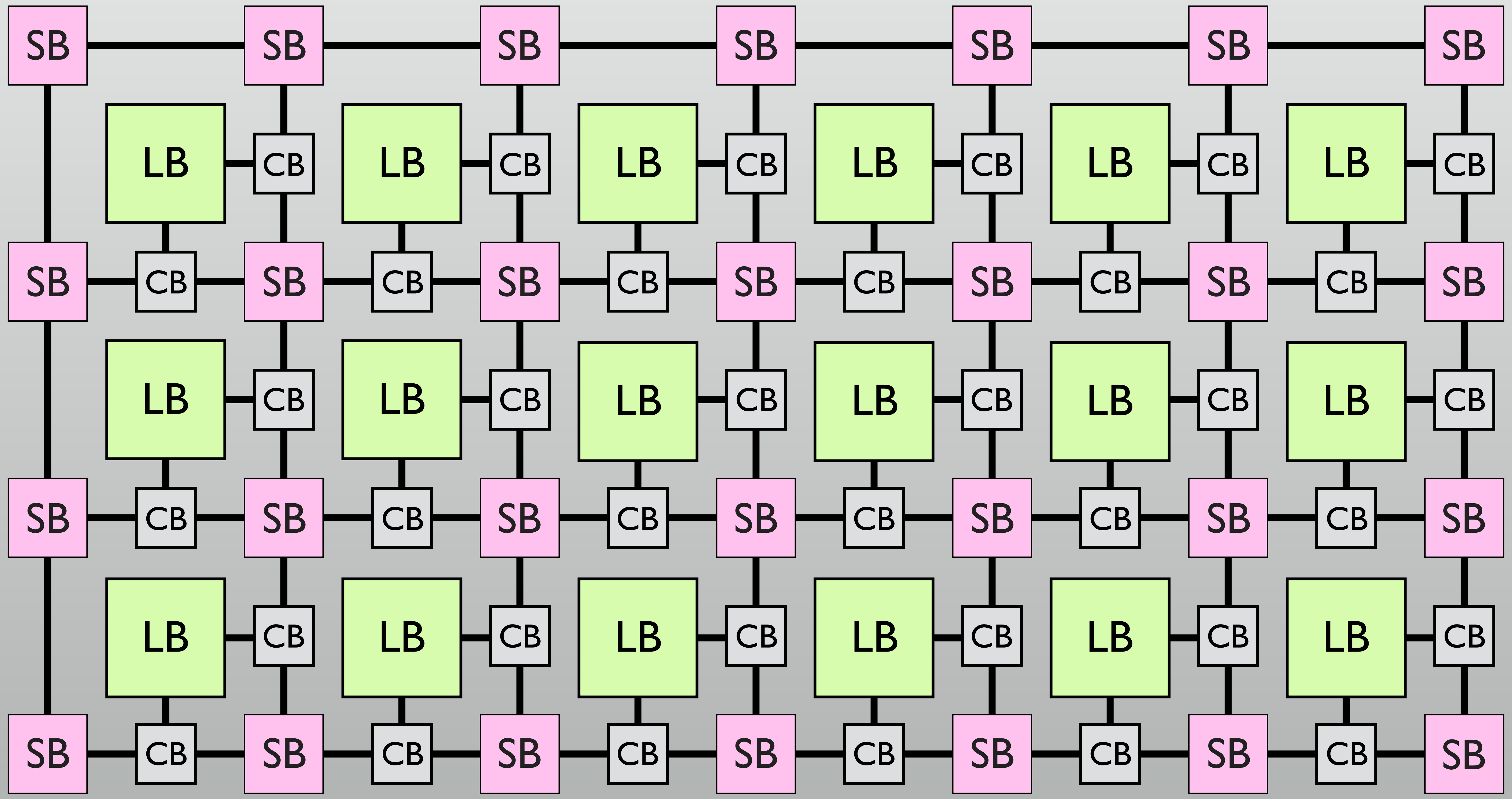
Reconfigurable Architecture (12)

FPGA Architecture

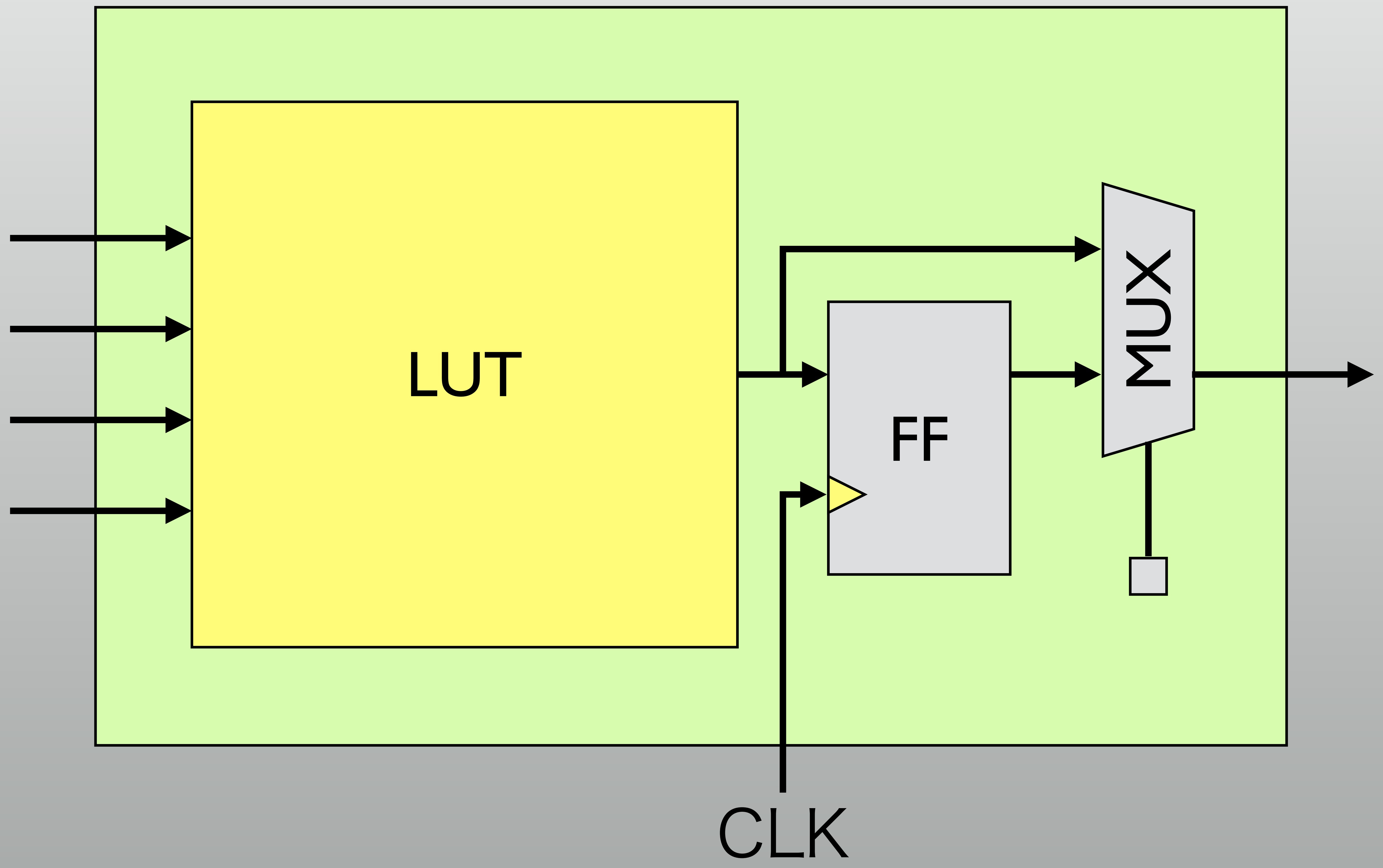
Basic components

* LB: Logic Block
* CB: Connection Block
* SB: Switch Block
* Interconnect wires


SB
LB
CB

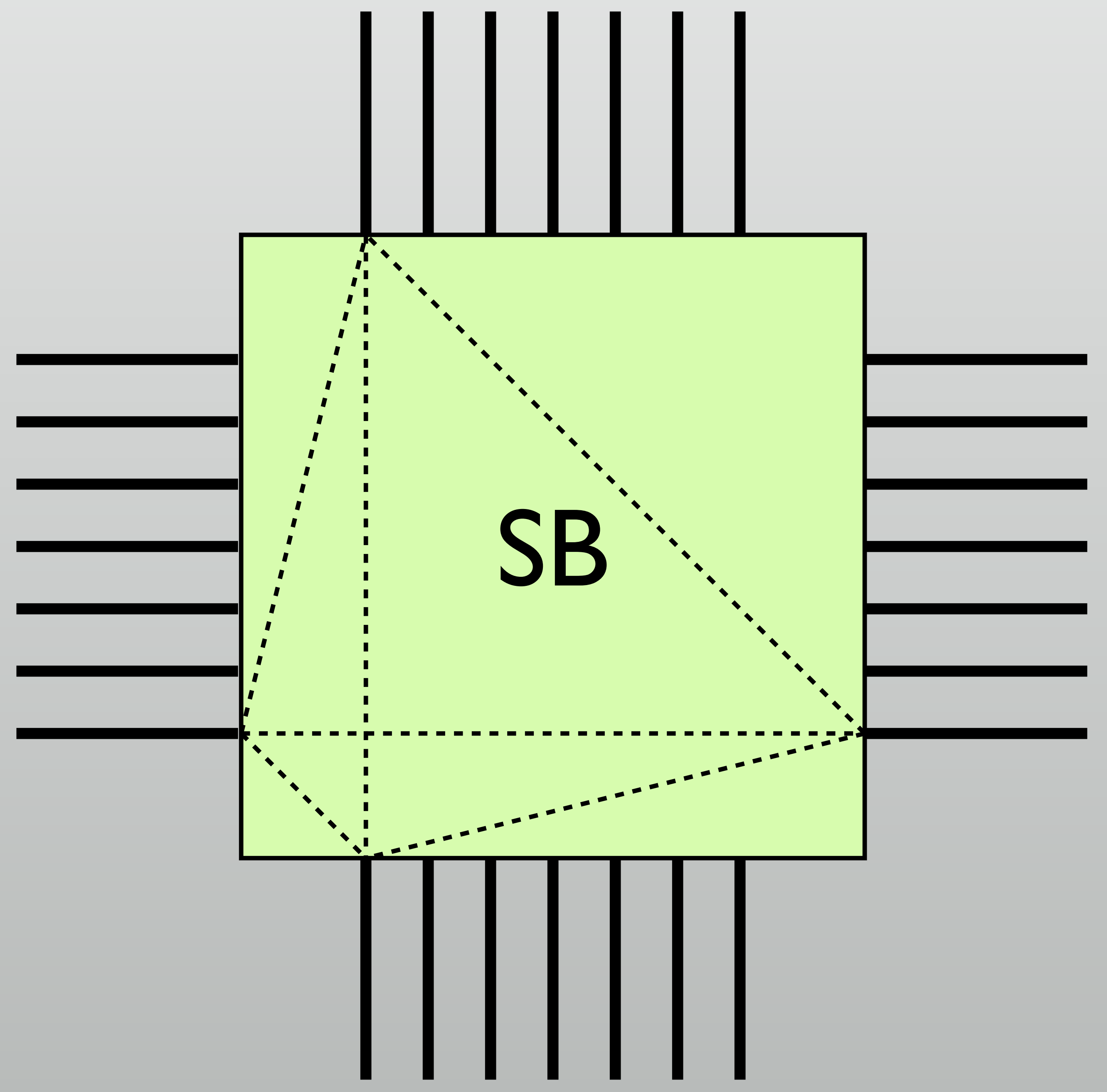
Logic Block

- The very basic structure: LUT + FF + MUX



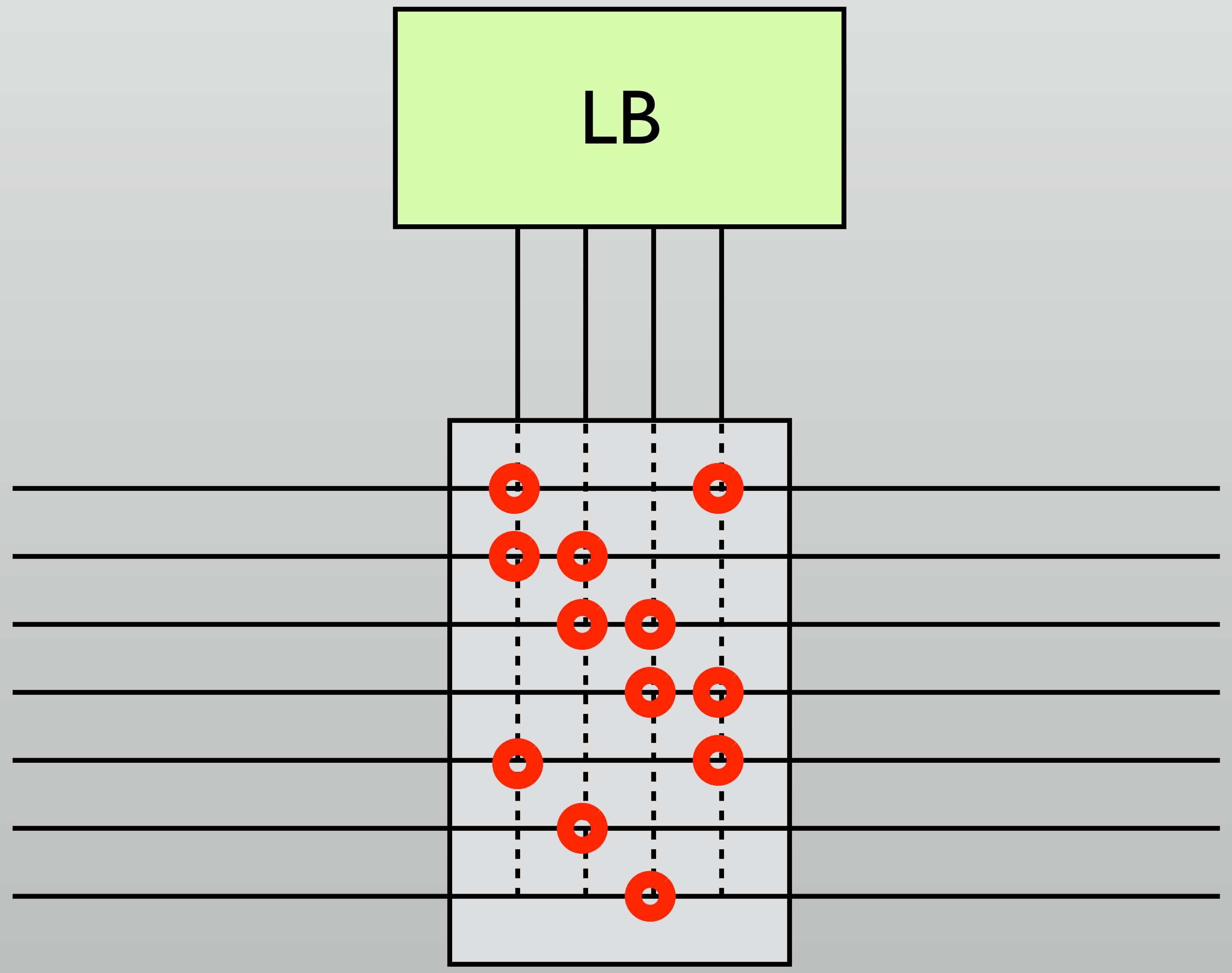
Interconnect and SB

- Multiple horizontal & vertical wires
    - Not fully-connected: destination wires are always limited to reduce # of switches in SB



LB, CB and wires

* Connect wires from logic block pins to interconnect tracks
  * Is not fully connected, because of the same reason

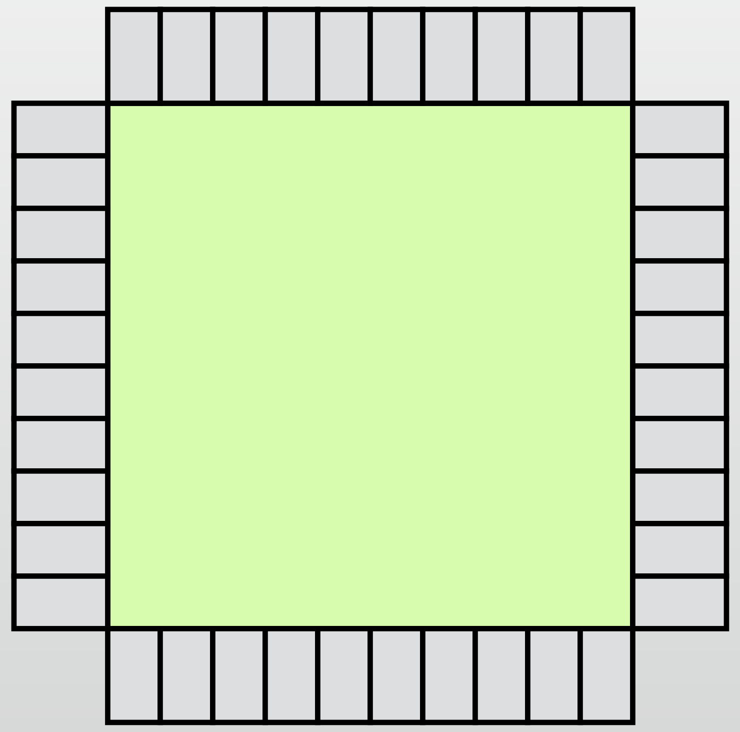


Other components

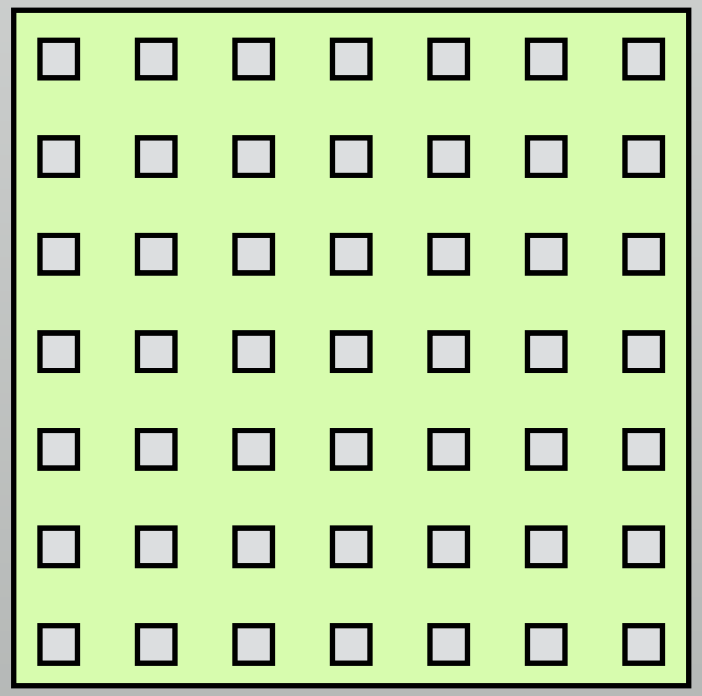
* IOB (I/O Blocks)
* Clock buffers and interconnects

I/O pads

- Wire bonding
  - I/O pads **around** the chip
  - Long wires to the pads
- Flip-chip
  - Distributed I/O pads
  - Shorter logic wires and better power delivery



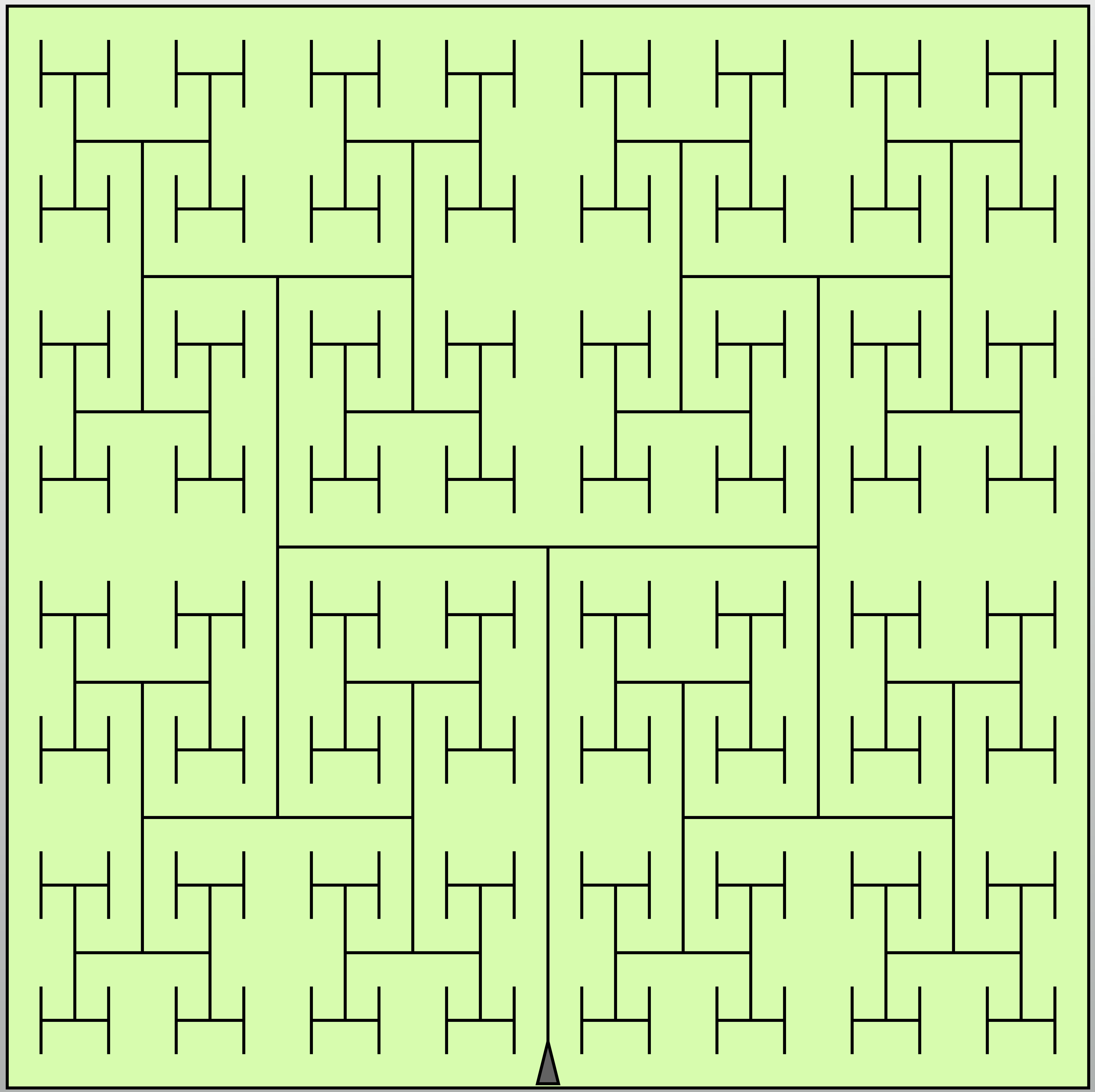
従来型



フリップチップ

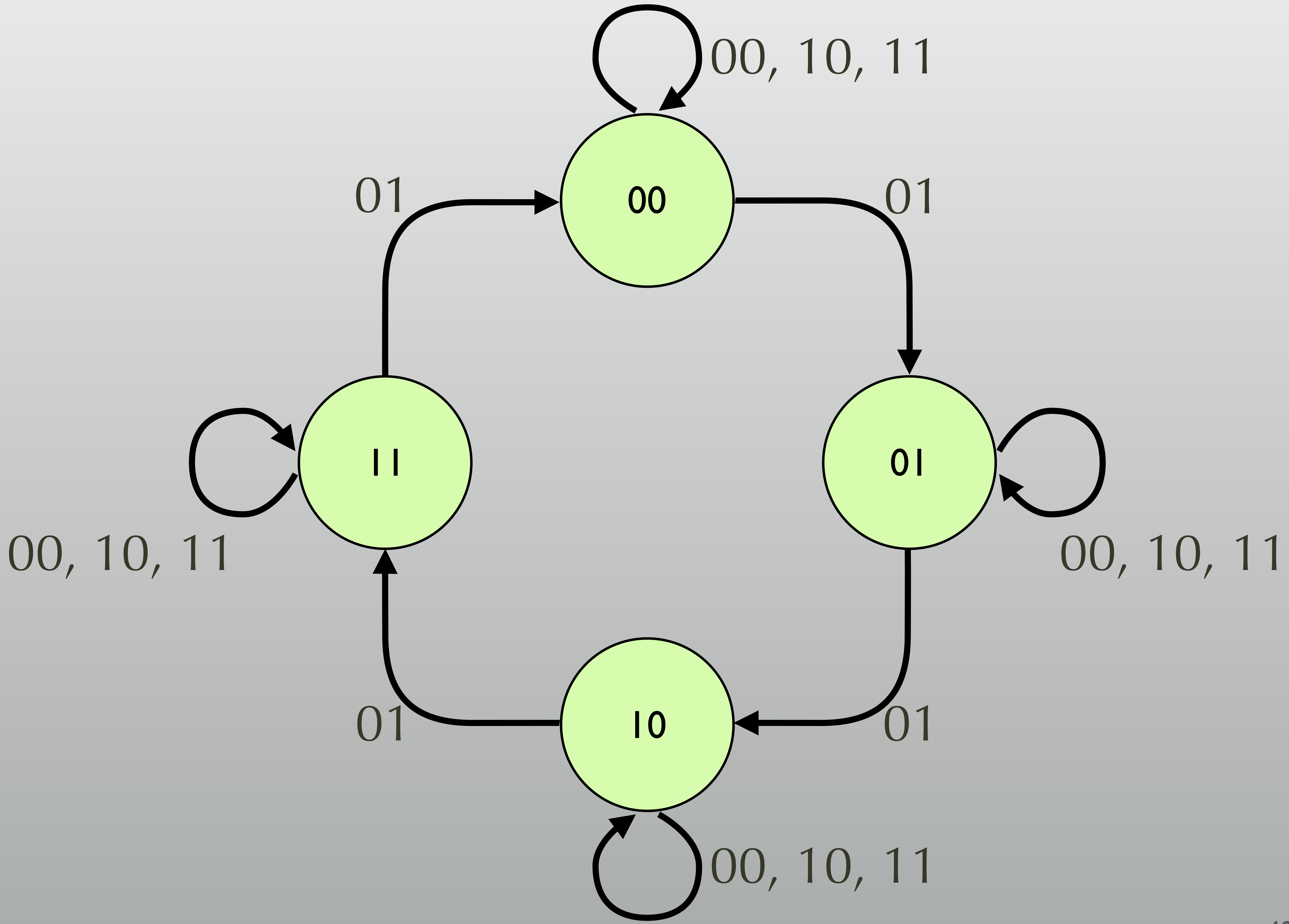
Clock tree

- Wires of exactly same length, from the clock inputs to all FFs on the FPGA
  - Dedicated wires for clock, not the wires for logic

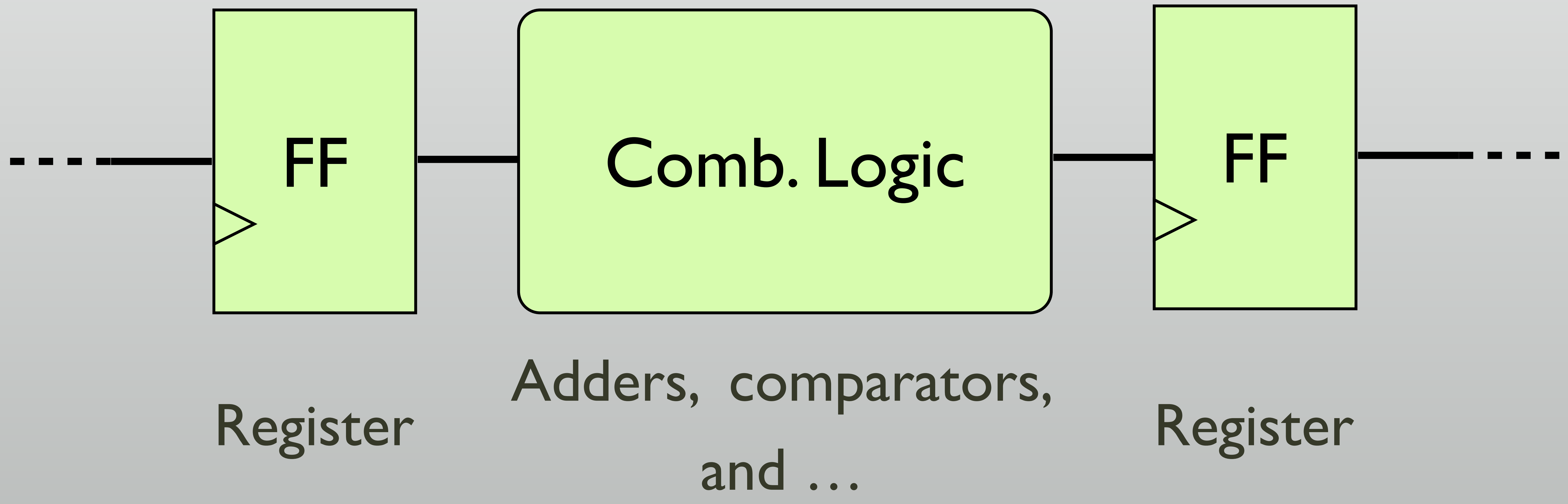


Logic circuit in textbooks

- Combinational logic and FFs
  - State transition diagrams



Logic circuits in action



Performance depends on:

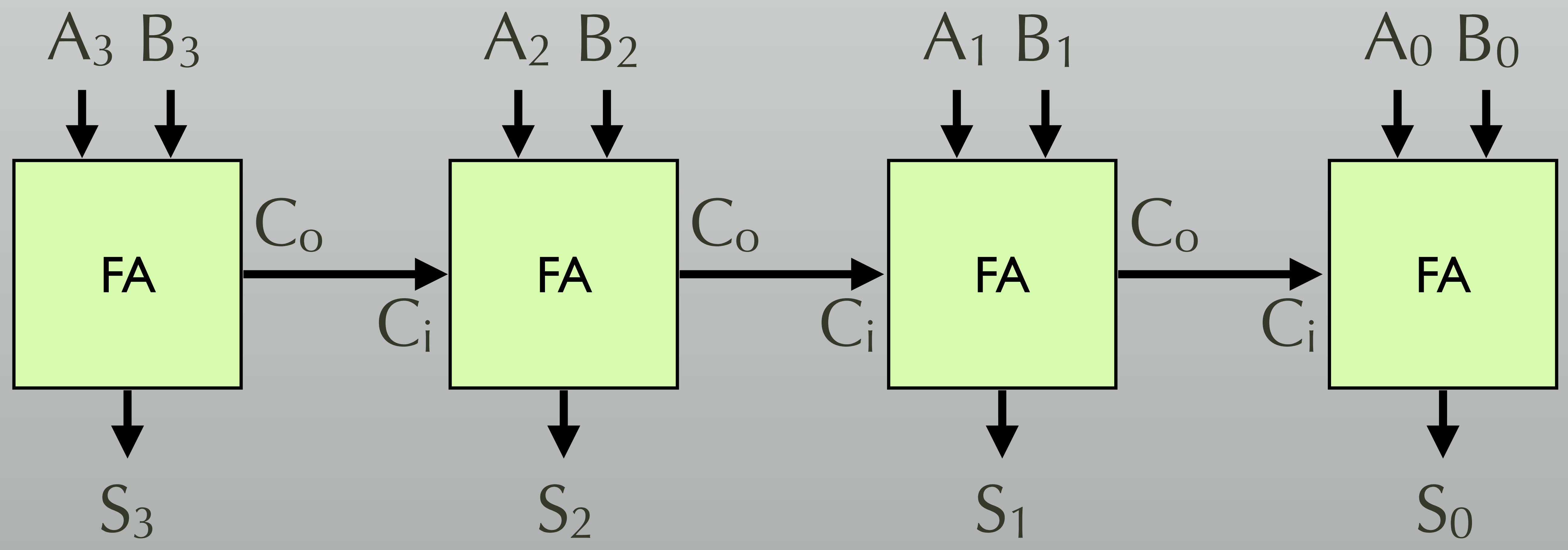
- Number of register (FF) stages
  - = # of clock cycles to get the result
- Combinational delay = logic delay + wire delay
  - Clock frequency depends on the worst combinational delay

Make common cases faster

* Effective way for basic building blocks, such as:
    * Adders
    * Shift registers
    * Memory elements

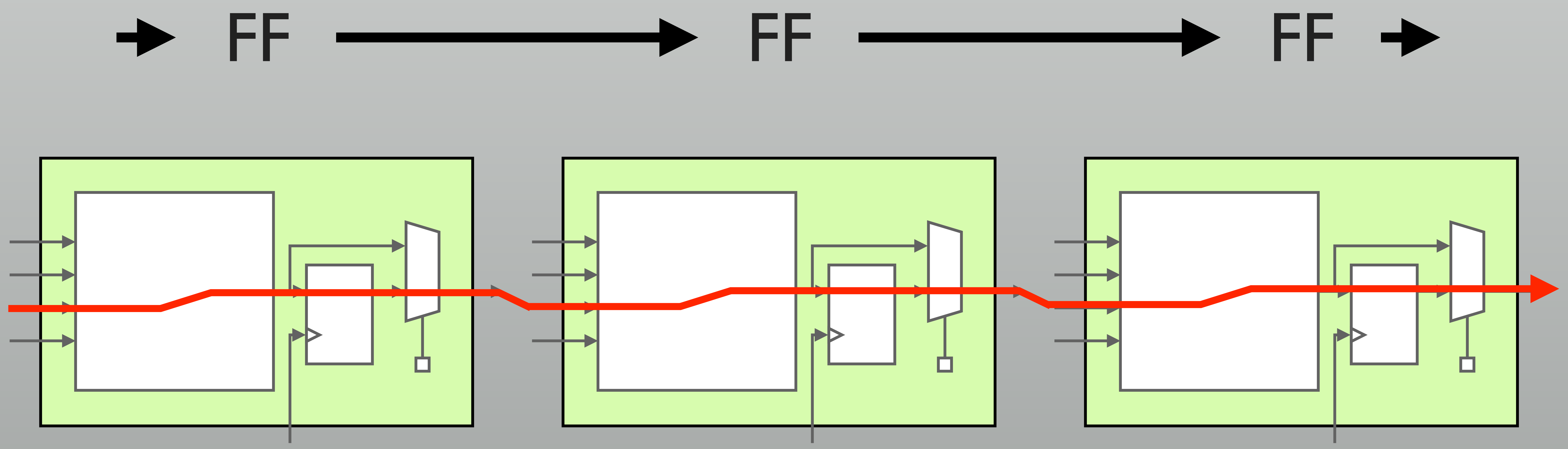
Adders

- Daisy-chained full adders
  - A full adder can be implemented with an LUT
  - Problem is the carry chain delay



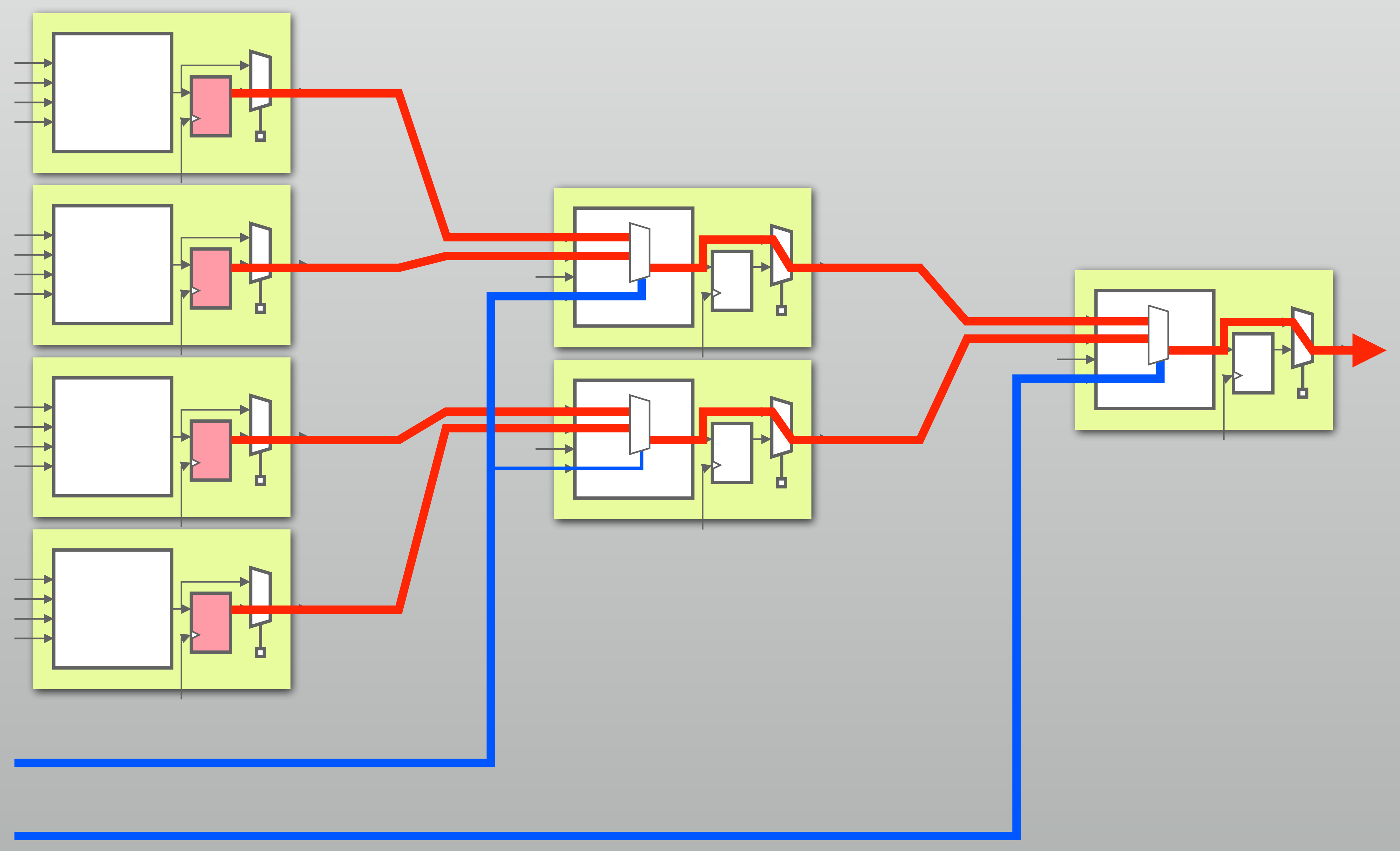
Shift registers

- Daisy-chain of LBs
  - Delay is not problem, because there are (many) registers
  - LUTs are just “wires”: not very effective in resource usage



Memory element

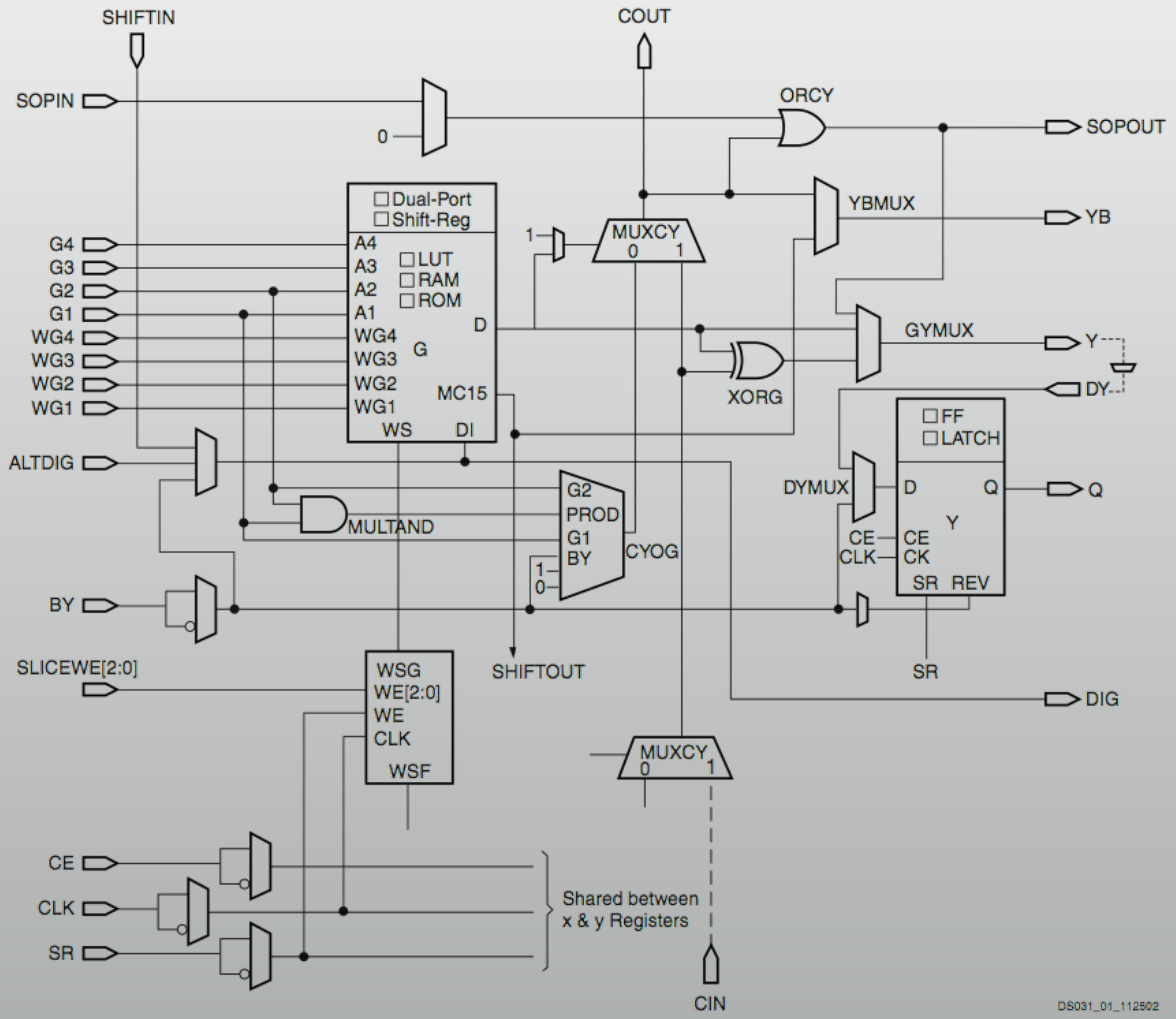
- LUTs are just to form address decoder…



Ineffective?

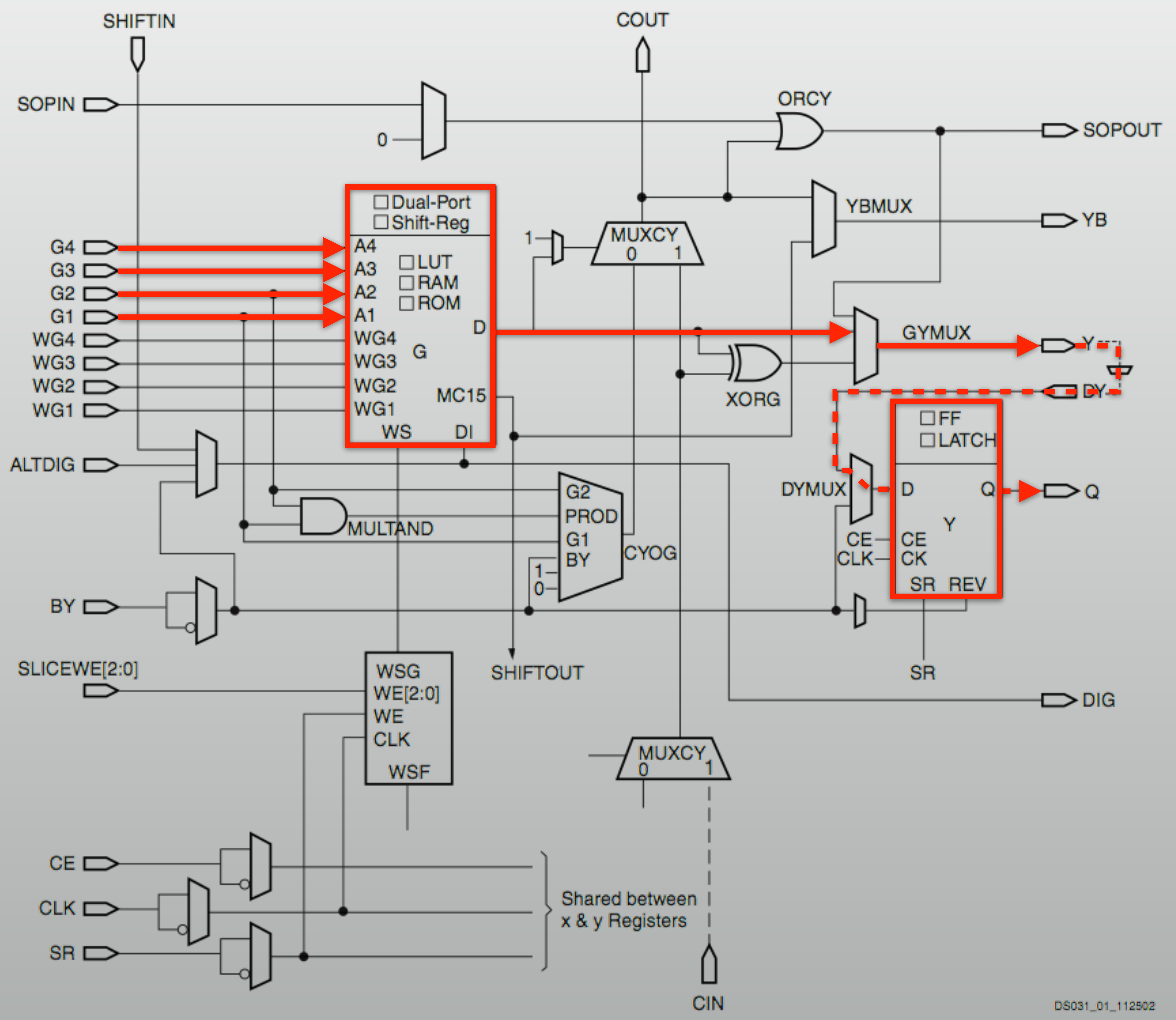
- LUTs are SRAM cell
  - A 4-LUT has 16bits, a 6-LUT in modern FPGAs has 64bits
  - Of course, there's built-in address decoder
- LUTs are ready to be RAM cells, and maybe for shift registers and FIFOs

Virtex-II LB organization



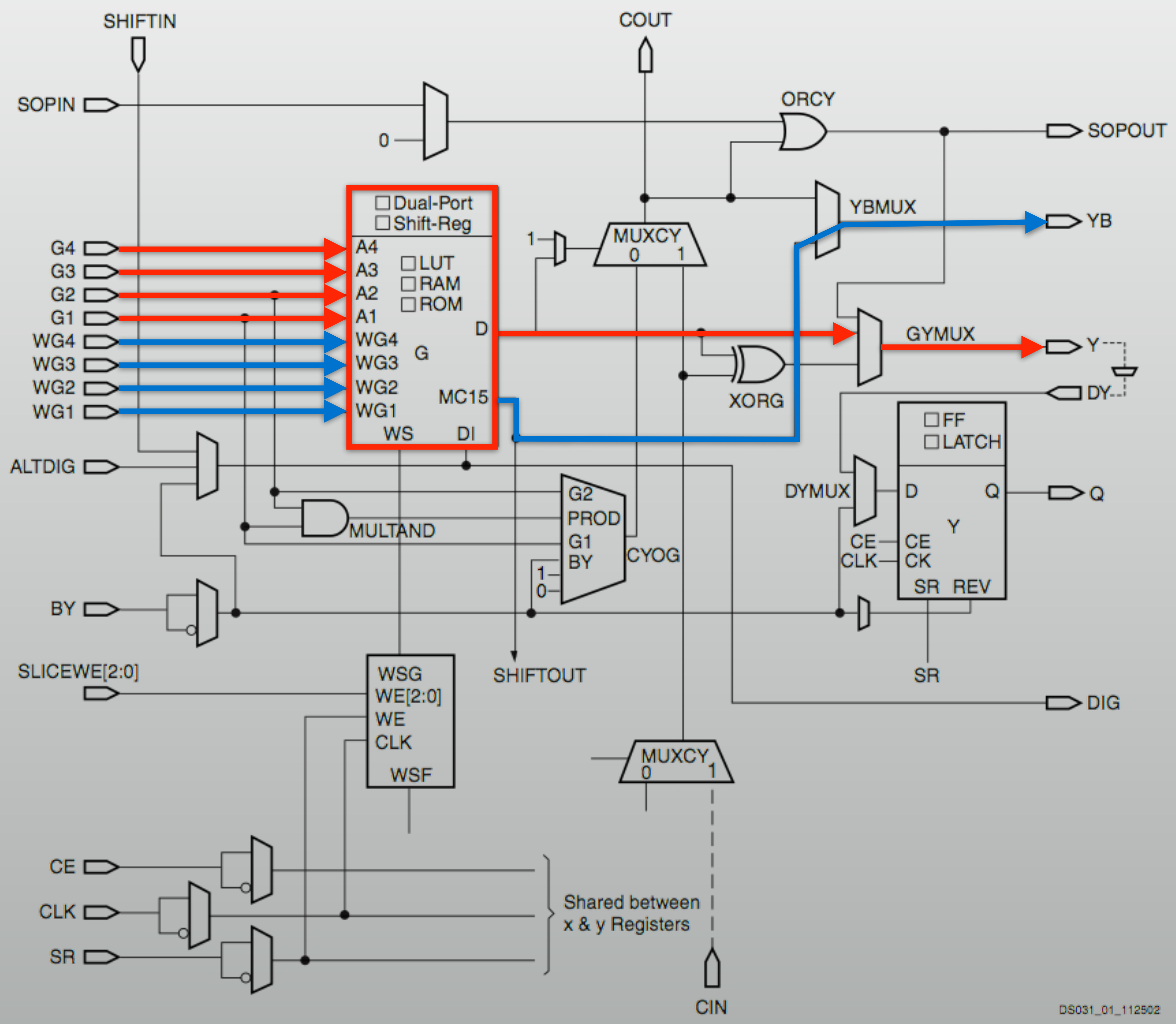
Xilinx Virtex-II FPGA Datasheet (DS031) より

As a basic LB…



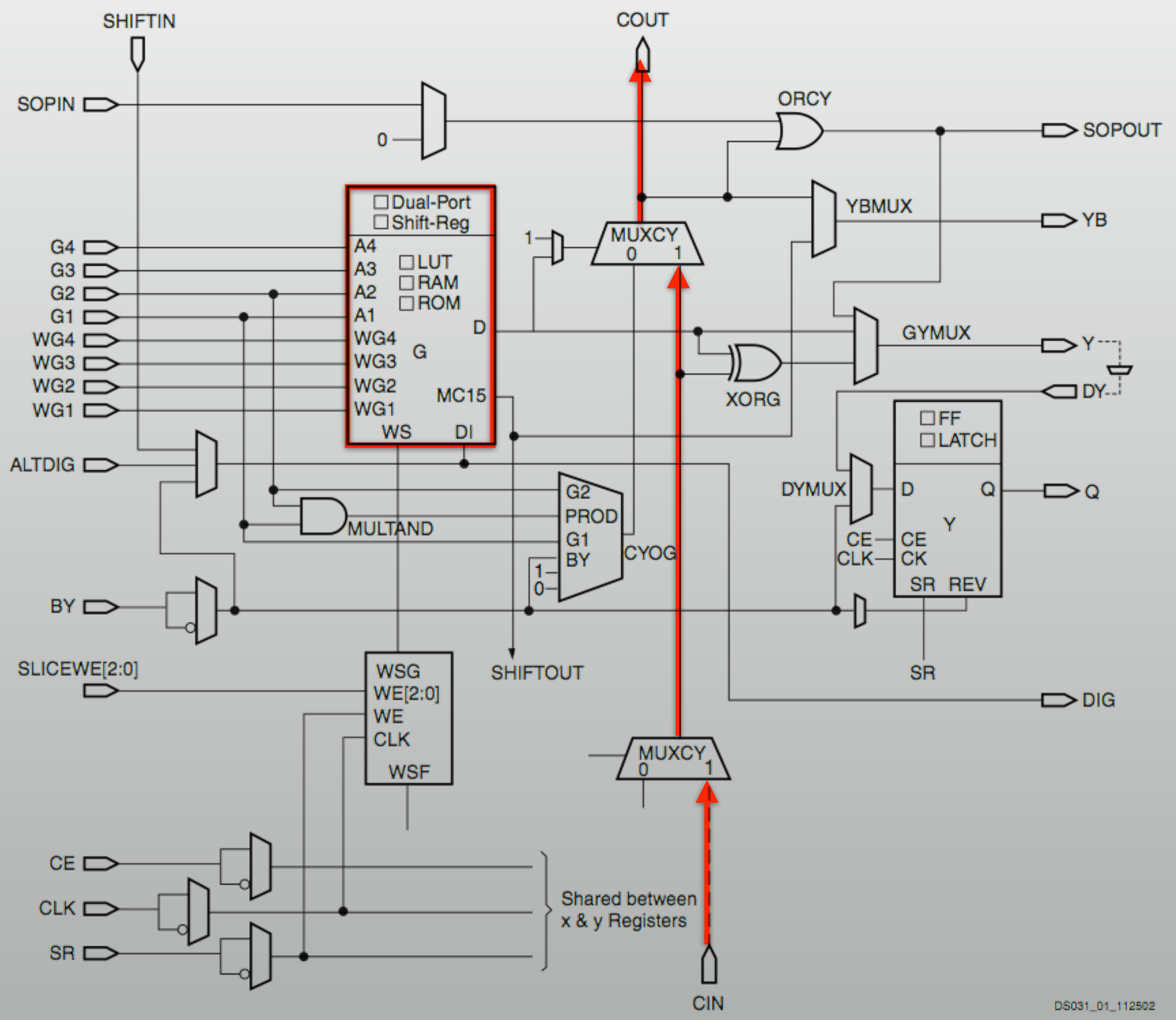
Xilinx Virtex-II FPGA Datasheet (DS031) より

As a dual-port memory cell



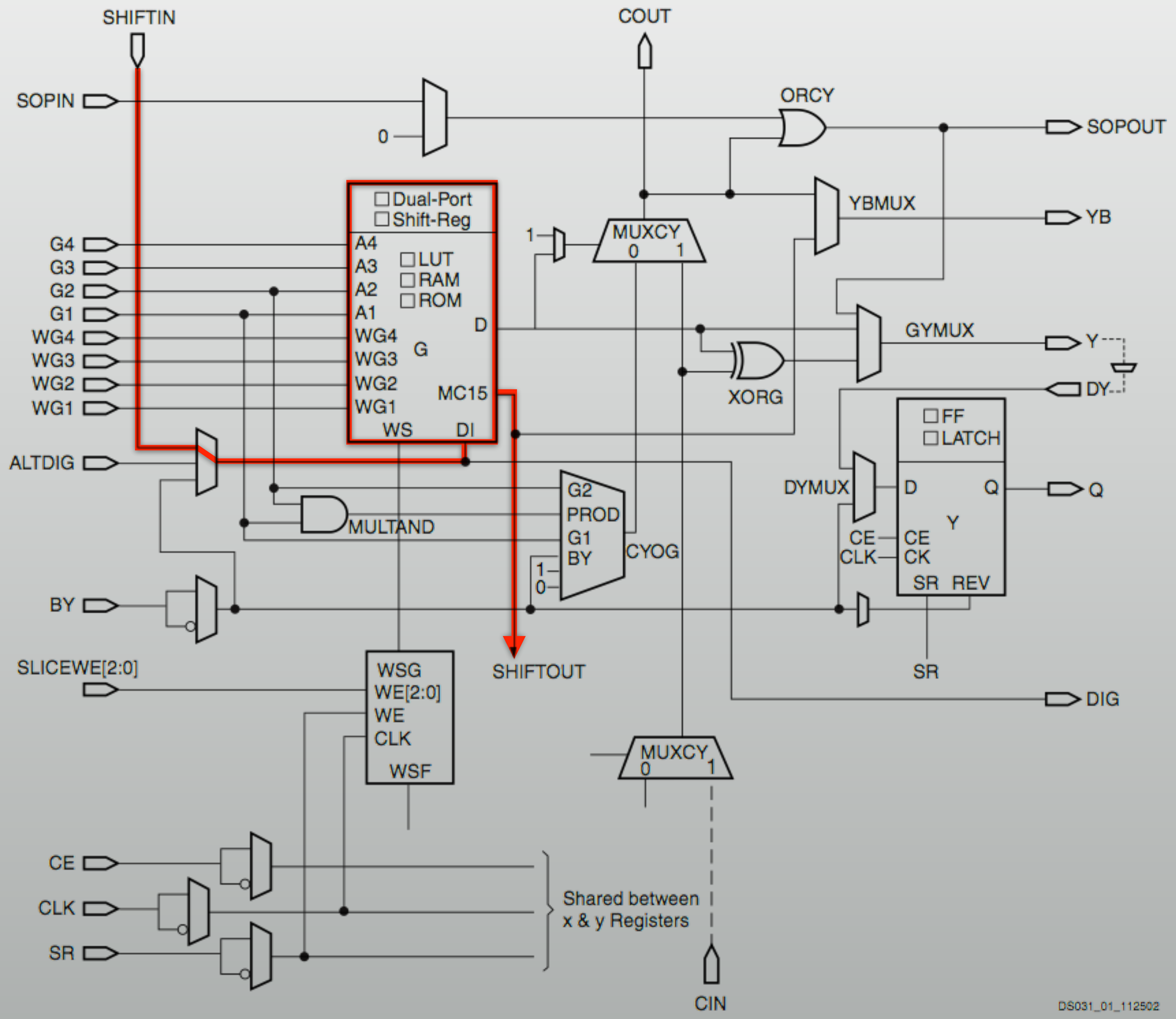
Xilinx Virtex-II FPGA Datasheet (DS031) より

Carry chain for adder



Xilinx Virtex-II FPGA Datasheet (DS031) より

Shift register chain



Xilinx Virtex-II FPGA Datasheet (DS031) より

LB+CB+SB=FPGA ?

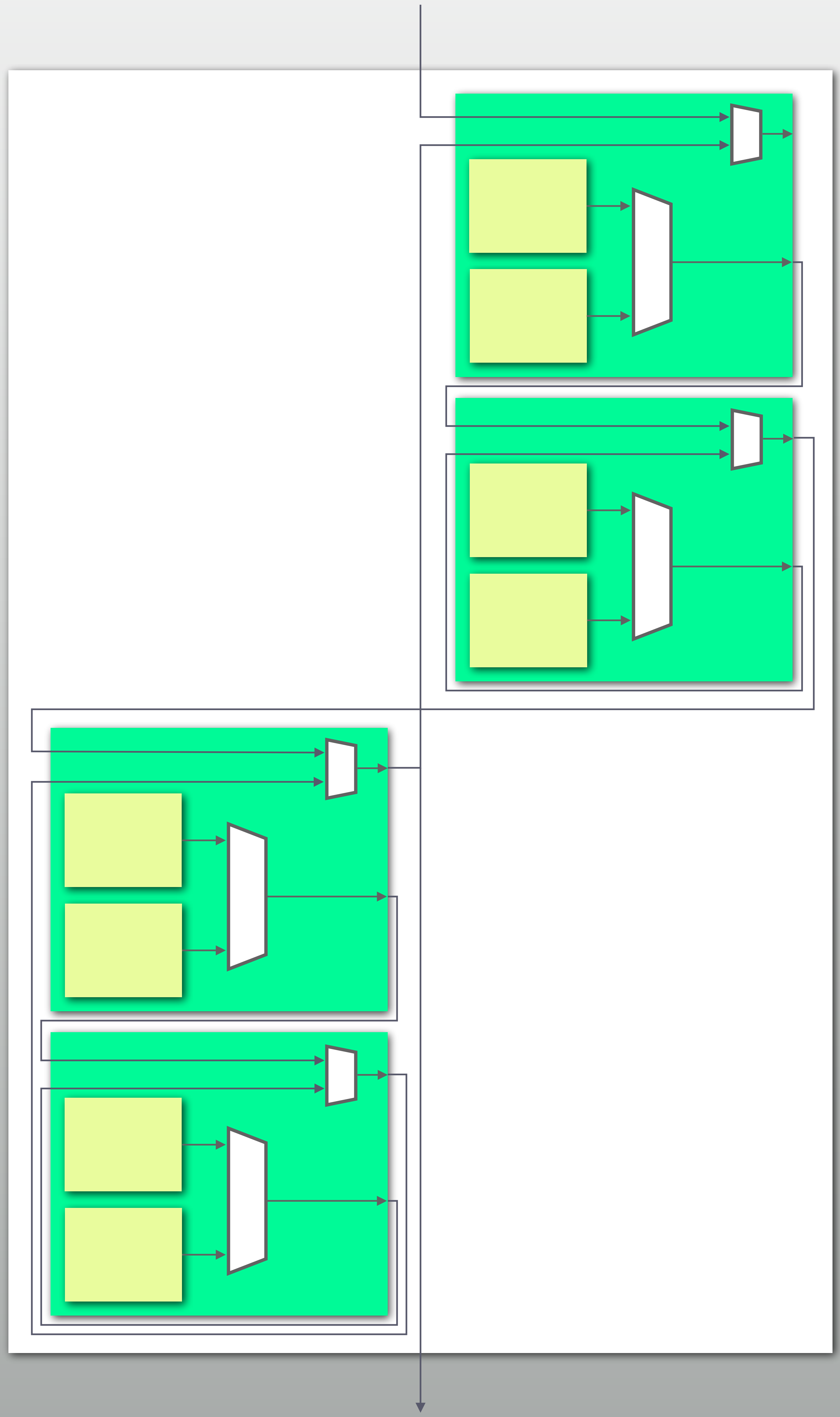
- Indeed, it’s not very effective
  - Go interconnection after every (LUT+FF+MUX) set
  - CB delay always follows to LB delay, is too slow
    - Sophisticated CLB (configurable logic block) provides shorter delay

Flexibility and Performance

- CBs are not always necessary:
  - Making **clusters** of several LBs (LUT+FF+MUX sets)
  - Within the cluster, connections between LB without CB is possible
  - Of course, the LBs has connections with CBs, regardless of the cluster

Clustering

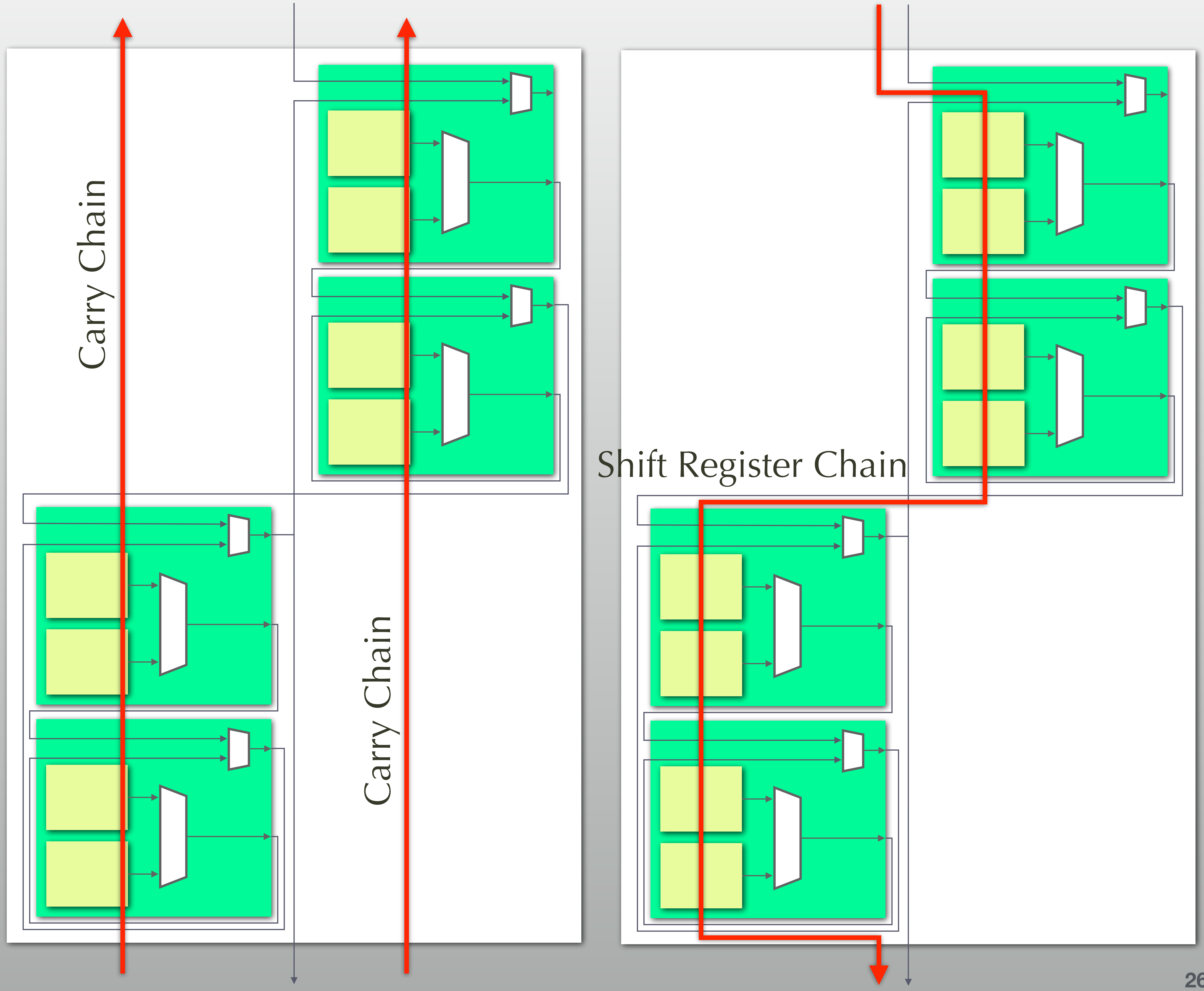
- Local wires have simple organization
- Carry and shift register chains + some local logic wires



Clustering

* Local wires have simple organization

* Carry and shift register chains + some local logic wires



Interconnect architecture

- Very uniform structure, with wire segments of all same length is not good:
  - Connecting more LBs in lower latency is important
  - Use of some wire segments with different length helps
    - Same to the trains: local trains, rapid trains, express trains, …

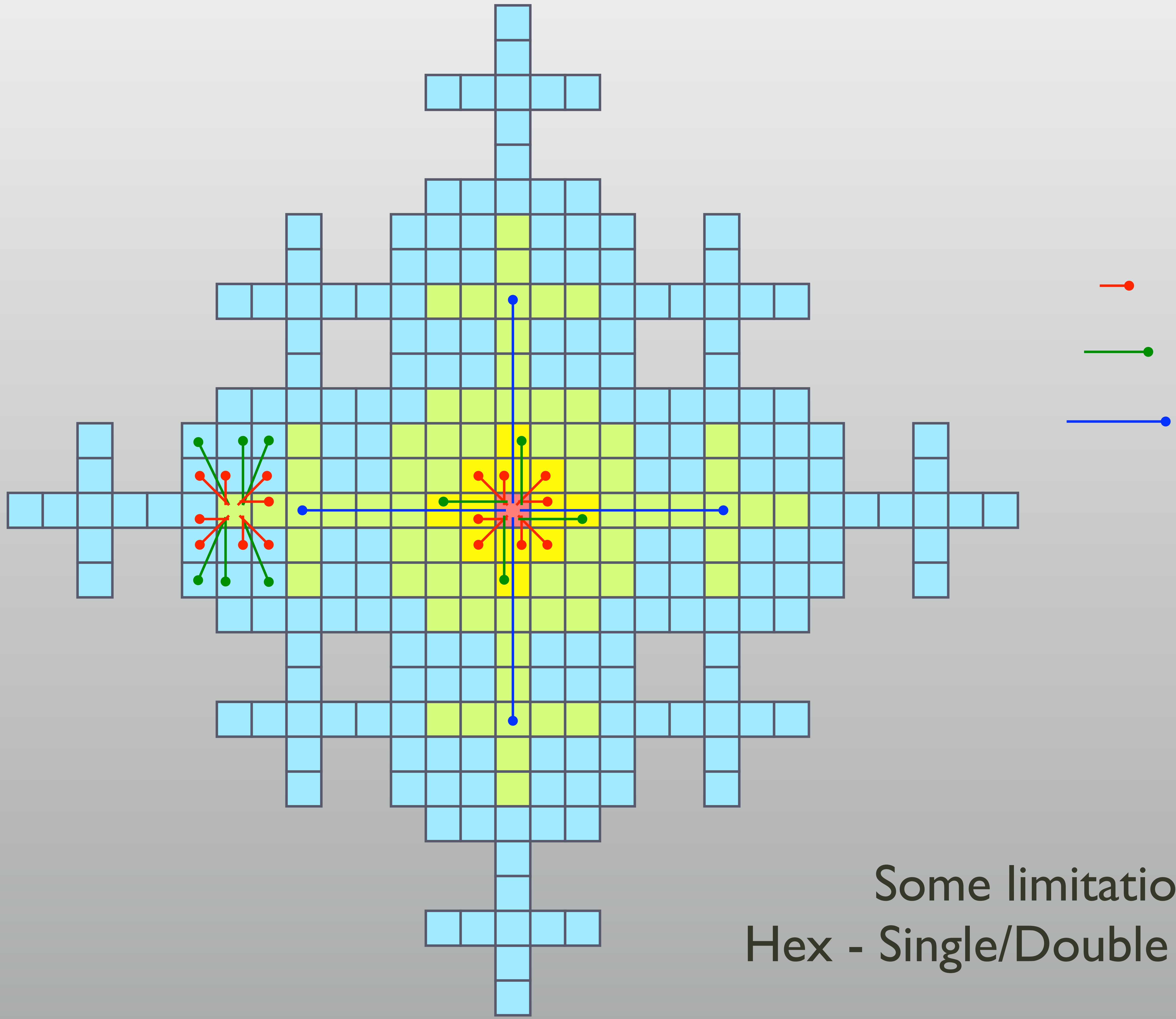
Virtex-4 & 5

* Details are not officially disclosed
* Following slides are estimations from several documents from Xilinx
    * Not very accurate, but just for understanding of modern FPGA architecture

Xilinx Virtex-4

- 3 wire types: Double (2 blocks), Hex (6 blocks) and Long (end-to-end)
  - Double and Hex enables low-latency connections to neighbor LBs
  - An extension to classical Xilinx FPGA architecture with double, quad and long

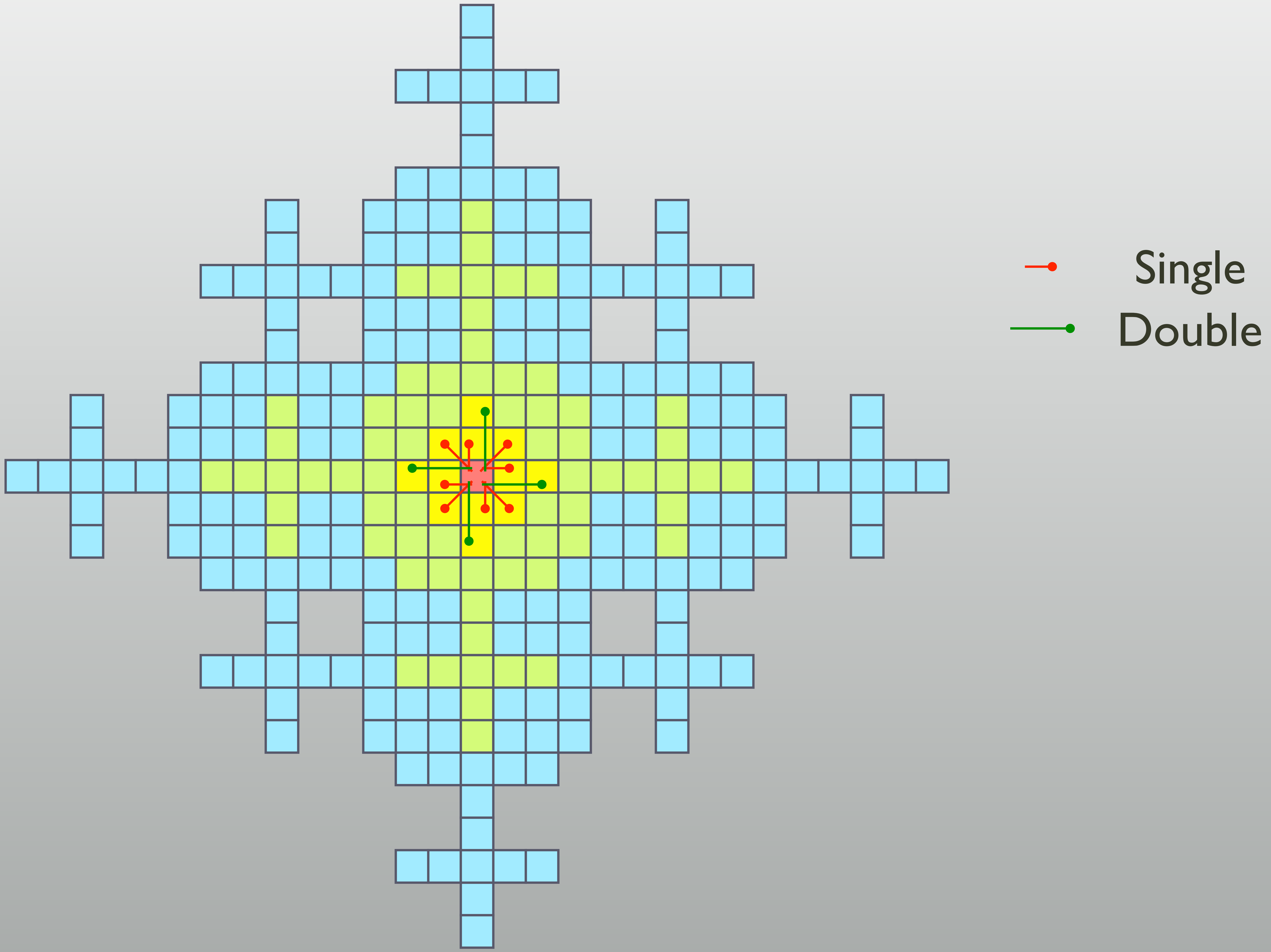
Virtex-4 Interconnect



Some limitations in
Hex - Single/Double transitions

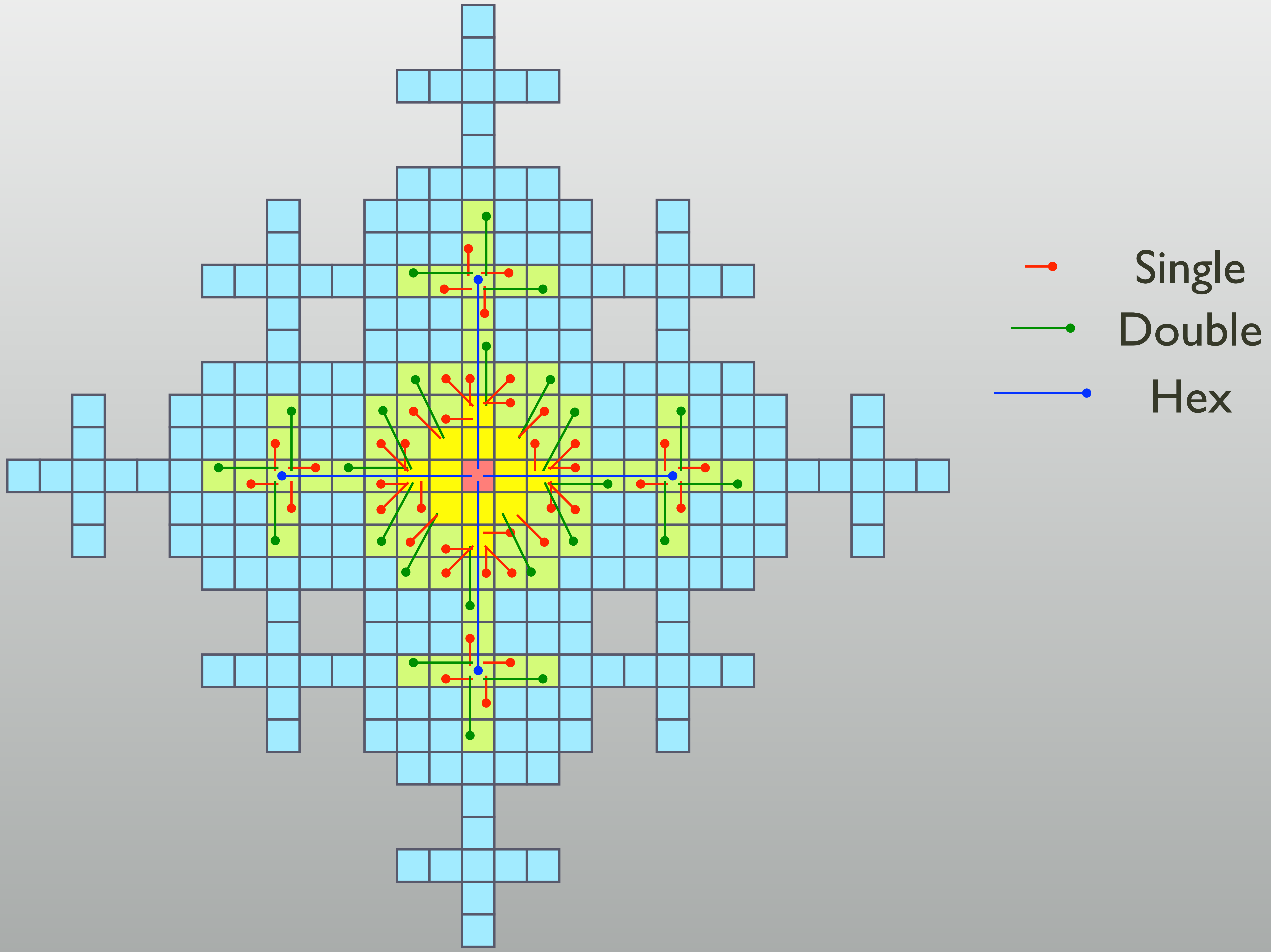
Virtex-4 Interconnect

1 Hop



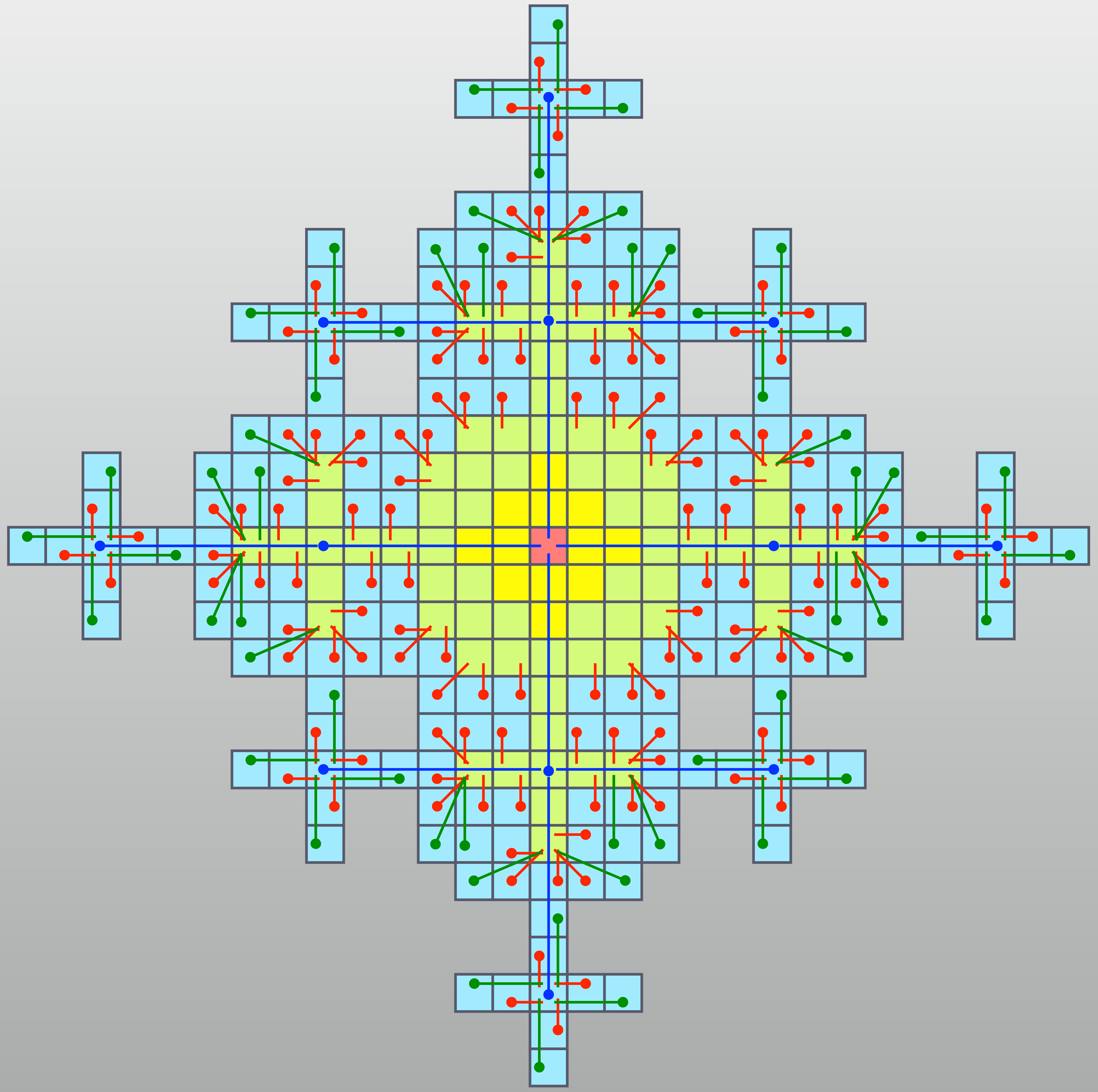
Virtex-4 Interconnect

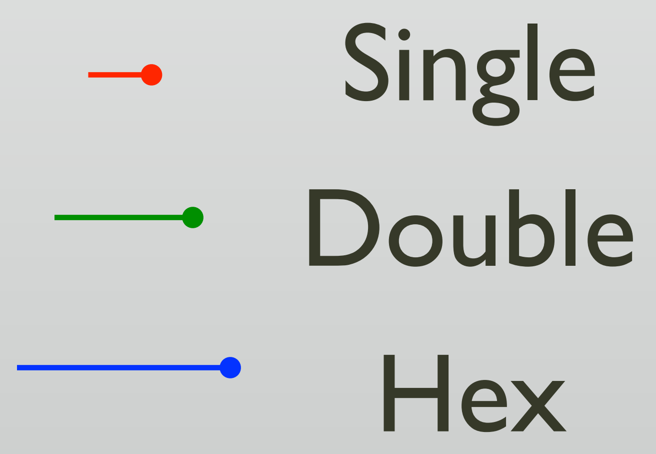
2 Hops



Virtex-4 Interconnect

3 Hops



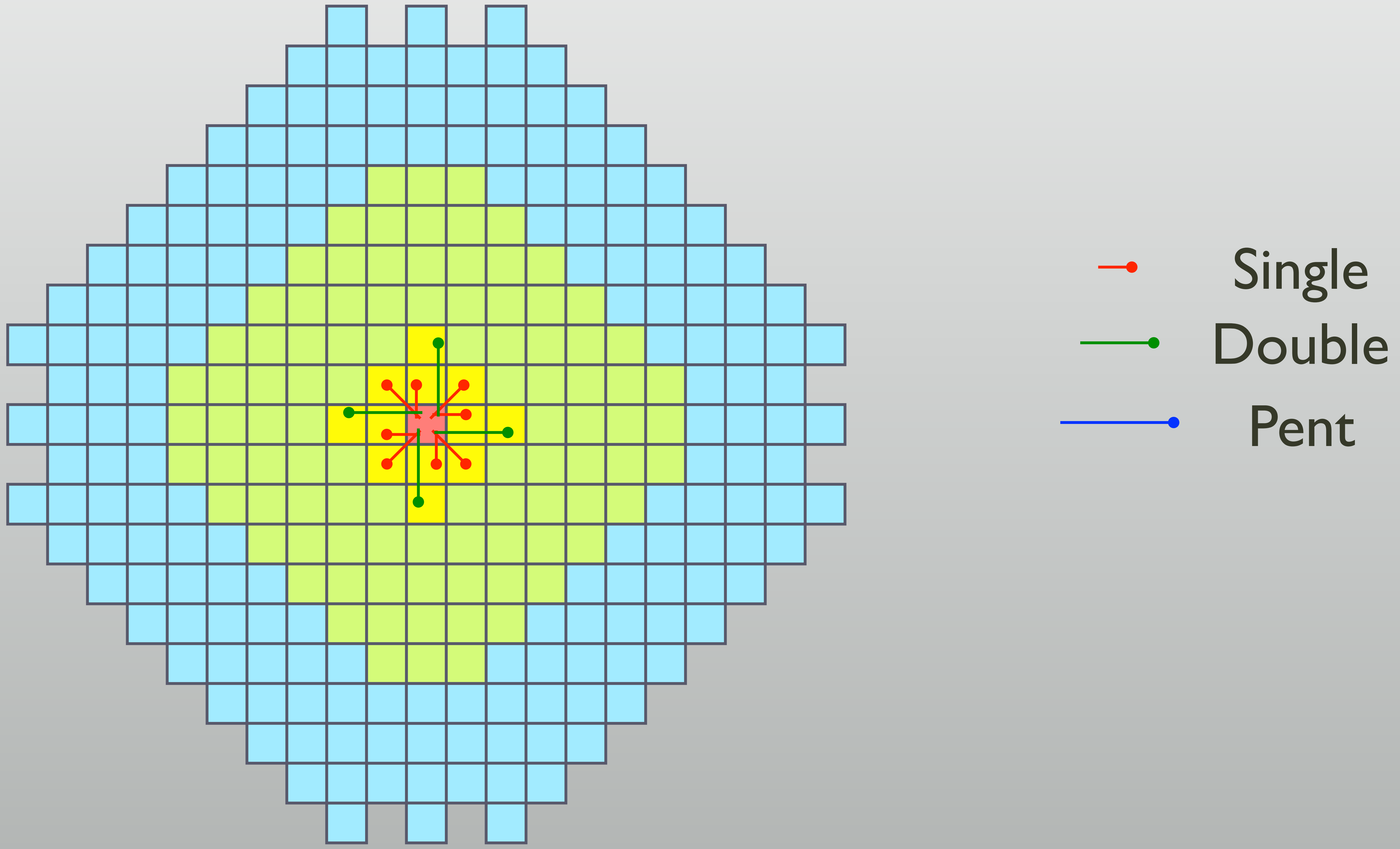


Xilinx Virtex-5

- Comes with 6-LUT
  - Simply, 50% more wires are required than 4-LUT
  - L-shaped wires (Pent) is introduced, instead of Hex wires
  - Enables connections with more neighbor LBs

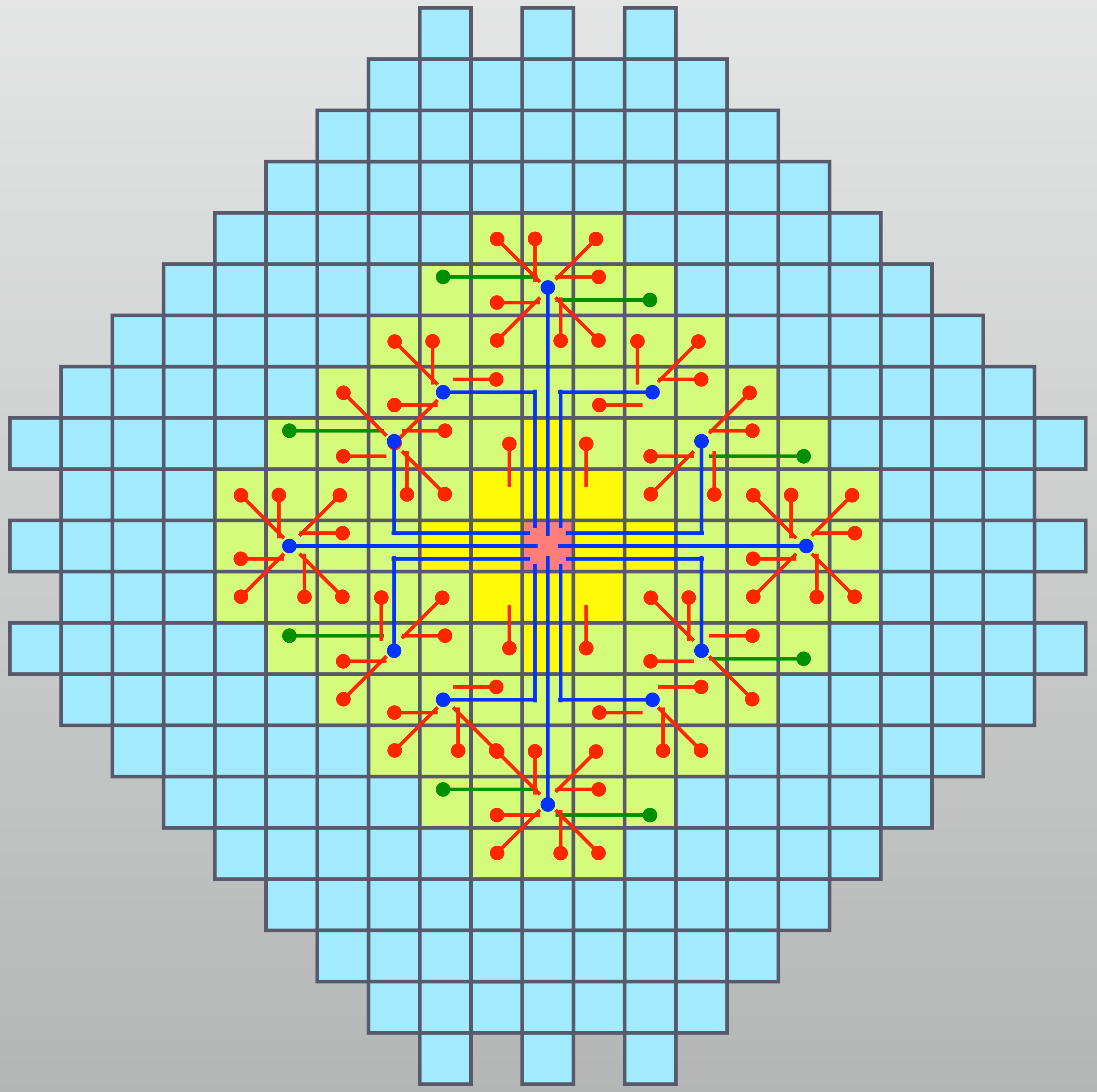
Virtex-5 Interconnect

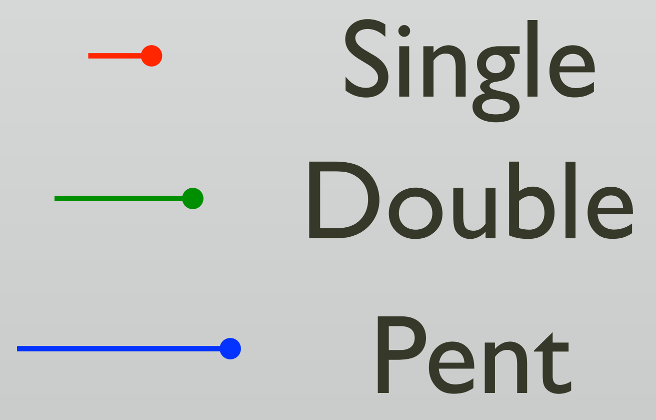
1 Hop



Virtex-5 Interconnect

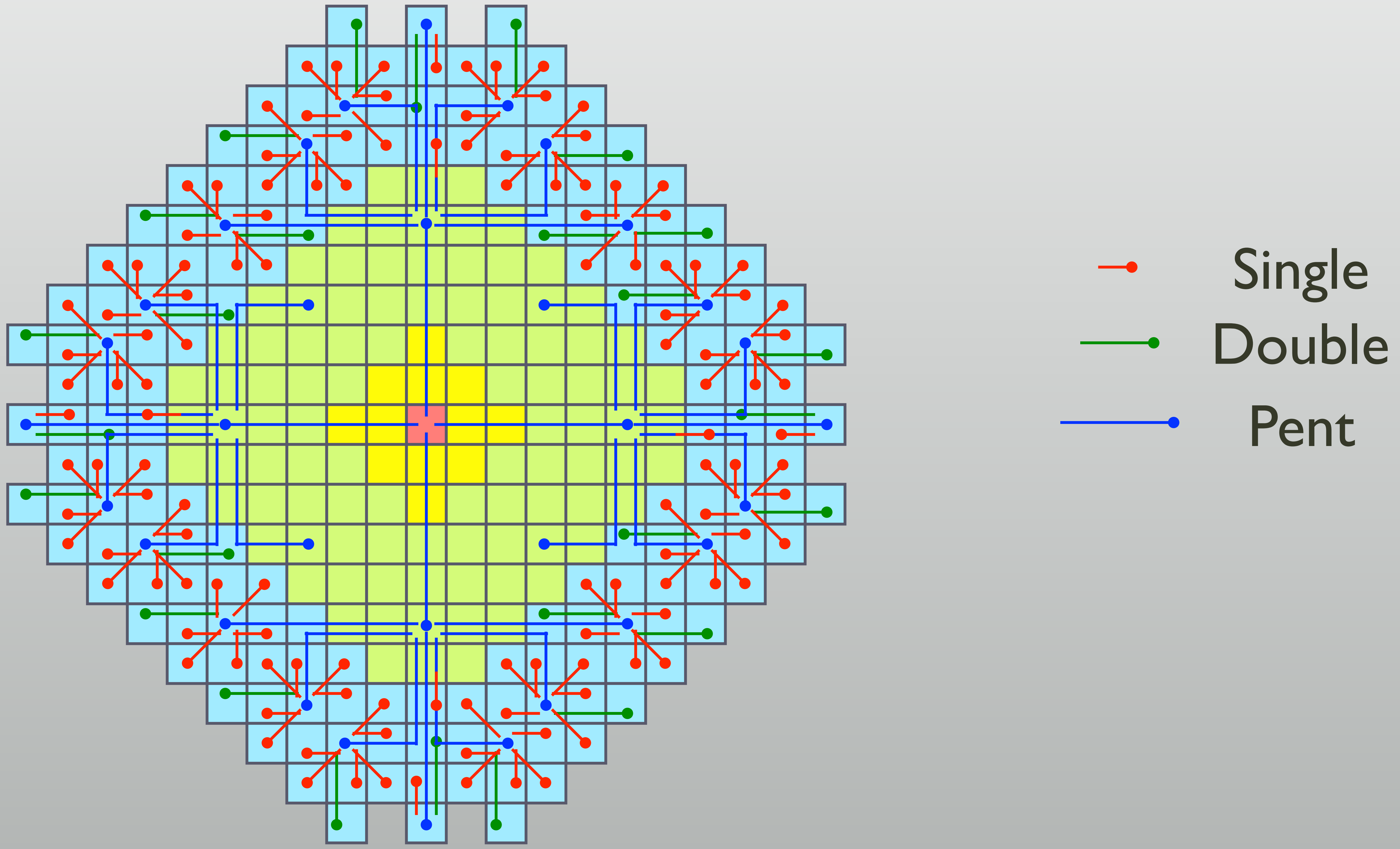
2 Hops





Virtex-5 Interconnect

3 Hops



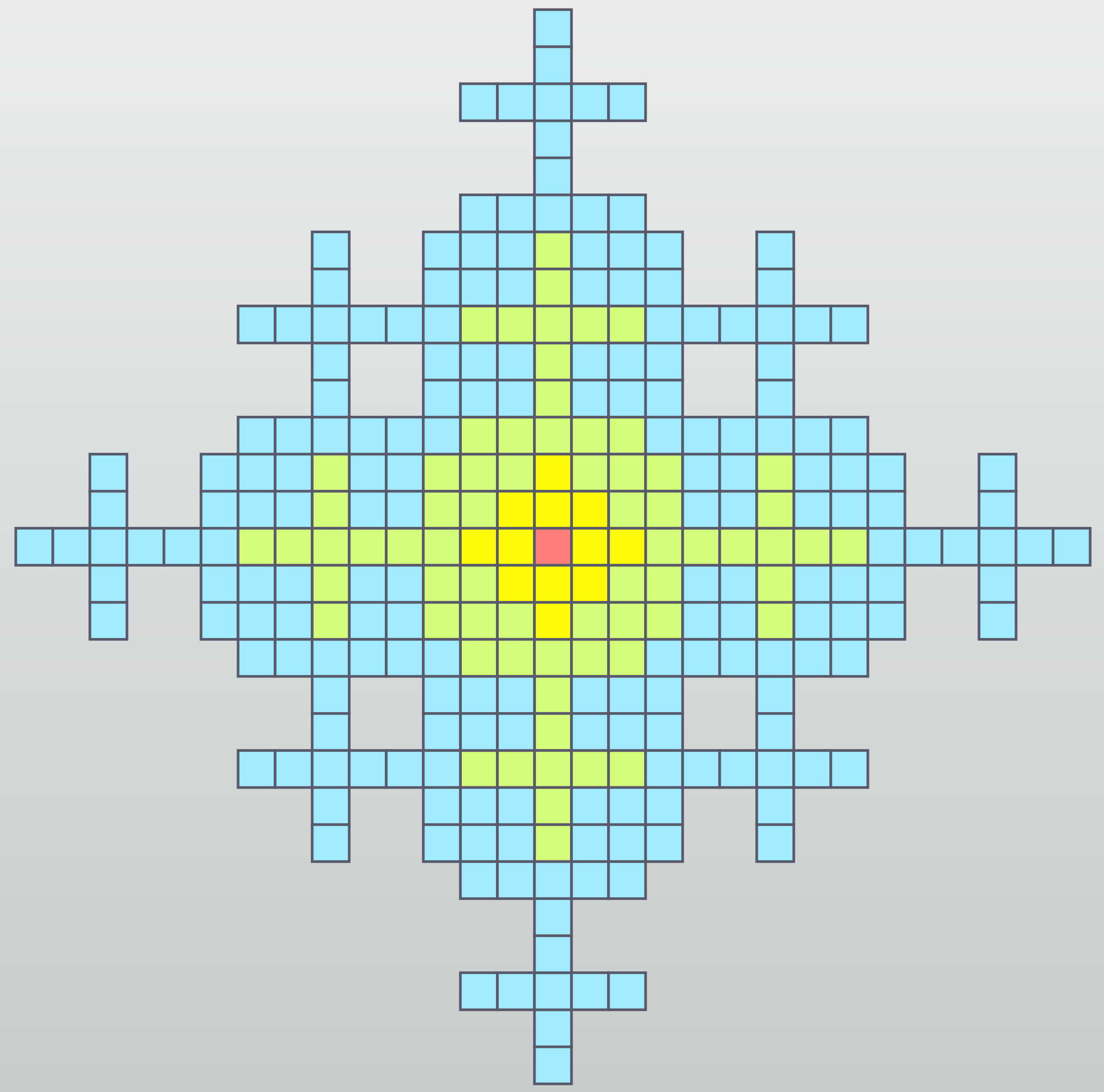
Virtex-4 and Virtex-5

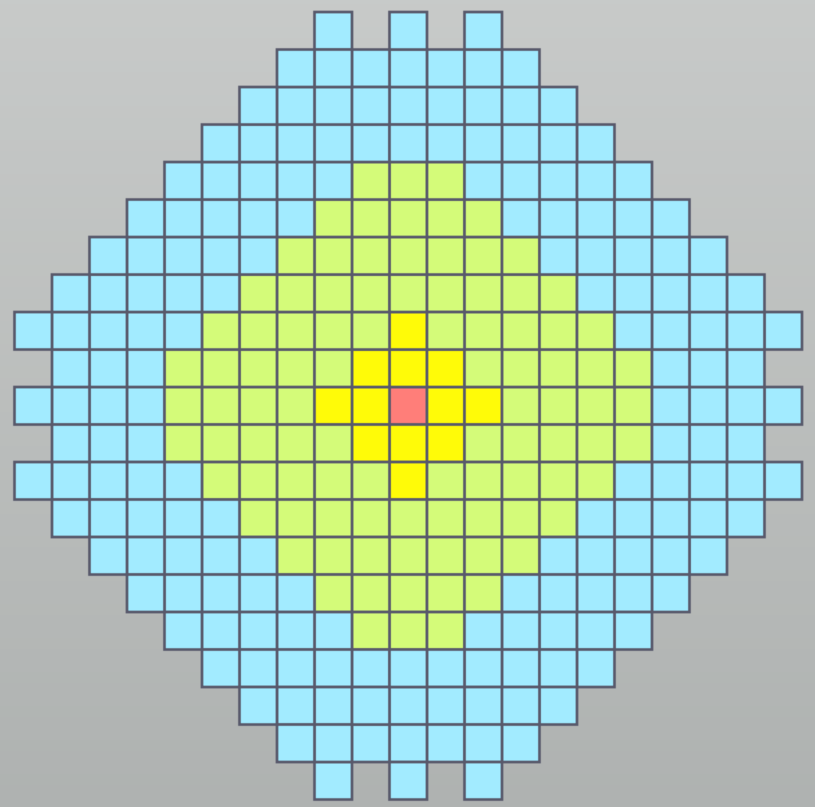
# Reachable CLBs

	Virtex-4	Virtex-5
1 Hop	12	12
2 Hops	68	96
3 Hops	200	180
Total	280	288

# Wire segments

	Virtex-4	Virtex-5
Double	40	42
Hex	120	-
Pent	-	120
Long	24	18
Total	184	180





More wirings?

- Metal layers increases along the process technology, but no more…?
  - Virtex-II 150nm, 6 metal layers (2001)
  - Virtex-II Pro 130nm, 7 metal layers (2002)
  - Virtex-4 90nm, 10 metal layers (2004)
  - Virtex-5 65nm, 11 metal layers (2008)
  - Virtex-6 40nm, 12 metal layers (2009)
  - Virtex-7 28nm (2011)
  - Virtex Ultrascale 20nm (2014) / Ultascale+ 16nm FinFET

Input and output

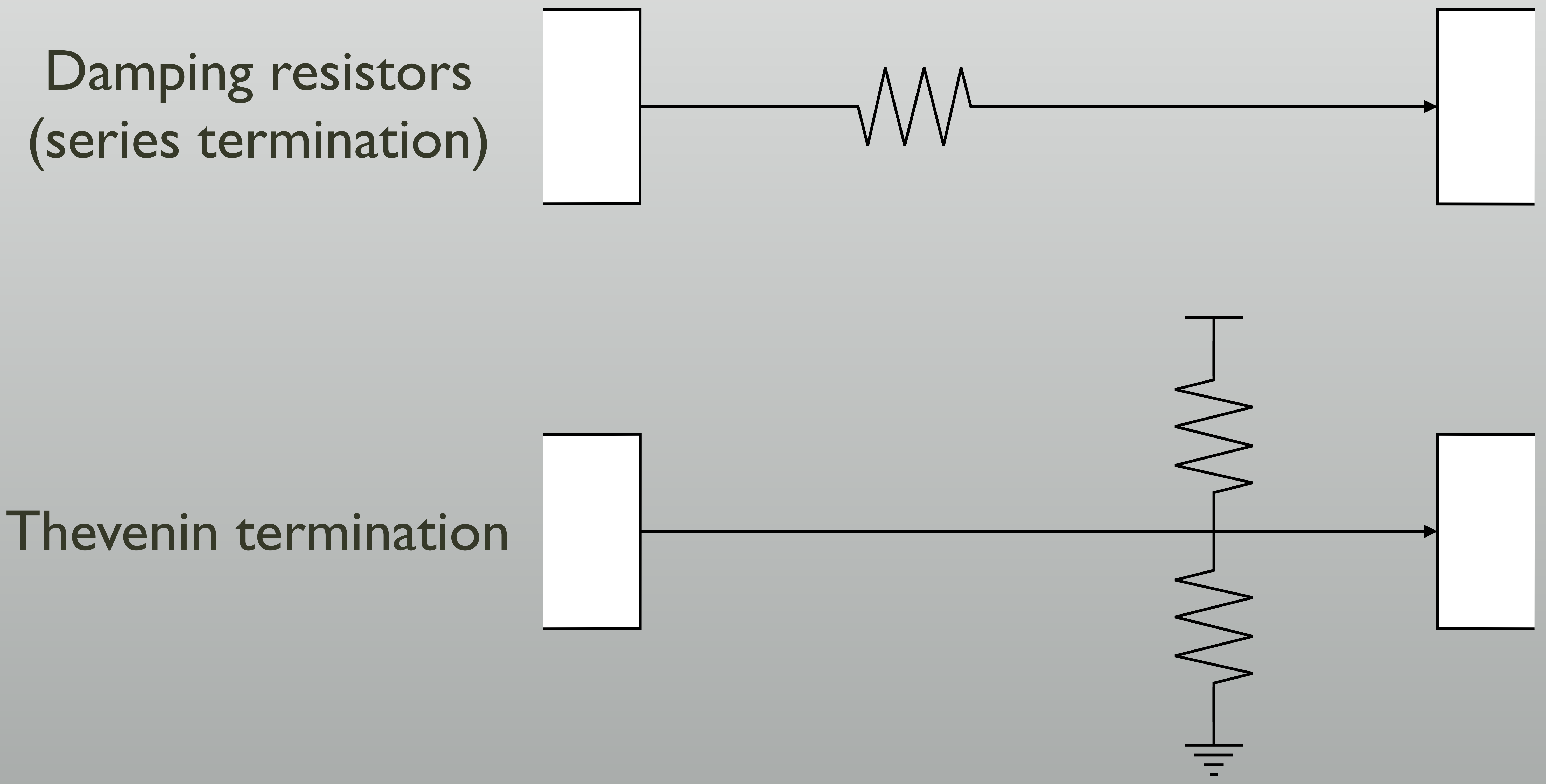
- Classically:
  - TTL (5V), LVTTL (3.3V)
  - CMOS (5V), LVCMOS (3.3V)
- Logic signals = board signals in these standards
  - With just drivers allowing more source/sink currents

Classic, single-ended I/Os

- TTL/CMOS signals in 3.3 or 5V
  - High and Low in the voltage between GND
  - Simple circuits, but weak for noise and ground-bounces
  - Signal integrity problem with 66+MHz, such as reflections

Faster single-ended I/Os

- Impedance matching and termination improves signal integrity



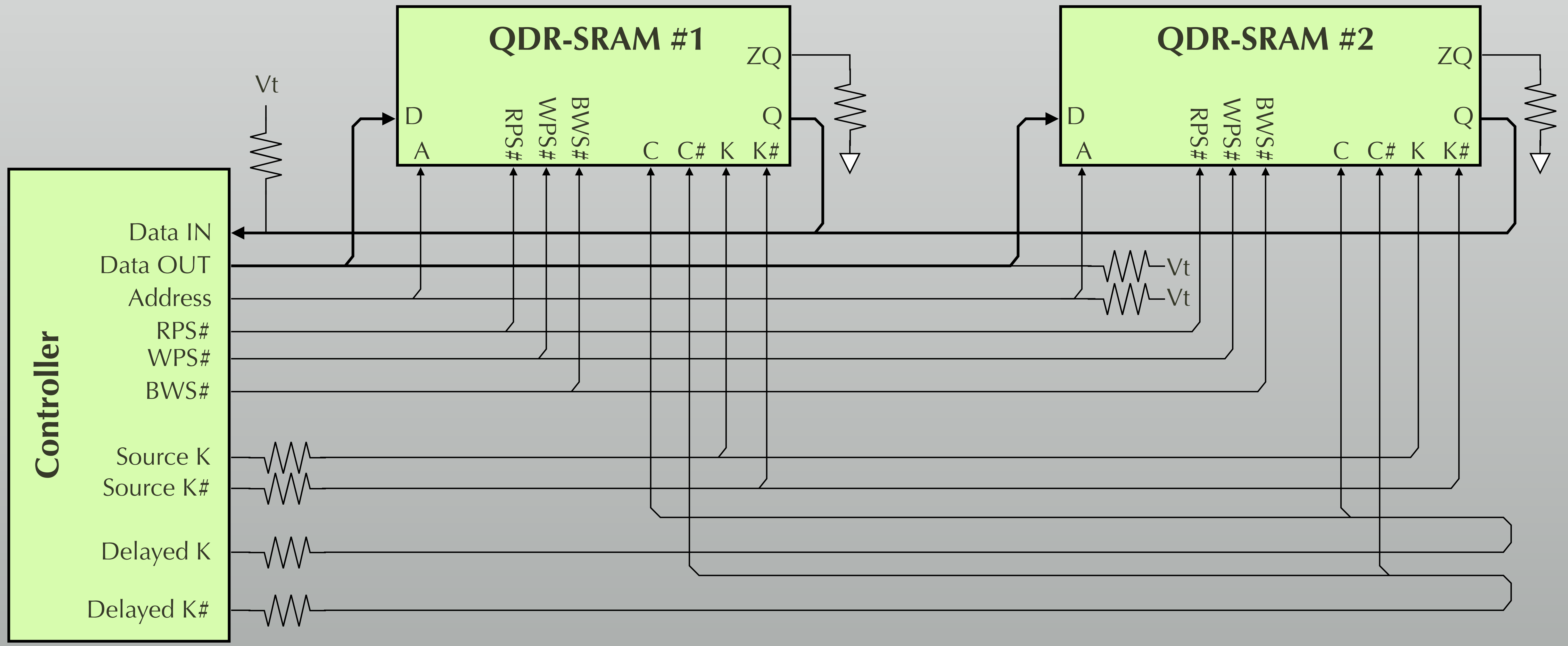
Faster single-ended I/Os

- Use of active terminations
  - HSTL (High-speed transceiver logic) and SSTL (stub series terminated transceiver logic)
  - SSTL is commonly used in DDRx SDRAMs
  - With HSTL and SSTL, signal rate of 800~3000Mbps are possible

HSTL

* Active terminations in all address + data signals, to reduce side effects of simultaneous switching

* Clock signals are with series terminations



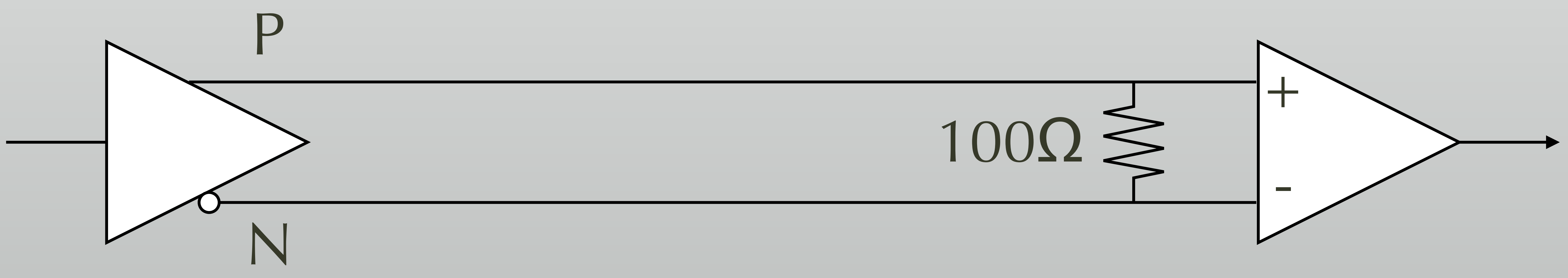
$Vt=V_{ddq}/2$

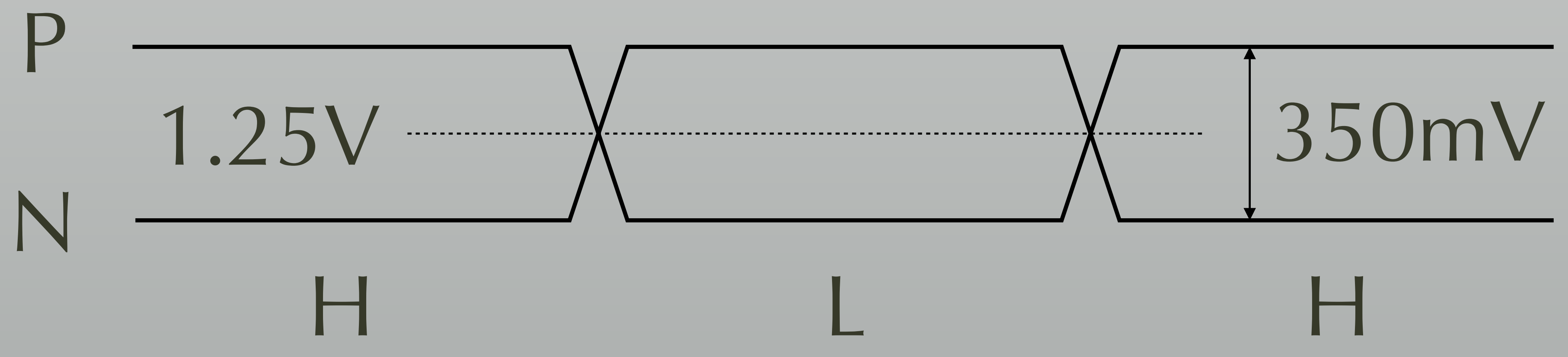
Differential signaling I/Os

- Much faster signal frequency with paired signal wires
  - LVDS, LVPECL, CML and more standards
  - Popular in super-speed I/Os: PCI Express, USB, HDMI, …
    - Several Gbps with 2 wires: 8~28Gbps per signal pair is available

LVDS

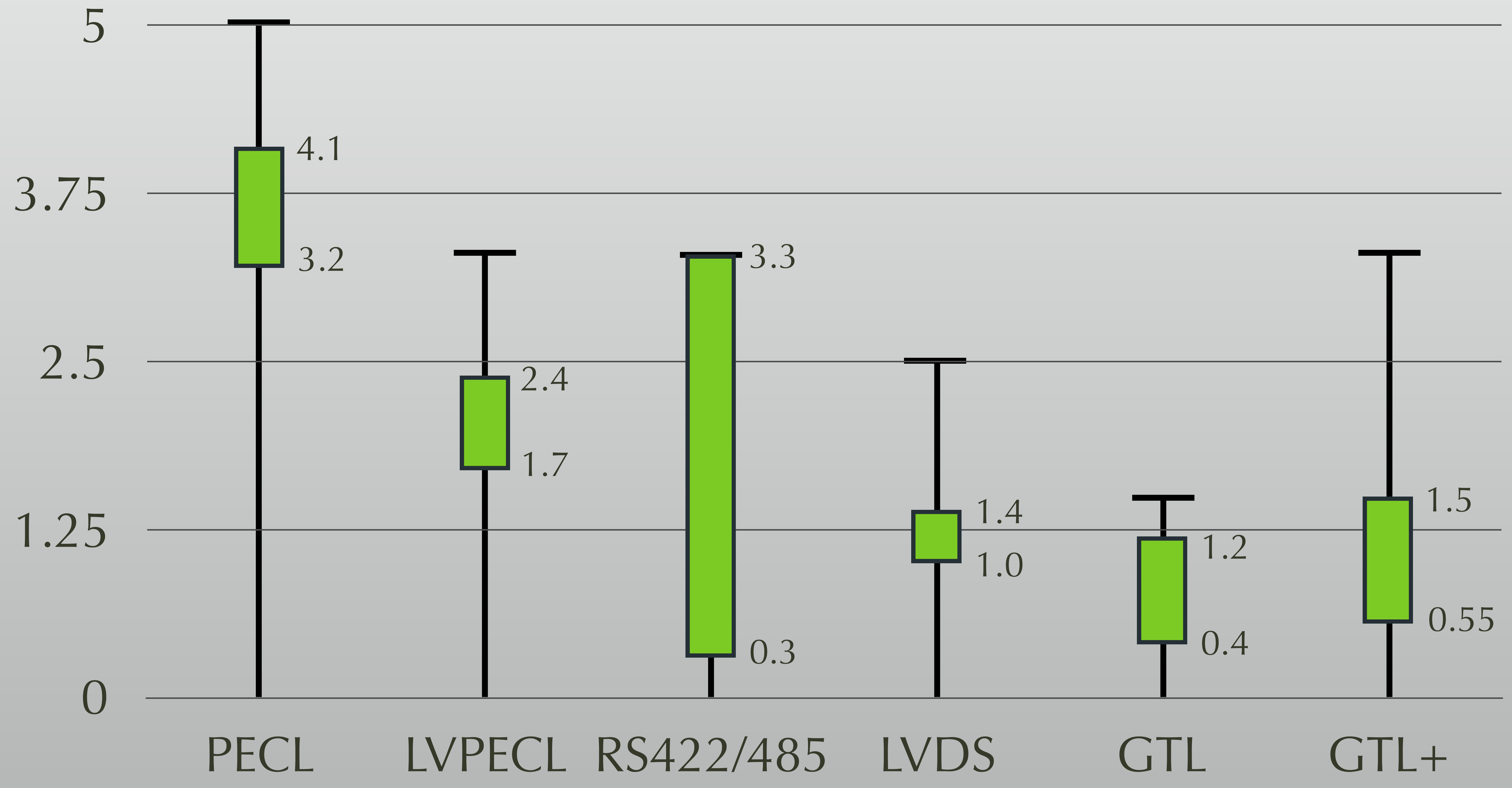
- 3.5mA, current mode signaling (with 1.25V of common mode)
  - With 100Ω termination: amplitude of 350mV





Differential I/O standards

- Power rails and amplitudes

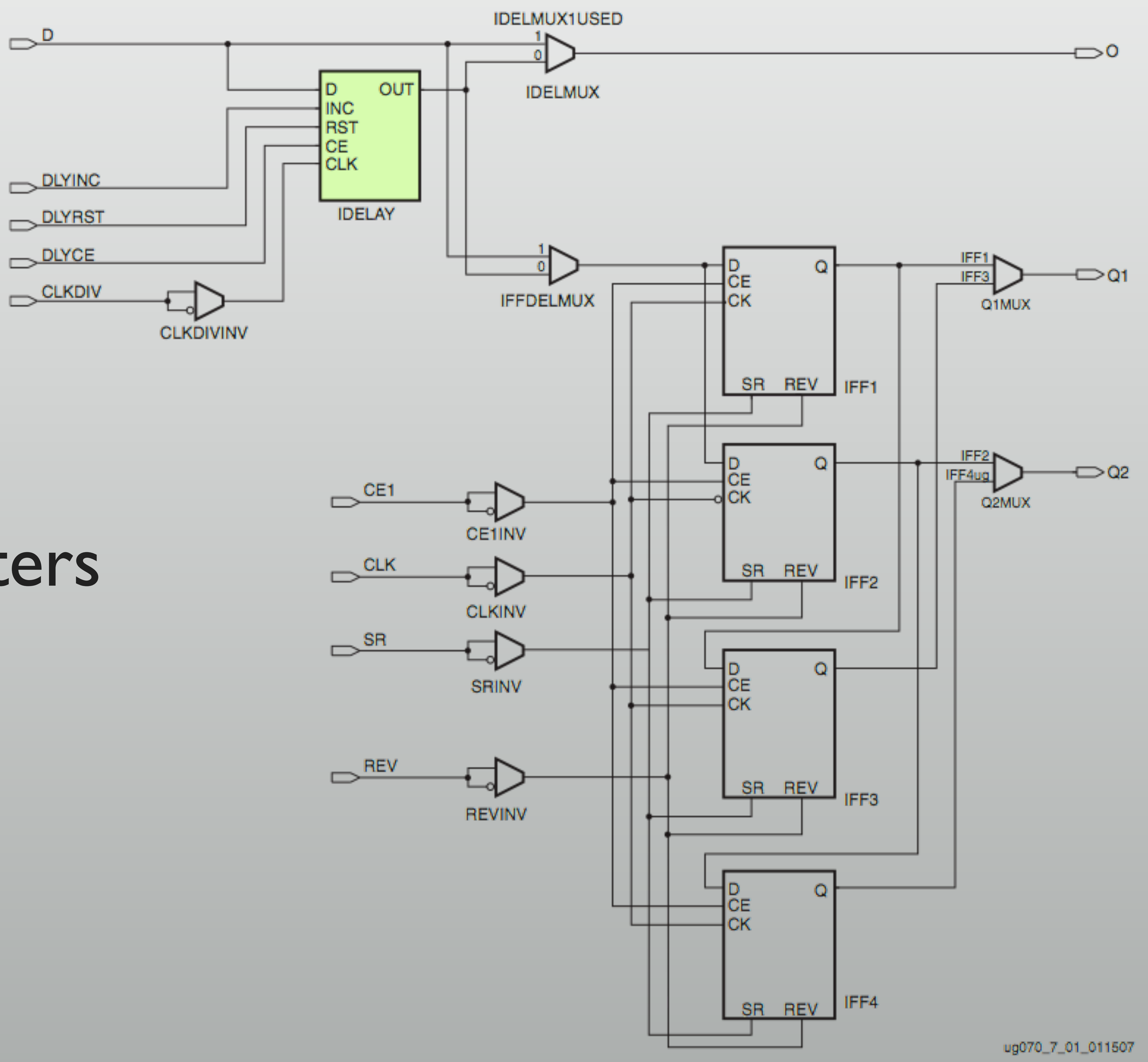


Many I/O standards are covered

* LVTTL (3.3V)
* LVCMOS (3.3/2.5/1.5/1.2V)
* HSTL (1.8/1.5/1.2V)
* SSTL (2.5V/1.8V/1.5V)
* LVDS
* LVPECL
* GTL, GTL+
* (PCI, PCI-X)

On-chip I/O tweaks

- ILOGIC block in Virtex-4
  - Not just an input buffer
  - Delay element (IDELAY) and 4 registers
    - Almost same for output



Xilinx UG070 Virtex-4 User Guide

Input registers

- Separating off-chip delays and on-chip delays
  - Off-chip delays are managed by the board designer
  - On-chip delays are managed by FPGA design tools such as Vivado
- These delays should be separated to make problems simple:
  - Placing registers on all I/O signals are highly recommended

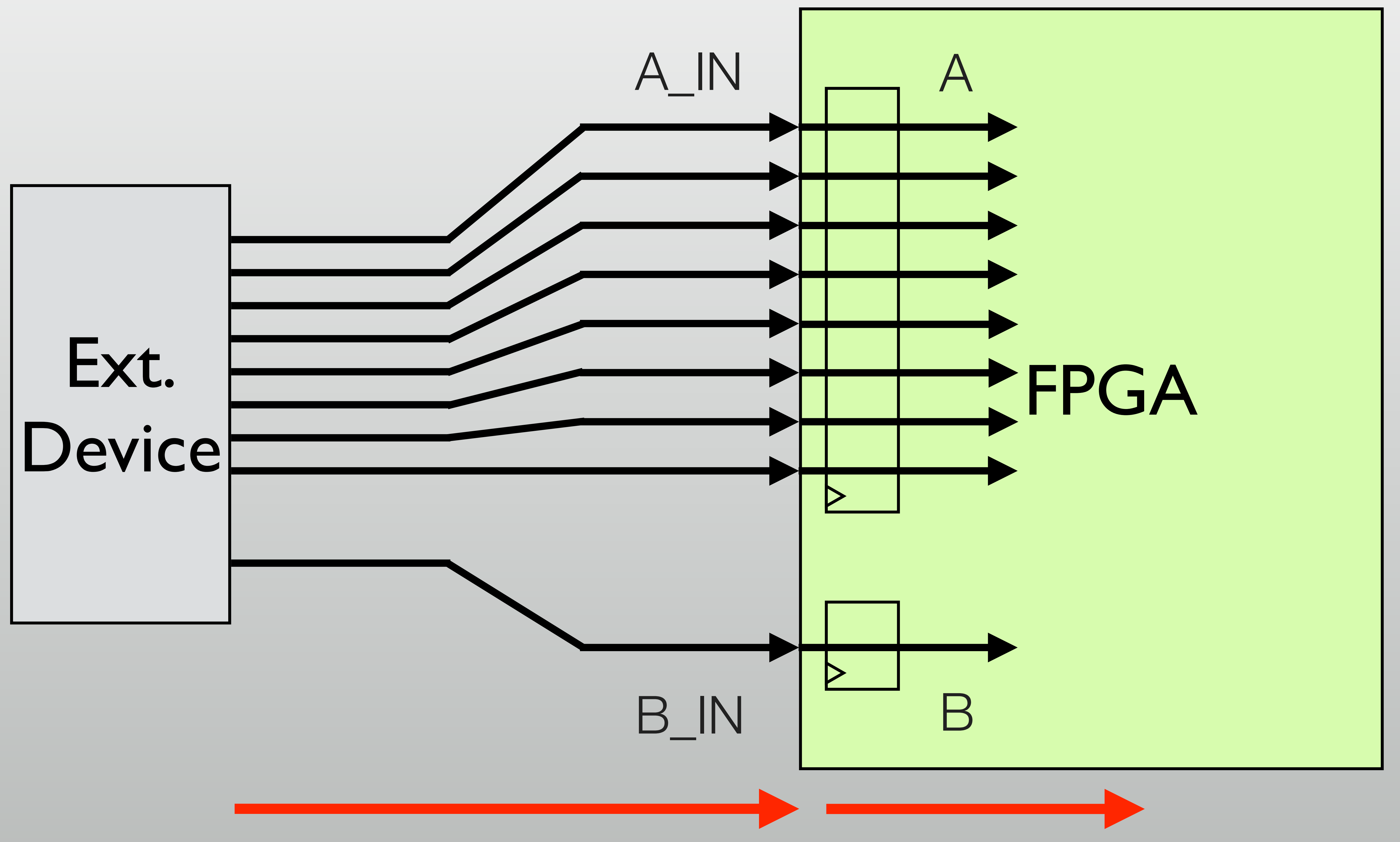
I/O registers

```
module some_fast_one
   ( input CLK, RST,
     input [7:0] A_IN,
     input       B_IN,
     . . . );

   reg [7:0] A,
   reg       B;

   always @ (posedge CLK) begin
      A <= A_IN;
      B <= B_IN;
   end

   . . .
endmodule
```



Board designer adjusts the delay to meet the target frequency

I/O registers align all the signals to clock edge: maximize the internal timing margins

DDR signals

- SDR (Single Data Rate):
  - Data signals are aligned on the positive edge of the clock signal
  - Data frequency = 0.5 x Clock frequency
- DDR (Double Data Rate):
  - Data signals are aligned on both clock edges: Data frequency = Clock freq.

DDR signals in FPGA

- always @ **(CLK)** works in DDR
  - Works, but requires 1/2 of logic delay
  - Instead, internal SDR signals with 2x bit width is popular
    - DDR-2x SDR conversion is done by the I/O block registers
    - More timing margin in FPGA

Example of input DDR

- 2 or 3 registers in IOB are used
  - Some of them operates @ negedges

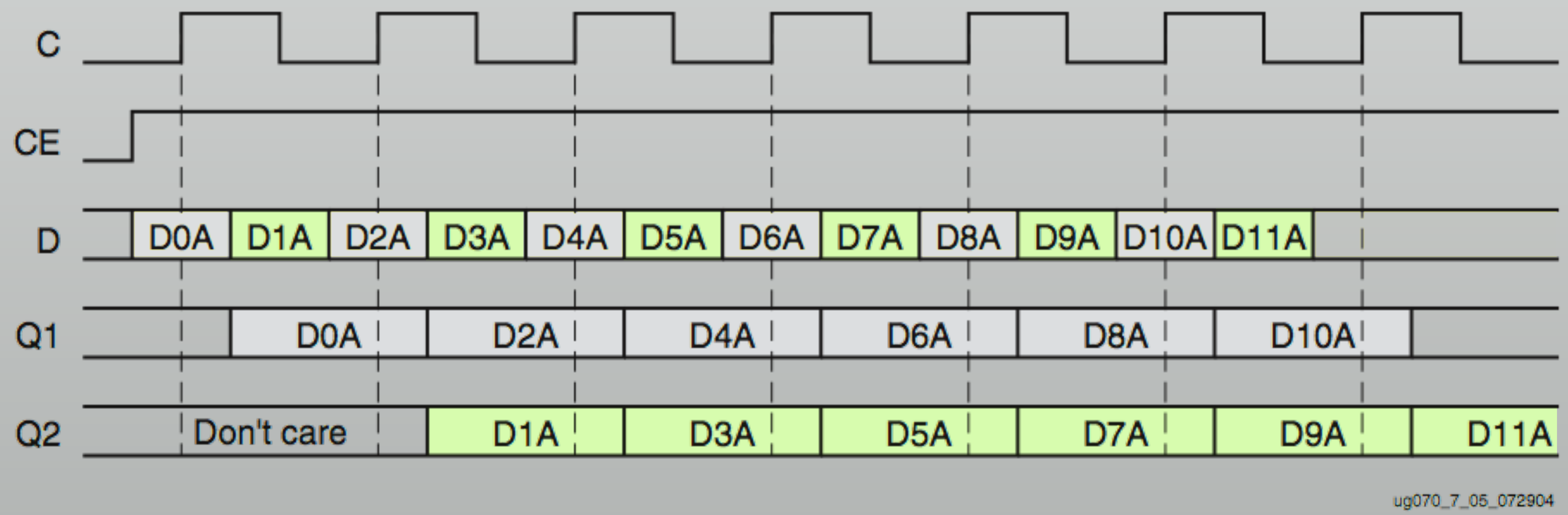


Figure 7-5: **Input DDR Timing in SAME_EDGE Mode**

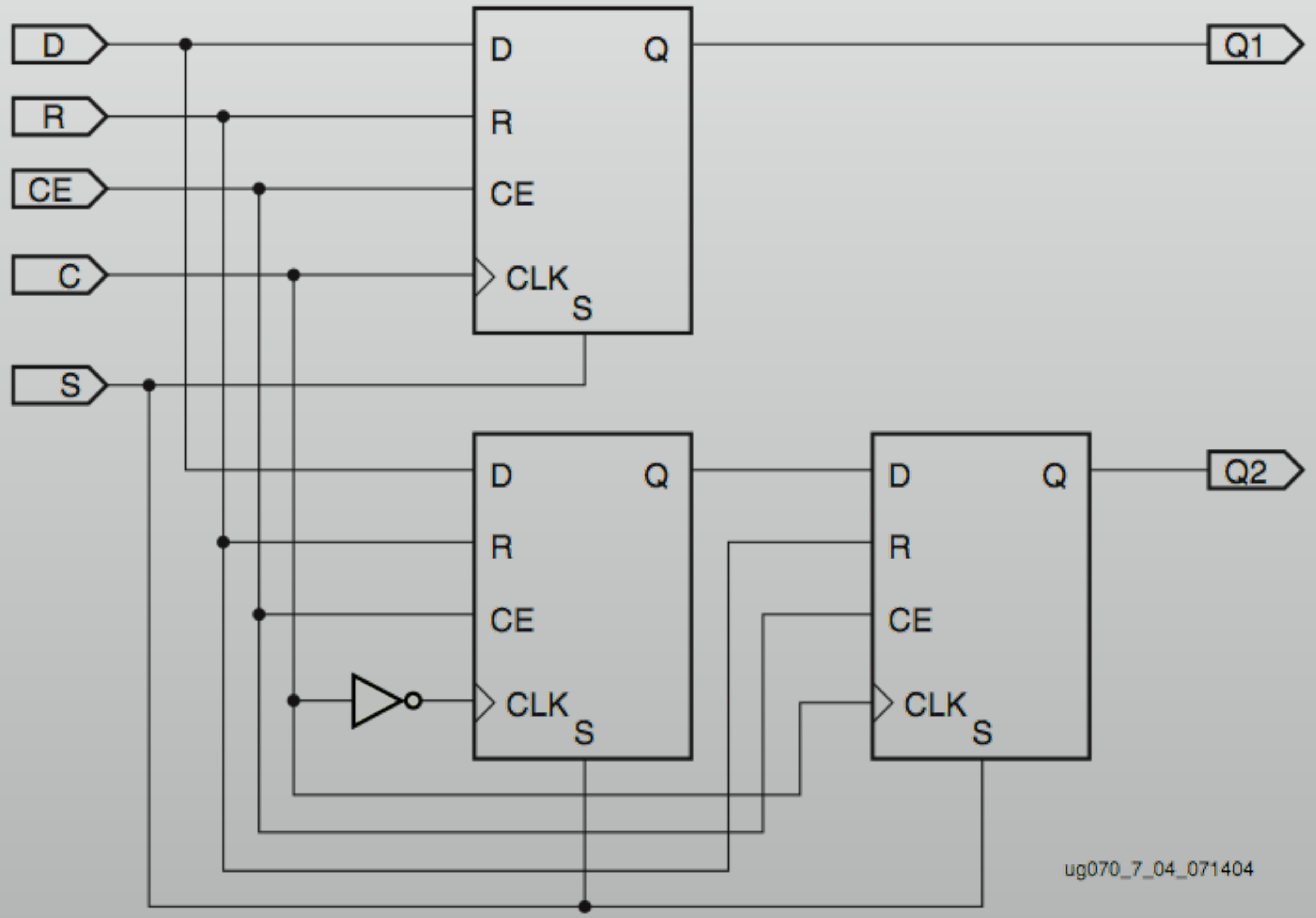


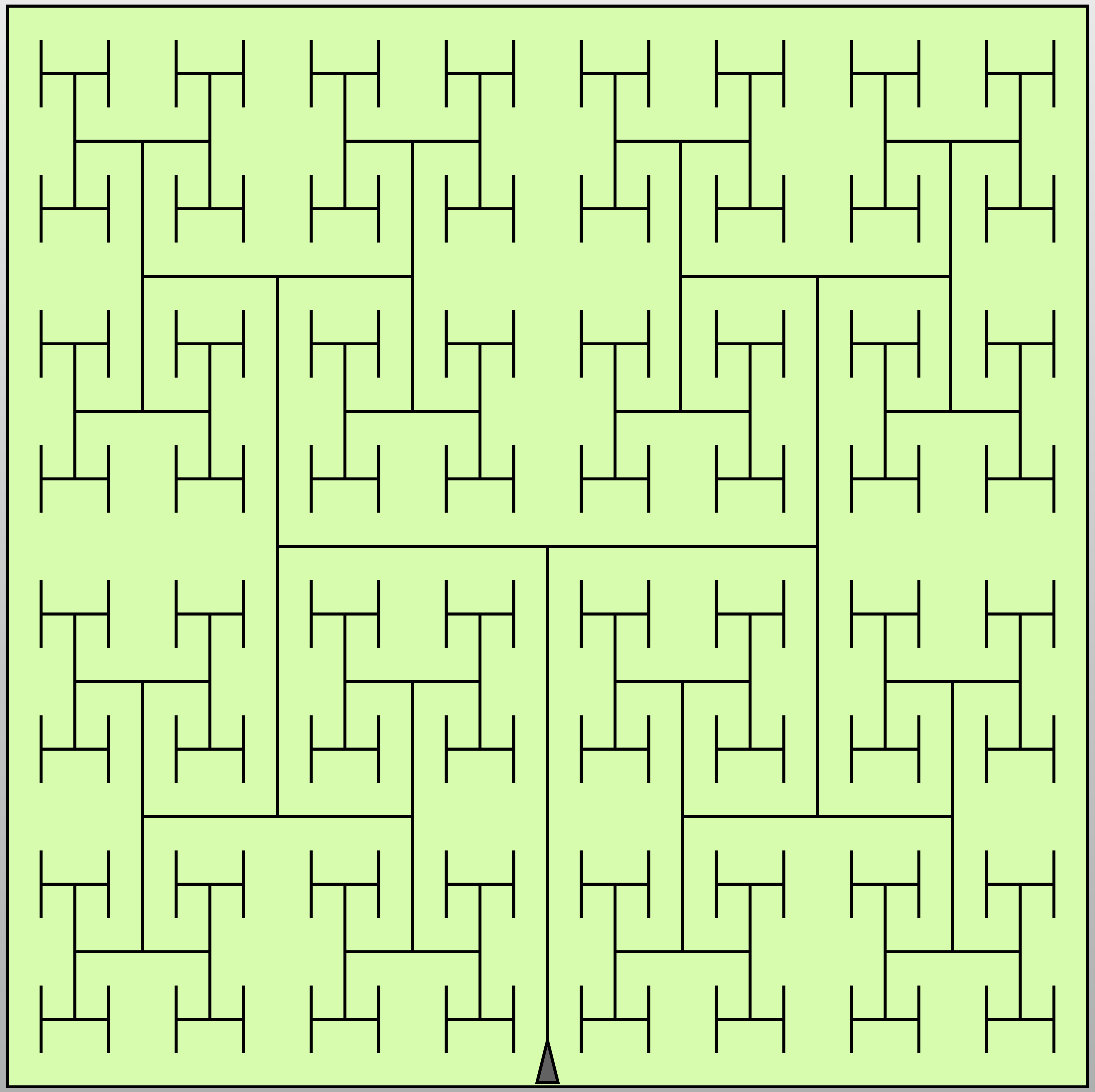
Figure 7-4: **Input DDR in SAME_EDGE Mode**

Delay element…?

* Shorter delay is better, but uniform delay is crucial
    * Wire delays are always slightly different in high-speed parallel bus
    * Fine-grain insertion of delay elements enables higher data rates or frequency
    * Independent delay elements on every input block: calibration with test pattern on system startup

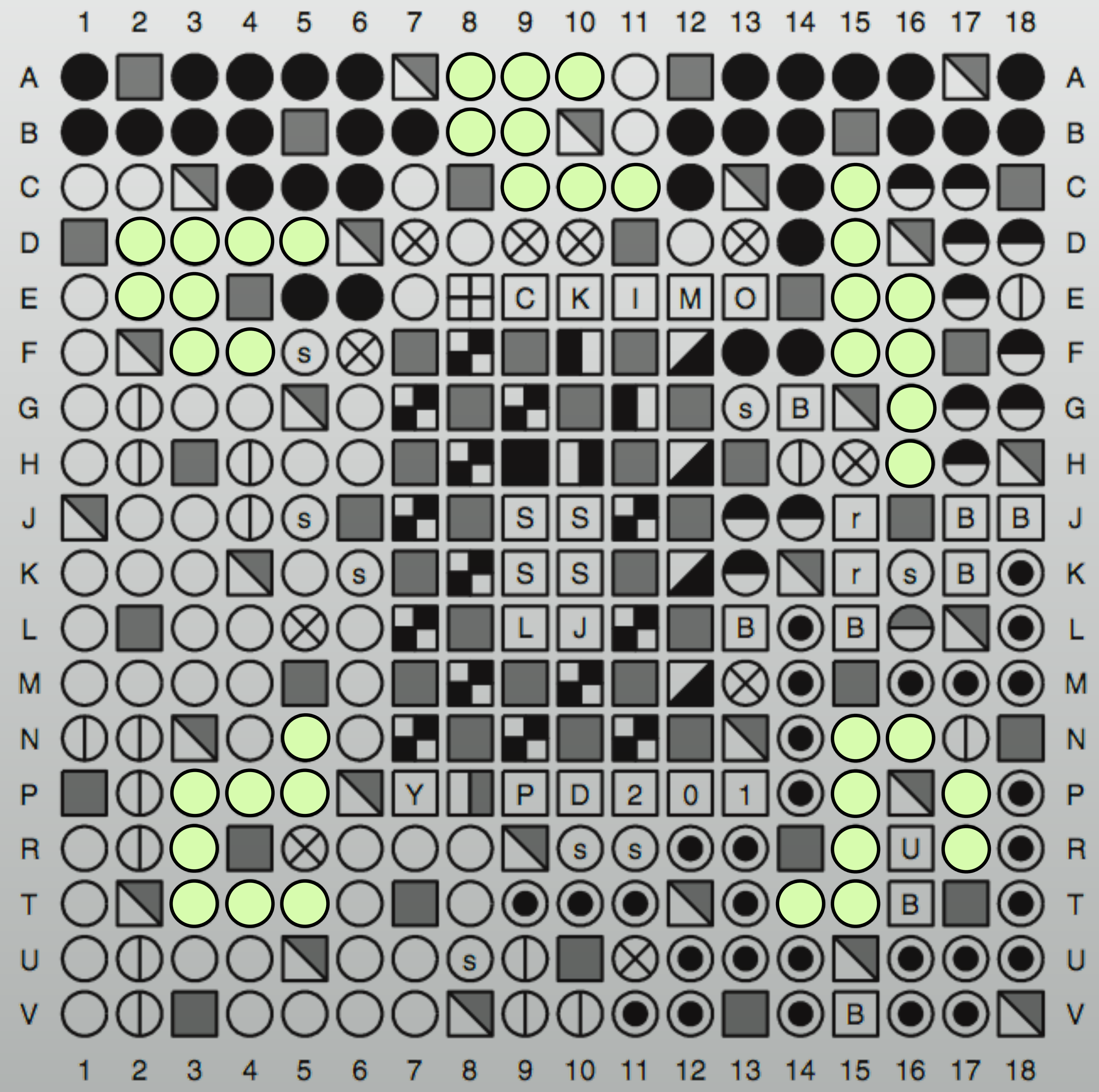
Clock tree

- A clock tree is driven by a clock buffer
- A clock buffer is driven by:
  - A clock capable pin, or special FPGA components such as CMT



Clock Capable Pin

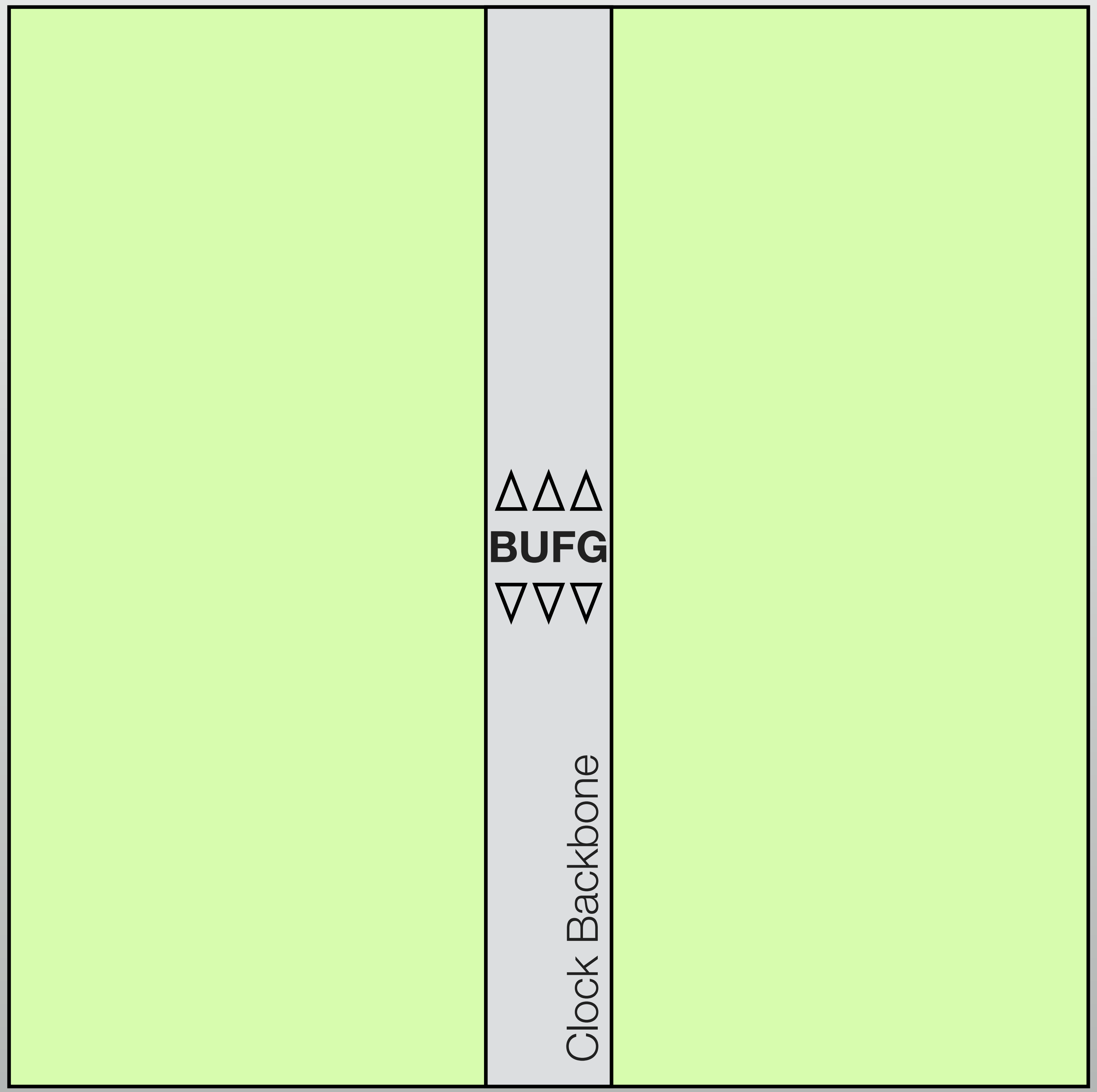
- Are very limited:
  - Only half of them are available in single-ended
  - Not all clock capable pins are same: detail follows



Xilinx UG475: 7 Series FPGAs Packaging and Pinout

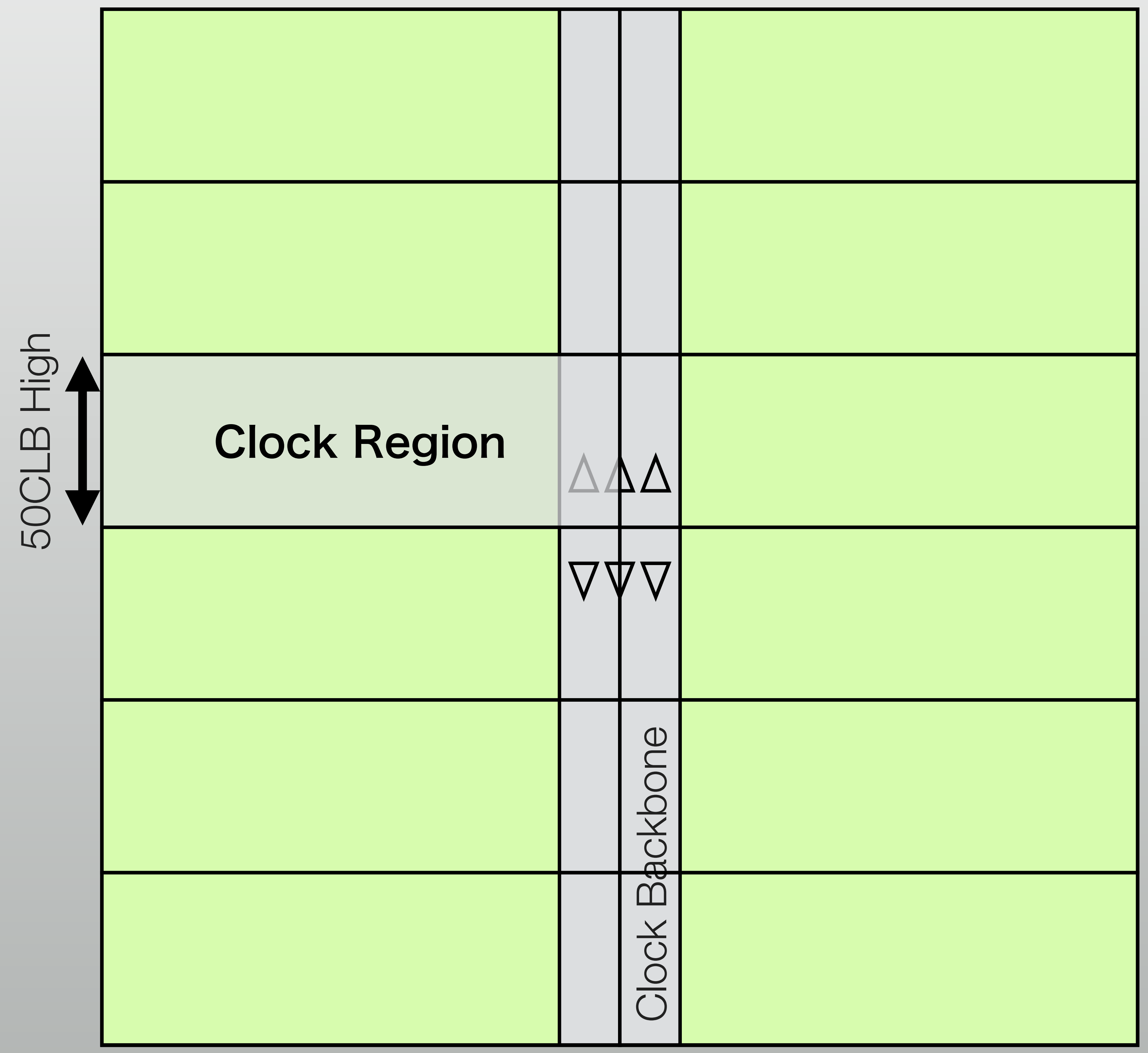
Global Clock

- Global Clock Buffer
  - Basically located at the center
  - Limited number



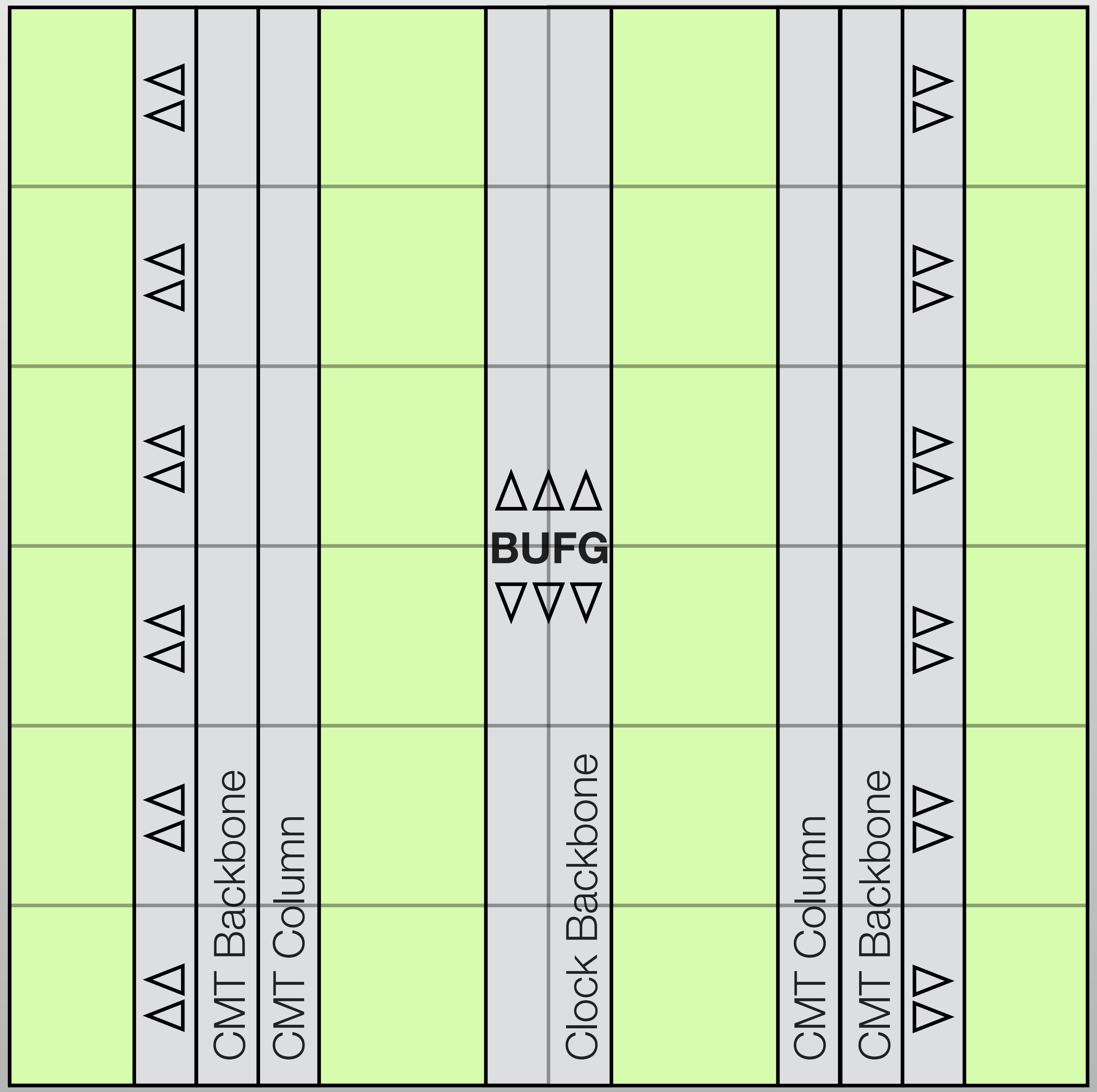
Regional Clock

- Not all clock signals are required in everywhere
  - For example, memory controller requires several clock signals between memory devices and controller
- Regional clock is a good alternative
  - More # of clock signals are available



Regional Clock

- Each clock region have:
  - Several clock capable I/O pins and
  - Several regional clock buffers
- In some devices, several clock region has their own clock management tiles



Summary

* FPGA has basically uniform structure: LB, CB and SB
  * LB for programmable logic, CB+SB for programmable wires
  * Several dedicated components are essential: IO blocks and clock buffers
* More dedicated blocks are on modern FPGAs:
  * RAMs, DSPs, High-speed I/Os and CPUs: had no time to talk about this…

Improves LB structure and functionality

- LUT + FF + MUX is the very basic
- Effectiveness for common building block is inportant
  - LUTs as memory elements or shift register
  - Carry chain for faster adder implementation

Clustering: improves LB connectivity

- Adding direct LB connections in a cluster without requiring CB or SB
  - Limited to simple structures, but powerful in:
    - Using 2 or more LUTs to form a larger LUT logic
    - Making longer shift-register chains or adder carry chains

Improving interconnections

- Neighbor-to-neighbor connection is not perfect
  - Long-distance and multiple-destination connections are important
  - Availability of more # of destination LBs is crucial for better performance
    - Interconnection topology is a key of latest FPGA devices

Off-chip I/Os

- Supports various single-ended and differential I/O standards
  - To connect with various external devices
  - In several cases, on-FPGA termination registers are available to reduce on-board resistors
  - Supports high-speed serial signals such as PCIe, USB or HDMI
    - Now FPGAs are used a hub of system

On-chip I/Os

- I/O blocks have delay elements and registers
  - Delay elements enables the delay calibration on startup
  - Registers separates on- and off-chip delay, also useful for DDR operations

Clock resources

- Dedicated clock input pins, clock buffers and clock-tree wires
  - Global clocks reaches to everywhere on FPGA
    - Global clock signals are synthesized in RTL without special cares
    - Explicit use of regional clock enables to use local clock signals for several modules, such as external memory controllers