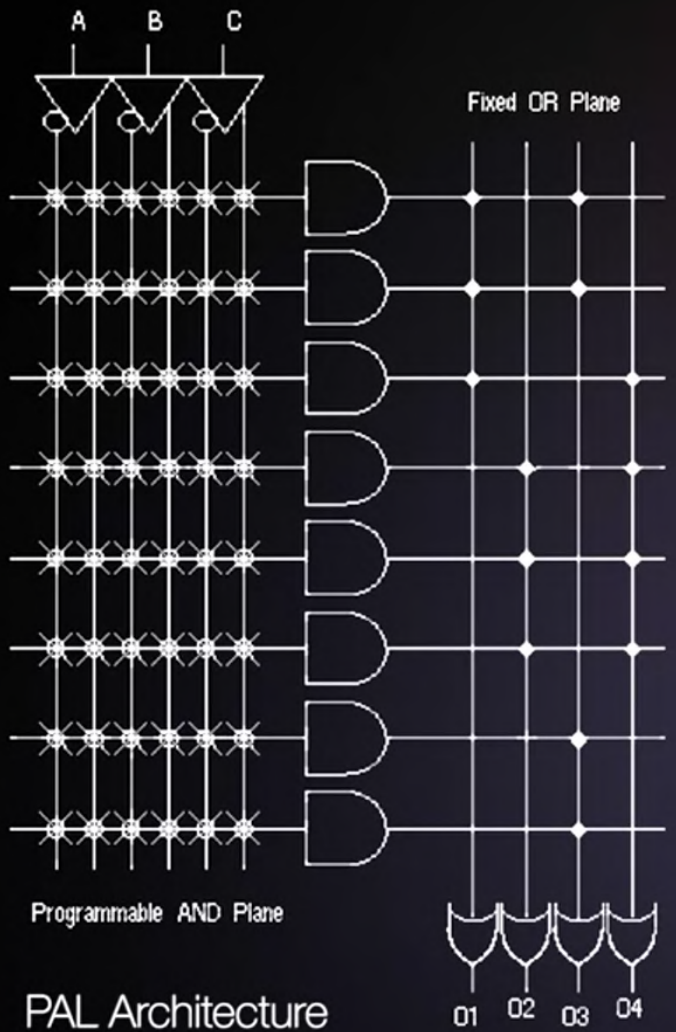


[1] PAL Architecture

Sum of Products Logic Functions

Programmable AND - Fixed OR

Replacement of 12
discrete logic packages

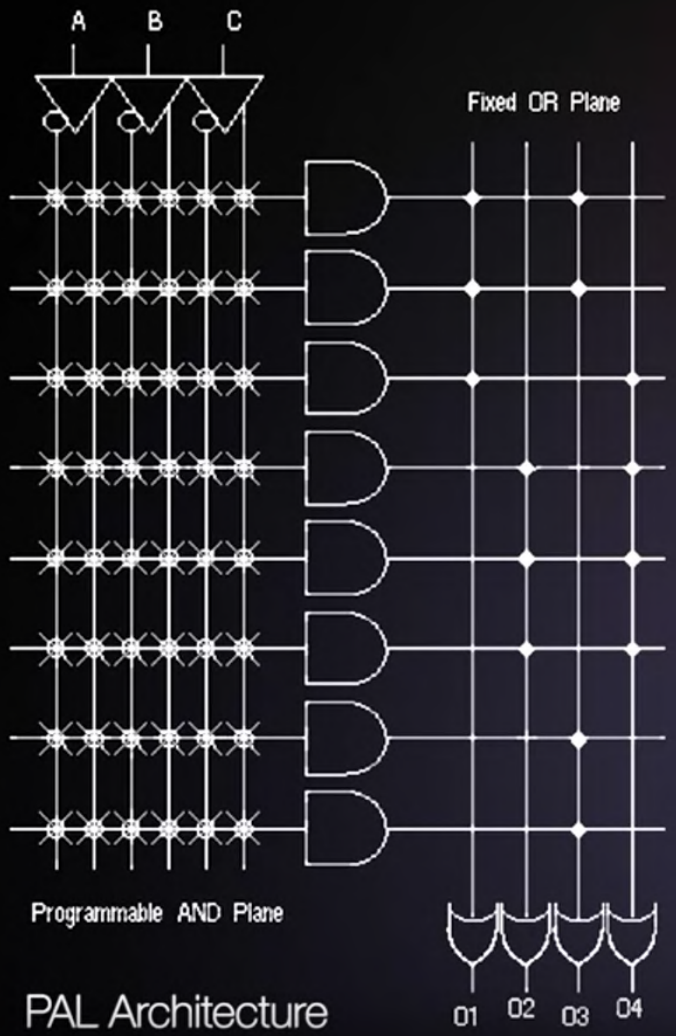


[1] PAL Architecture

Macrocells in a single integrated circuit package:

– all the input pins were available to each macrocell

– any output pin could be driven by any macrocell

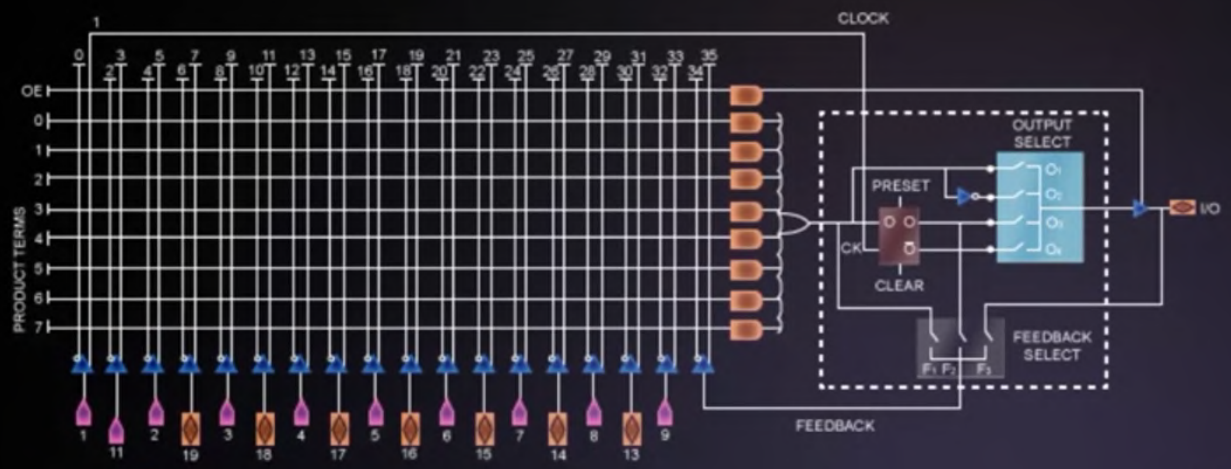


[1] PAL Architecture

CPLDs built in CMOS
to be reprogrammable

Bi-polar PALs were not

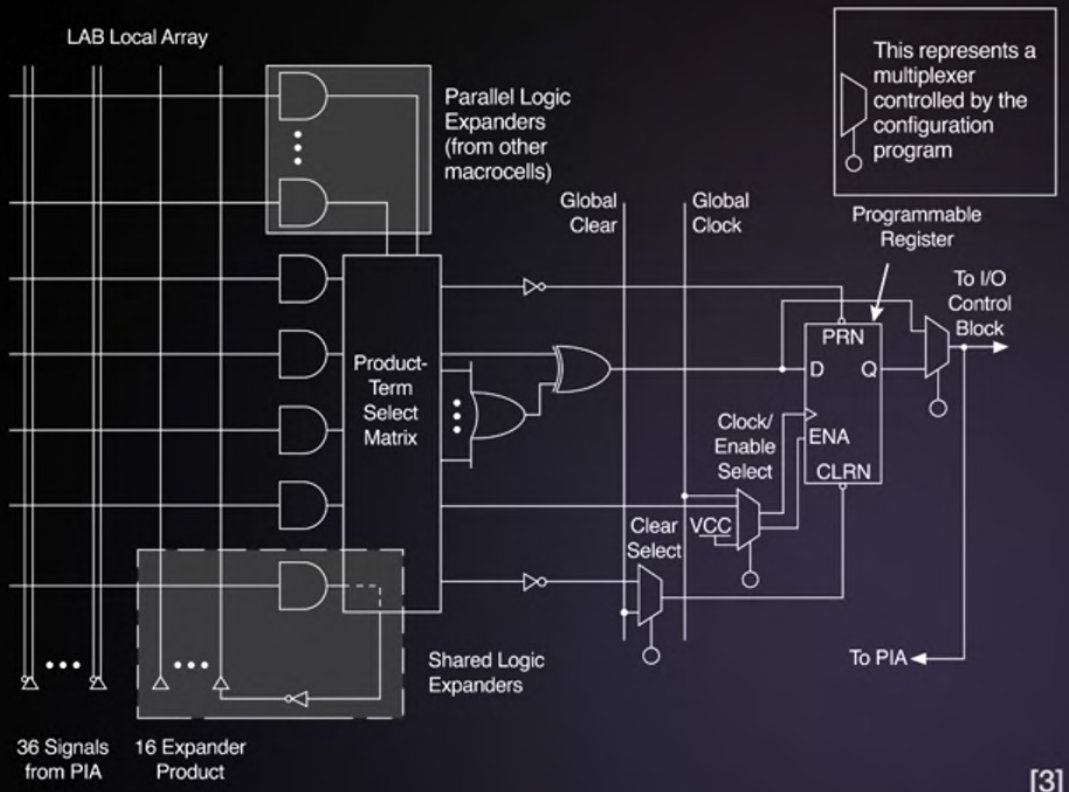
CPLD Early Device Architecture



[2]

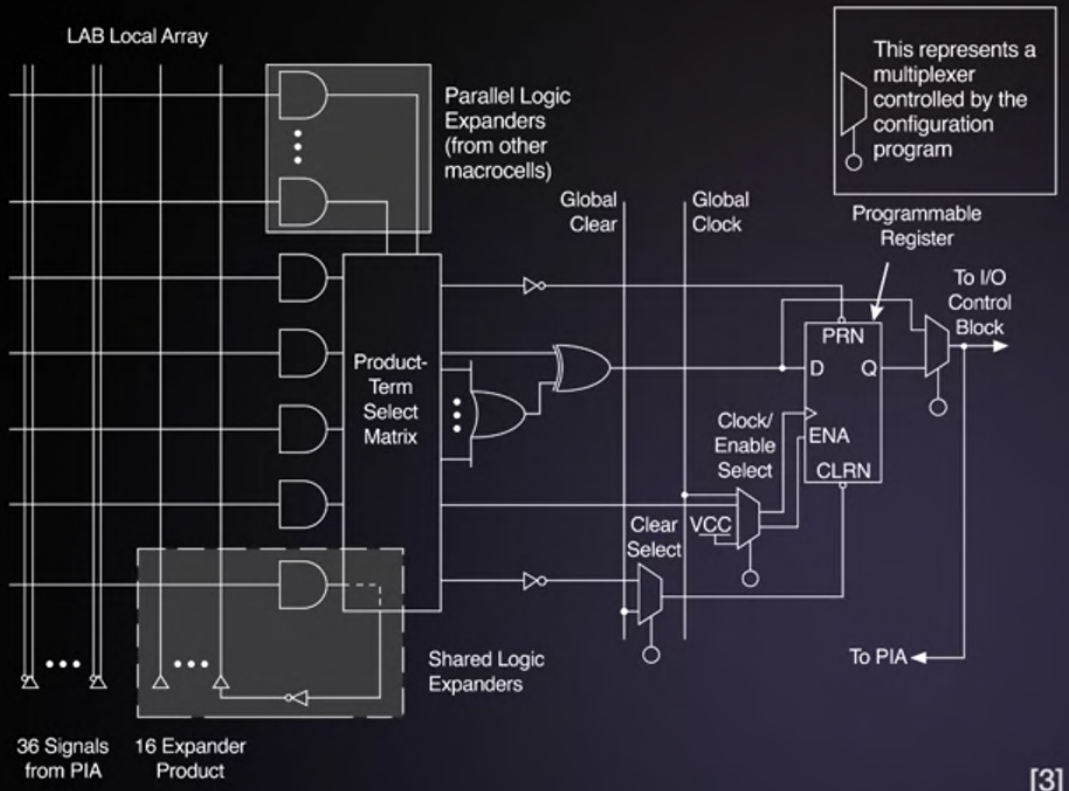
Multiple macrocells in a single integrated circuit package

Each macrocell had a dedicated output pin



[3]

CPLD Macrocell is basically just a registered PAL array

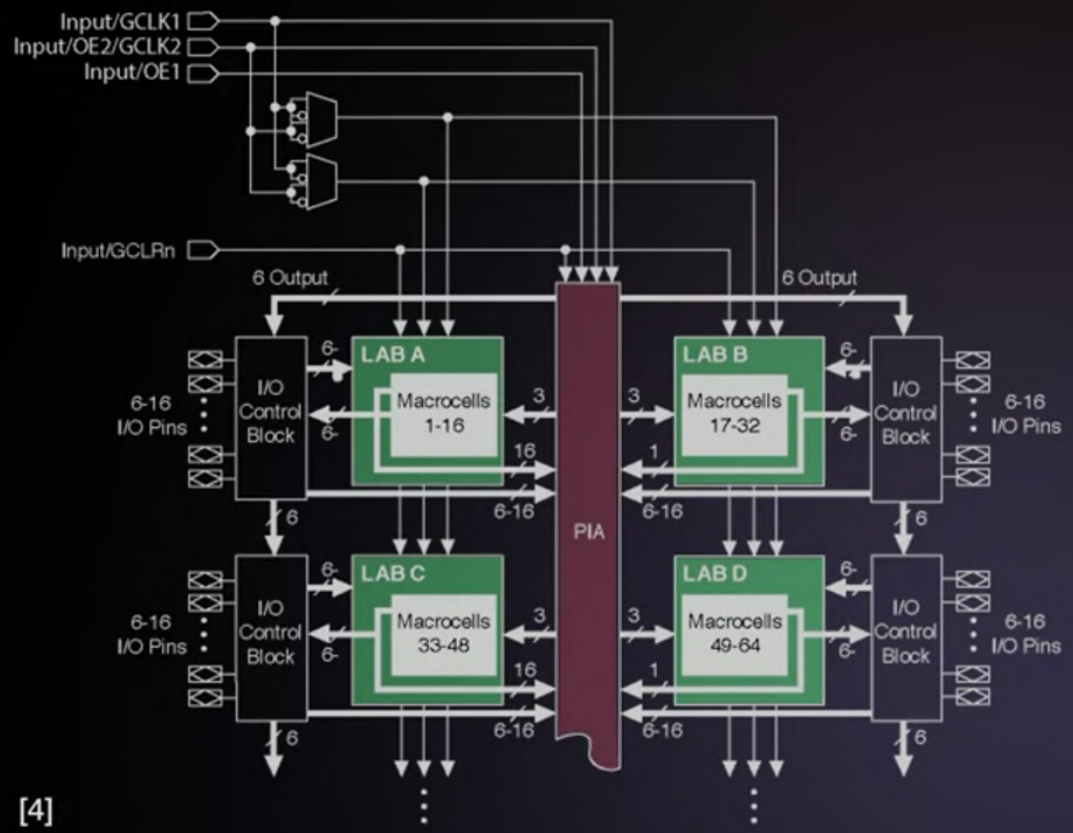


[3]

Good for problems that require a good deal of logic for every flip-flop:

– Complex 16 state state machine

– Wide input decoder



[4]

LAB – Logic Array Block

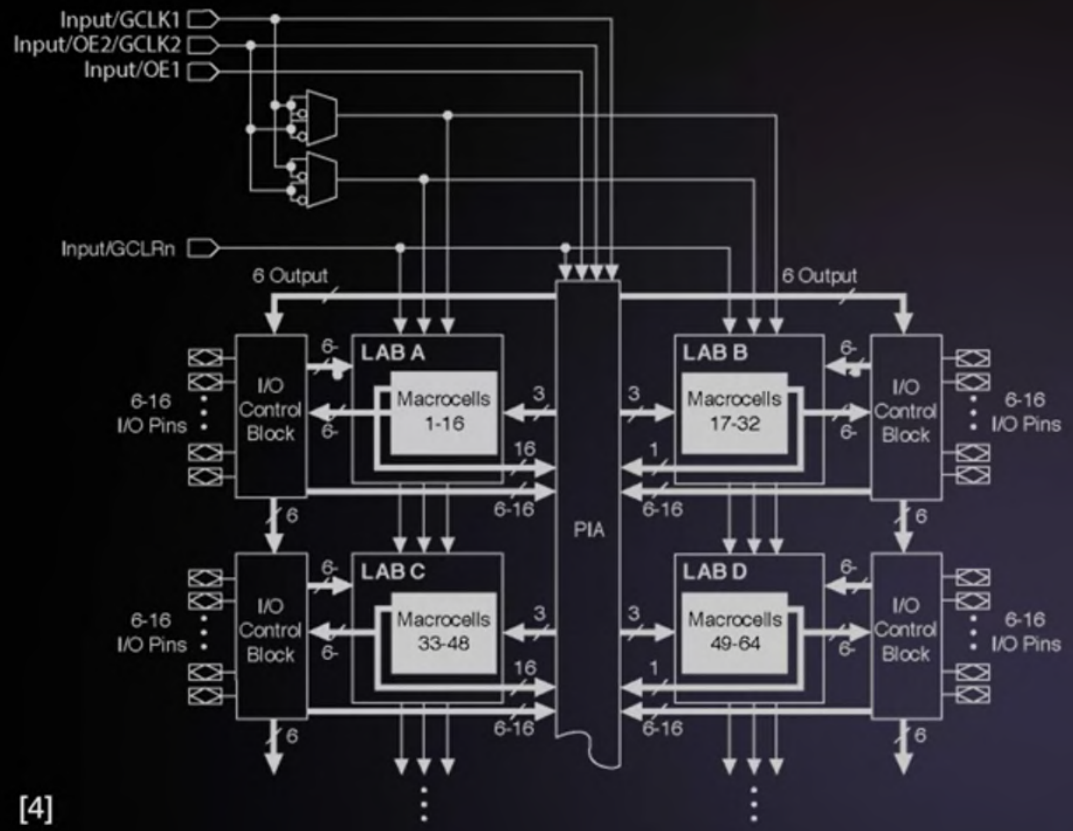
Made of 16 macrocells with local routing between the macrocells

PIA - Programmable Interconnect Array

– Routing network between labs

– CPLDs are inherently heirarchical devices

Larger logical systems from smaller logical functions



[4]

Designers now had to plan:

– Which macrocells implement particular logic equations and which need result

– Account for delays possible through interconnections in hierarchy

CPLDs

Easy generation of wide input functions

Easy to calculate timing that is very predictable, even called "deterministic"

Still a good choice for glue logic applications today

CPLDs

For designs that requried many:

– Registers
– Data transfers
– Bus interfaces

CPLDs did not scale well.

This led to the development of the FPGA.

CPLD Architecture Summary

– CPLDs introduced reprogrammability to programmable logic devices, an important new feature.

– Also reinforced heirarchical design methods

– The architecture of the CPLD allowed for easy design of wide input combinational logic functions

  – Address decoders

  – State machines with deterministic timing

LUTS and FPGA Architecture

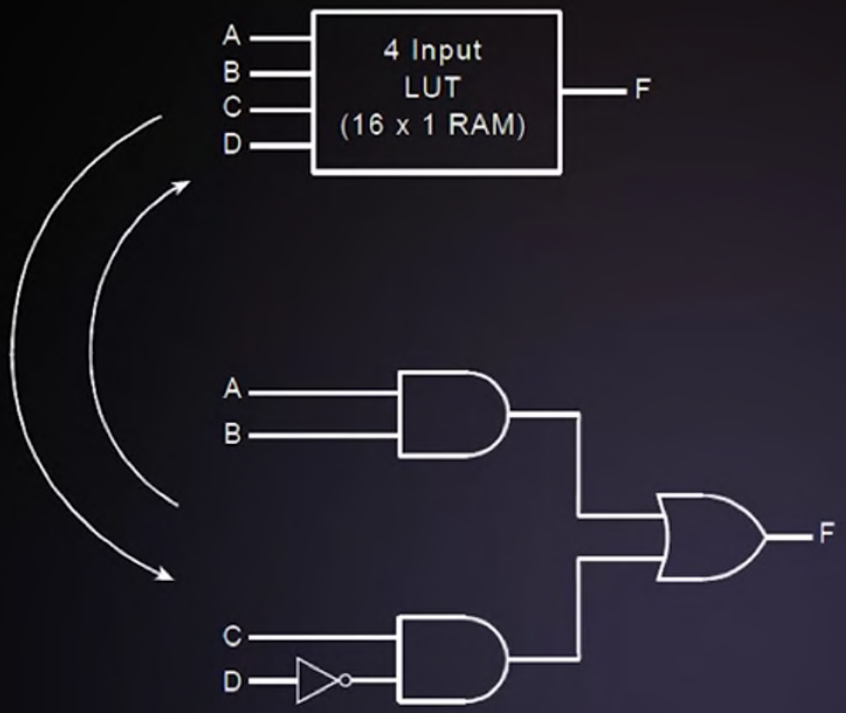
– Invented to provide a general purpose digital logic device with great flexibility and utility

We will learn about the developement of the FPGA:

- How the fundamental logic cells work using look up table (LUTs)
- How routing becomes a bigger factor in performance

FPGA Goal : Flexible general purpose logic device

Logic Implementation : Look up tables (memory)

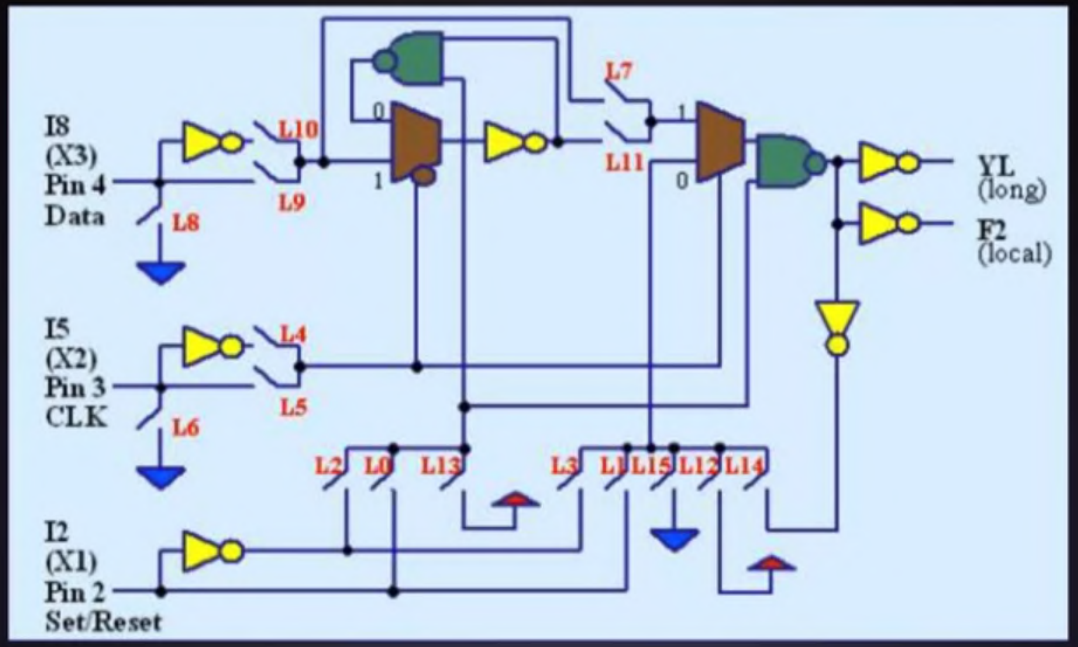


RAM Contents				
Address				Data
A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

[1]

Using a LUT to model a gate network.

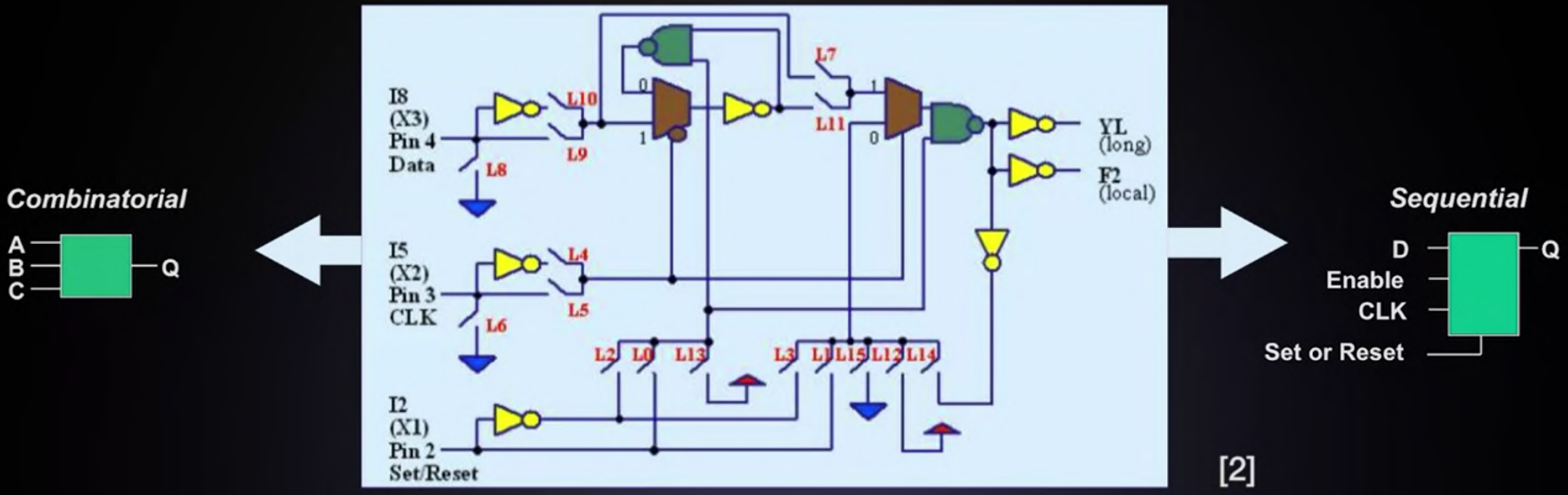
FPGA Based on LUTs and Routing



[2]

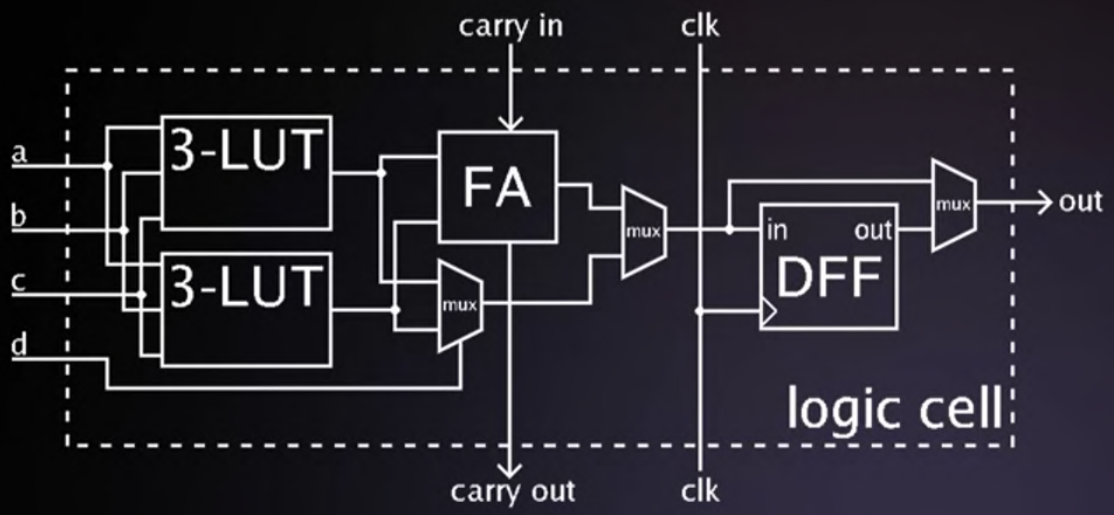
Logic element is smaller, usually implemented as a LUT.

FPGA Based on LUTs and Routing



[2]

IGLOO VersaTile can implement:
3-input Combinational Gate / Latch / D-Flip-Flop with Enable



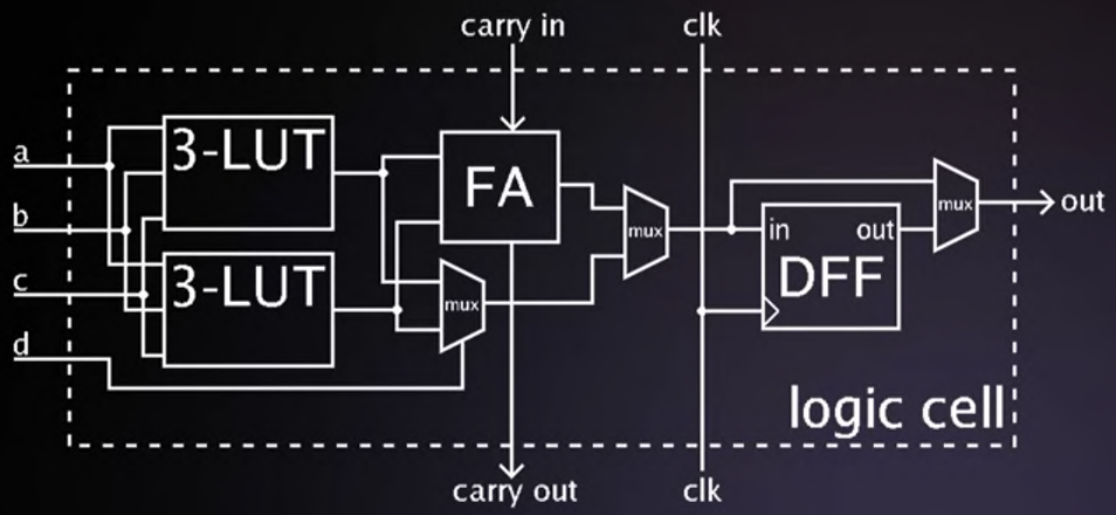
[3]

FPGA Logic Cell

4-input LUT

Full Adder (FA)

D-type Flip-Flop



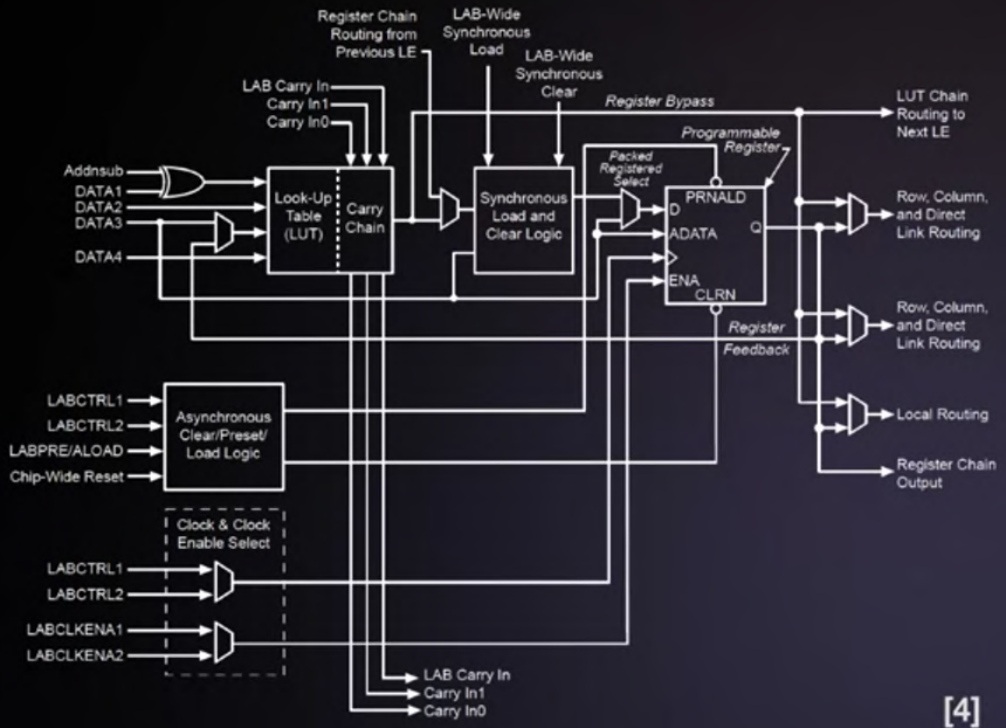
[3]

FPGA Logic Cell

Normal Mode:
Combined into
4-input LUT
via left mux

Arithmetic Mode:
Outputs are fed
to FA

Selection of mode
programmed into
middle multiplexer

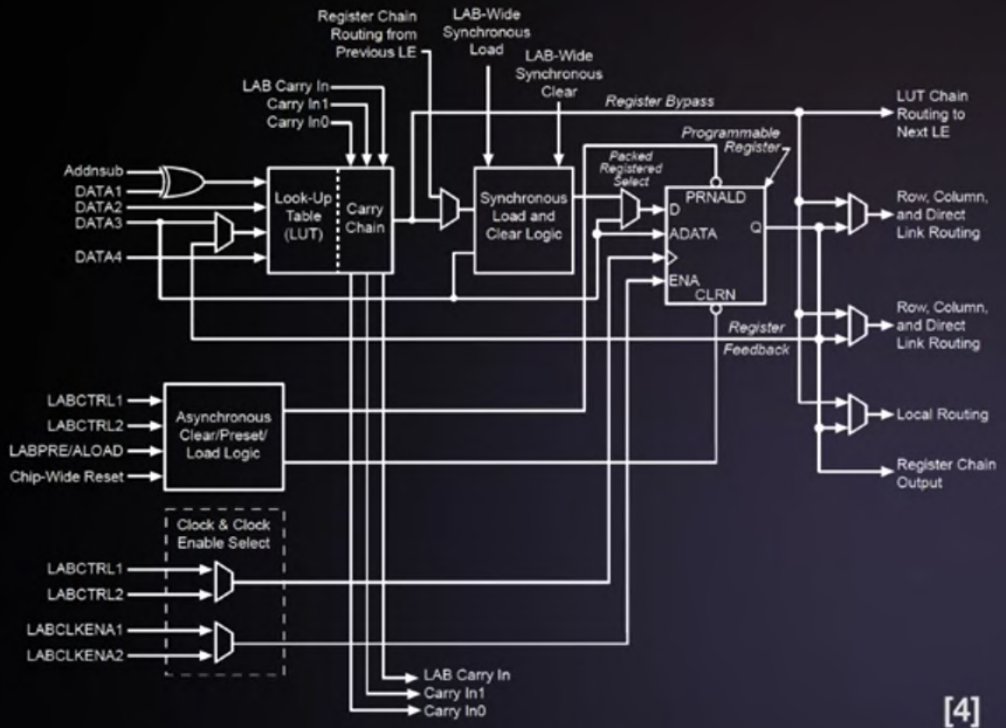


[4]

LUT output is part of a carry chain structure used to speed up arithmetic computations

Output can be routed out directly or registered by the flip-flop

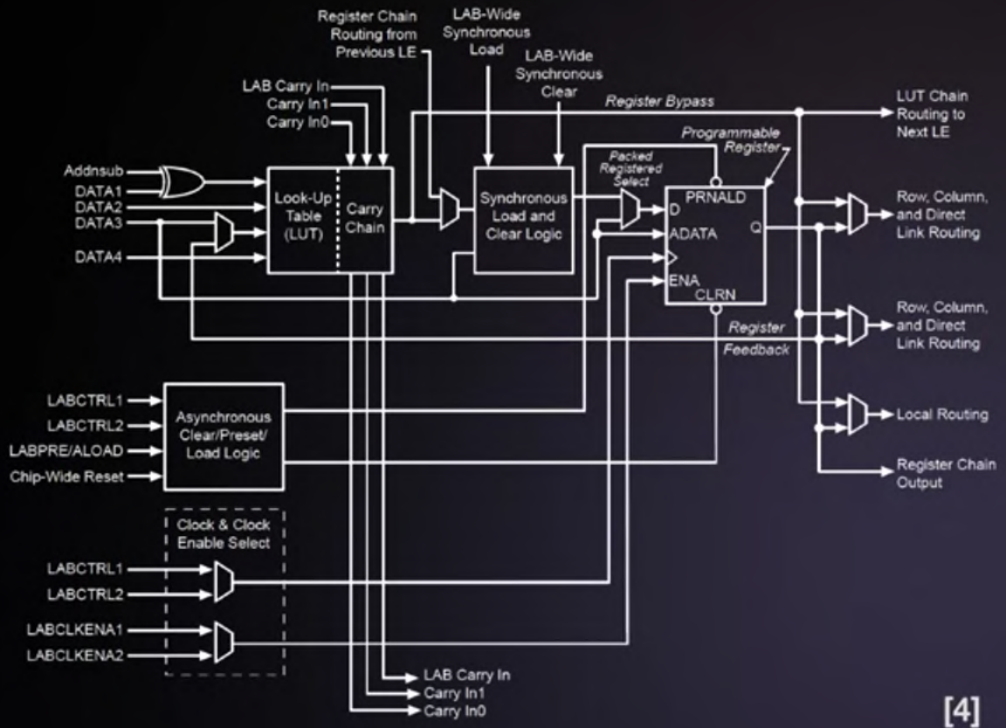
Circuit is designed with versatility in mind.



[4]

FPGA user can determine:

- the function of small clusters of gates
- the logic cell
- how that cluster is connected to other clusters on the chip

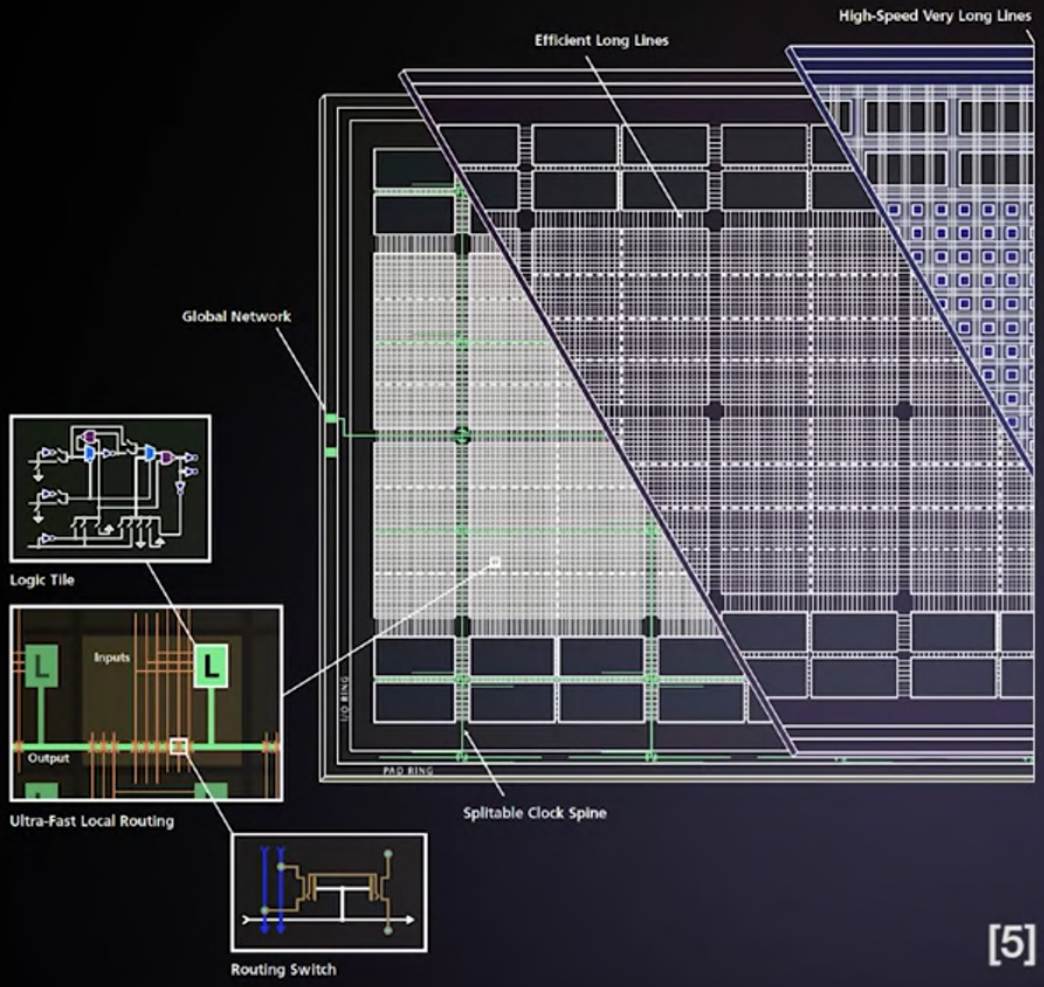


[4]

To implement a high fan-in function:

- FPGAs need to cascade many stages of logic elements
- Can lead to excessive and somewhat unpredictable delays

Timing analyzer tools and design software help resolve problems

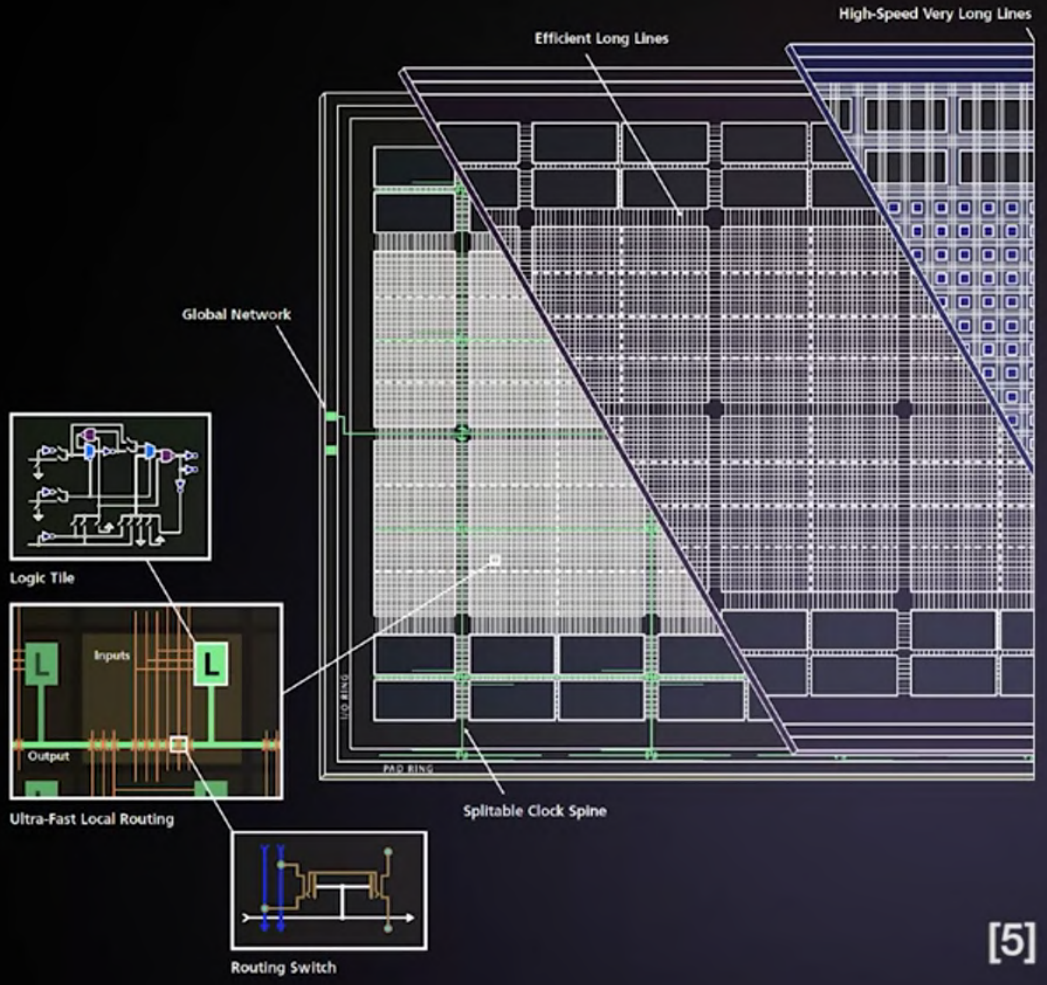


[5]

Routing Switch controlled by SRAM, flash bit, or fuse

Many routing switches in local routing to connect various logic elements or blocks

FPGA is made up of a sea of logic blocks



[5]

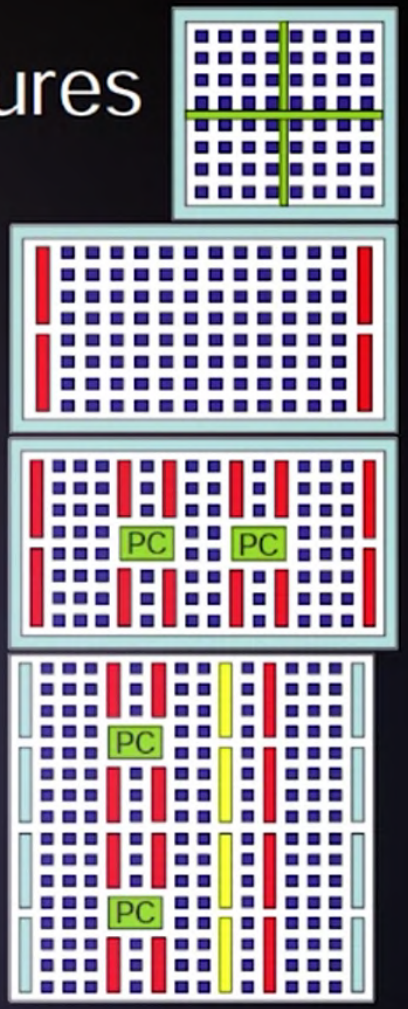
Groups of blocks are connected by long lines or very long lines

Clocks are routed on special networks called Global Networks

Xilinx FPGA Architectures

- 4000/Spartan ~1995
  - *N*x*N* array of unit cells
    - Unit cell = CLB + routing
      - Special routing along center axes
  - I/O cells around perimeter
- Virtex/Spartan-2 ~2000
  - *M*x*N* array of unit cells
  - Added block 4K RAMs at edges
- Virtex-2/Spartan-3 ~2005
  - Block 18K RAMs in array
  - Added 18x18 multipliers with each RAM
  - Added PowerPCs in Virtex-2 Pro
- Virtex-4/Virtex-5 ~2007
  - Added 48-bit DSP cores w/multipliers
  - I/O cells along columns for BGA

C. Stroud 10/11



[6]

Configuration Memory Choice affects Architecture

Antifuse interconnects are highly reliable, but OTP and expensive.

FLASH interconnects are highly reliable, reprogrammable, but more expensive than SRAM FPGAs.

SRAM interconnects reprogrammable and highest density, lowest cost.

Many Programmable Logic Choices

Product	Xilinx	Altera	Lattice	Microsemi
FLASH CPLD	X	X		
SRAM CPLD			X	
AntiFuse FPGA				X
FLASH FPGA	X			X
SRAM FPGA	X	X	X	
FLASH SoC				X
SRAM SoC	X	X		

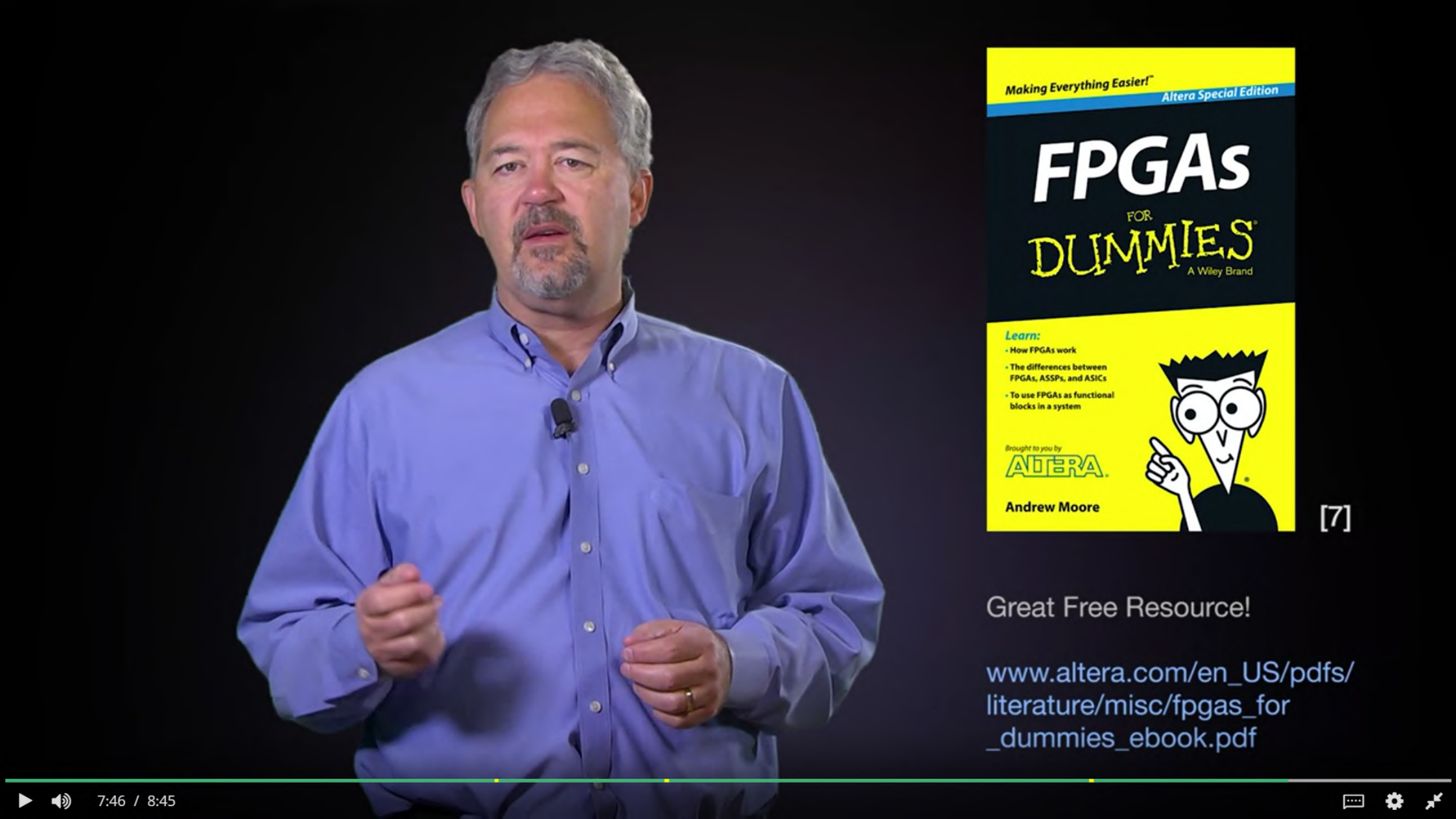
Why use a CPLD instead of an FPGA?

Save money in design requiring just a little logic.

Add more I/O in a small space.

Meet very tight and deterministic timing requirements.

FPGA Development Process Design Flow


Making Everything Easier!™
Altera Special Edition
FPGAs
FOR
DUMMIES
A Wiley Brand
Learn:
• How FPGAs work
• The differences between FPGAs, ASSPs, and ASICs
• To use FPGAs as functional blocks in a system
Brought to you by
ALTERA
Andrew Moore
[7]
Great Free Resource!
www.altera.com/en_US/pdfs/literature/misc/fpgas_for_dummies_ebook.pdf
7:46 / 8:45

FPGA Architecture Summary

FPGAs are highly flexible general purpose digital logic devices. LUTs used as base logic cell.

FPGAs scale better than CPLDs
Lower cost for larger designs, many flip-flops.

Finer grained than CPLDs - Routing has more impact on performance.

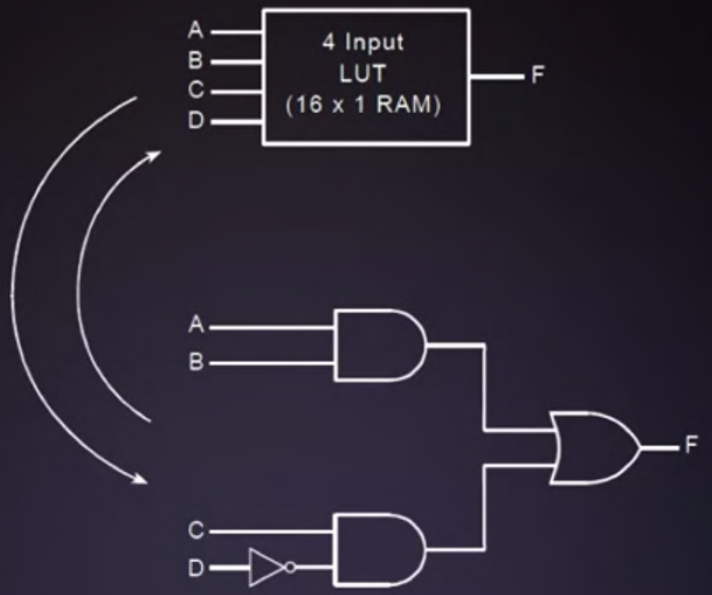
Can be implemented using several technologies:
SRAM, FLASH and Antifuse.
Choice impacts cost and performance.

LUTS and Logic Design

- FPGAs use Look up tables (LUTs) as the base logic element

- LUTs are quite versatile for logic design.

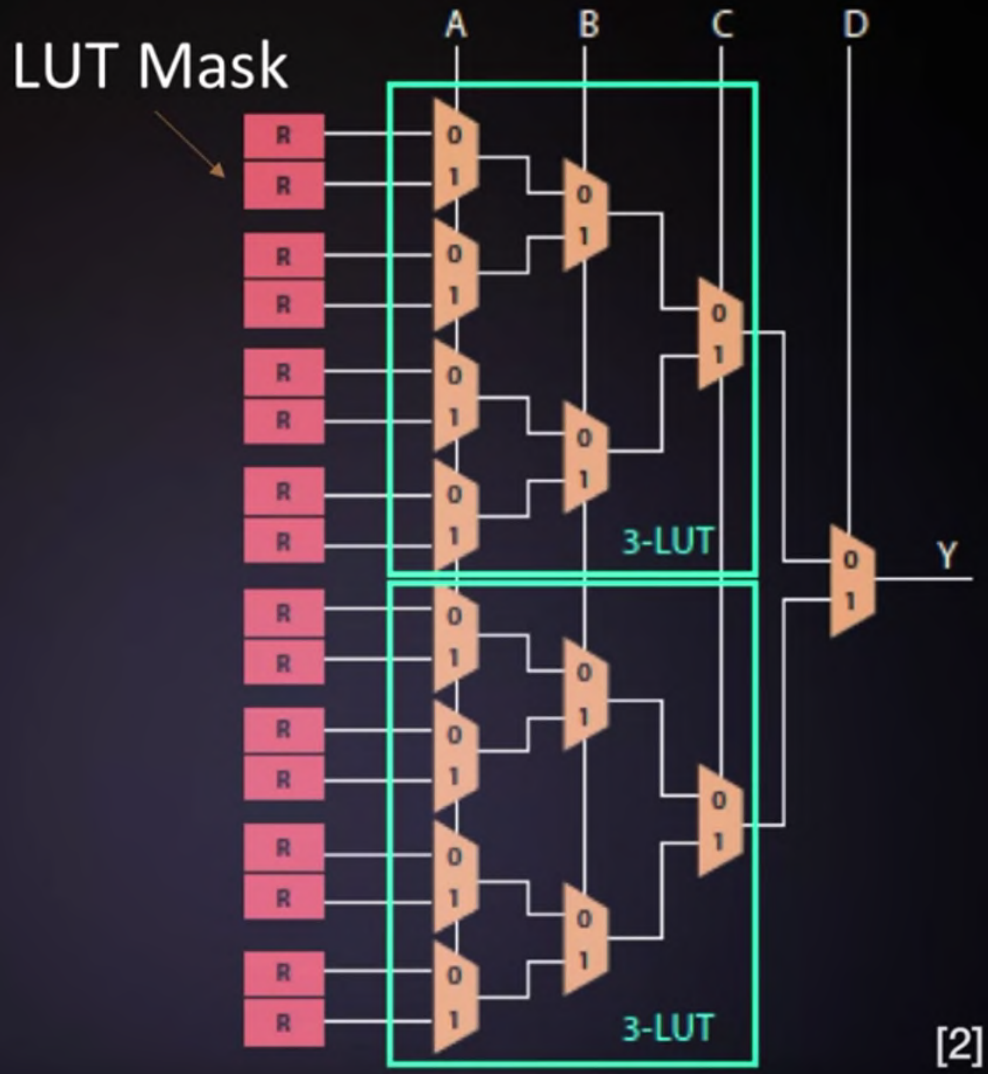
LUT can be used
To implement
any 4-input logic.



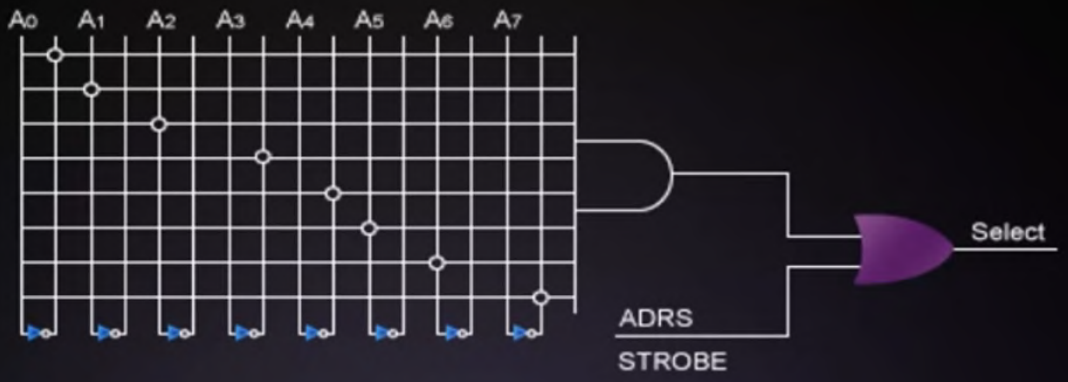
RAM Contents				
Address				Data
A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

[1]

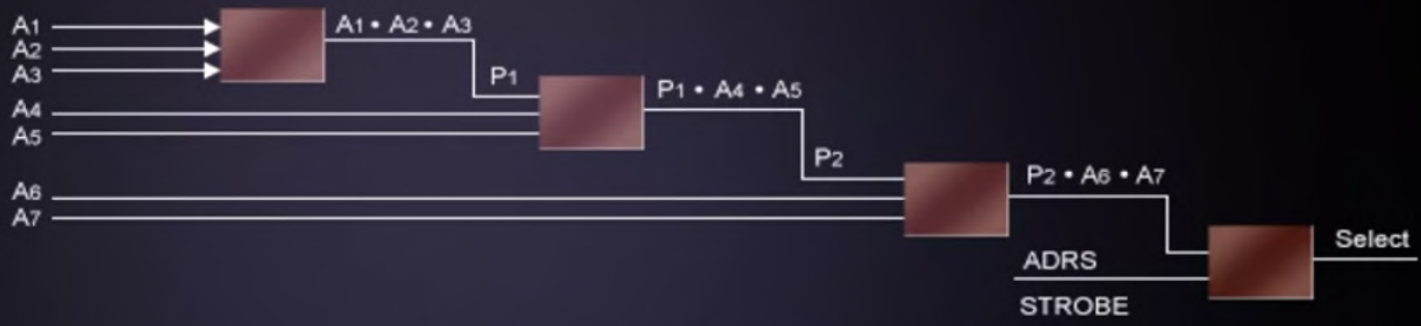
FPGA Implementation of a 4-input LUT



8-bit address decoder using 8-input logic versus 3-input LUTs



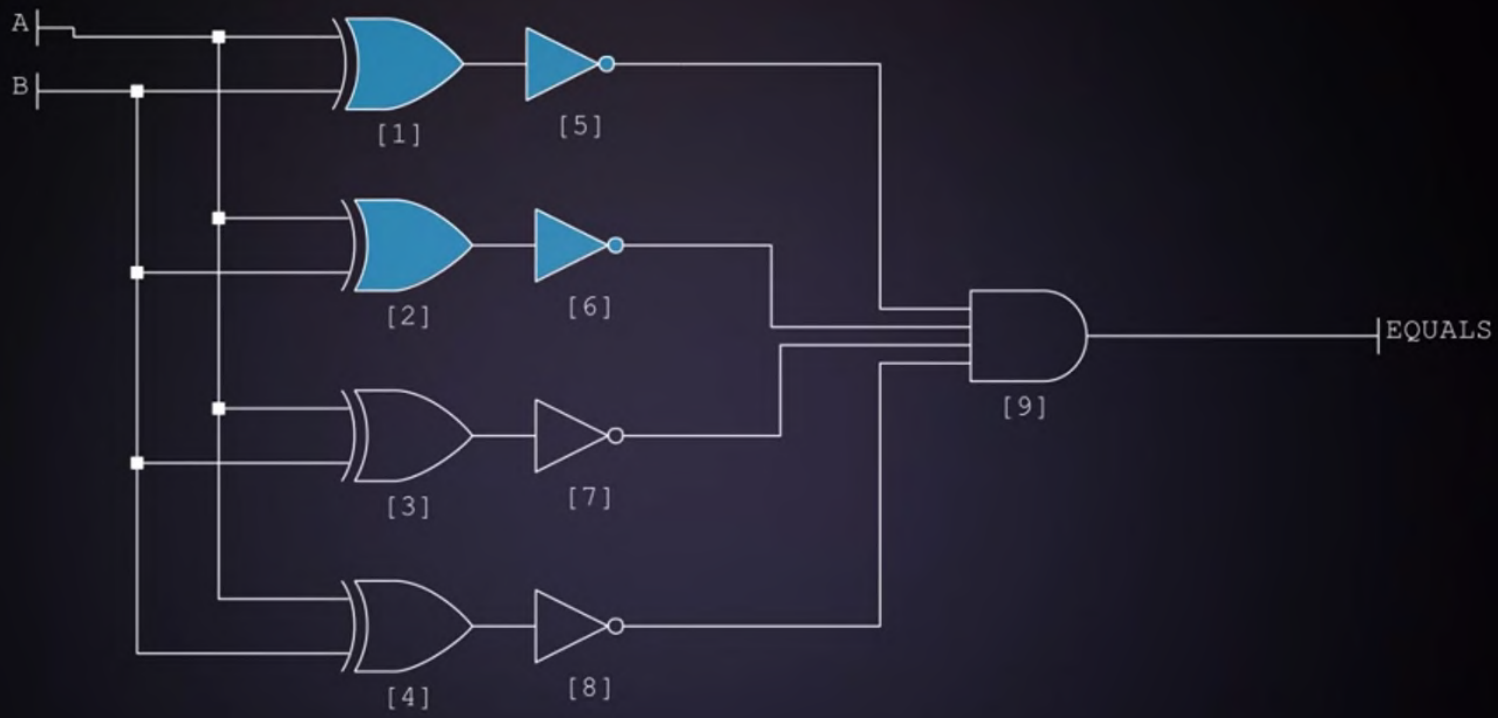
PAL Implementation : 1 p.term



FPGA Implementation - May Have Timing Issues 4 LEs

[3]

How many 4-input LUTs will a 4-bit comparator require?



LUTs and Logic Design Summary

FPGAs use LUTs of various sizes to implement logic.

Tradeoff between size of LUT and amount of routing.

Larger LUTs can create more logic and require less routing.

Smaller LUTs require more routing, but offer better efficiency.

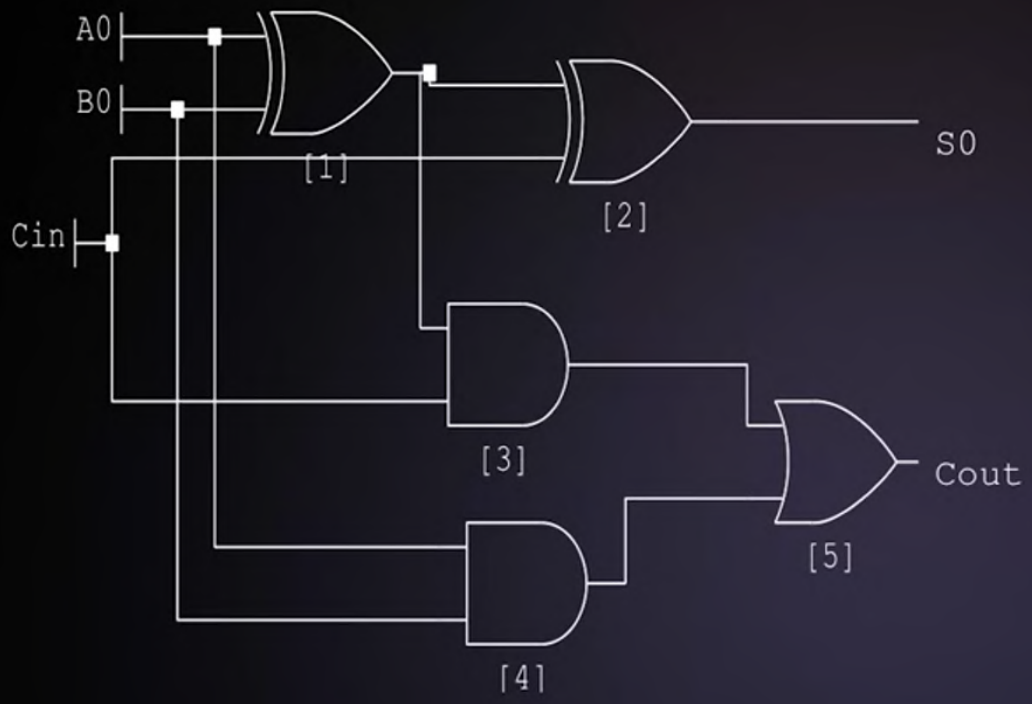
LUTs can be used to create a variety of combinatorial logic functions.

Designing Adders in FPGAs

The heart of every digital computer is the CPU.
The heart of every CPU is the ALU.
The heart of every ALU is the adder circuit.

Arithmetic circuits can easily be built in FPGAs using logic elements or special arithmetic circuitry if available.

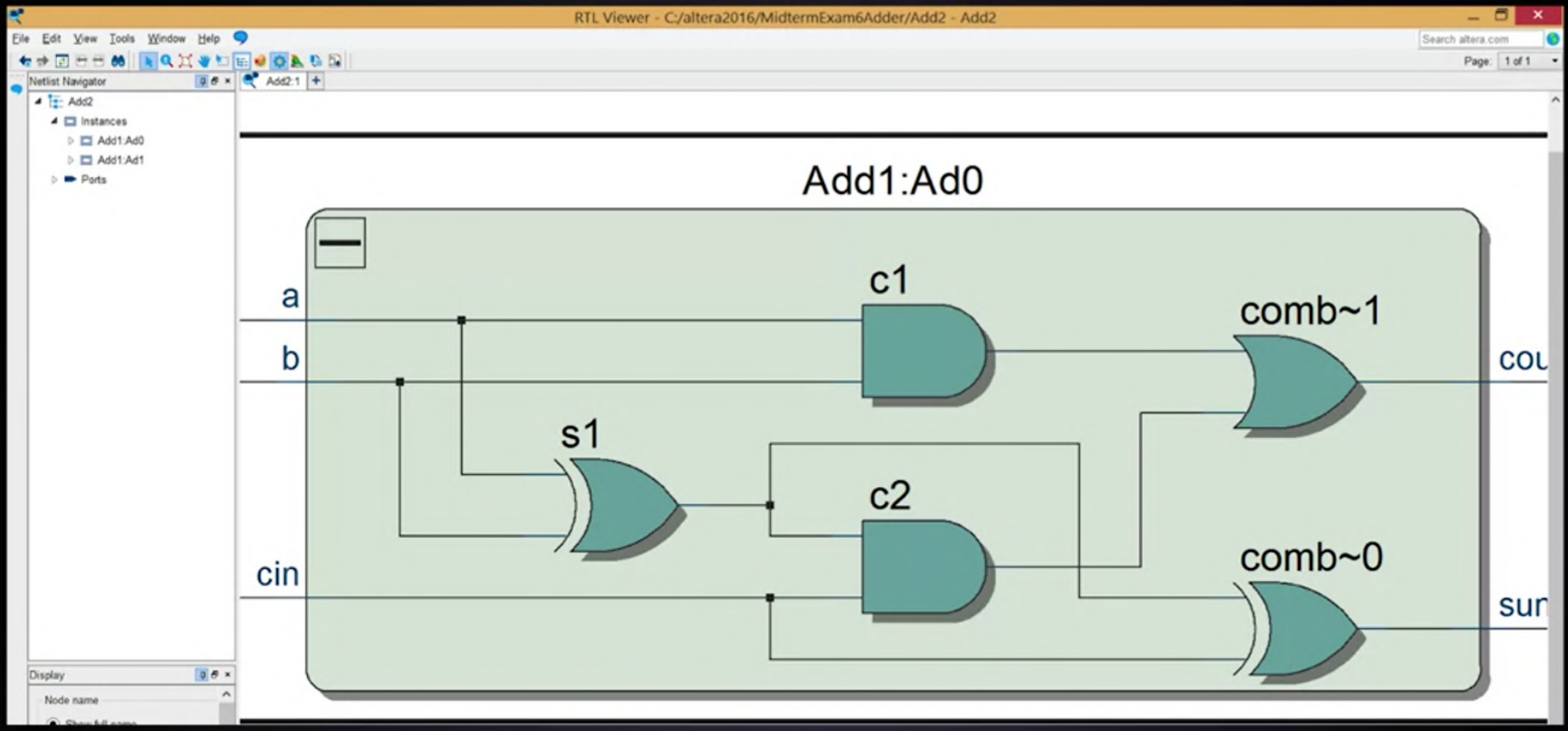
Full 1-bit Adder in Gates



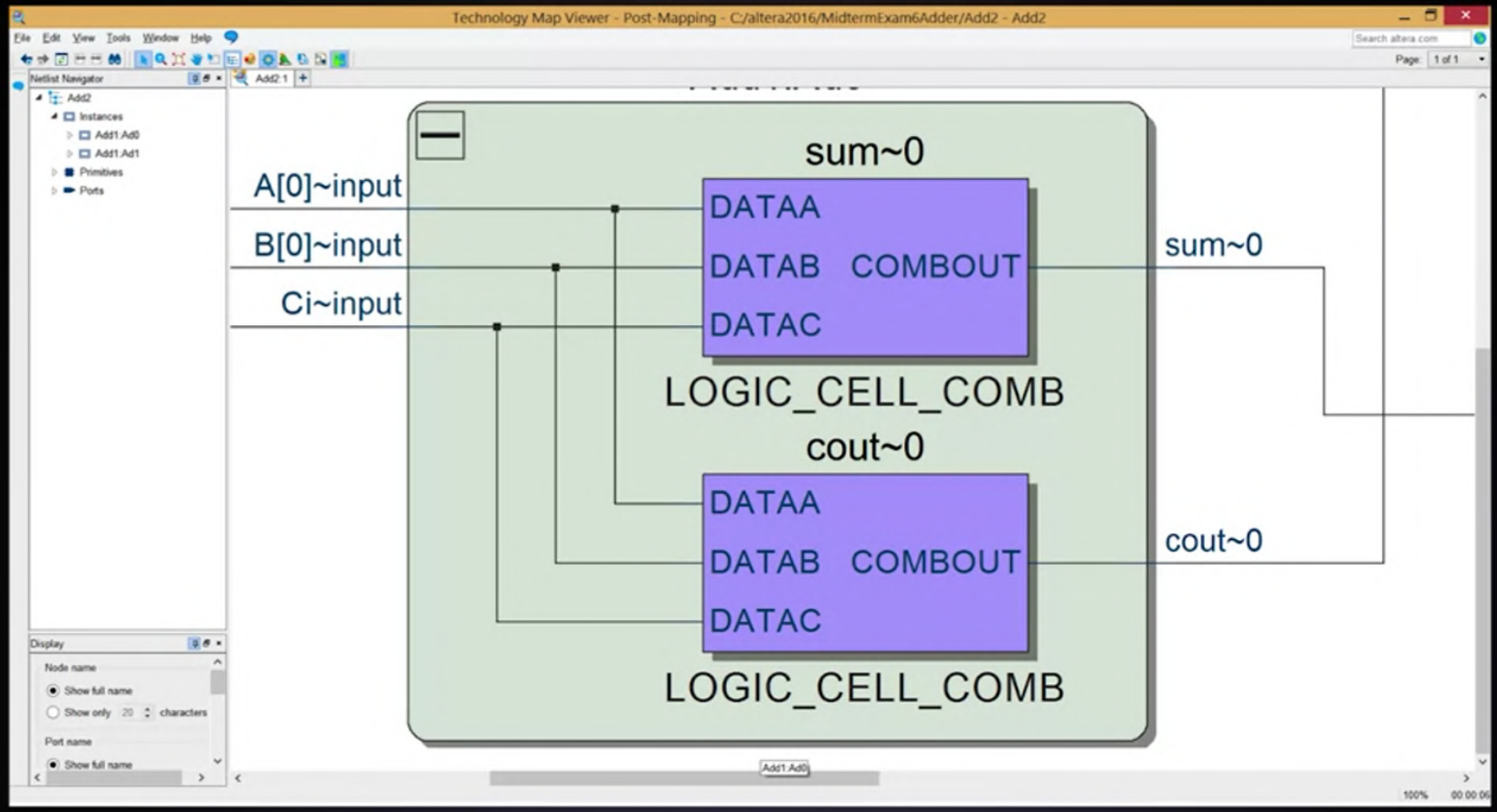
Truth Table

A0	B0	Cin	So	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Full 1-bit Adder implemented in FGPA , RTL View



Full 1-bit Adder implemented in FGPA , Technology View



Ripple Carry Adder Delay

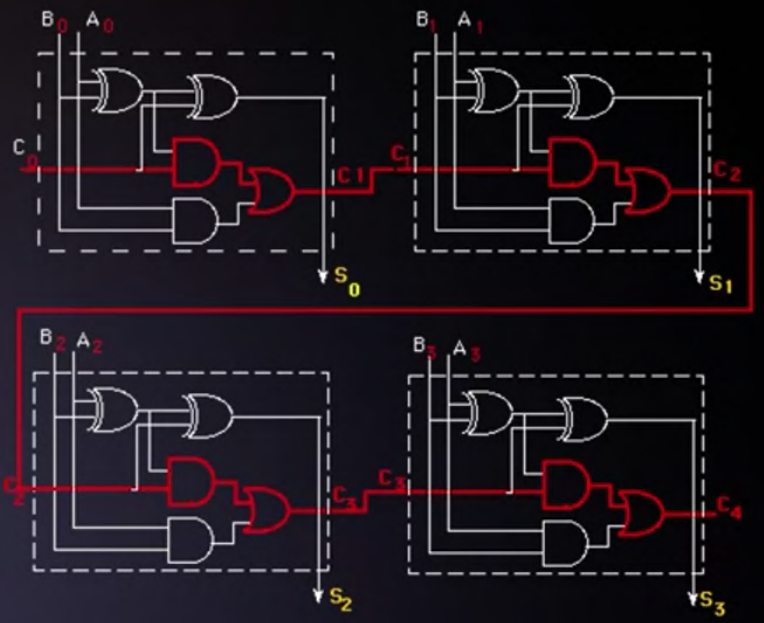
Total delay:

Tadder(n) = (n-1) * Tc + Ts = (n-1)*3D+2D = (3n-1)D, where D is the delay through one gate, and n is the number of bits in the adder.

How many gate delays would there be for a 32-bit adder constructed this way?

Answer: 95 Gate Delays

[2]



[1]

Full 4-bit Adder made of using Carry Look Ahead (CLA) Adders to reduce delay

The Carry bit can be calculated as generate and propagate terms:

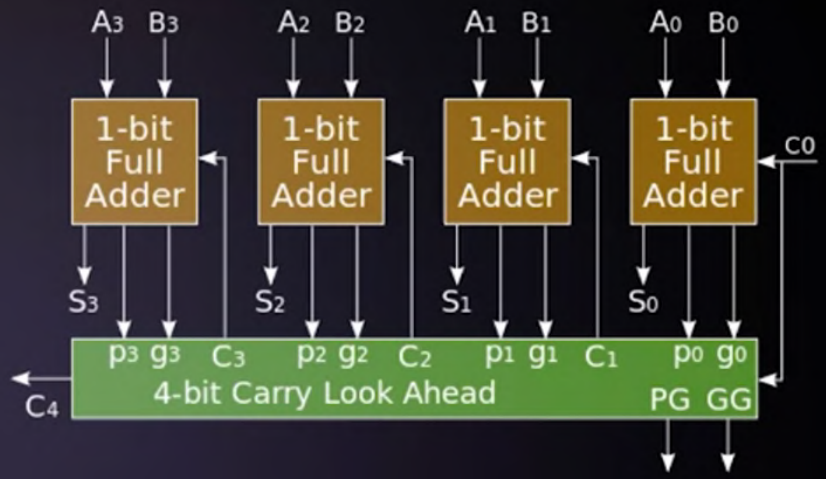
$C_{i+1} = G_i + P_i.C_i \qquad (1)$

in which

$G_i = A_i.B_i \qquad (2)$

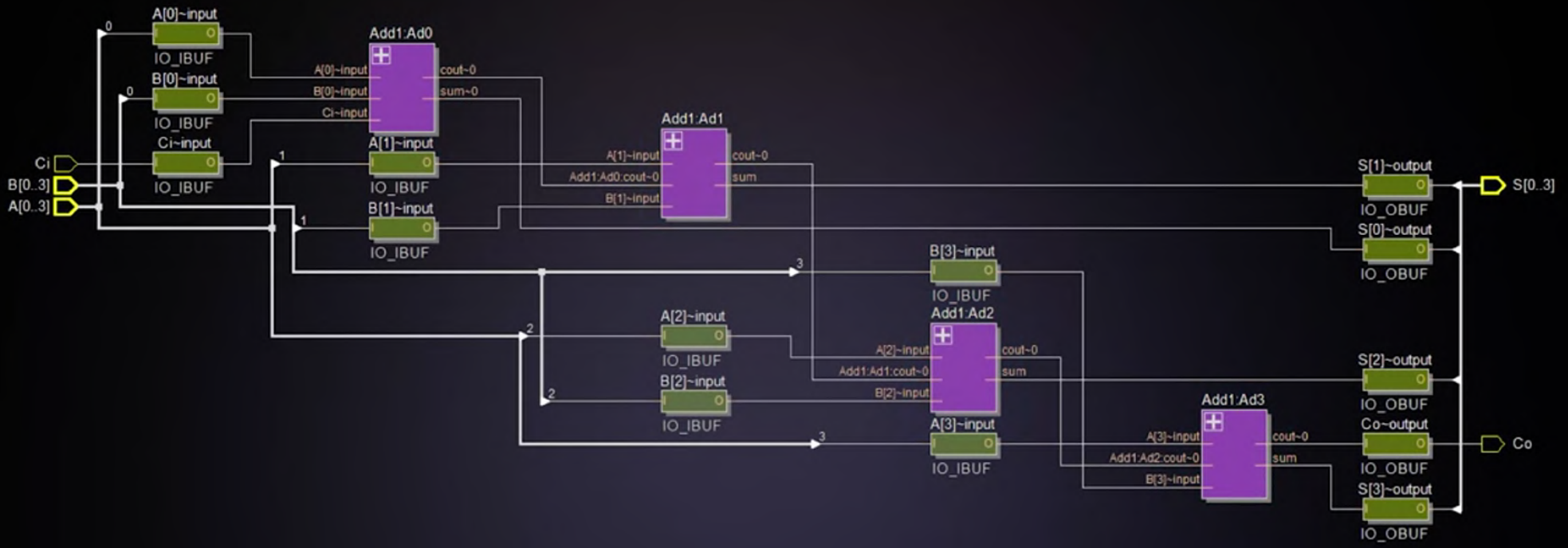
$P_i = (A_i \text{ Å } B_i) \qquad (3)$

Now the delay is roughly equal to n

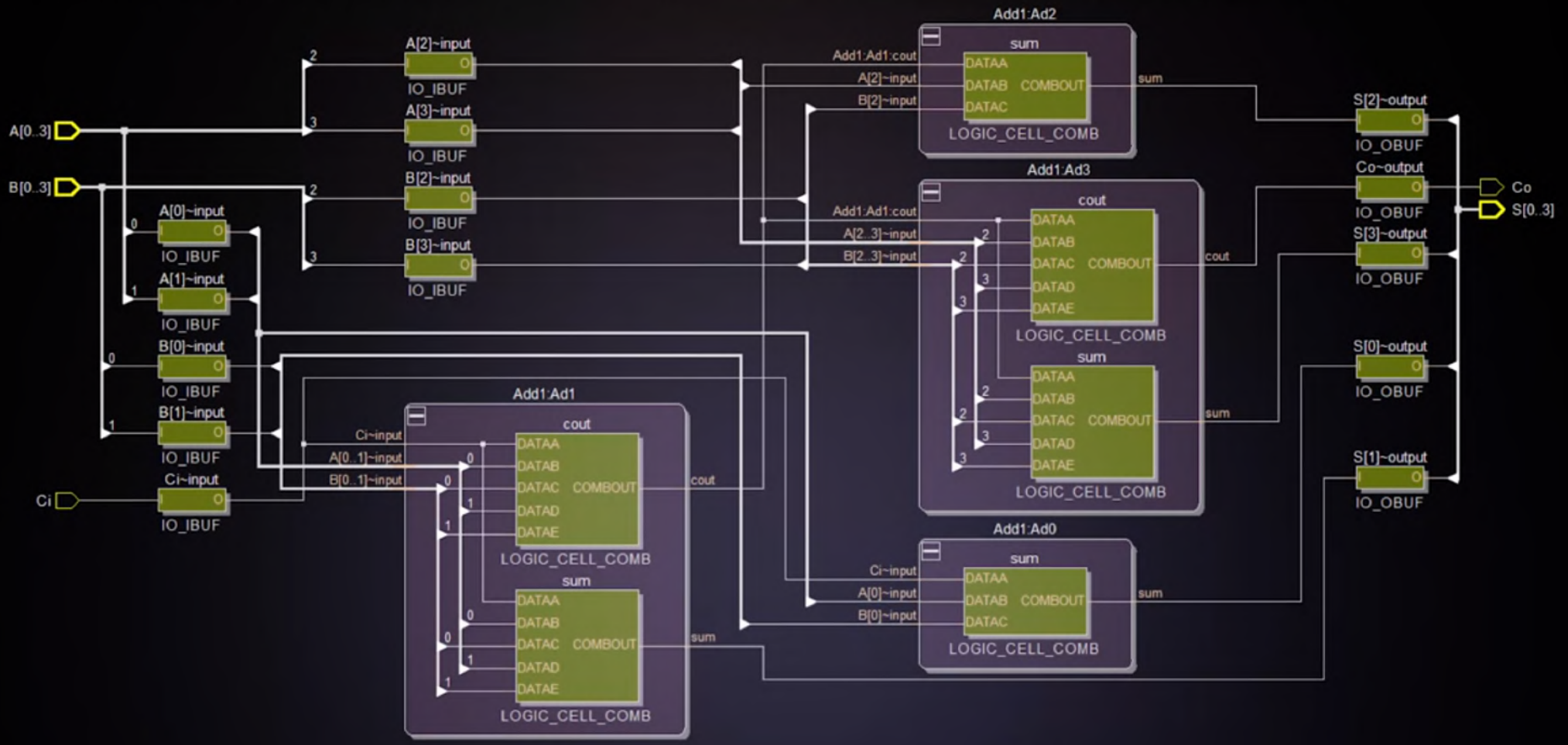


[3]

Full 4-bit Adder implemented in FGPA , Technology View.

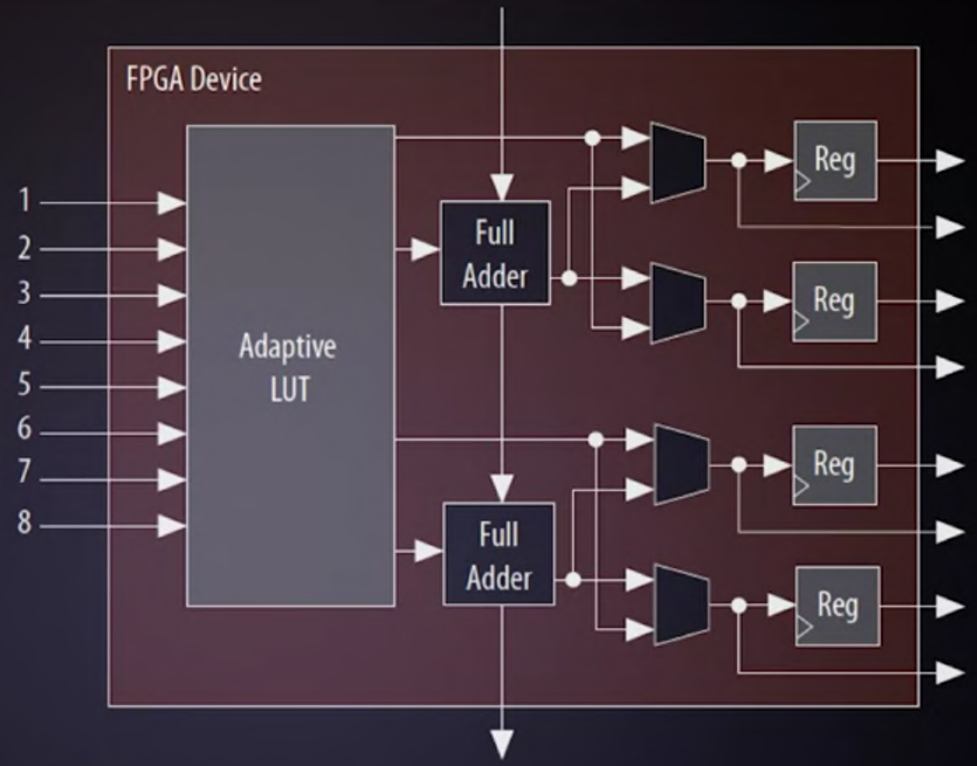


Full 4-bit Adder implemented in Cyclone V FGPA , Technology View.



Altera

Figure 8: ALM for Cyclone V Devices



Designing Adders in FPGAs Summary

Adders are a fundamental digital circuit function, easily implemented in FPGAs

Standard ripple-carry adders are easy to design, slow in performance. Other circuits can perform better, but require more logic.

Use of LUTs, carry chains and full adders make FPGAs efficient at implementing adders.
Standard designs can have excellent performance.

[4]

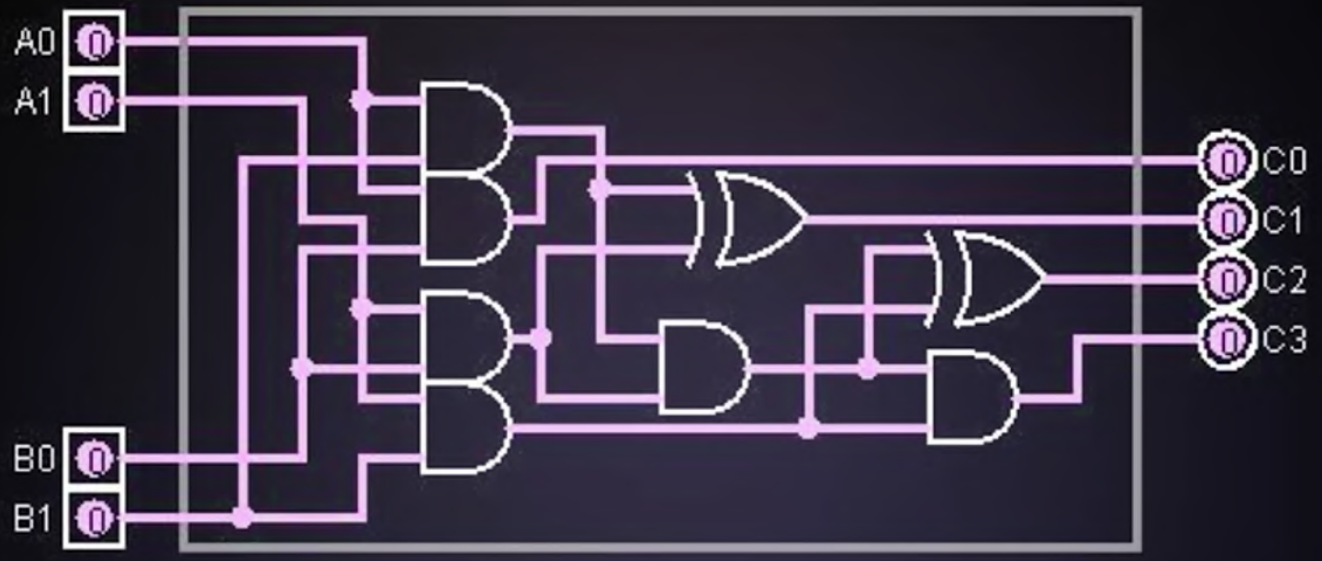
Designing Multipliers in FPGAs

Multiplying 2 numbers is a common digital logic or ALU function that can take considerable amounts of circuitry.

There are a variety of ways to implement multiplier circuits within FPGAs.

Multipliers in Digital Logic: Gates

Adders are built with Gates, can we build multipliers this way, too?



A 2-bit by 2-bit binary multiplier [1]

Multipliers in Digital Logic: Gates

Array multipliers
- Partial products
- Partial product summation

Gates increases as n^2

Beyond 4x4, a sequential design is more efficient

# bits	Approximate # gates
2x2	8
3x3	54
4x4	96
5x5	150
6x6	216
7x7	294
8x8	384

[2]

Many Ways to Build a Multiplier in an FPGA

1. Combinational circuits – fast but big
2. Sequential shift and add (state machine approach)
3. Specialty algorithms, like Booth's Algorithm or the Dadda Multiplier or Wallace Tree Multipliers
4. Memories (Look up tables)
5. Some combination of the above
6. Hard Multiplier Blocks if available

Many Ways to Build a Multiplier in an FPGA

Sequential shift and add (state machine approach)

Smaller in area than combinational multipliers beyond 4x4 multiplies, but slower as 2n clock cycles are required for shifting and adding.

Many Ways to Build a Multiplier in an FPGA

Memories for Multiplication

A 4x4 multiplication can be achieved with a memory with an 8-bit address.

A7…A4 is the multiplier and A3…A0 is the multiplicand. 2^8 or 256 locations are needed in the memory.

Although memory cells are inexpensive, beyond 8x8 this becomes too expensive.

Many Ways to Build a Multiplier in an FPGA

Hard Multipliers provide the greatest speed of any FPGA implementation, running at 200 MHz or more (50 ns per multiplication).

Typical MAX 10 or Cyclone V parts will have dozens to hundreds of 18 x 18 multiplier circuits available for your use.

Designing Multipliers in FPGAs Summary

Creation of Digital Multipliers is a challenging design problem. Most solutions have significant drawbacks

Implement multipliers in FPGAs:
Combinational circuits
Sequential circuits (state machines)
Specialty circuits (Booth's algorithm, etc.)
Memories
Hard multiplier blocks

Many FPGAs contain multiple hard multiplier blocks which allow very fast 18x18 multiplication.

[3]