Digital System Design
ELEC3342

Introduction to Field Programmable Gate Array (FPGA)

Dr. Hayden So

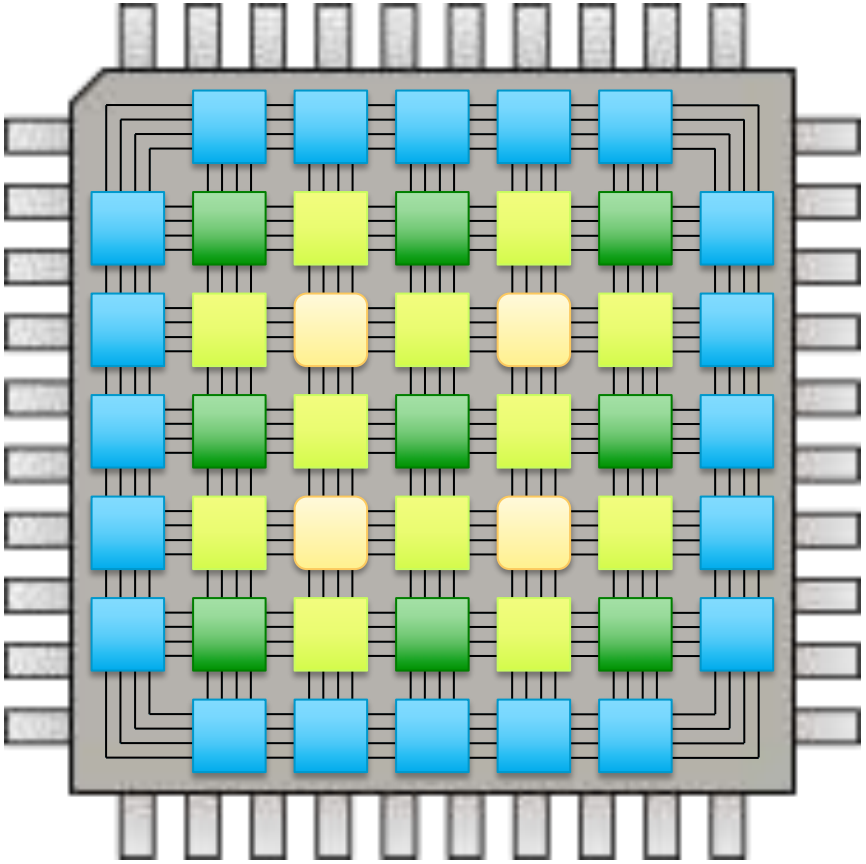
Department of Electrical and Electronic Engineering

http://www.eee.hku.hk/~elec3342

FPGA – Not long ago…

- Field Programmable Gate Array
- Began as ASIC alternatives
  - ASIC that can be configured “in the field”
  - At power up, configuration is load to the chip
  - Chip acts as an ASIC until power down
- Used as…
  - Glue logic
  - ASIC pre-fabrication “Prototype”
  - ASIC replacement for small volume, fast time-to-market

Simplified View of an FPGA



Island-style Layout

Basic Idea:

- Logic blocks implement combinational logic + register
- Connect logic blocks through switches and connection boxes

Logic Block

Switch Block

Connection Block

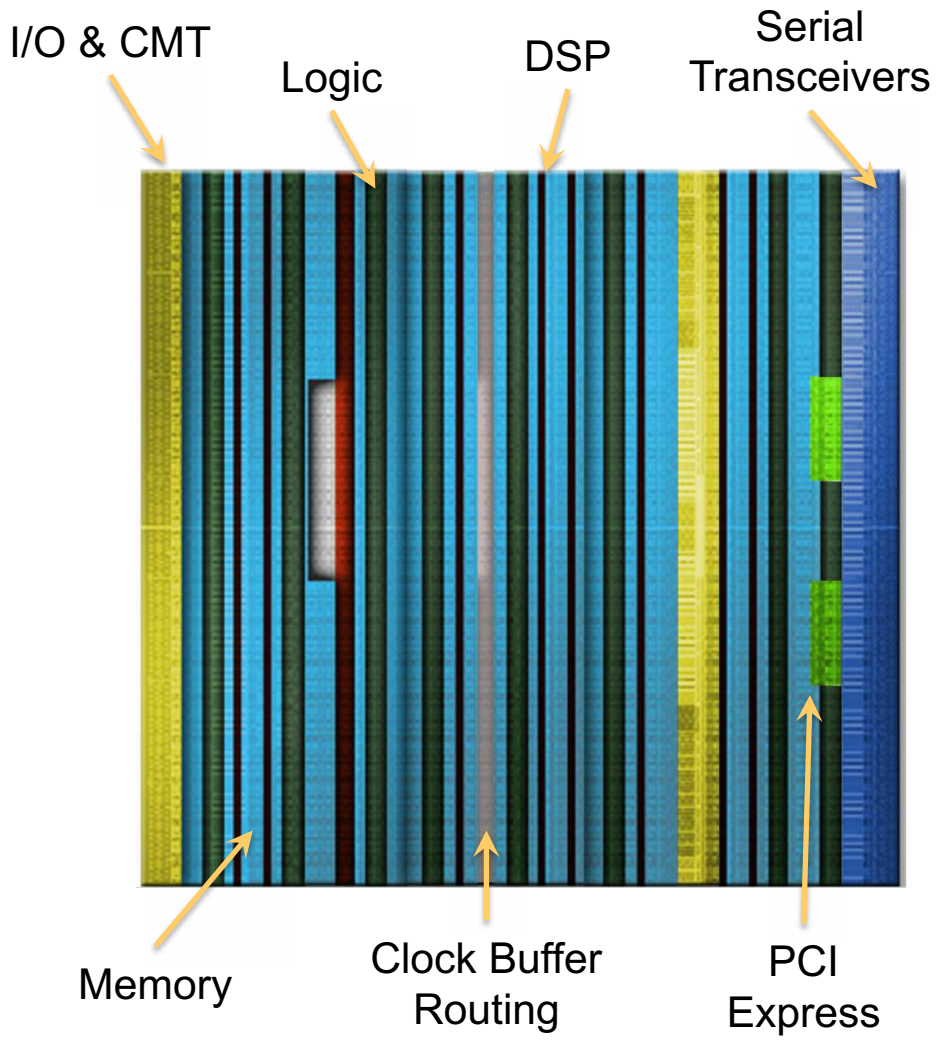
I/O Block

Modern FPGAs

- Embedded processors
  - Soft+Hard: PowerPC/ARM
- On-chip memory
  - Soft+Hard
- On-chip FIFO
- Dedicated DSP blocks
  - 23x18 multipliers, pre adder
  - MAC
  - Floating point
- High speed Serial I/O
- Plenty of parallel I/O
  - Configurable impedance
  - Configurable delay
- On-chip clock management
  - MMCM, PLL, multiple clock regions
- On-chip hard IP:
  - Ethernet MAC
  - PCI endpoint
- On-chip ADC
- Size
  - > 1 million “equivalent ASIC agates”
- AI core
- System-level reconfiguration
- Dynamic Partial *Re*configuration
  - Reconfigure part of the chip while the rest continues to operate

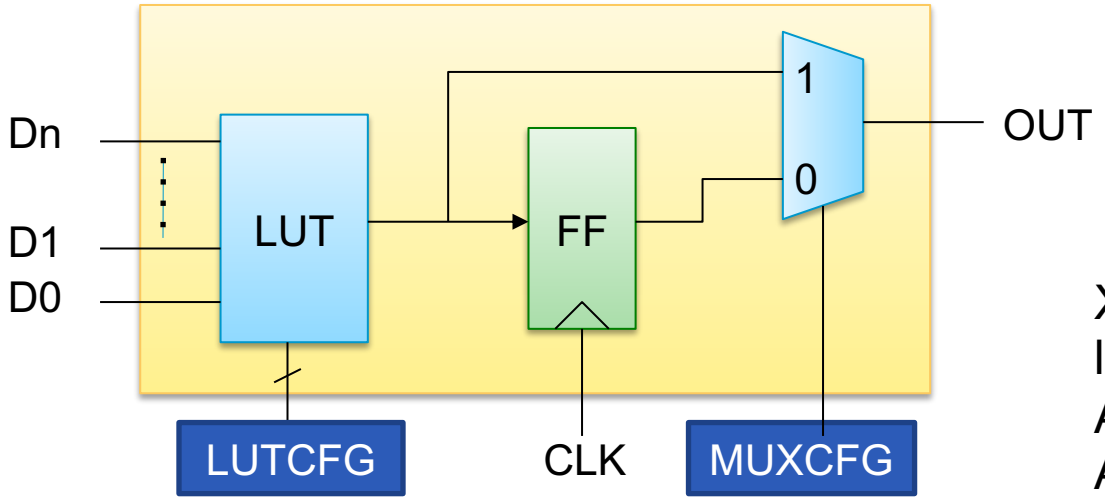
Modern FPGA Layout

- Island style connection
- Features arranged as columns
  - I/O columns on peripheral
  - Clock management next
  - Logic and Memory I/O next
- Modular design
- Clock routing in middle



Xilinx 7-series FPGA Layout (source Xilinx)

Logic Block

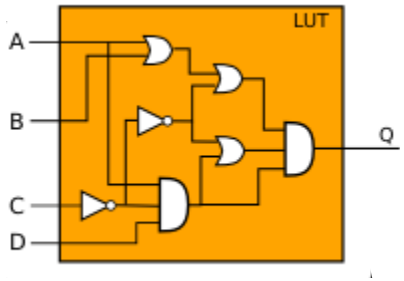


Xilinx: Configurable logic block (CLB)
Altera: Logic and Arithmetic Block (LAB)

- Each n-input lookup table (LUT) may be configured to perform ANY n-input combinational logic
- The result of the LUT is optionally registered by the FF
- Function controlled by configuration bits (LUTCFG, MUXCFG, etc)
  - Store in configuration memory

The LUT

- LUT: ***L***ook***u***p ***T***able
- A direct implementation of a truth table
  - Recall a TT uniquely defines a circuit



A	B	C	D	Q
0	0	0	0	0
0	0	0	1	0
		...		
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

- An n-LUT implements any n-input combinational logic
  - Depends on LUT configuration

Making a 2-LUT from Truth Table

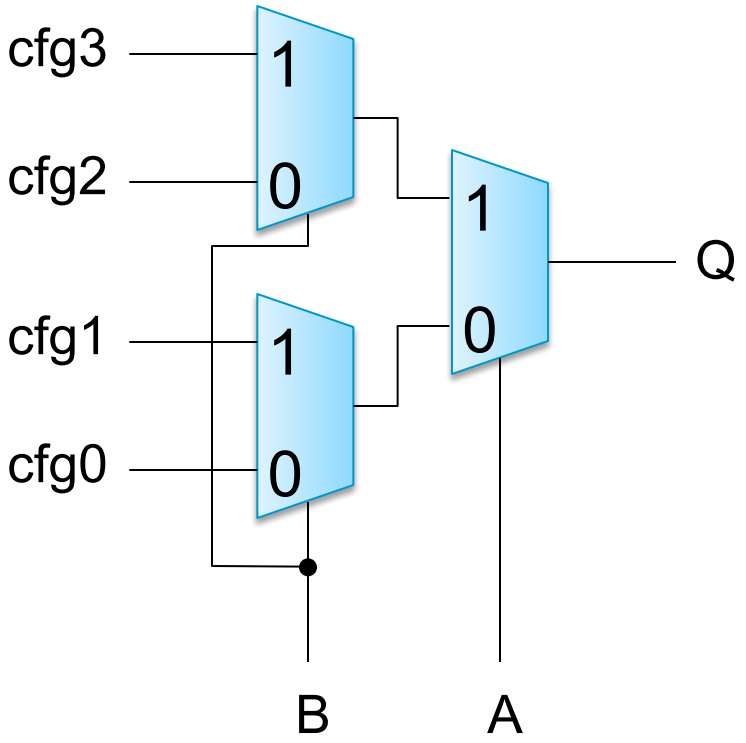
A	B	FALSE	AND	…	OR	…	NAND	TRUE
0	0	0	0		0		1	1
0	1	0	0		1		1	1
1	0	0	0	…	1	…	1	1
1	1	0	1		1		0	1

16 possible ways

4 configuration bits

2-LUT: MUX Based

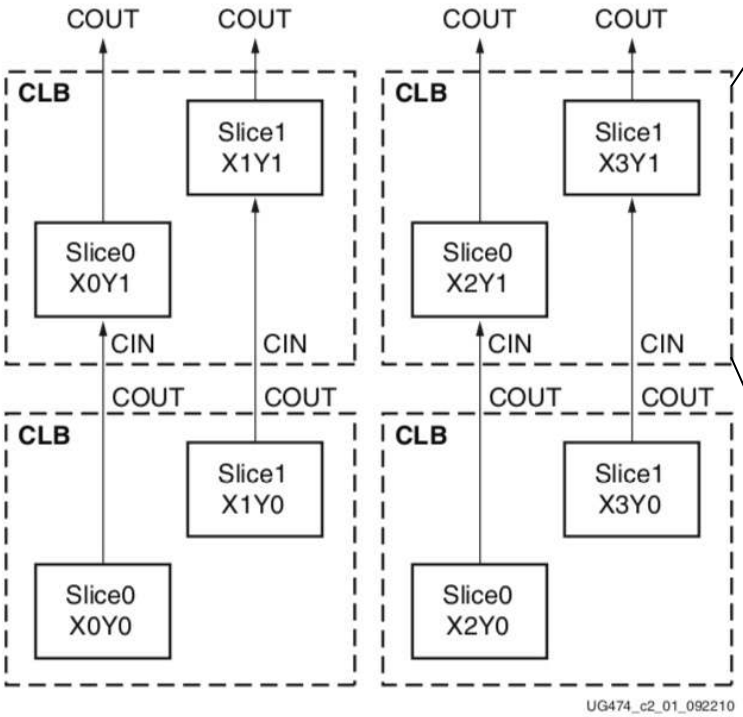
cfg3	cfg2	cfg1	cfg0	A	B	Q
0	0	0	0	0	0	0
0	0	0	0	0	1	0
0	0	0	0	1	0	0
0	0	0	0	1	1	0
0	0	0	1	0	0	0
0	0	0	1	0	1	0
		...				
1	1	1	1	1	0	1
1	1	1	1	1	1	1

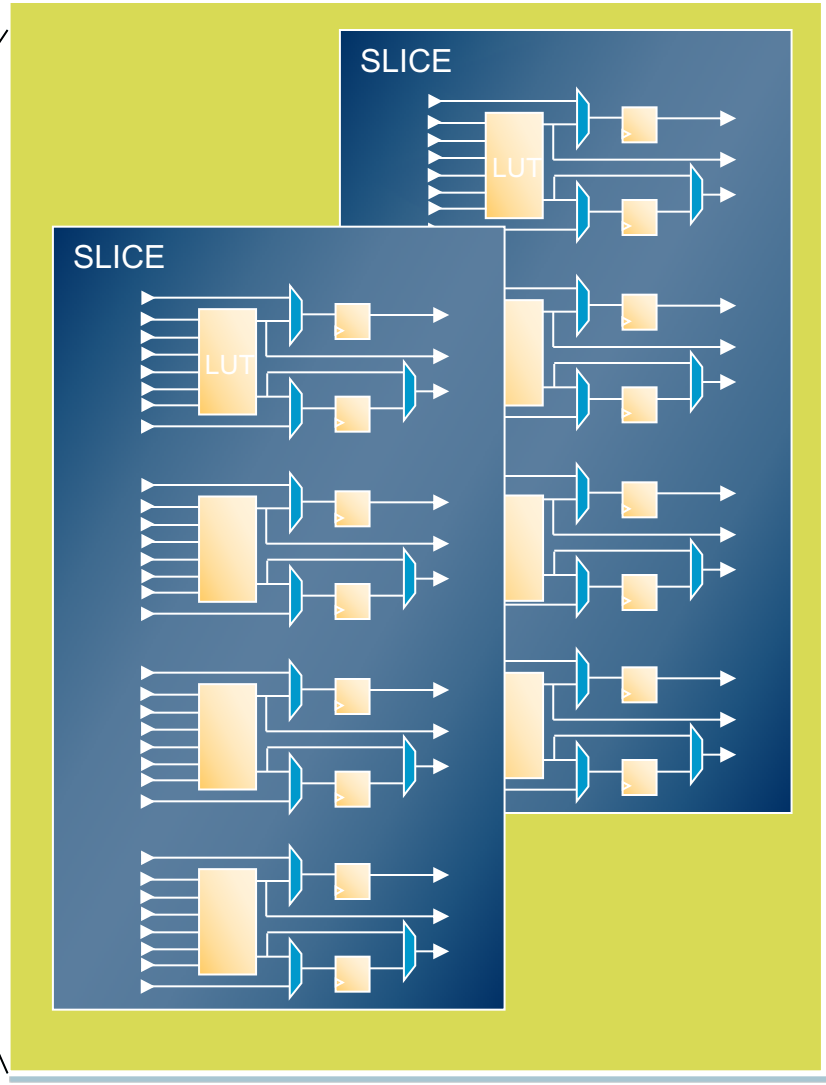


LUT in Real Life

- 3-LUT and 4-LUT are most common
- 6-LUT in latest Xilinx FPGA (Virtex-5, Virtex-6)
- SRAM based
  - Configurations are stored as SRAM “content”
  - Input signal acts like “address” that reads the SRAM content
- SRAM-based LUT can be reusable as a real SRAM if needed
  - Xilinx calls it Distributed RAM

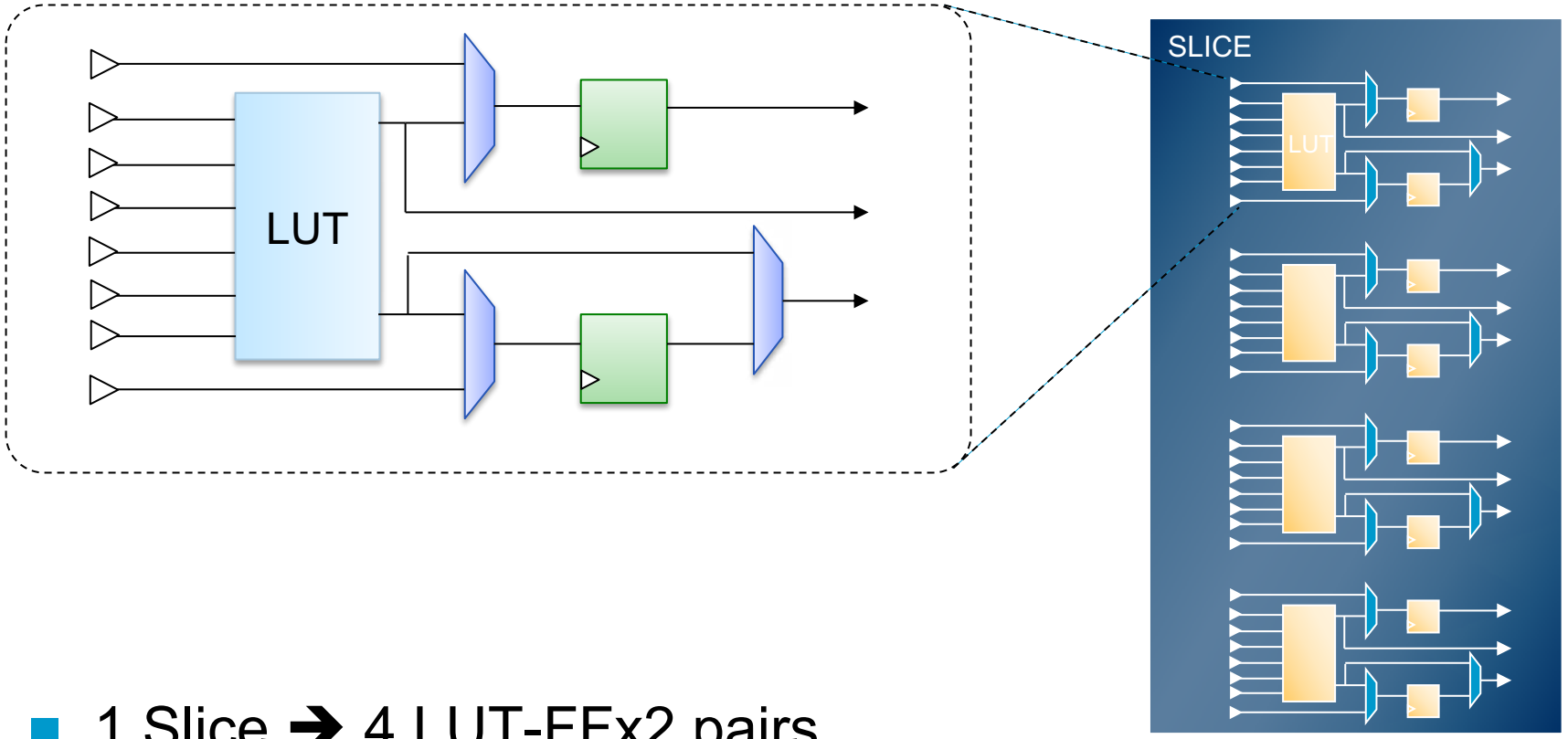
Xilinx CLB





- 1 CLB ➔ 2 Slices
- Slices connected vertically with dedicated carry chain

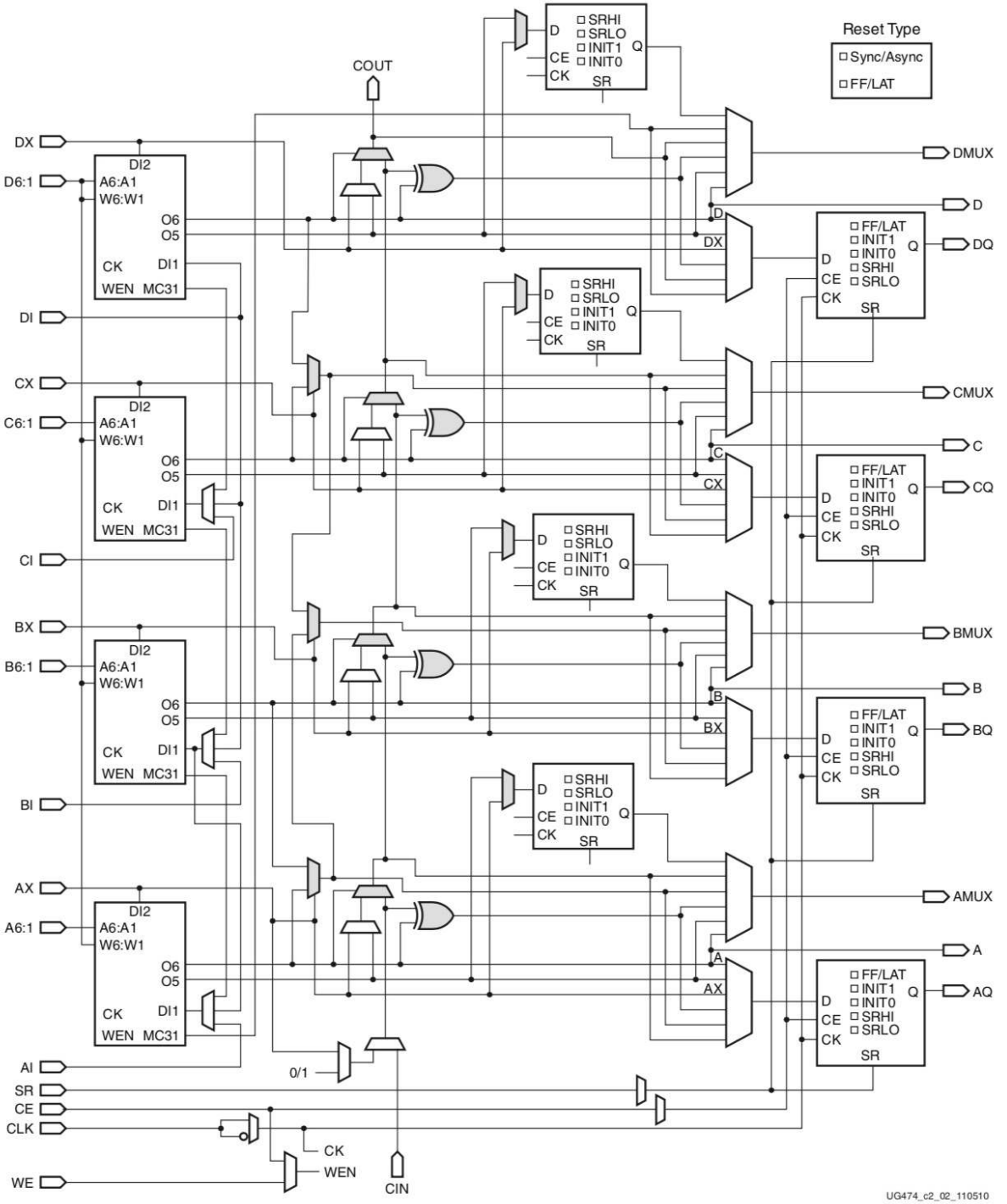
Xilinx Logic Slice (Overview)



- 1 Slice ➔ 4 LUT-FFx2 pairs
- 6-LUT with additional muxes between LUTs within a slice
- Dedicated carry logic to create fast adders
  - Intra- and inter slices

Source: Xilinx

Xilinx Logic Slice



Source: Xilinx

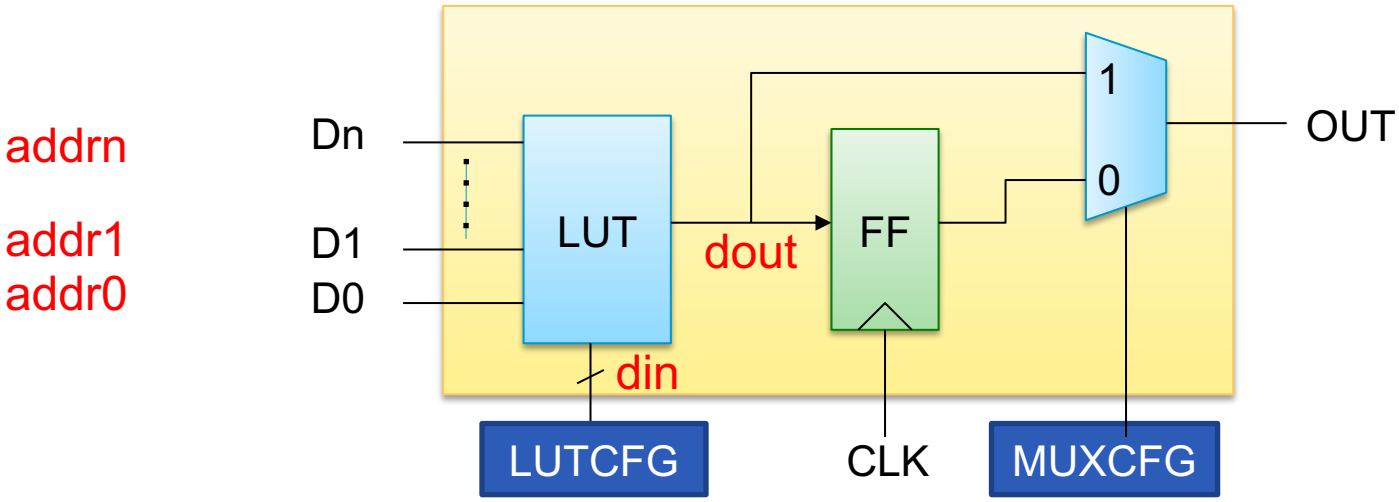
On-Chip Memory

- Modern FPGAs have 2 main types of on-chip memory
  - Dedicated on-chip memory (Block RAM)
  - LUT-based memory
- Both implemented with SRAM technologies
- Dedicated Block RAM has higher capacities, higher density due to dedicated internals
- LUT-based memory make memory out of the LUTs in Logic Blocks

LUT-based Memory

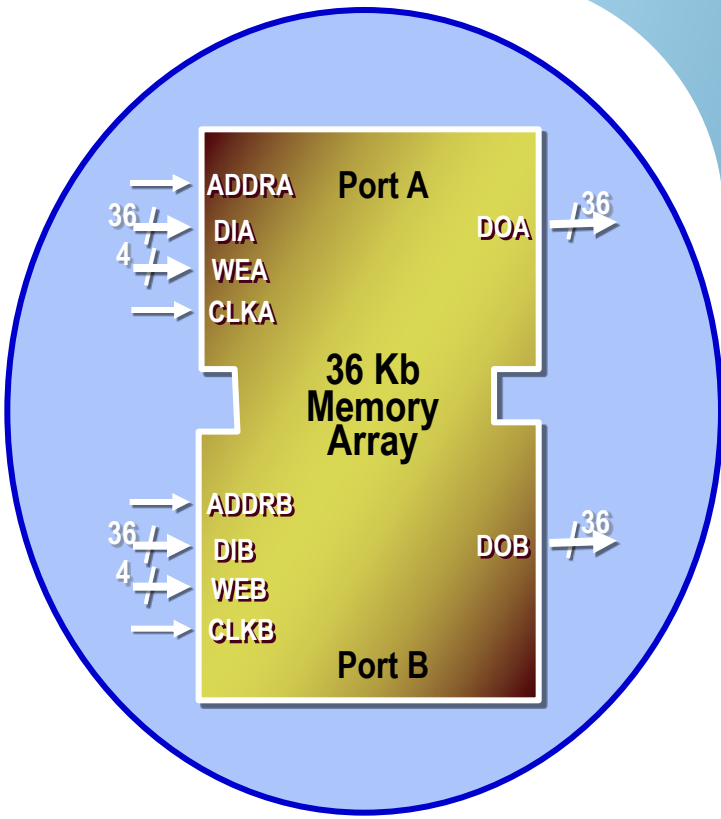
- Recall that the core of a logic block is 6-LUT
  - A look up table = a ROM
  - Configuring the content of the LUT turns it into a combinational circuit
- Standard FPGA configure LUT content through dedicated configuration logic
- Idea: expose the configuration logic to the LUT to a user application:
  - Configuration ➔ writing to memory
  - Configuration bit ➔ address
  - LUT data out ➔ read data port
  - Configuration data ➔ write data port

Turning a LUT into Memory



- Each 6-LUT becomes a 1-bit memory with 6 bit address (64 locations)
- b 6-LUTs in parallel ➔ 64 x b memory

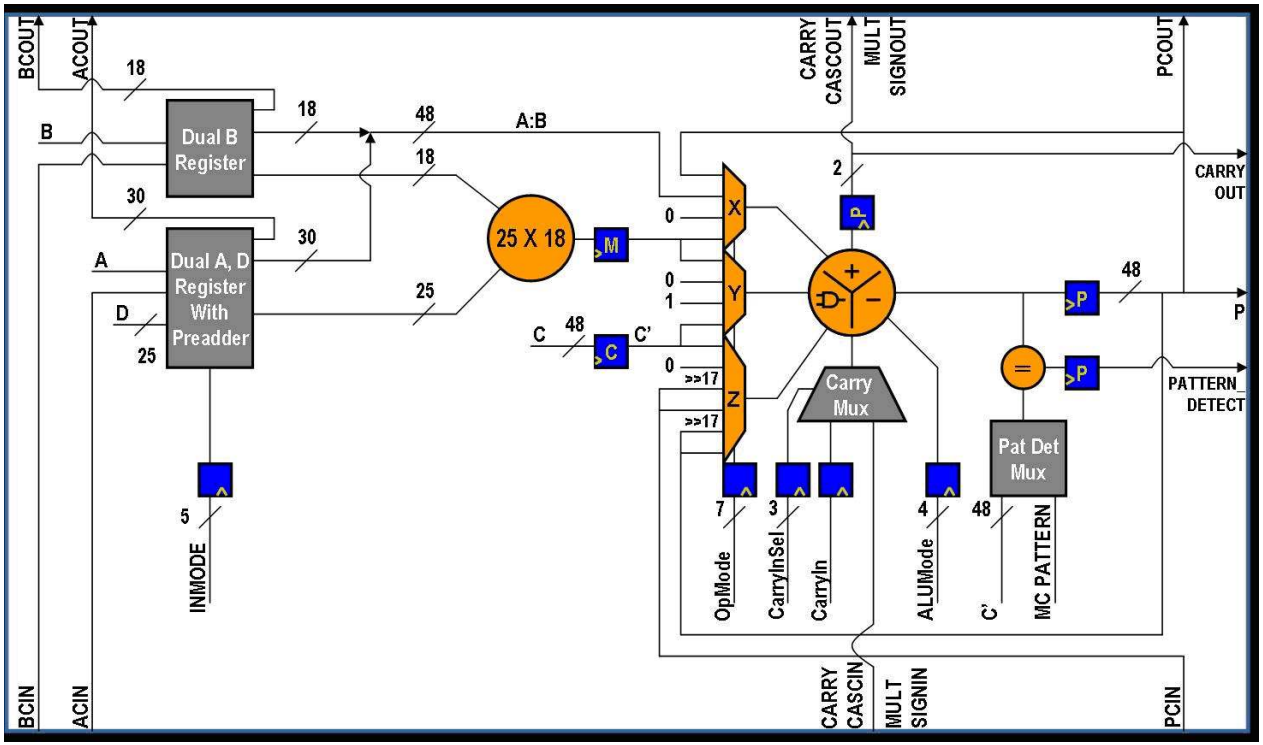
Hard Memory Block



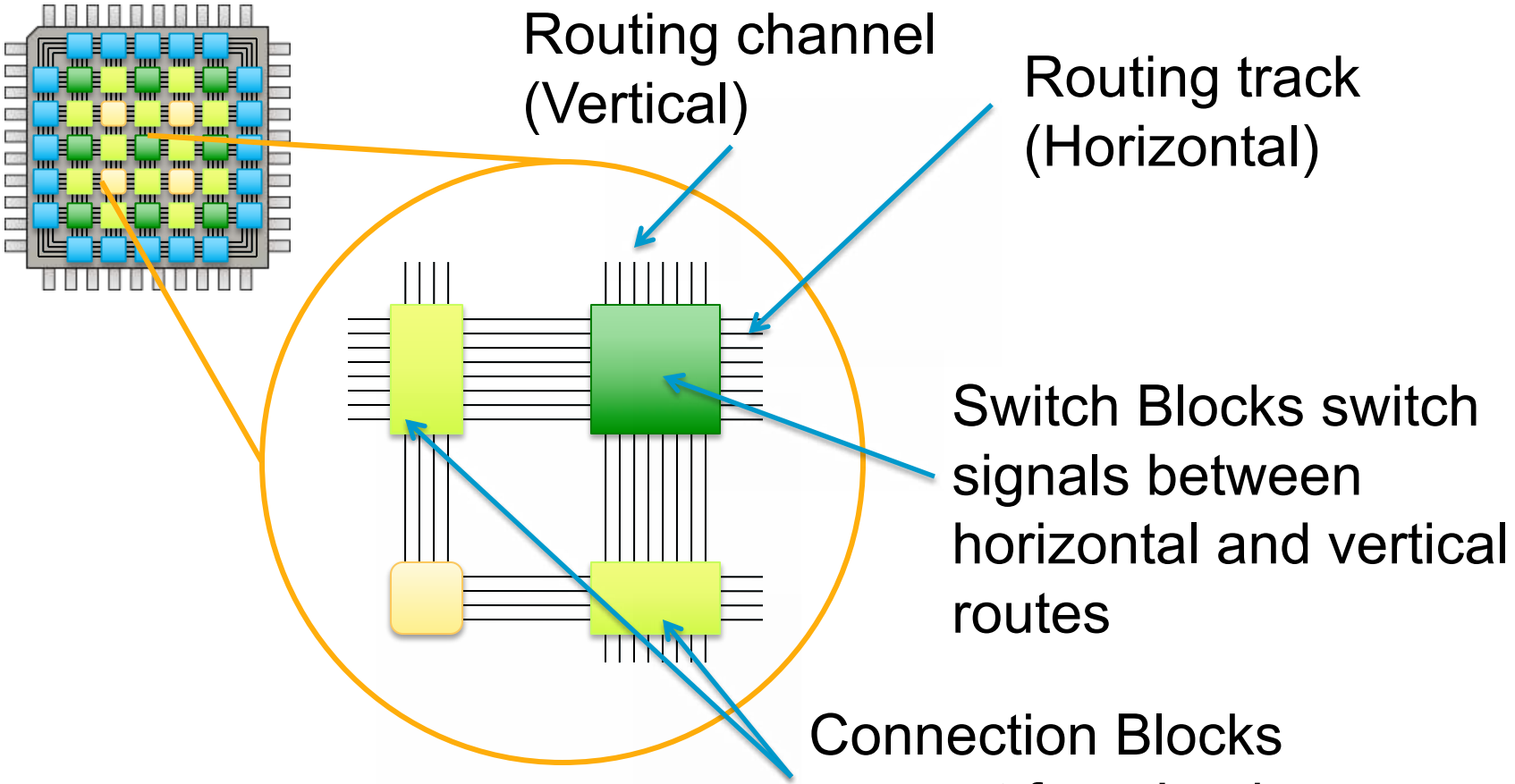
- Harden on-chip configurable memory
  - Xilinx: BRAM
- True dual-port memory
- Configurable:
  - 32k x 1 to 512 x 72 in one 36K block
  - Simple dual-port and true dual-port configurations
  - Built-in FIFO logic
  - 64-bit error correction coding per 36K block
  - Adjacent blocks combine to 64K x 1 without extra logic

DSP Slice

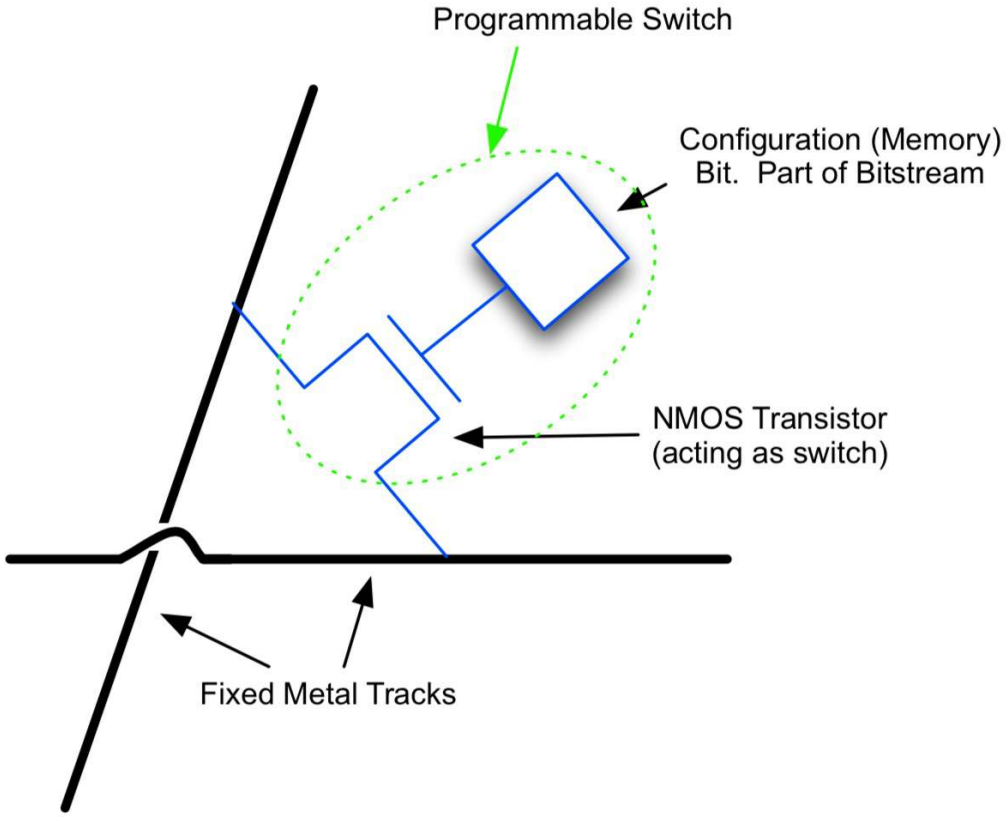
- All 7 series FPGAs share the same DSP slice
  - 25x18 multiplier
  - 25-bit pre-adder
  - Flexible pipeline
  - Cascade in and out
  - Carry in and out
  - 96-bit MACC
  - SIMD support
  - 48-bit ALU
  - Pattern detect
  - 17-bit shifter
  - Dynamic operation (cycle by cycle)



FPGA Routing

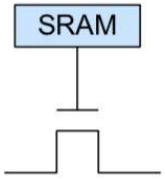


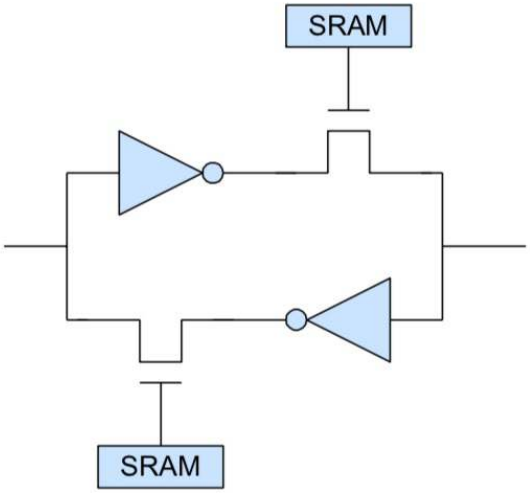
Programmable Switches

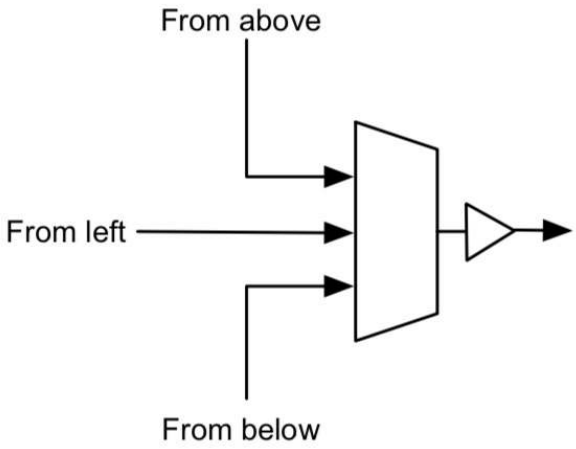


- Simple connection:
  - Connect 2 wires by a single transistor
- 1 configuration bit per connection

Programmable Switches







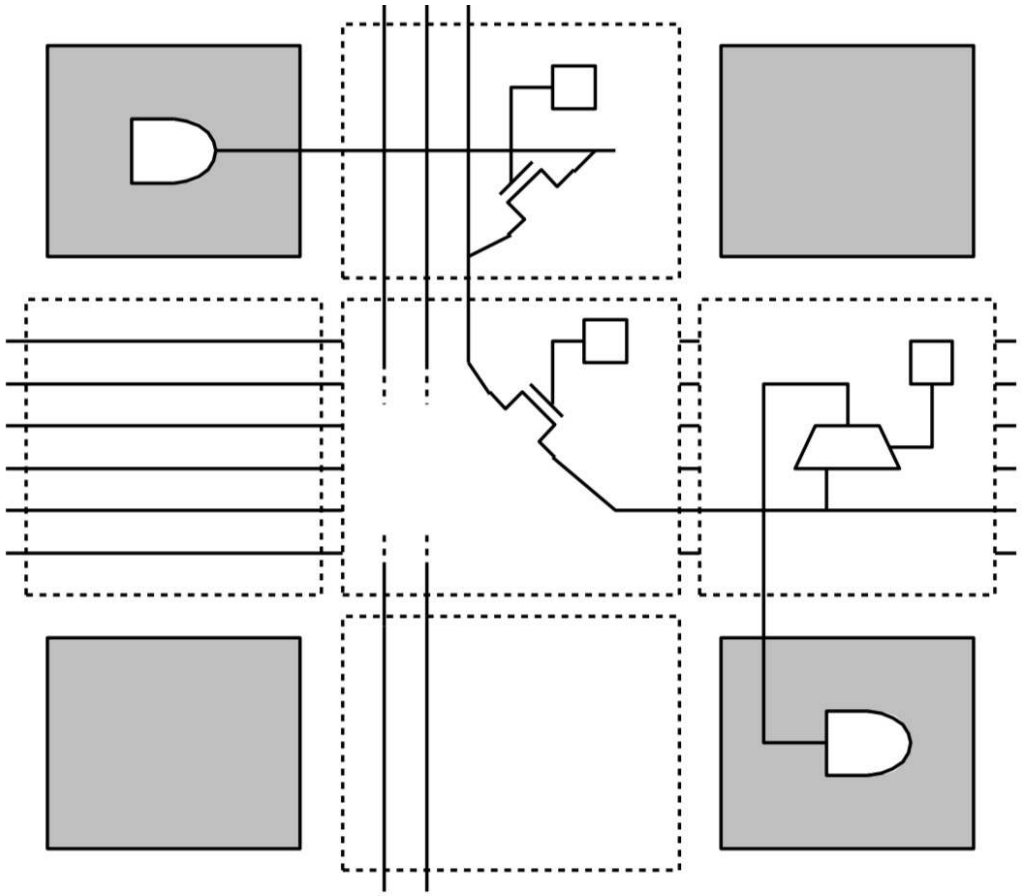
Unbuffered Connection

Buffered Connection

Buffered Uni-Directional Connection

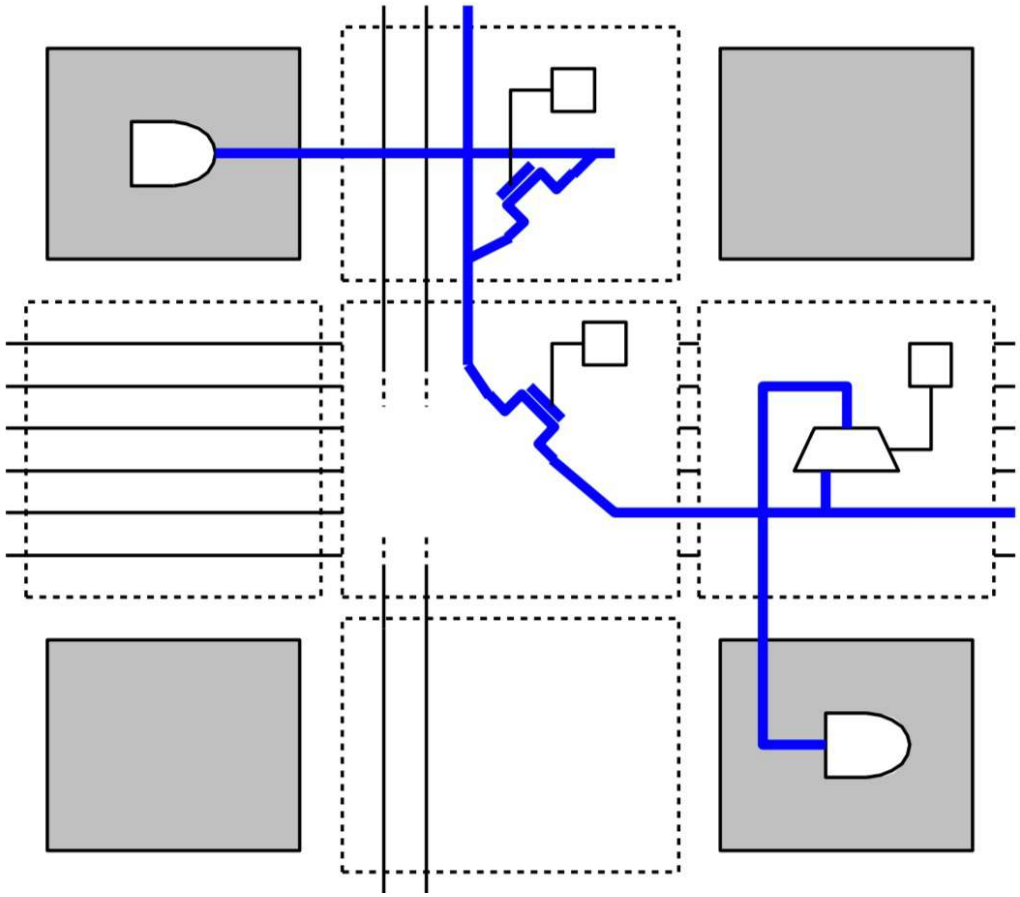
- Common to have **buffered**, **unidirectional** connection in modern FPGAs

Setting up a route



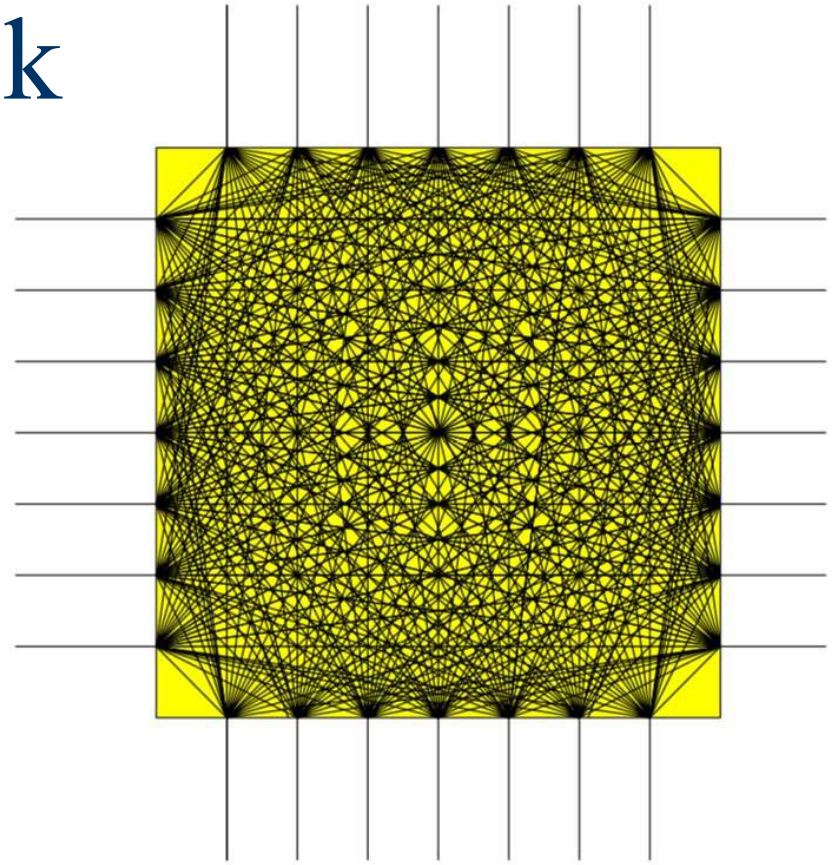
- Make connection by setting switches a along route between logic blocks

Setting up a route



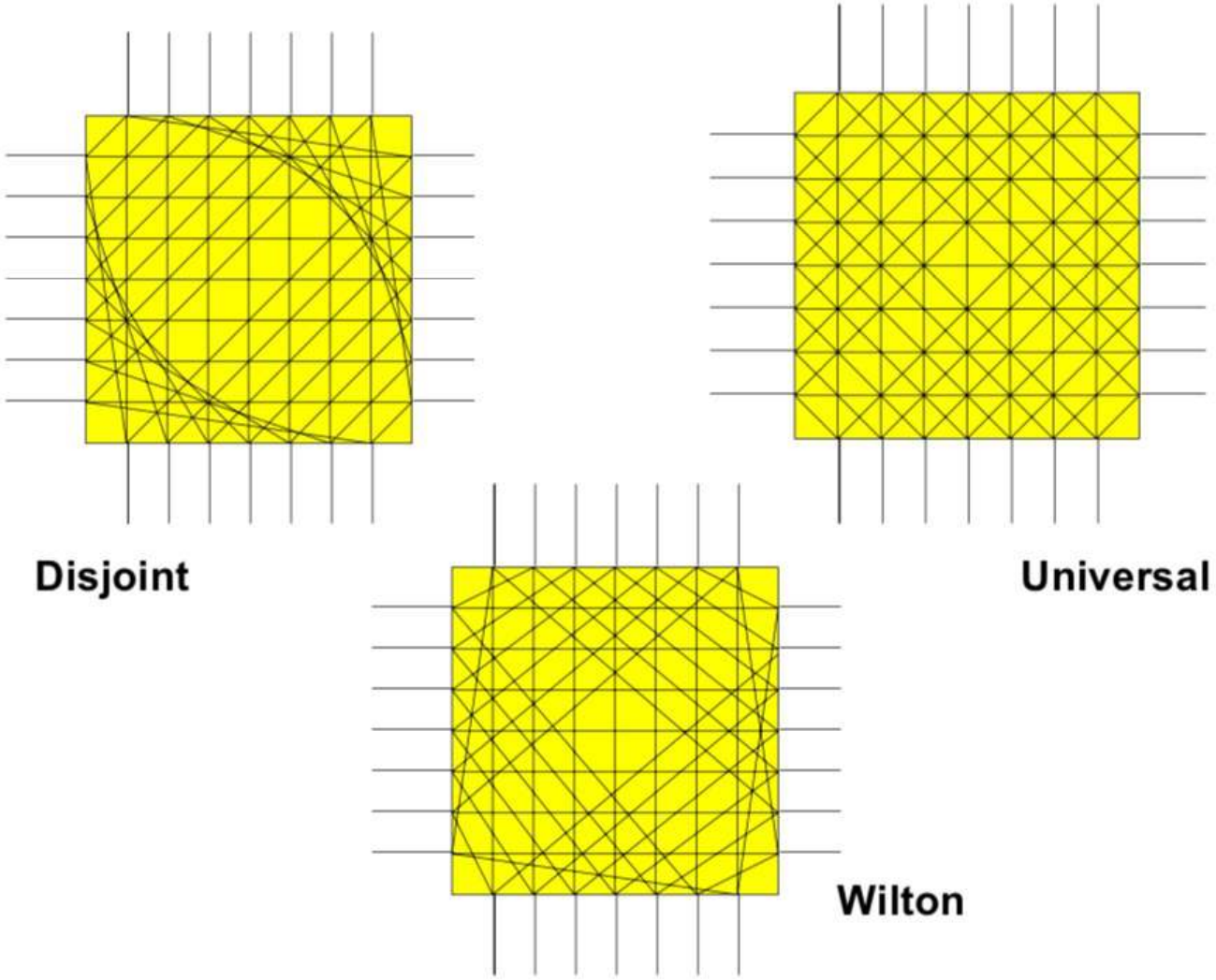
- Make connection by setting switches a along route between logic blocks

Switch Block

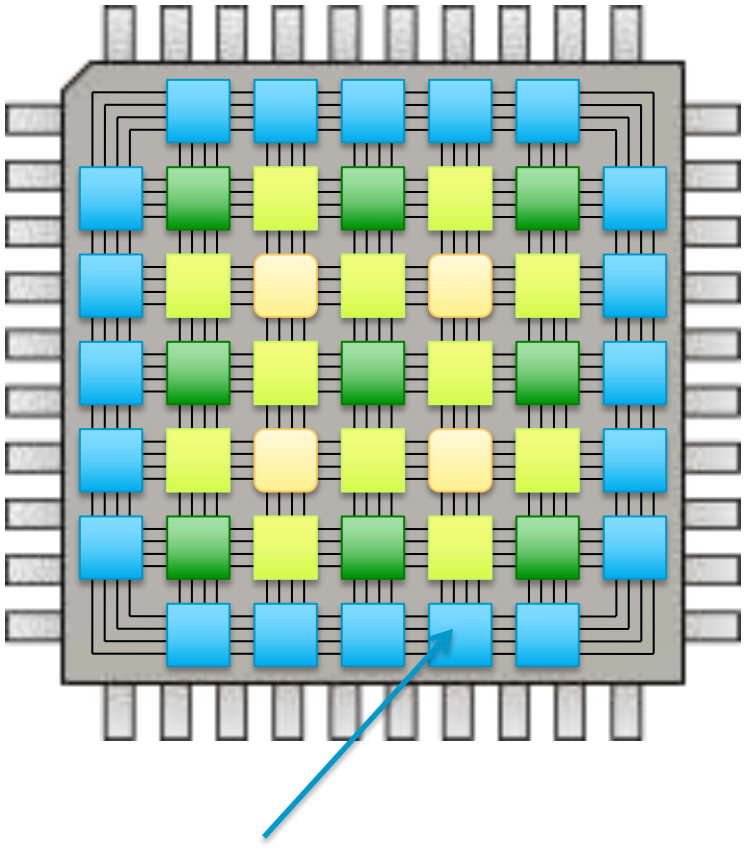


- Goal of a switch block is to connect physical wires.
- Problem: too many possible connections
- Fully connected graph:
  - Too large, too expensive, too much power

Many Topologies



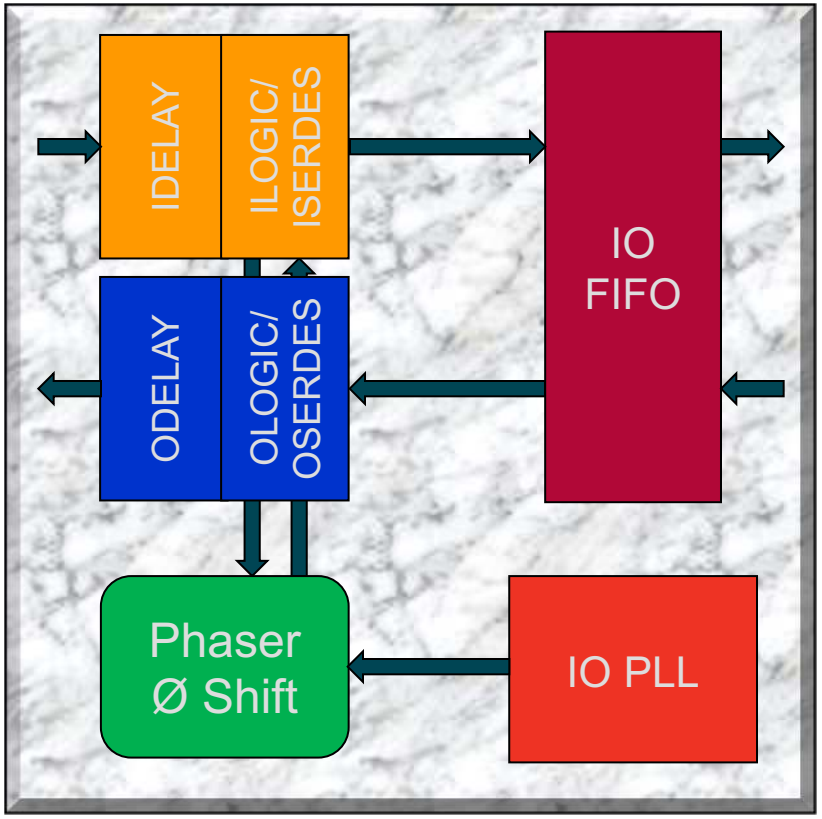
I/O Block



- Configure each pin to be input, output, or input/output
- Different voltage standard
  - LVTTL, PCI, etc
- Single-end signal ➔ diff signals
- Internal routing make sure that any internal signal can be routed to any external pin
  - Allow PCB fixup

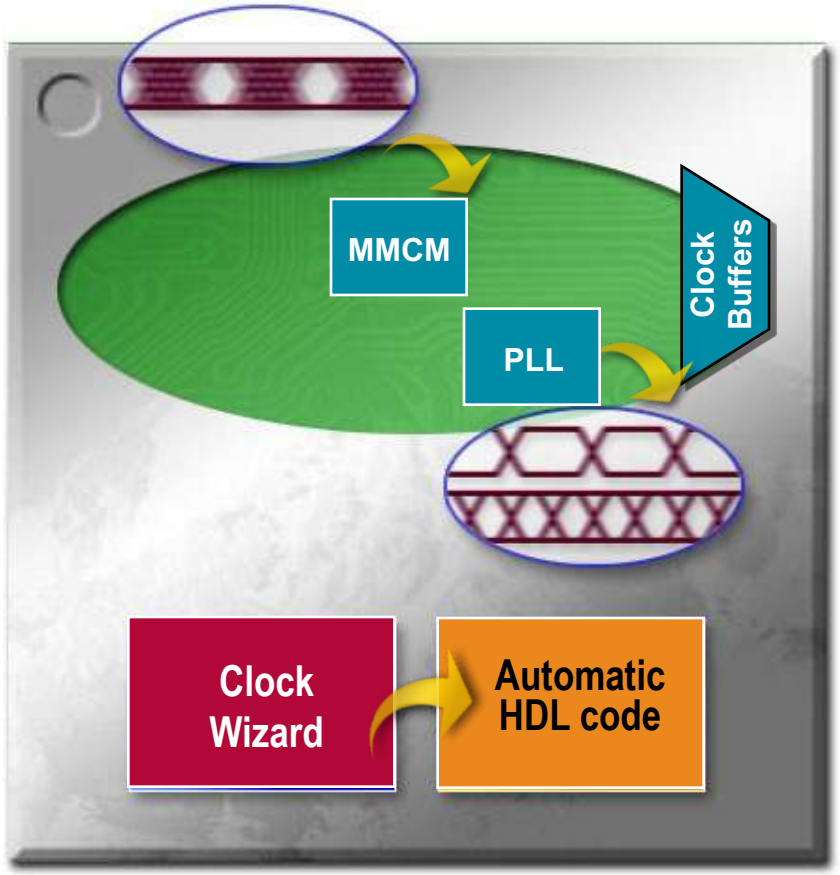
Input/Output Blocks

- Two distinct I/O types
  - High range: Supports standards up to 3.3V
  - High performance: Higher performance with more I/O delay capability
    - Supports I/O standards up to 1.8V
- Extension of logic layer functionality
  - Wider input/output SERDES
  - Addition of independent ODELAY
- New hardware blocks to address highest I/O performance
  - Phaser, IO FIFO, IO PLL

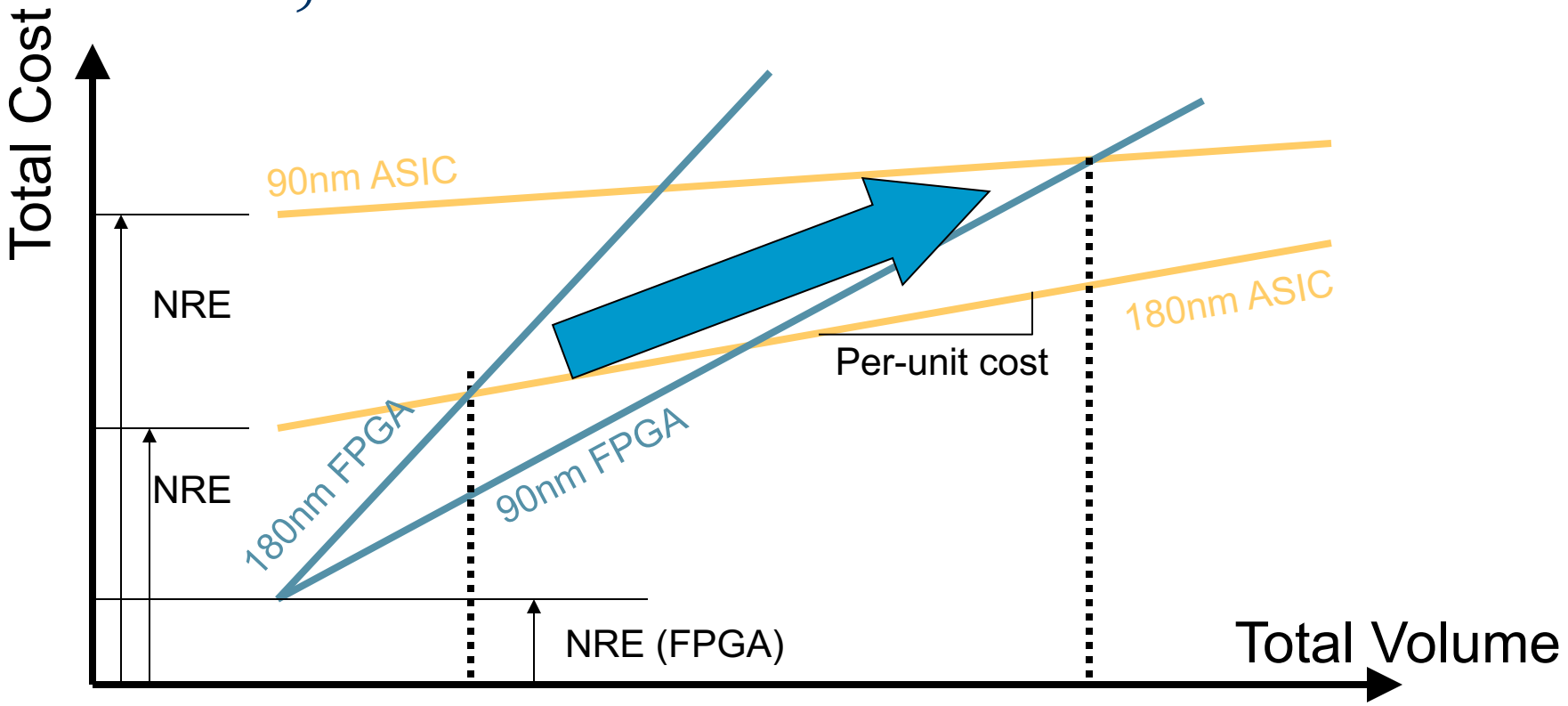


Clocking Resources

- Based on the established Virtex-6 FPGA clocking structure
  - All 7 series FPGAs use the same unified architecture
- Low-skew clock distribution
  - Combination of paths for driving clock signals to and from different locations
- Clock buffers
  - High fanout buffers for connecting clock signals to the various routing resources
- Clock regions
  - Device divided into clock regions with dedicated resources
- Clock management tile (CMT)
  - One MMCM and one PLL per CMT
  - Up to 24 CMTs per device



FPGA, ASIC Crossover



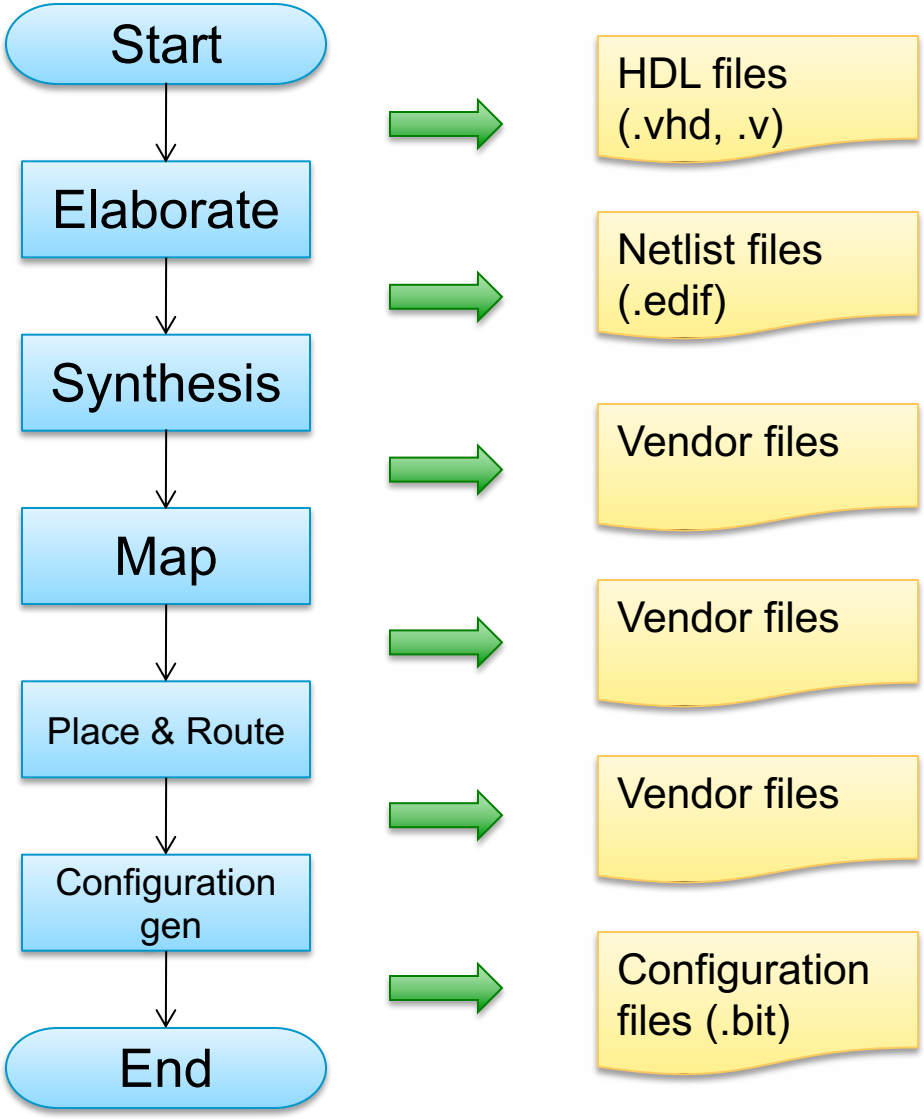
- Spartan-3E
  - 90nm
  - ~300 MHz
- Crossover point at ~250,000 units
- 1.2M system gates for under US$9.00 (XC3S1200E*)

FPGA Design Flow

Traditional Design Flow

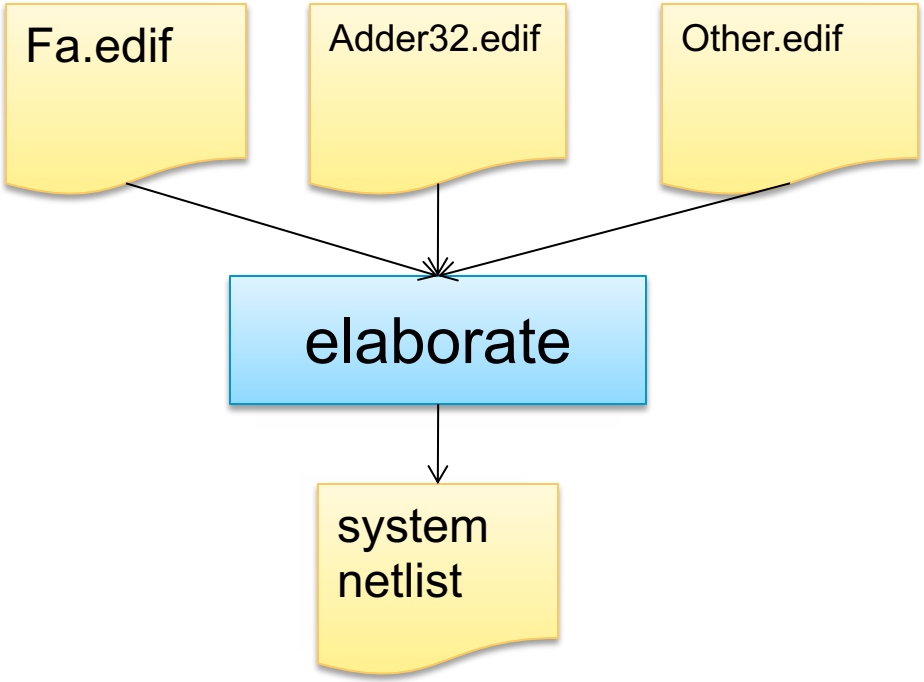
- With its root as an ASIC replacement, traditional FPGA design flow resembles closely that of an ASIC design.
- Start with schematics or high level hardware description languages, the design flow translate the input designs into low level entities
- FPGA's programmability make the low-level flow slightly different

FPGA design flow overview



Elaborate

- Assemble netlists from different sources
- Basic syntax check
- Basic inference
  - First step of real synthesis



Synthesis (Behavioral HDL)

- e.g.

```
if (d > e) then
    c <= a * b
else
    c <= a * e
end if
```

- Extract hardware structures from the code
- E.g. muxes, adders, multipliers, RAM, Flip-flops, etc
- Requires careful “writing styles” so that the synthesis tool can successfully extract the “correct” structure
- Many strange errors can be attributed to incorrect synthesis results.
- Synthesis tools *may* optimize for the target device

Synthesis (Structural HDL)

```
FA0: FULLADDER port map (a(0), b(0), Cin, sum(0), c(1));
FA1: FULLADDER port map (a(1), b(1), C(1), sum(1), c(2));
FA2: FULLADDER port map (a(2), b(2), C(2), sum(2), c(3));
FA3: FULLADDER port map (a(3), b(3), C(3), sum(3), c(4));
Cout <= c(4);
```

- Hardware structure is already presented in the code
- Synthesis tools simply translate and assemble the structures into a correct netlist
- Low-level blocks may be instantiated directly
  - But the code will no longer be “portable”

Map

- Convert netlist into LUTs, FFs, on-chip RAM, hardcore multipliers, DSP blocks, etc
- Generate the actual LUT equations
- May reorganize logics combine signals from different parts of the design into 1 LUT

Place and Route

- Place:
  Select the site to where each LUT and FF should locate
- Route:
  Select the optimal route to connect all LUTs and FFs
- Most time consuming
- Iterative process
  - Placement affects Route, and vice versa

Configuration Generation

- Generate final configuration for the device
- Download to the FPGA during run time
  - E.g. through JTAG