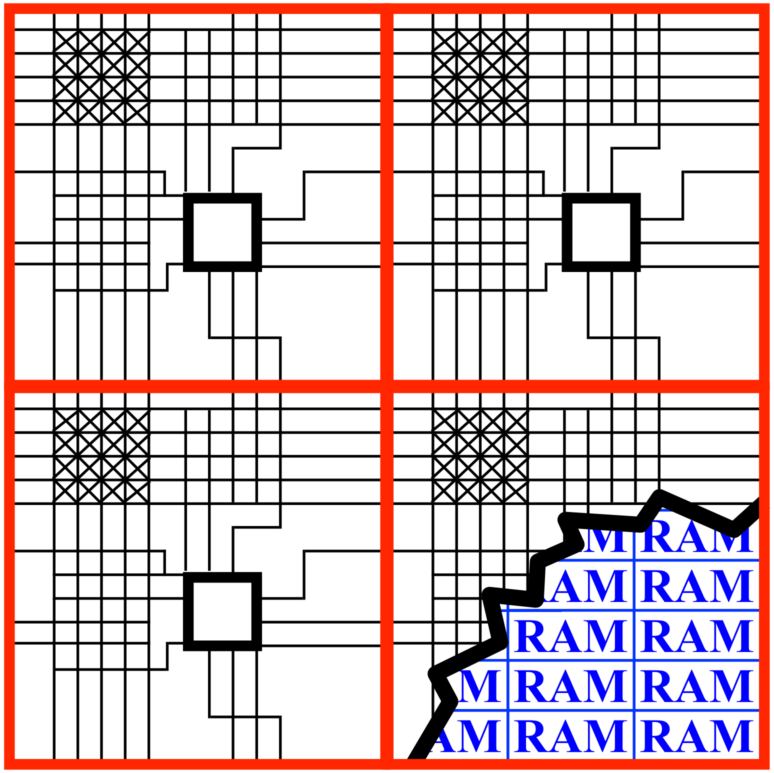
Field Programmable Gate Arrays (FPGAs)

- Readings: B.6-B.6.5



Logic cells imbedded in a general routing structure

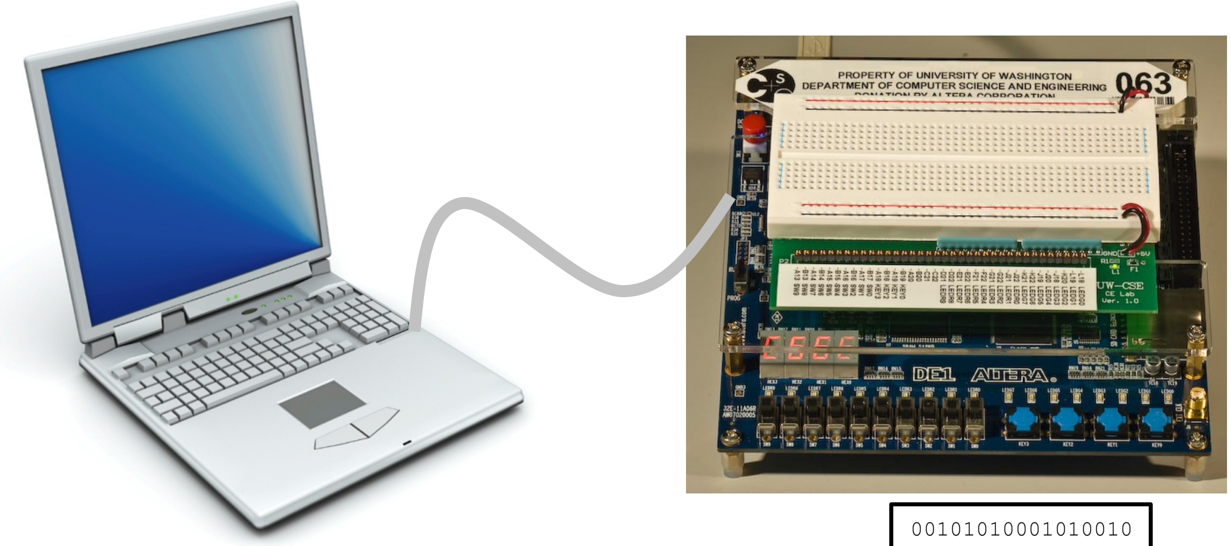
□

Logic cells usually contain:

- **6-input Boolean function calculator**
- **Flip-flop (1-bit memory)**

All features electronically (re)programmable

Using an FPGA


PROPERTY OF UNIVERSITY OF WASHINGTON
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
063
UW-CSE
CE Lab
Ver. 1.0
DE1
ALTERA

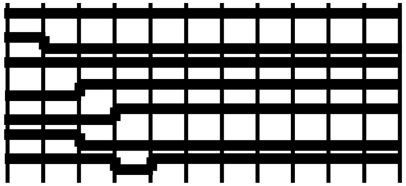
```
// Verilog code for 2-input
multiplexer

module AOI (F, A, B, C, D);
  output F;
  input A, B, C, D;

  assign F = ~((A & B) | (C & D));
endmodule

module MUX2 (V, SEL, I, J);    //
2:1 multiplexer
  output V;
  input SEL, I, J;
  wire SELB, VB;

  not G1 (SELB, SEL);
  AOI G2 (VB, I, SEL, SELB, J);
  not G3 (V, VB);
endmodule
```

Verilog

FPGA
CAD
Tools

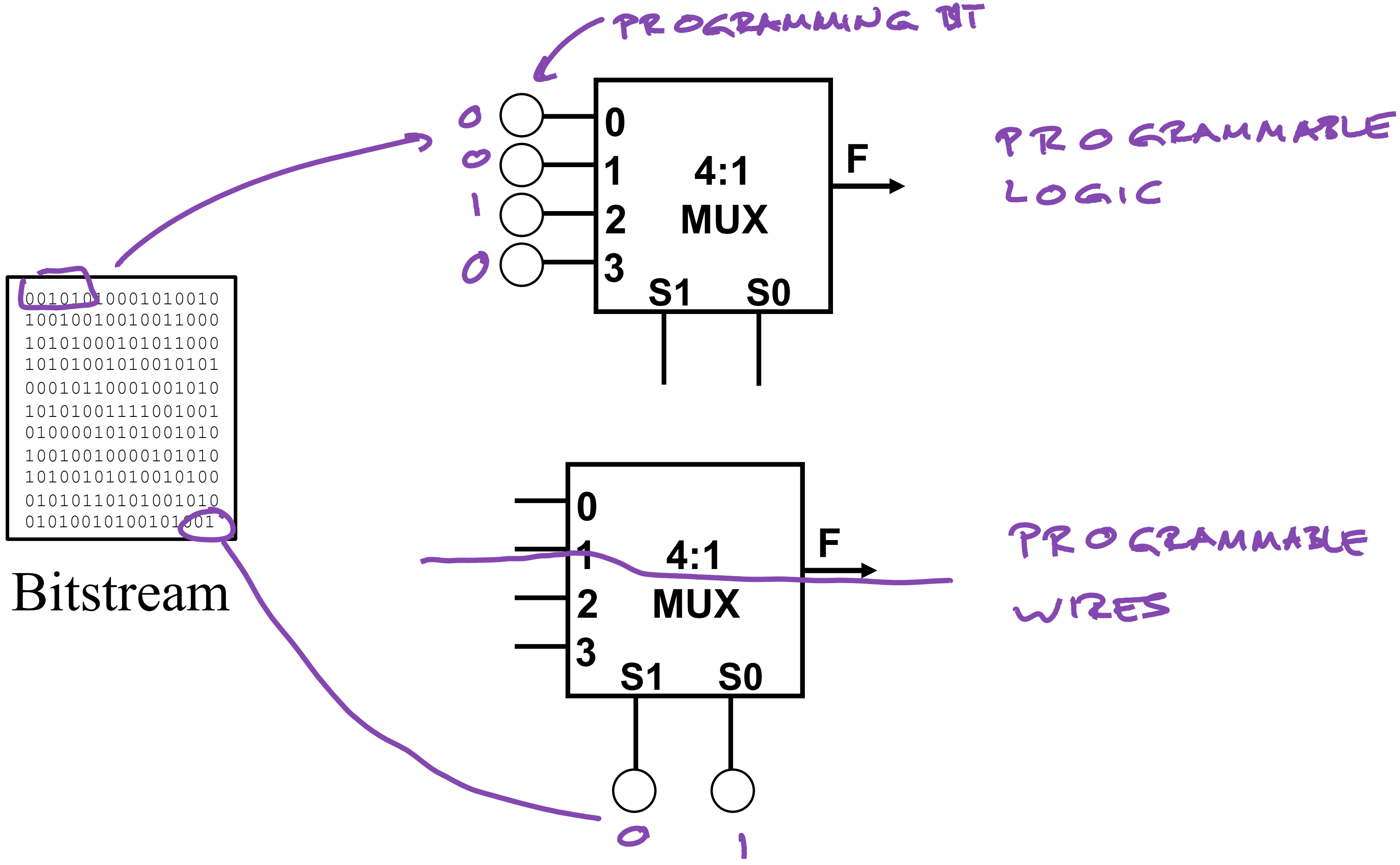
```
00101010001010010
10010010010011000
10101000101011000
10101001010010101
00010110001001010
10101001111001001
01000010101001010
10010010000101010
10100101010010100
01010110101001010
01010010100101001
```

Bitstream



Simulation

FPGA Programming



```
0010101000101010010
1001001001001011000
1010100010101011000
1010100101010010101
0001011000100101010
1010100111110010001
0100000101010010101
1001001000001010100
1010010101010010100
0101011010101001010
0101001010010101001
```

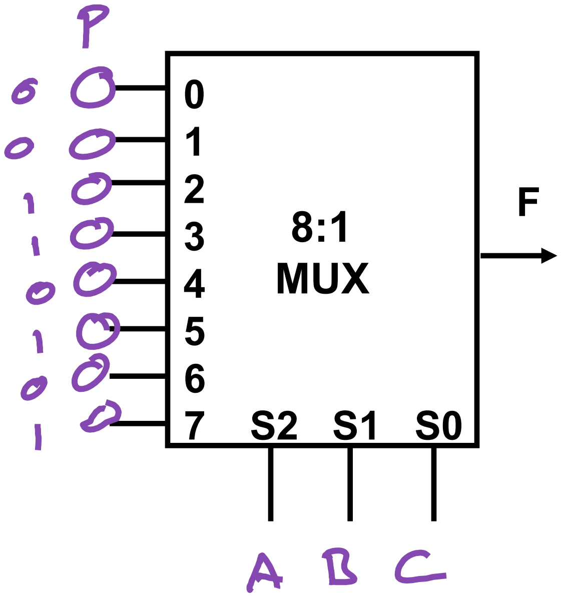
Bitstream

○ = 1 memory cell (stores 1 bit of info)

FPGA Combinational Logic

- How can we use Muxes and Programming bits to compute combinational binary function F(A,B,C)?

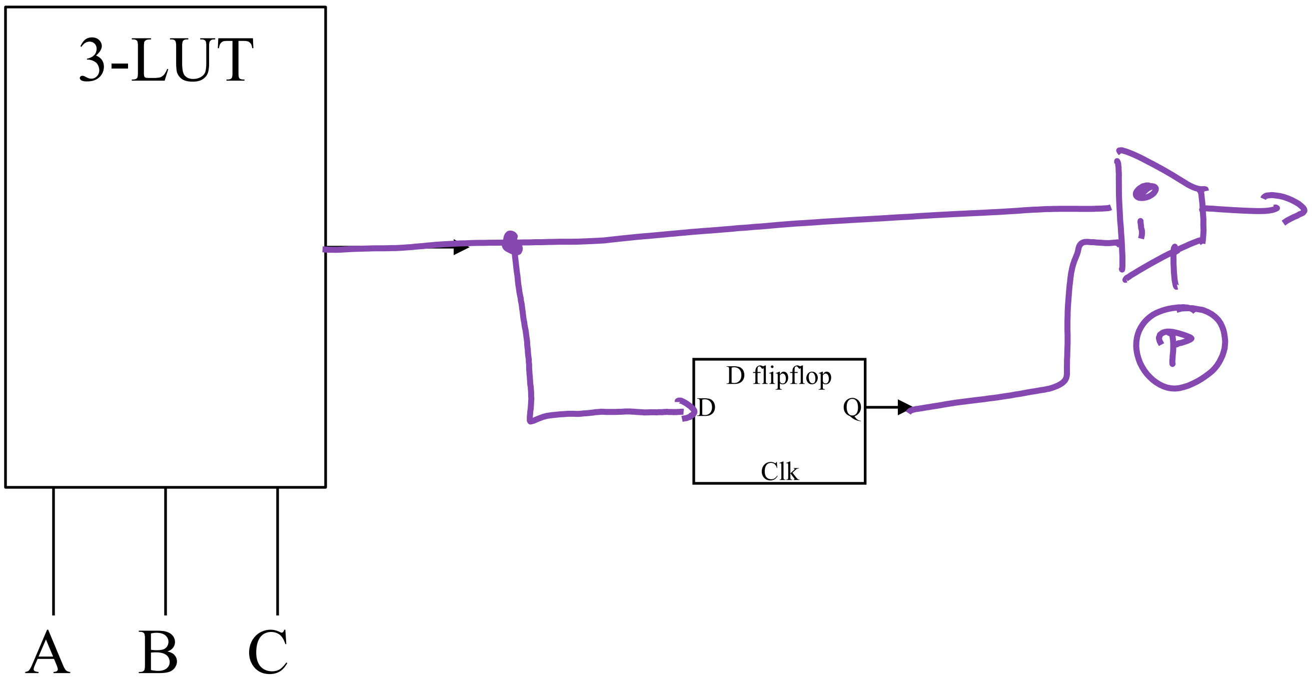
ABC	F
	0
	0
	1
	1
	0
	1
	0
	1



- Creates a "LUT" or lookup table.

FPGA Sequential Logic

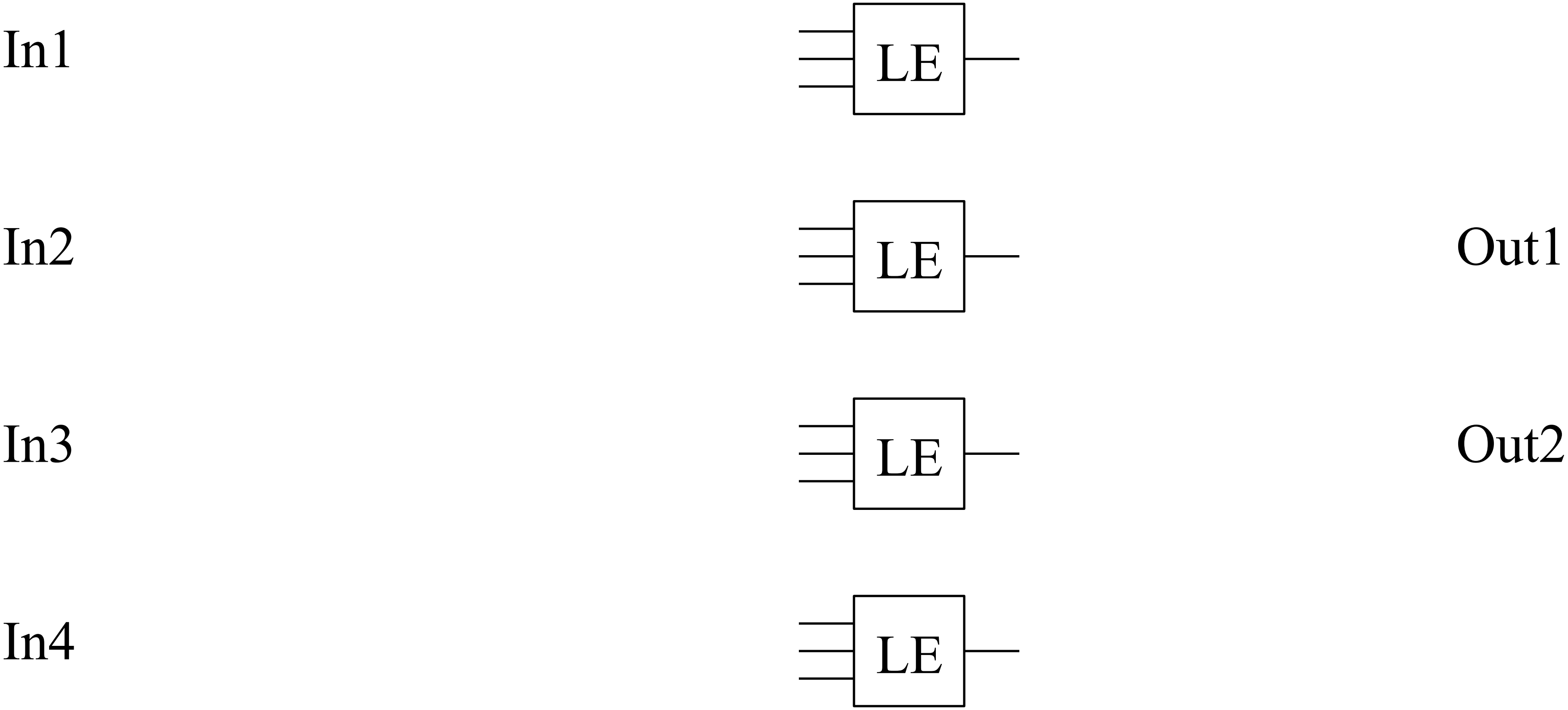
- How do we put DFF’s onto LUT outputs only when we need them?



- Creates a “LE” or logic block

FPGA Local Routing

- How do we combine LE’s to build larger functions?



In1

In2

In3

In4

LE

LE

LE

LE

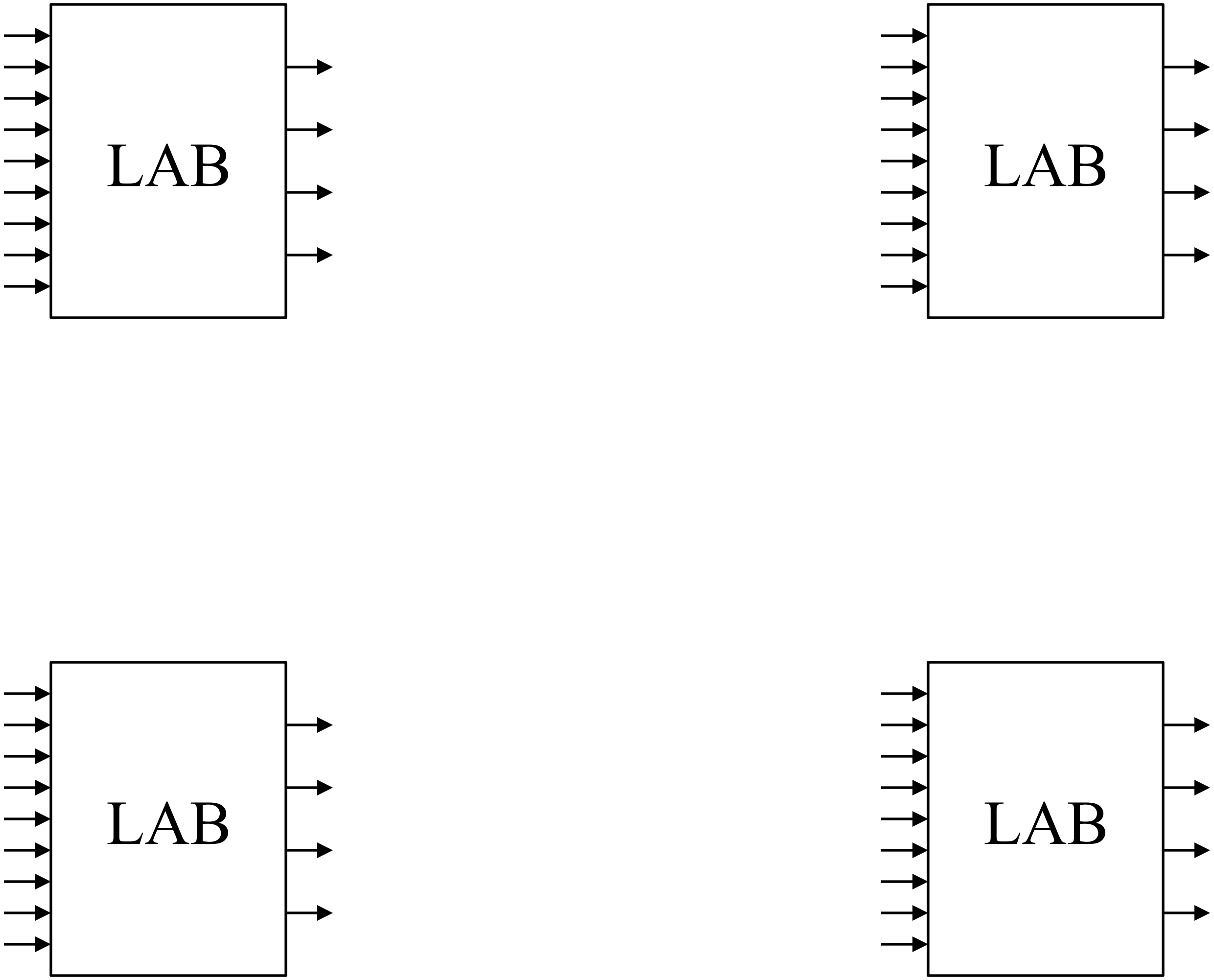
Out1

Out2

- This is an Altera “LAB”.

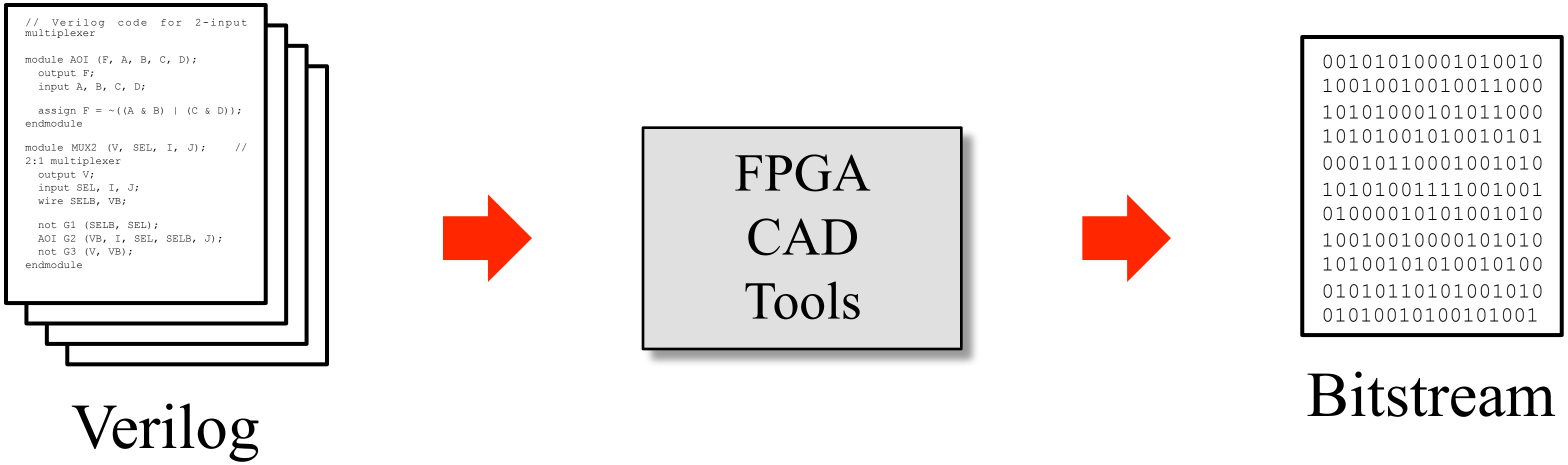
FPGA Global Routing

- Can’t do all-to-all/crossbar routing, so what?



FPGA CAD

- CAD = “Computer-Aided Design”



```
// Verilog code for 2-input
multiplexer

module AOI (F, A, B, C, D);
  output F;
  input A, B, C, D;

  assign F = ~((A & B) | (C & D));
endmodule

module MUX2 (V, SEL, I, J);    //
2:1 multiplexer
  output V;
  input SEL, I, J;
  wire SELB, VB;

  not G1 (SELB, SEL);
  AOI G2 (VB, I, SEL, SELB, J);
  not G3 (V, VB);
endmodule
```

Verilog

FPGA CAD Tools

```
00101010001010010
10010010010011000
10101000101011000
10101001010010101
00010110001001010
10101001111001001
01000010101001010
10010010000101010
10100101010010100
01010110101001010
0101001010010100 1
```

Bitstream

- Tech Mapping: Convert Verilog to LUTs
- Placement: Assign LUTs to specific locations
- Routing: Wire inputs to outputs
- Bitstream Generation: Convert mapping to bits