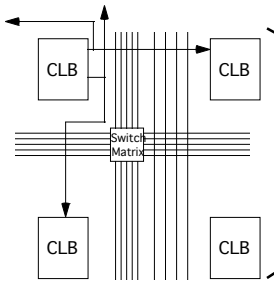
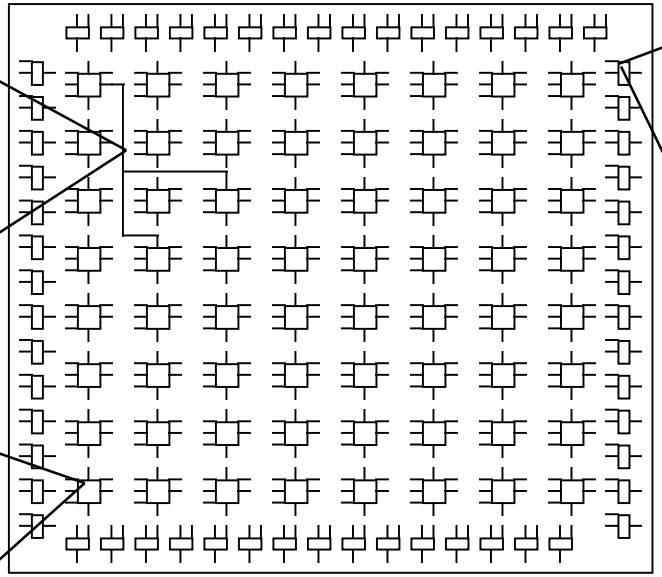
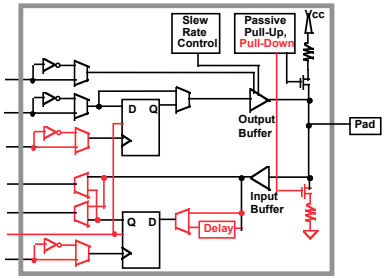
XILINX®

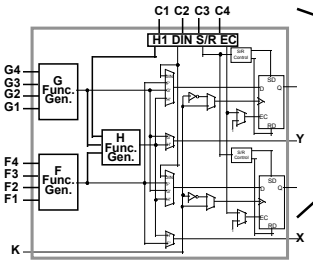


Programmable Interconnect

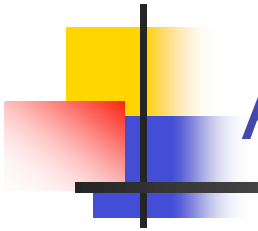




I/O Blocks (IOBs)



Configurable Logic Blocks (CLBs)



Applications of FPGAs

- Implementation of random logic
  - easier changes at system-level (one device is modified)
  - can eliminate need for full-custom chips
- Prototyping
  - ensemble of gate arrays used to emulate a circuit to be manufactured
  - get more/better/faster debugging done than possible with simulation
- Reconfigurable hardware
  - one hardware block used to implement more than one function
  - functions must be mutually-exclusive in time
  - can greatly reduce cost while enhancing flexibility
  - RAM-based only option
- Special-purpose computation engines
  - hardware dedicated to solving one problem (or class of problems)
  - accelerators attached to general-purpose computers

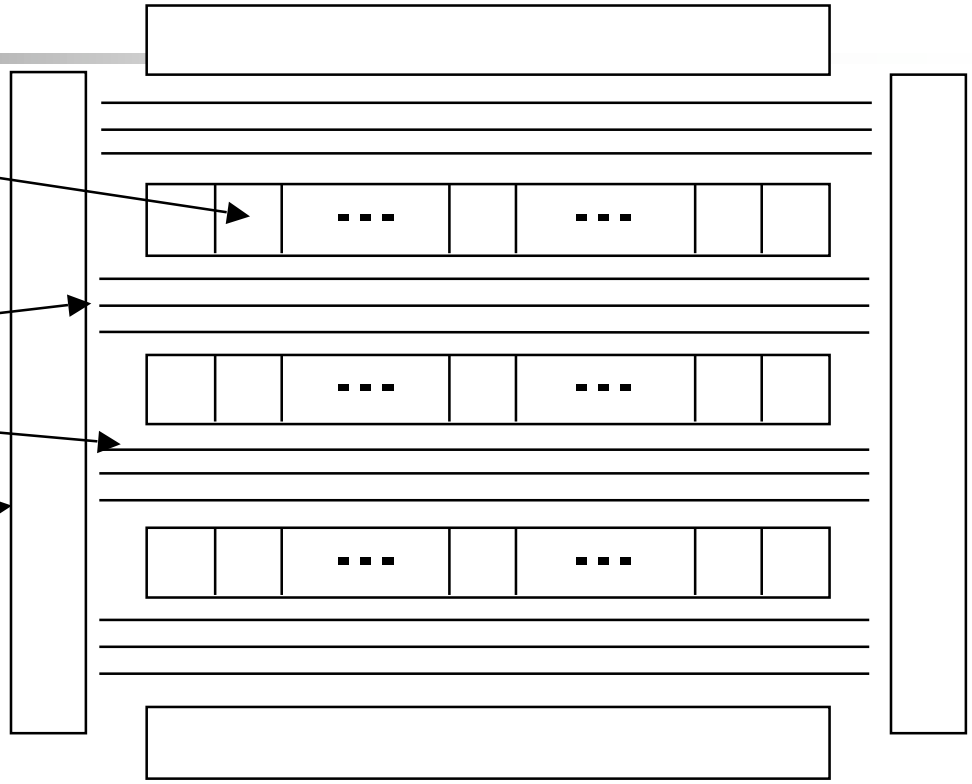
Evolution of implementation technologies

- Logic gates (1950s-60s)
- Regular structures for two-level logic (1960s-70s)
  - muxes and decoders, PLAs
- Programmable sum-of-products arrays (1970s-80s)
  - PLDs, complex PLDs
- Programmable gate arrays (1980s-90s)
  - densities high enough to permit entirely new class of application, e.g., prototyping, emulation, acceleration

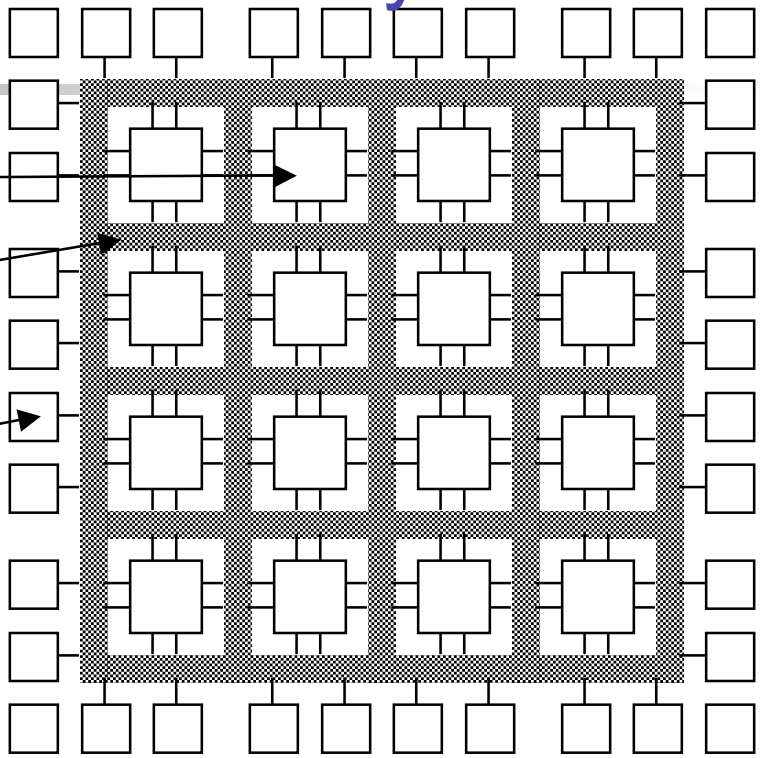
trend toward higher levels of integration

Gate Array Technology (IBM - 1970s)

- Simple logic gates
  - combine transistors to implement combinational and sequential logic
- Interconnect
  - wires to connect inputs and outputs to logic blocks
- I/O blocks
  - special blocks at periphery for external connections
- Add wires to make connections
  - done when chip is fabbed
    - “mask-programmable”
  - construct any circuit

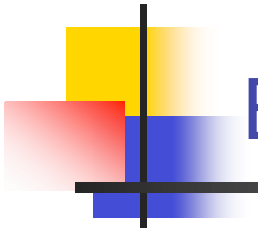


Field-Programmable Gate Arrays



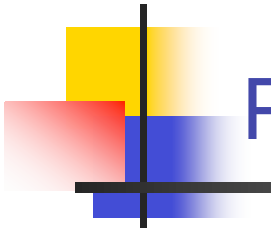
- Logic blocks
  - to implement combinational and sequential logic
- Interconnect
  - wires to connect inputs and outputs to logic blocks
- I/O blocks
  - special logic blocks at periphery of device for external connections

- Key questions:
  - how to make logic blocks programmable?
  - how to connect the wires?
  - *after the chip has been fabbed*



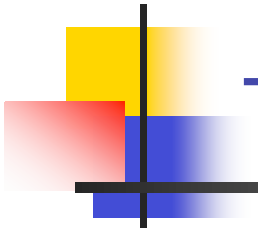
Enabling Technology

- Cheap/fast fuse connections
  - small area (can fit lots of them)
  - low resistance wires (fast even if in multiple segments)
  - very high resistance when not connected
  - small capacitance (wires can be longer)
- Pass transistors (switches)
  - used to connect wires
  - bi-directional
- Multiplexors
  - used to connect one of a set of possible sources to input
  - can be used to implement logic functions



Programming Technologies

- Fuse and anti-fuse
  - fuse makes or breaks link between two wires
  - typical connections are 50-300 ohm
  - one-time programmable (testing before programming?)
- EPROM and EEPROM
  - high power consumption
  - typical connections are 2K-4K ohm
  - fairly low density
- RAM-based
  - memory bit controls a switch that connects/disconnects two wires
  - typical connections are .5K-1K ohm
  - can be programmed and re-programmed easily (tested at factory)

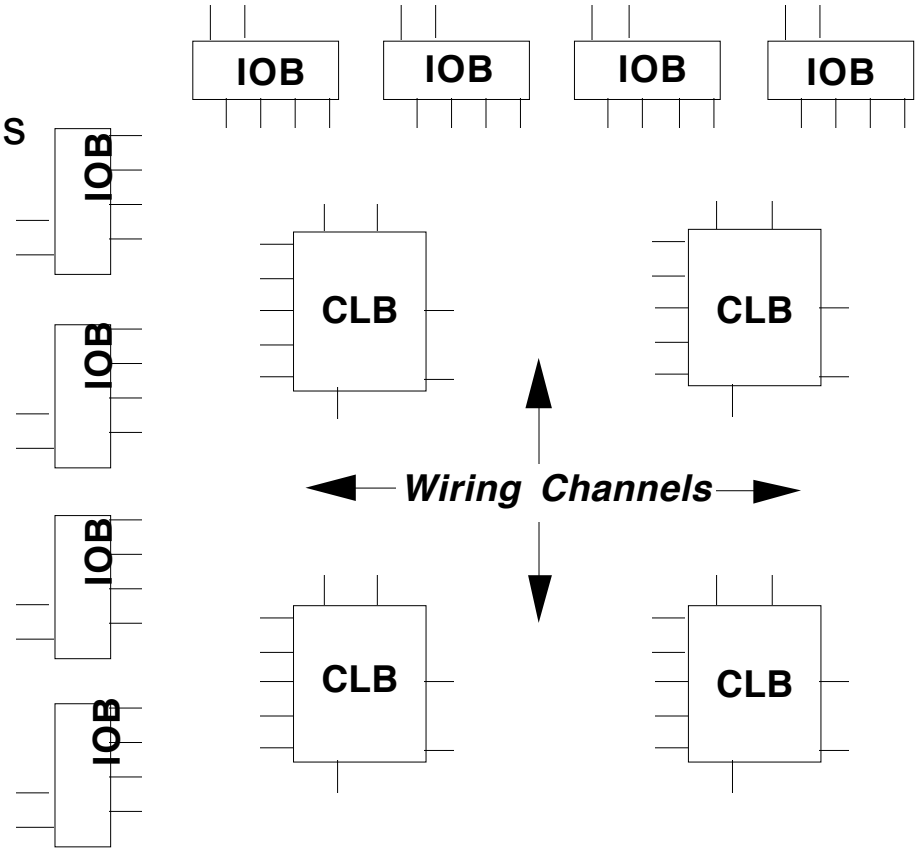


Tradeoffs in FPGAs

- Logic block - how are functions implemented: fixed functions (manipulate inputs) or programmable?
  - support complex functions, need fewer blocks, but they are bigger so less of them on chip
  - support simple functions, need more blocks, but they are smaller so more of them on chip
- Interconnect
  - how are logic blocks arranged?
  - how many wires will be needed between them?
  - are wires evenly distributed across chip?
  - programmability slows wires down – are some wires specialized to long distances?
  - how many inputs/outputs must be routed to/from each logic block?
  - what utilization are we willing to accept? 50%? 20%? 90%?

Xilinx Programmable Gate Arrays

- CLB - Configurable Logic Block
  - 5-input, 1 output function
  - or 2 4-input, 1 output functions
  - optional register on outputs
- Built-in fast carry logic
- Can be used as memory
- Three types of routing
  - direct
  - general-purpose
  - long lines of various lengths
- RAM-programmable
  - can be reconfigured



The Xilinx 4000 CLB

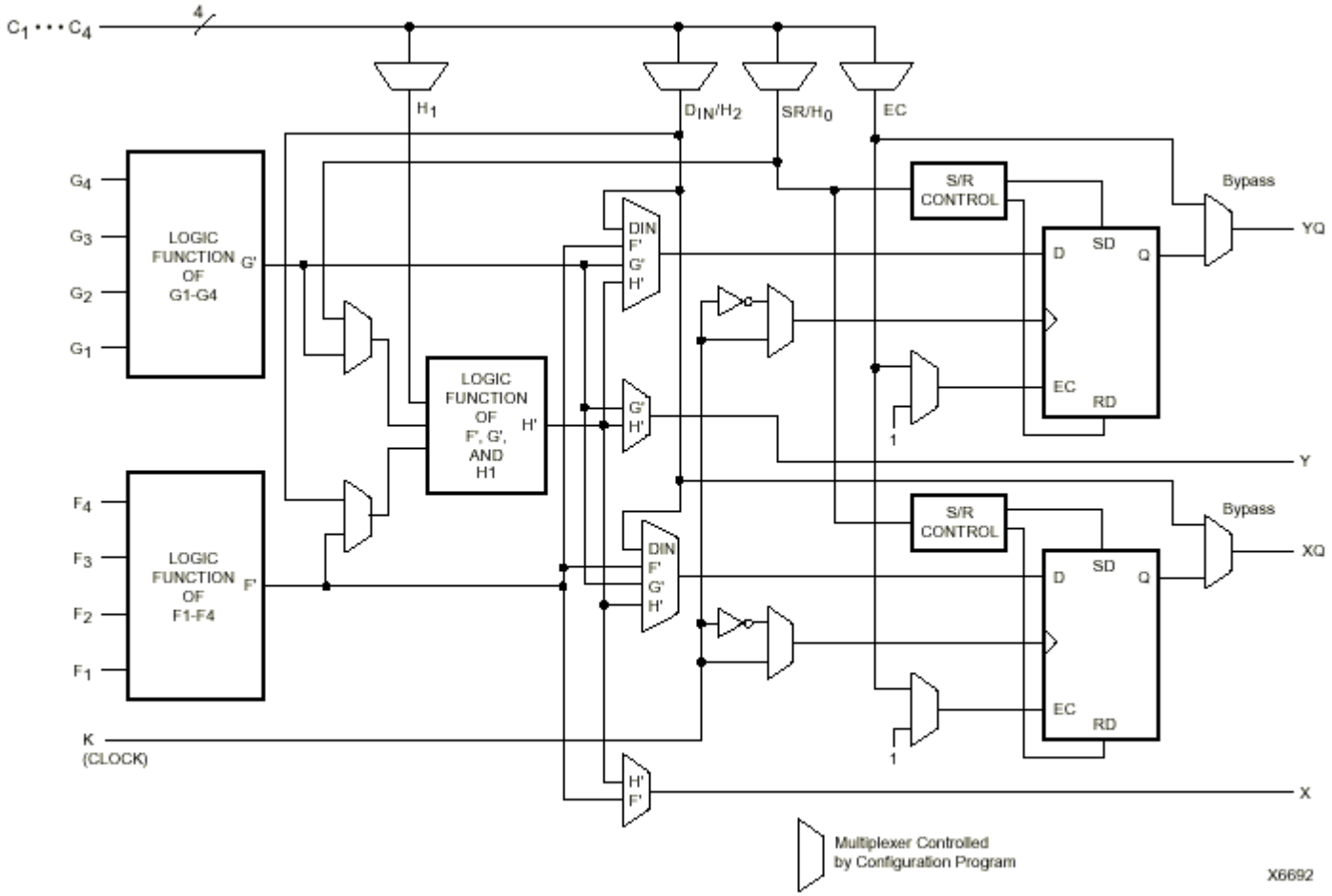


Figure 1: Simplified Block Diagram of XC4000 Series CLB (RAM and Carry Logic functions not shown)

Two 4-input functions, registered output

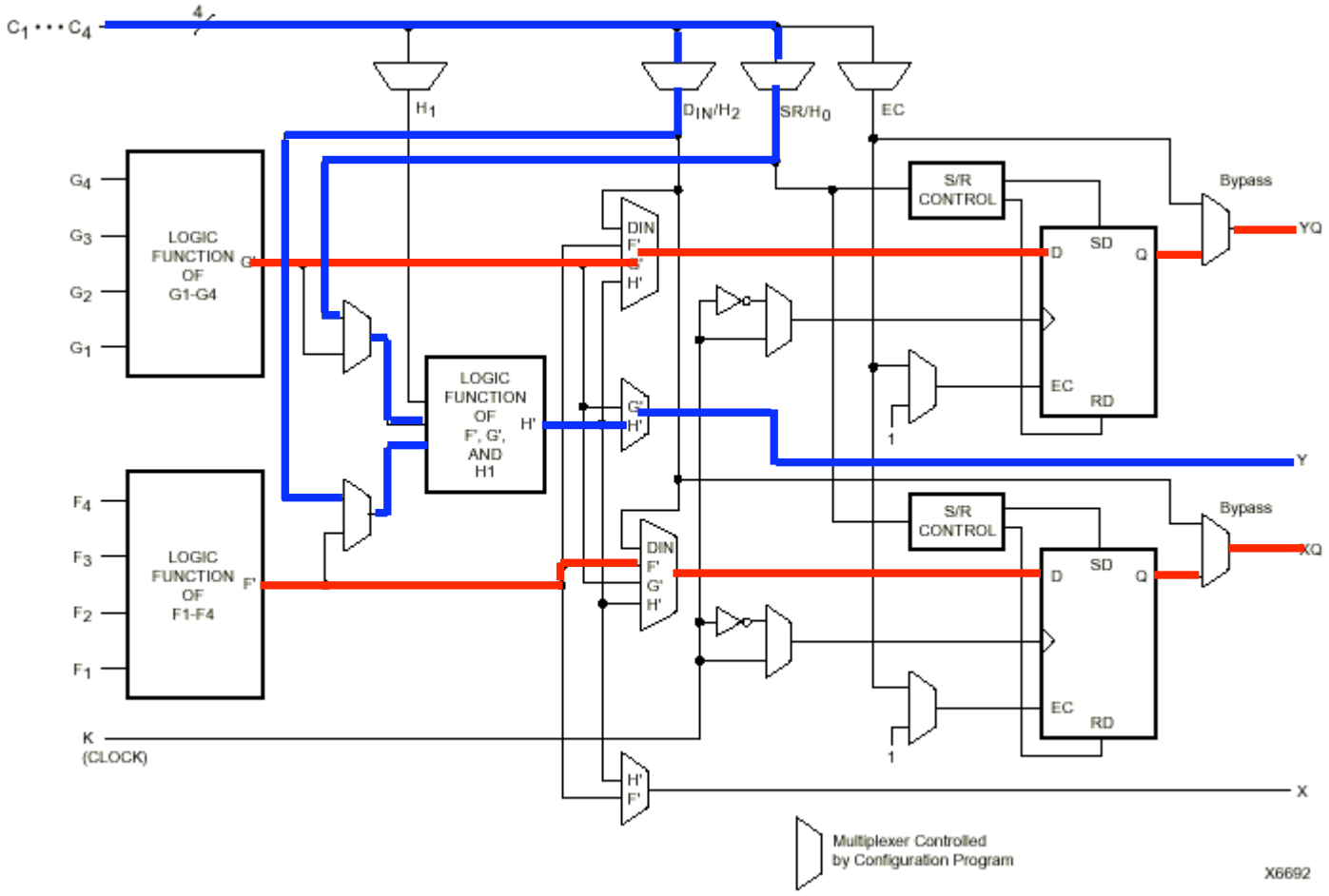


Figure 1: Simplified Block Diagram of XC4000 Series CLB (RAM and Carry Logic functions not shown)

5-input function, combinational output

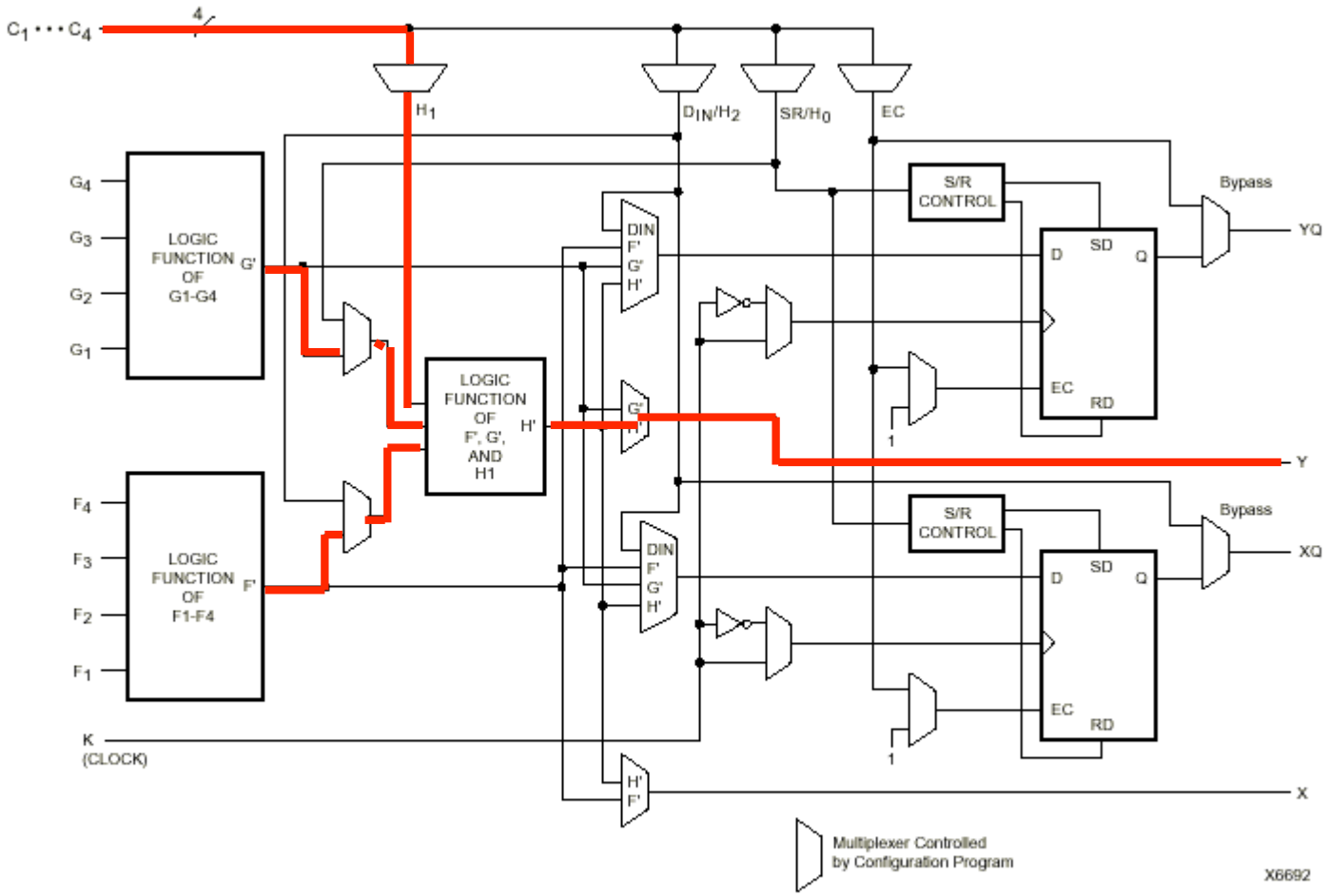


Figure 1: Simplified Block Diagram of XC4000 Series CLB (RAM and Carry Logic functions not shown)

CLB Used as RAM

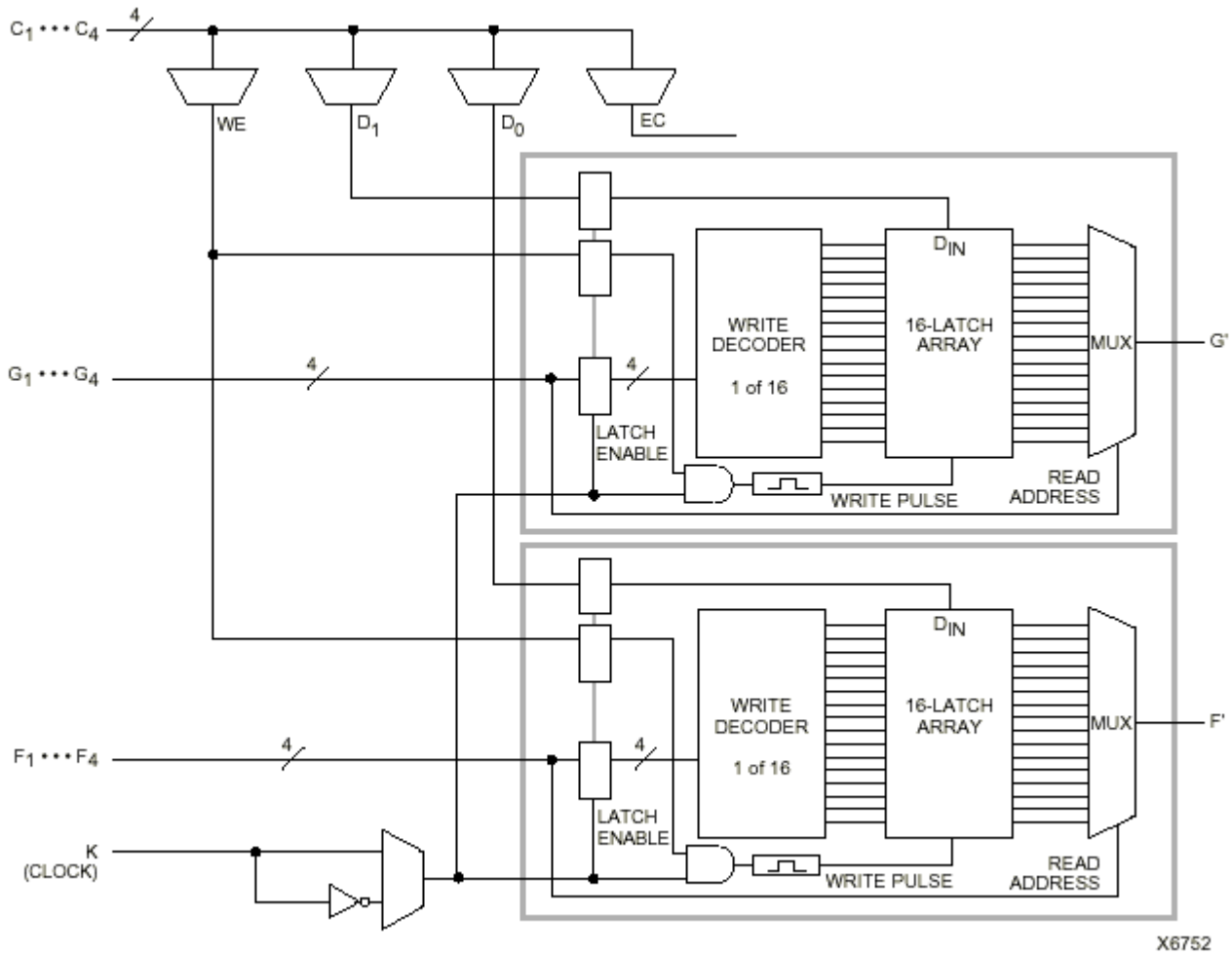
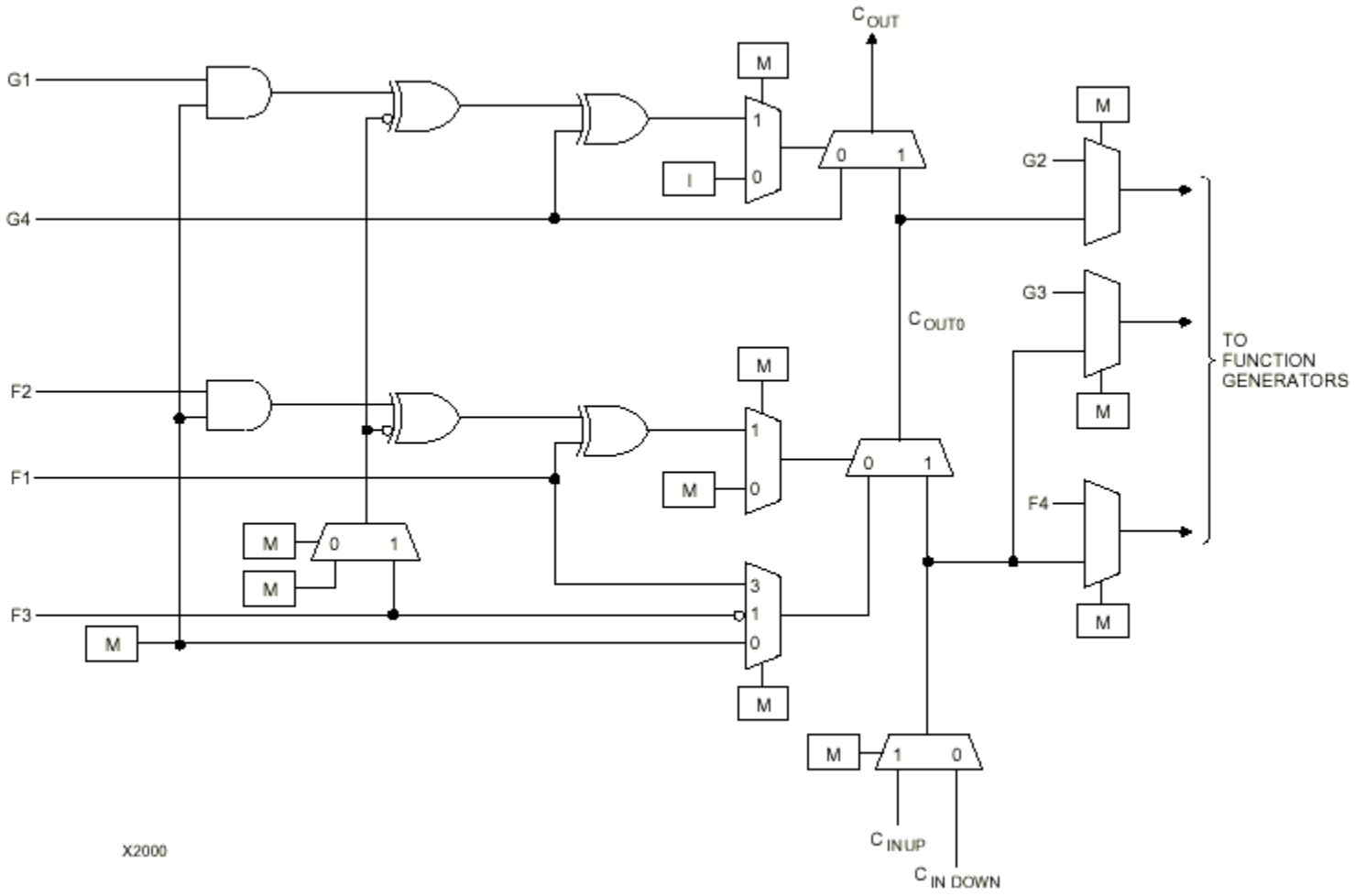


Figure 4: 16x2 (or 16x1) Edge-Triggered Single-Port RAM

Fast Carry Logic



Xilinx 4000 Interconnect

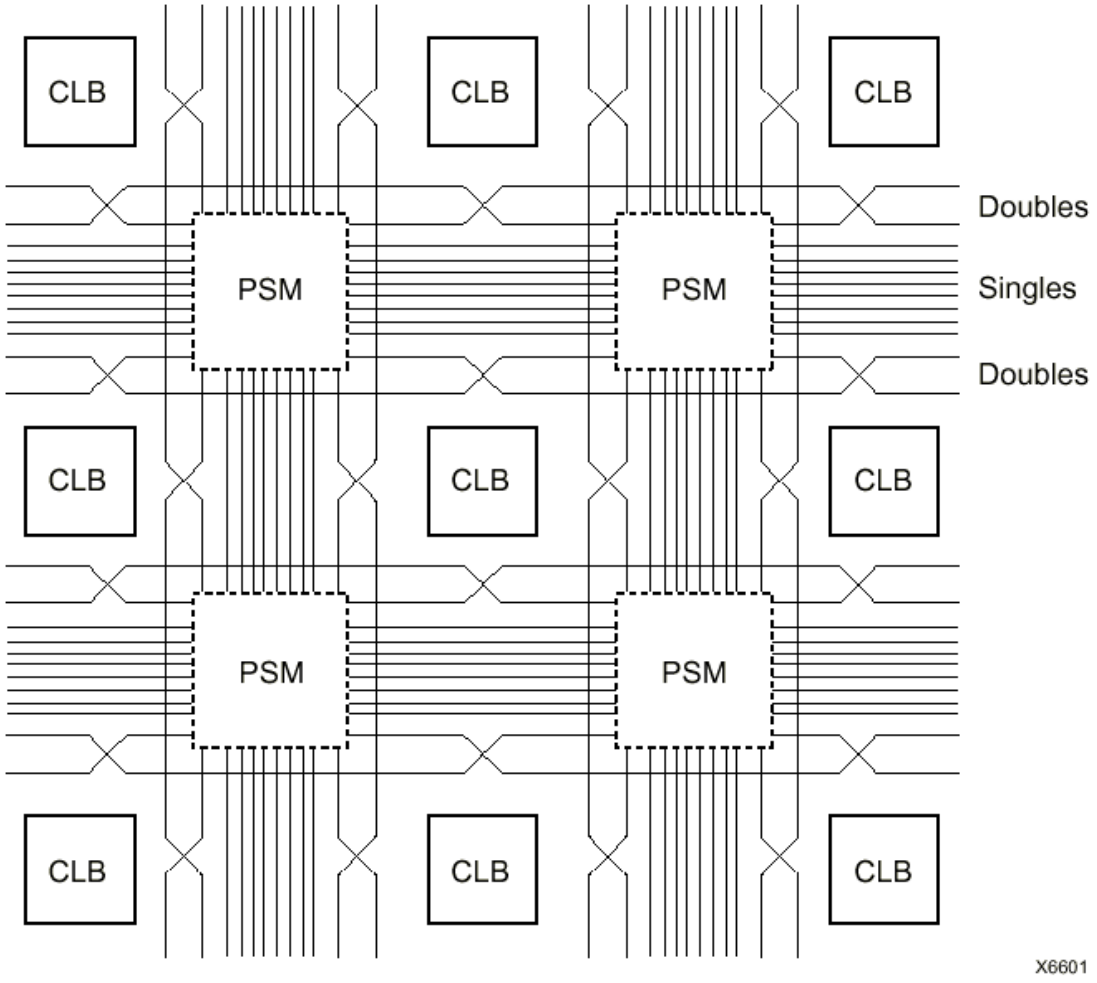


Figure 28: Single- and Double-Length Lines, with Programmable Switch Matrices (PSMs)

Switch Matrix

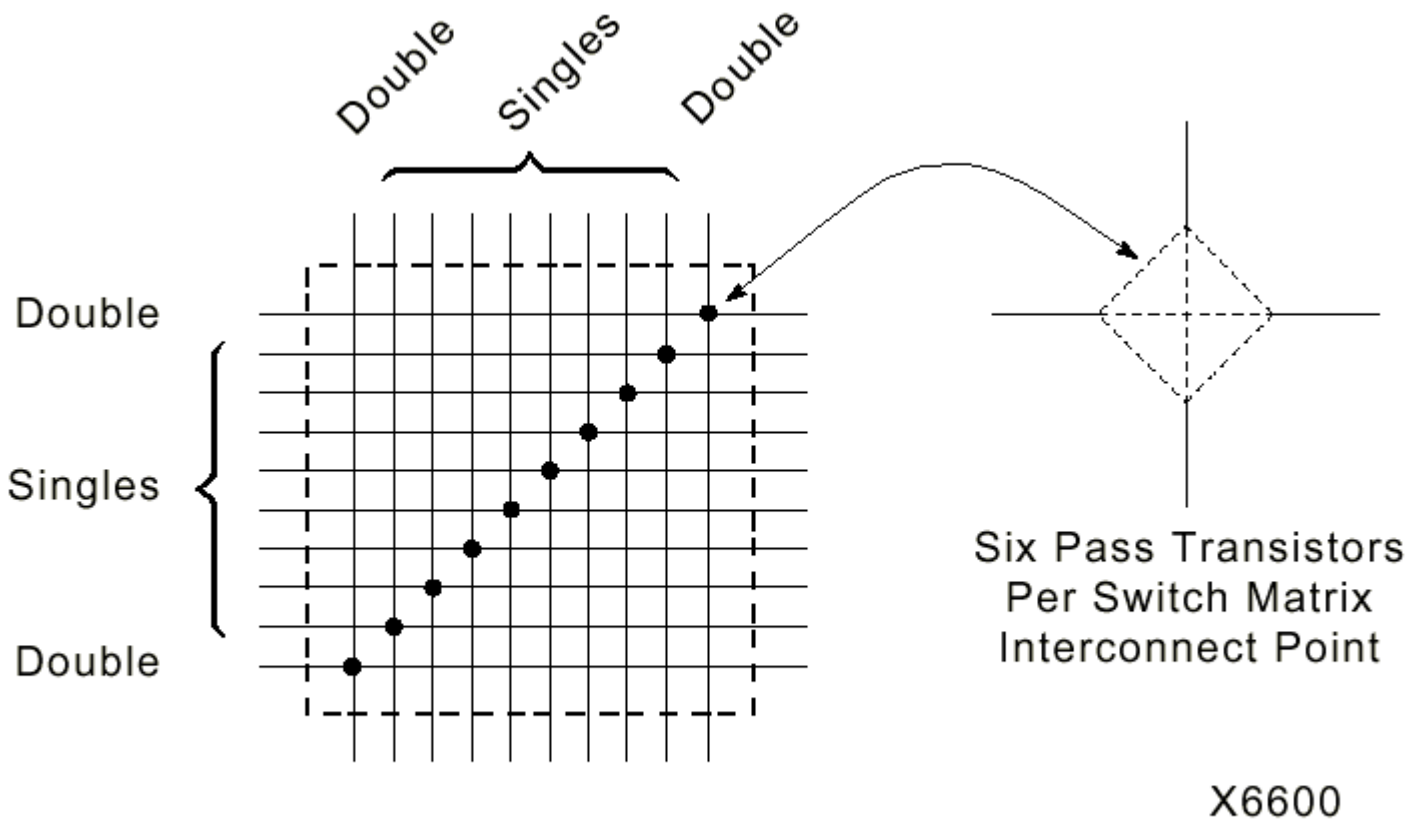
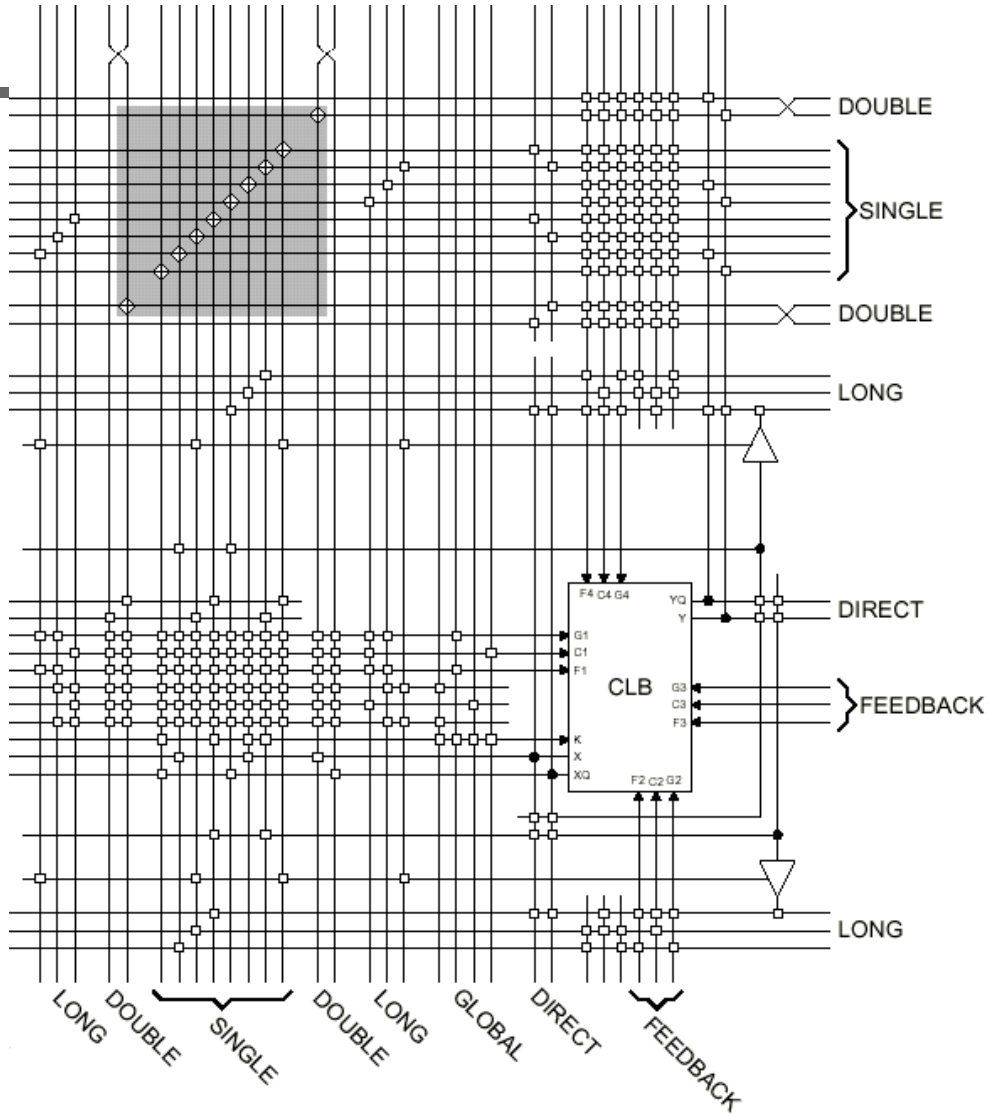
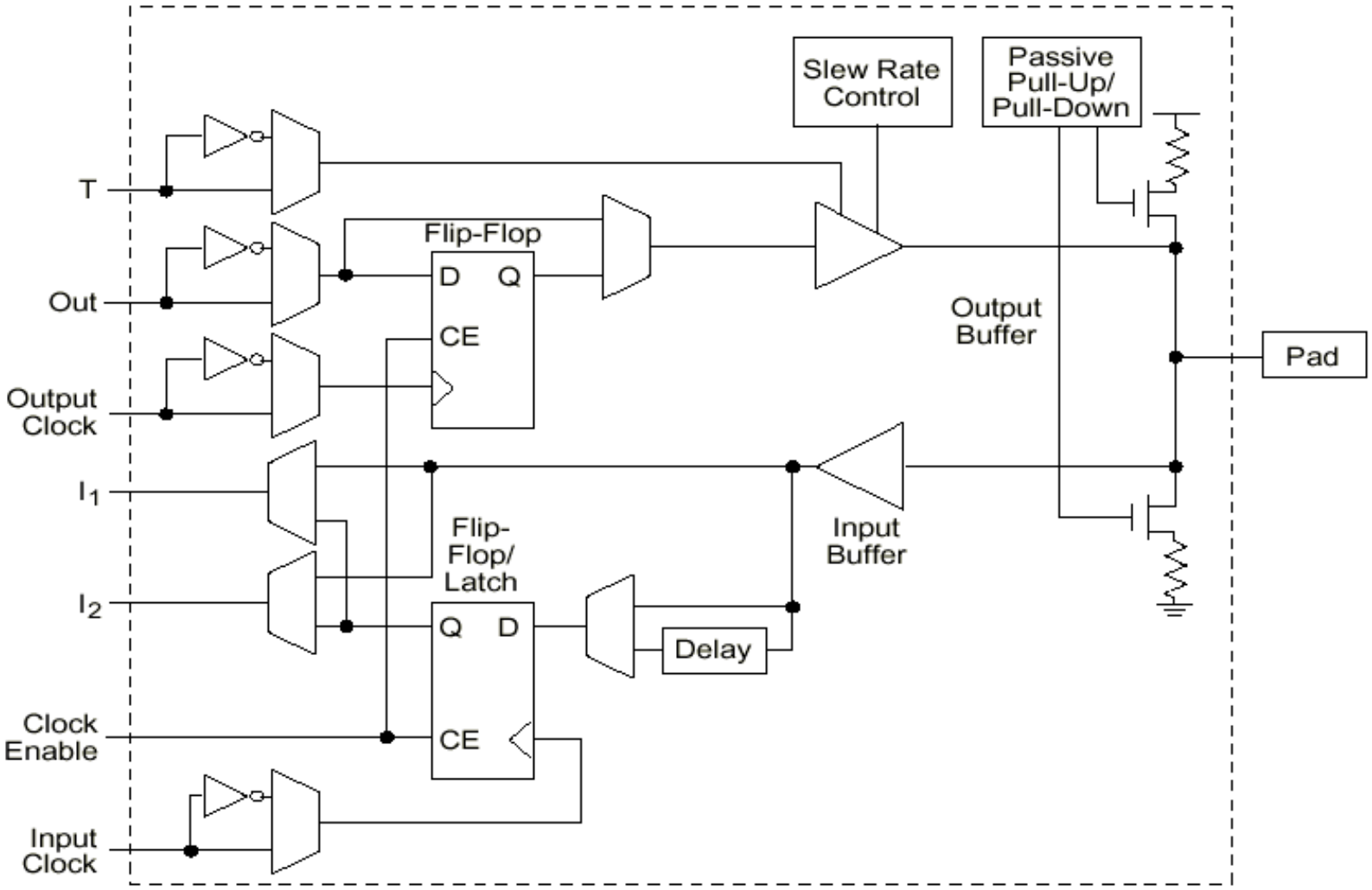


Figure 26: Programmable Switch Matrix (PSM)

Xilinx 4000 Interconnect Details



Xilinx 4000 IOB



Xilinx FPGA Combinational Logic Examples

- Key: Functions are limited to 5 inputs (4 even better)
  - No limitation on function complexity
- Examples:
  - 5-input parity generator implemented with 1 CLB

```
F = (A xor B xor C xor D xor E)'
```

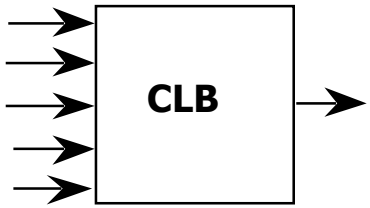
  - 2-bit comparator: A B = C D or A B > C D implemented with 1 CLB

```
(GT)   F = A C' +  A B D' +  B C' D'
(EQ)   G = A' B' C' D'  +  A' B C' D
           + A  B' C  D'  +  A  B C  D
```

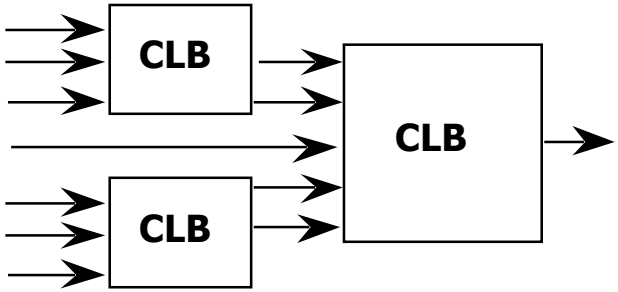
Xilinx FPGA Combinational Logic

- Examples
  - n-input majority function: 1 whenever n/2 or more inputs are 1

5-input Majority Circuit

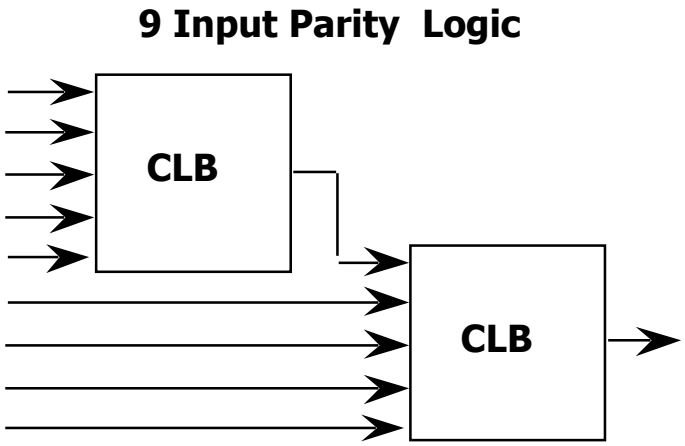


7-input Majority Circuit



Xilinx FPGA Combinational Logic

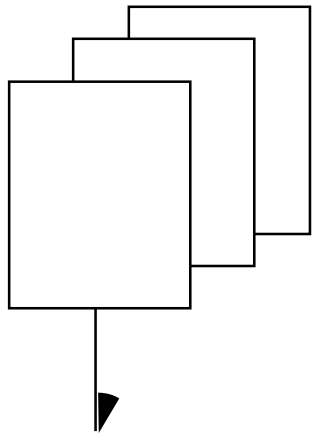
- Examples
  - n-input parity function: 5 input = 1 CLB, 2 levels yield up to 25 inputs



Xilinx FPGA Adder Example

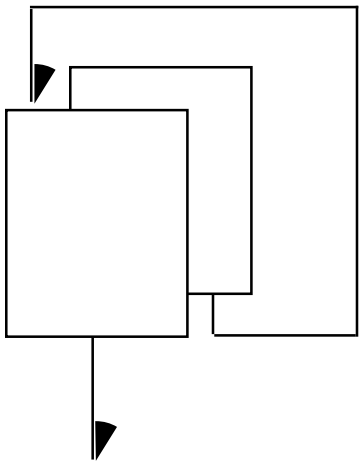
- Example
  - 2-bit binary adder - inputs: A1, A0, B1, B0, CIN
    outputs: S0, S1, Cout

3 CLB solution
single CLB delay for Cout of second bit

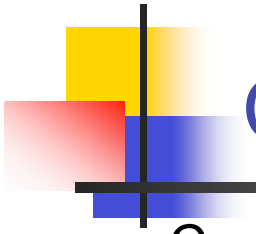


Cout

2 CLB solution
double CLB delay for Cout of second bit

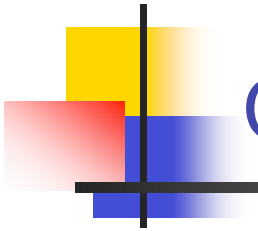


Cout



Computer-aided Design

- Can't design FPGAs by hand
  - way too much logic to manage, hard to make changes
- Hardware description languages
  - specify functionality of logic at a high level
- Validation - high-level simulation to catch specification errors
  - verify pin-outs and connections to other system components
  - low-level to verify mapping and check performance
- Logic synthesis
  - process of compiling HDL program into logic gates and flip-flops
- Technology mapping
  - map the logic onto elements available in the implementation technology (LUTs for Xilinx FPGAs)



CAD Tool Path (cont'd)

- Placement and routing
  - assign logic blocks to functions
  - make wiring connections
- Timing analysis - verify paths
  - determine delays as routed
  - look at critical paths and ways to improve
- Partitioning and constraining
  - if design does not fit or is unroutable as placed split into multiple chips
  - if design it too slow prioritize critical paths, fix placement of cells, etc.
  - few tools to help with these tasks exist today
- Generate programming files - bits to be loaded into chip for configuration