FPGA

FIELD PROGRAMMABLE GATE ARRAY

CONFIGURABLE BLOCKS OF DIG. LOGIC

&

CONFIGURABLE INTERCONNECTS

CAN BE PROGRAMMED / REPROGRAMMED MANY TIMES

TYPES OF PROGRAMMABLE DIGITAL LOGIC

SPLD — SIMPLE PROGRAMMABLE LOGIC DEVICE

CPLD — COMPLEX PROG. LOGIC DEVICE

→ HIGHLY CONFIGURABLE, W/ FAST DESIGN & MOD. TIMES, BUT LIMITED FUNCTIONALITY

ASIC — APPLICATION SPECIFIC INTEGRATED CIRCUIT

HIGH DEGREE OF FUNCTIONALITY, BUT NOT CONFIGURABLE

FPGA [XILINX 1984 ... CATCH ON IN '90s]

ARRAY OF

→ CONFIGURABLE LOGIC BLOCKS [CLBs]

PROGRAMMABLE INPUT/OUTPUT BLOCKS [IOB]

BLOCK RAM

DLL TO SYNCHRONIZE FUNCTIONAL ELEMENTS

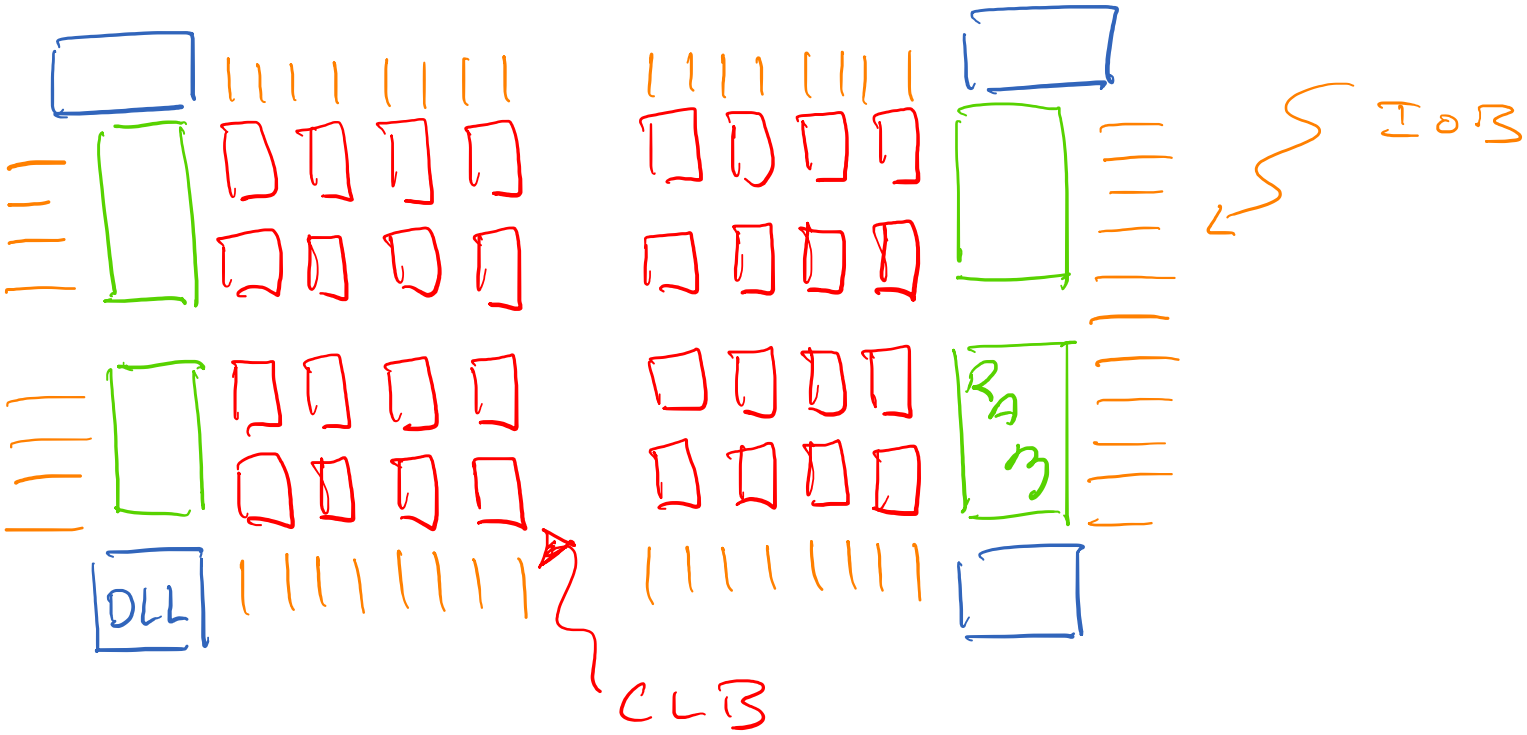
ALL LINKED BY PROGRAMMABLE INTERCONNECT FABRIC

CLB

IOB

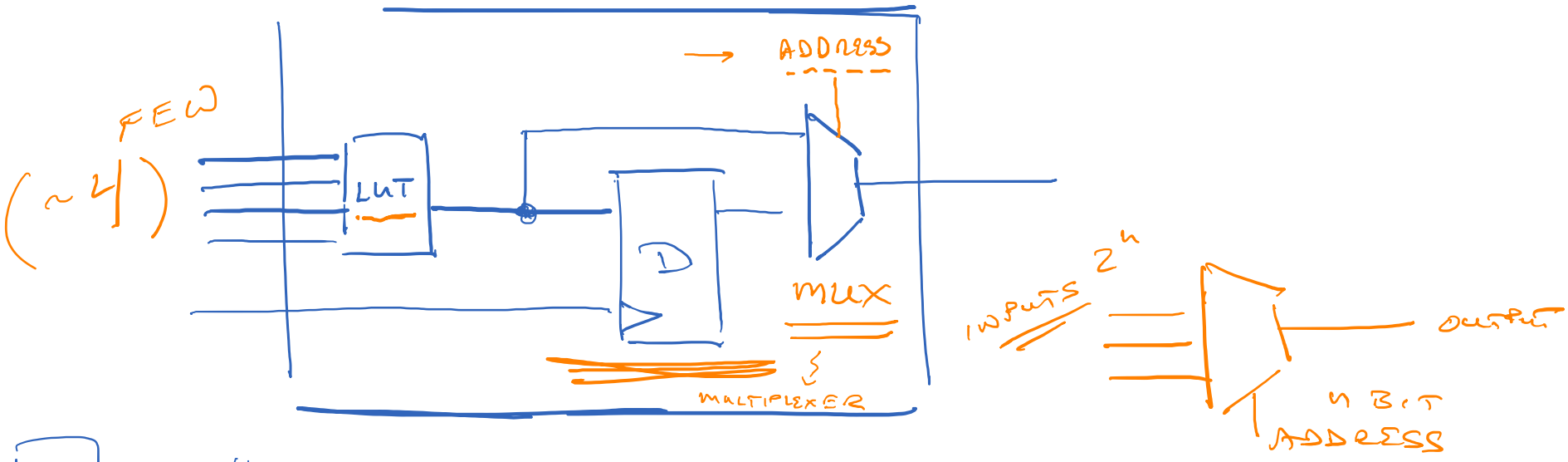
RAM

DLL

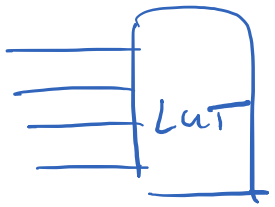


INSIDE CLB's

→ "LOGIC CELL" [XILINX] "LOGIC ELEMENT" [ALTERA]



LUT → "LOOK UP TABLE": REPRESENTS TRUTH TABLE FOR ARBITRARY BINARY FUNCTION

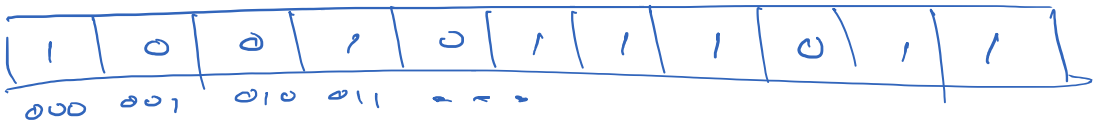


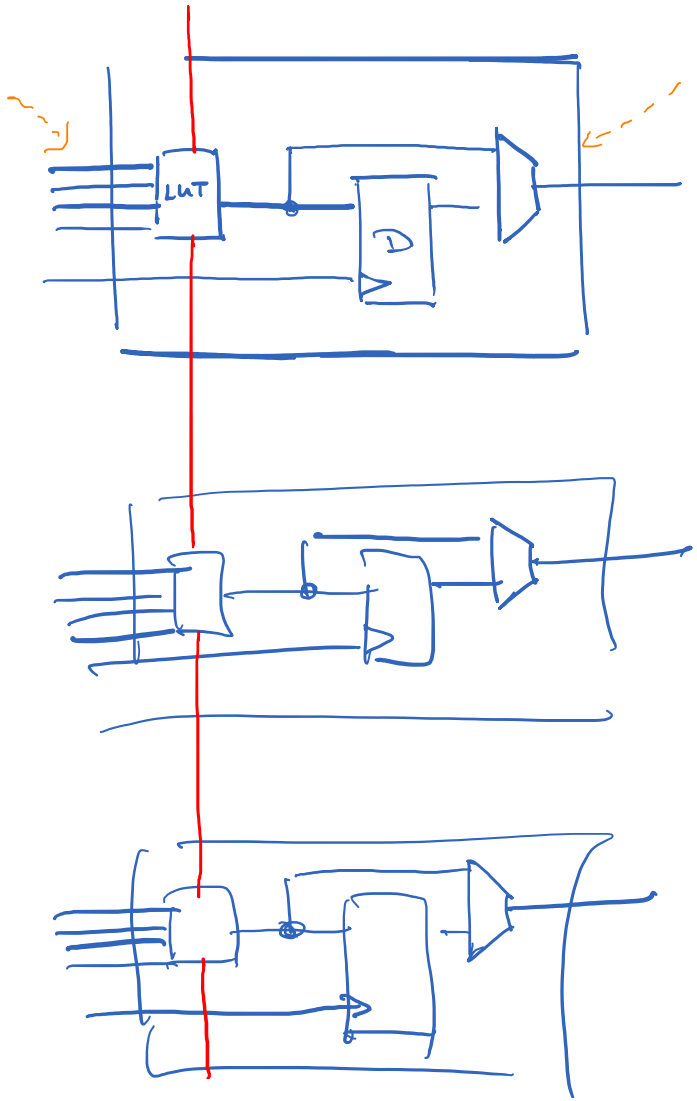
N-INPUT LUT:

2^N POSSIBLE STATES

N INPUT LUT CAN BE CONFIGURED AS $2^N \times 1$ RAM

- LUTs IN CLBs CAN BE REPURPOSED AS DISTRIBUTED RAM





FAST CARRY LOGIC

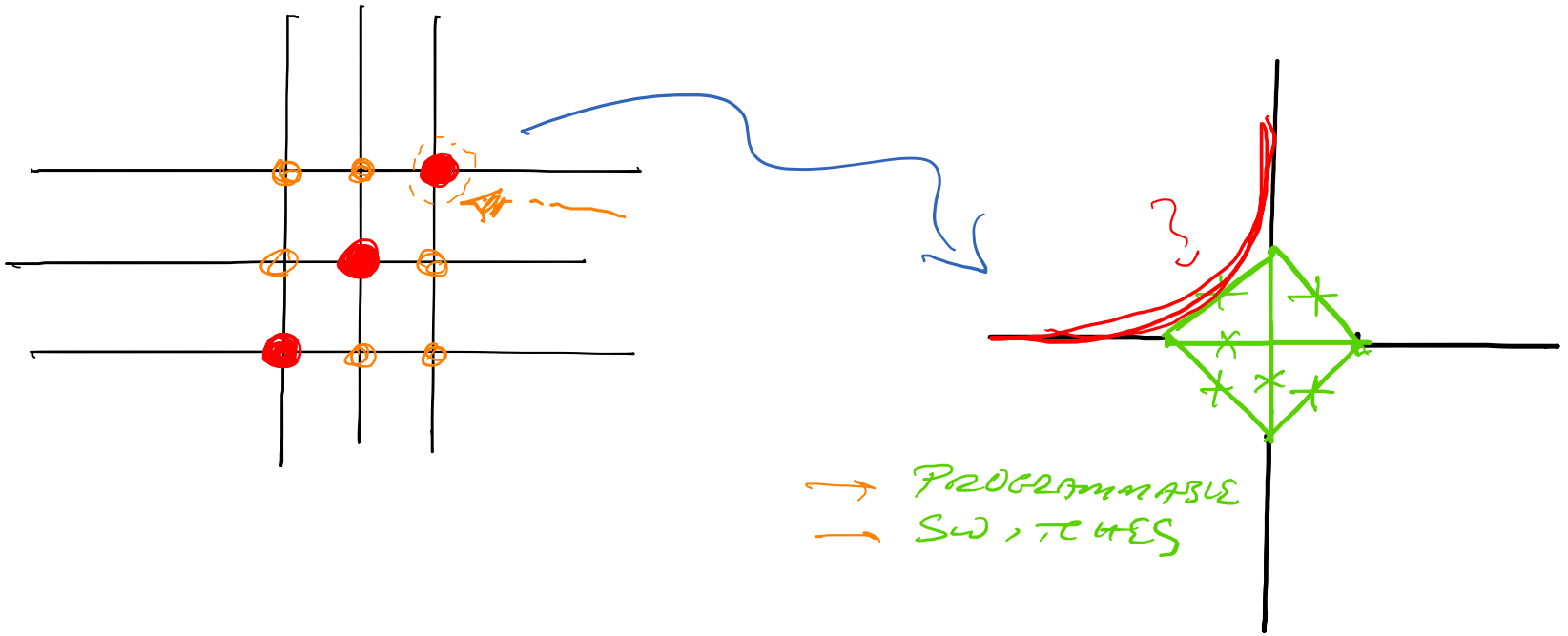
INTERCONNECTING LOGIC CELLS

ALLOWS EFFICIENT IMPLEMENTATION OF COUNTERS / ADDERS

[REDUCED HARDWARE RESOURCES & HIGHER OPERATING SPEED]

"CARRY CHAINS"

PROGRAMMABLE WIRING



XILINX

→ "SLICE" : 2 × LOGIC CELL

→ CLB : 4 × SLICE

∴ CLB INVOLVES 8 LOGIC CELLS

IF 4 INPUT LUT PER LOGIC CELL [16 × 1 RAM]

1 CLB CAN YIELD

SINGLE PORT

16 × 8
32 × 4
64 × 2
128 × 1

RAM

DEPTH

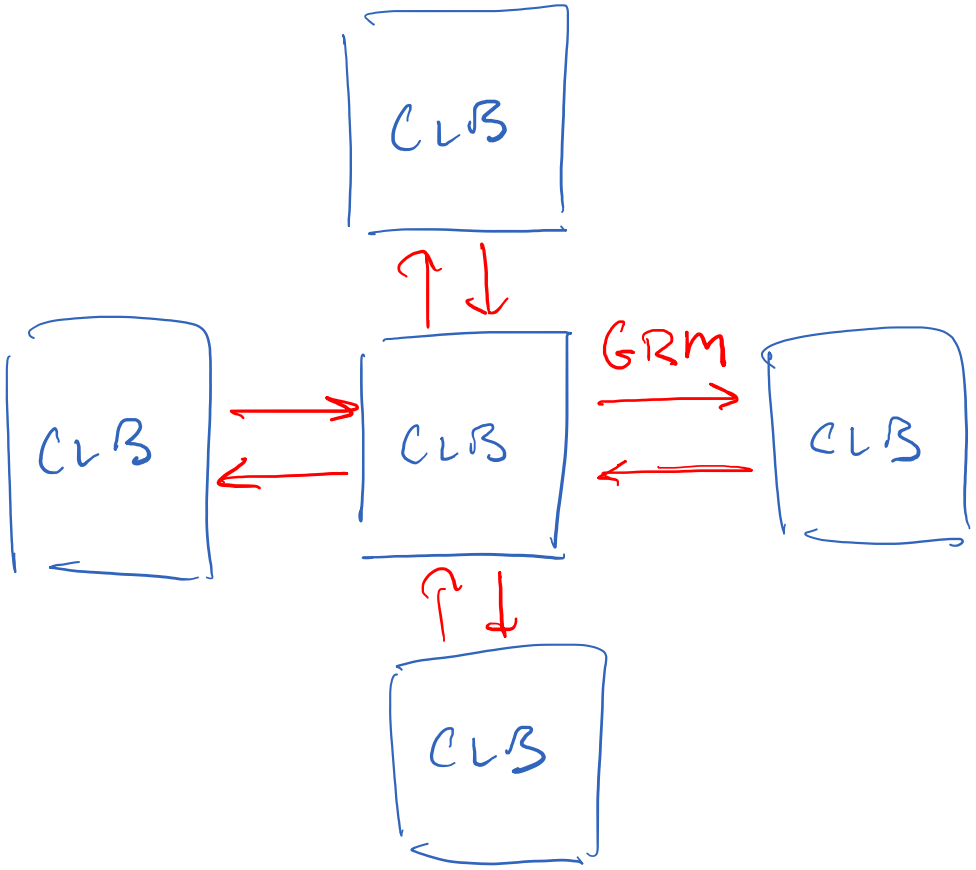
WIDTH

EMBEDDED MULTIPLIERS, ADDERS,

MACs

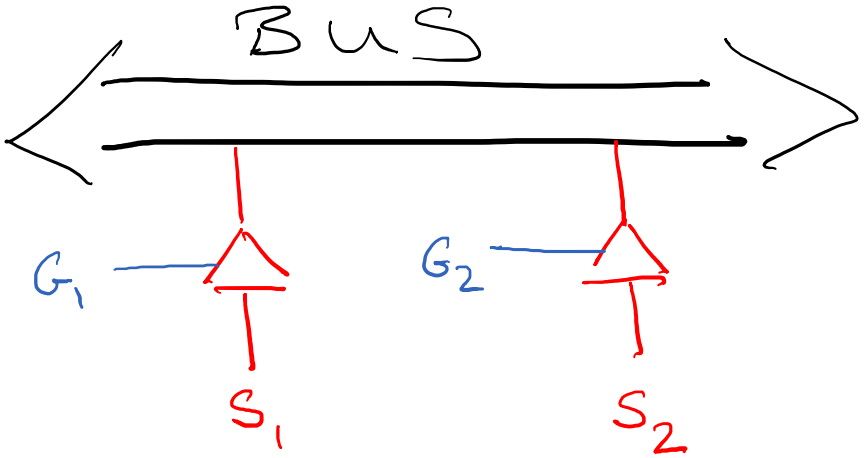
→ "MULTIPLY AND ACCUMULATE"

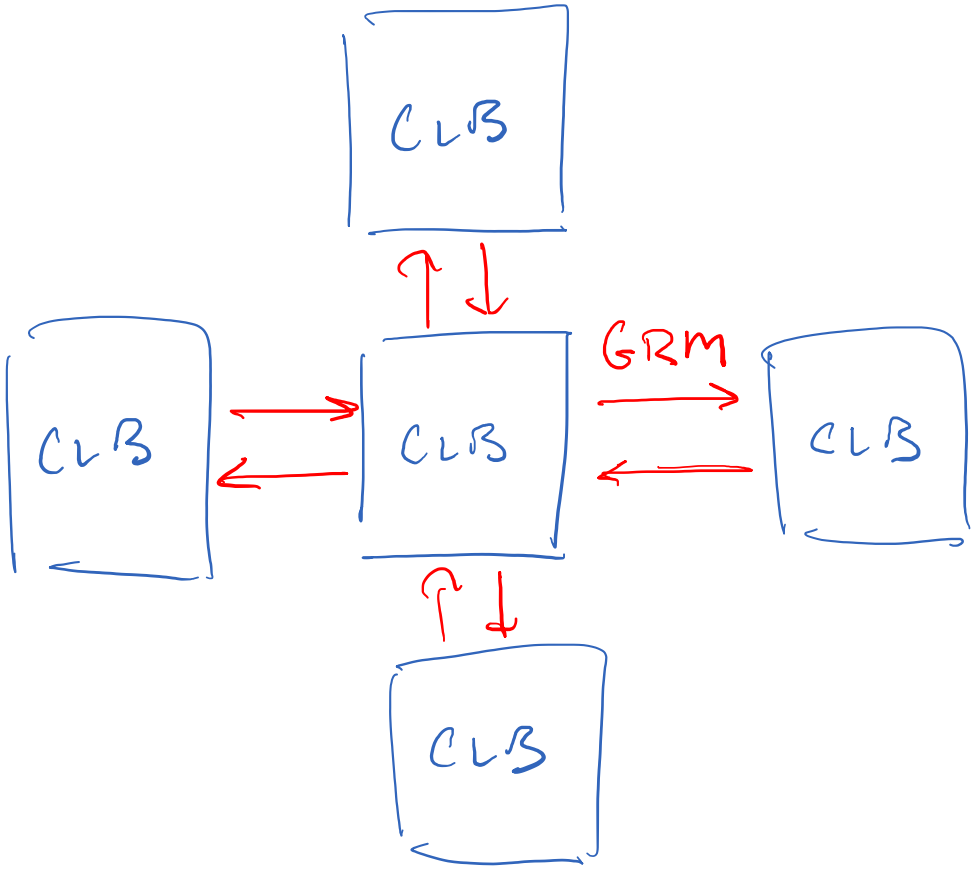
EMBEDDED MICROPROCESSORS / CORES

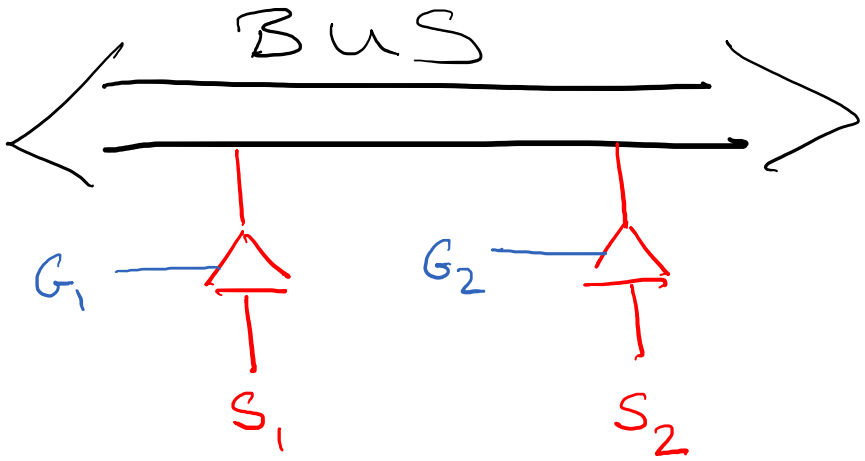


GENERAL ROUTING MATRIX (GRM)

3-STATE CONNECTIONS BET'N CLB & GRM







TRISTATE LOGIC :

IF G IS HIGH, OUTPUT OF GATE IS HIGH OR LOW

IF G IS LOW, OUTPUT IS

OPEN CIRCUIT

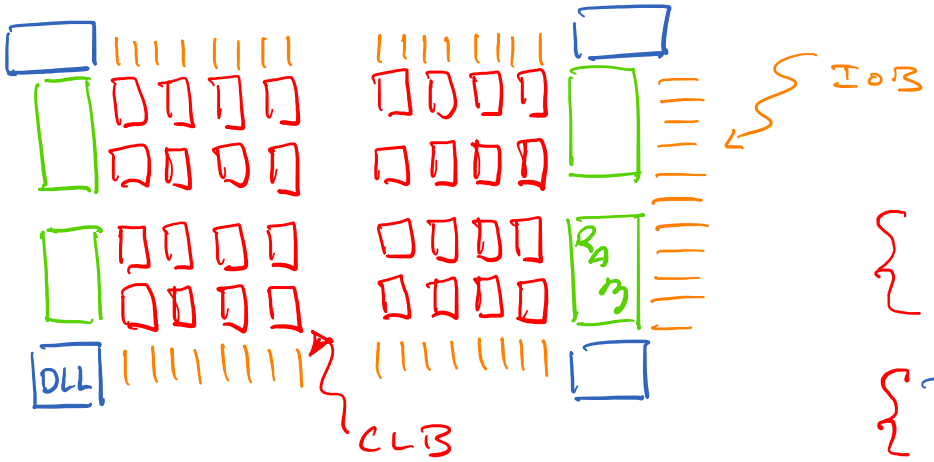
STATIC RAM

DATA STORES IN FLIP FLOP REGISTERS

BLOCK RAM vs DISTRIBUTED RAM

STATIC RAM

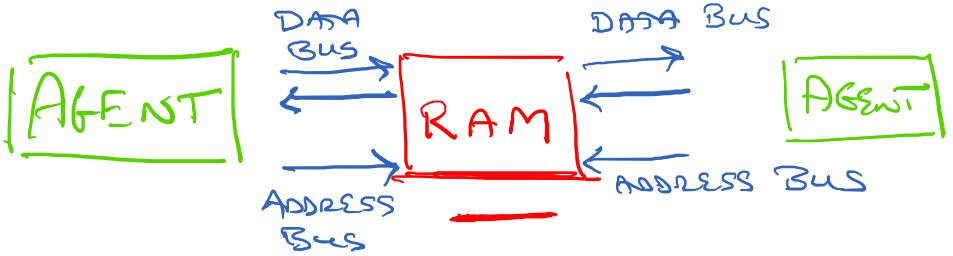
DATA STORED IN FLIP FLOP REGISTERS

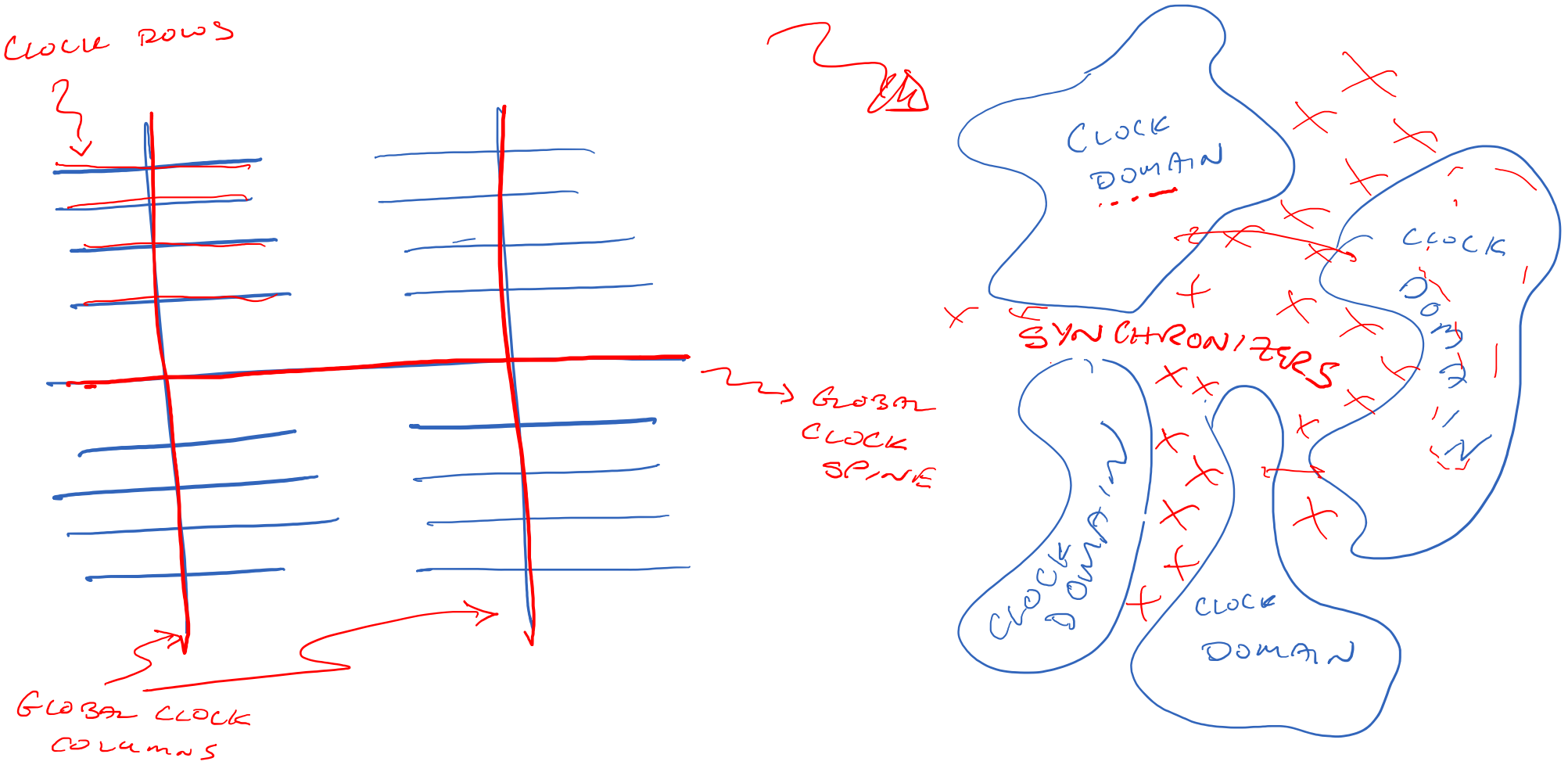


BLOCK RAM vs DISTRIBUTED RAM

{ SINGLE-PORT : ONLY ONE AGENT CAN READ/WRITE

{ DUAL-PORT : 2 AGENTS CAN READ/WRITE





DLL → CLOSED LOOP CORRECTION OF DELAY BET'N INPUT CLOCK & DISTRIBUTED CLOCK

```
module addbit(
  a,
  b,
  ci,
  sum,
  co
);

  input a;
  input b;
  input ci;

  output sum;
  output co;

  wire a;
  wire b;
  wire ci;
  wire sum;
  wire co;

  assign {co,sum} = a + b + ci;

endmodule;
```

HDL

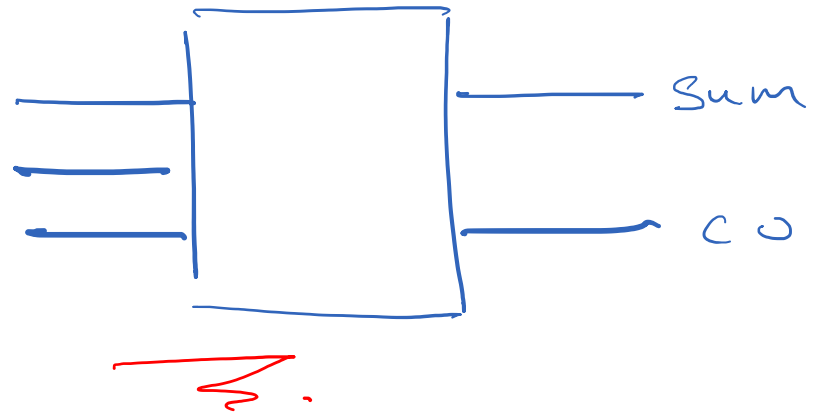
HARDWARE DESCRIPTION LANGUAGE

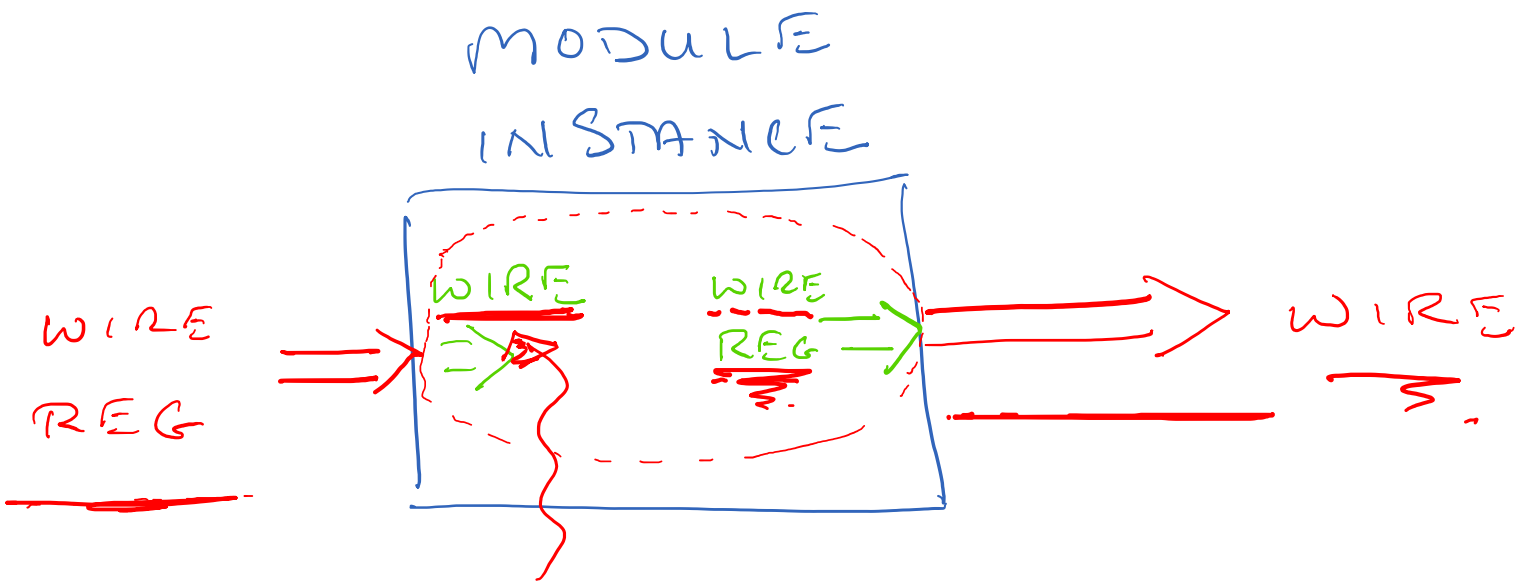
→ VHDL (EUROPE)

→ VERILOG (US)

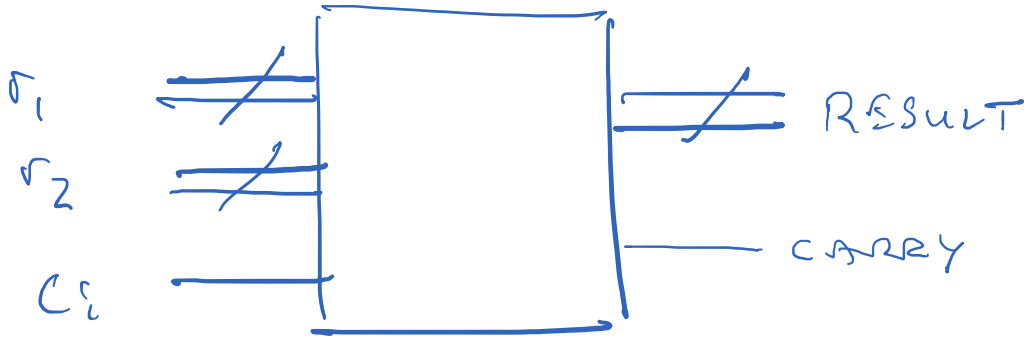
DECLARE PORTS

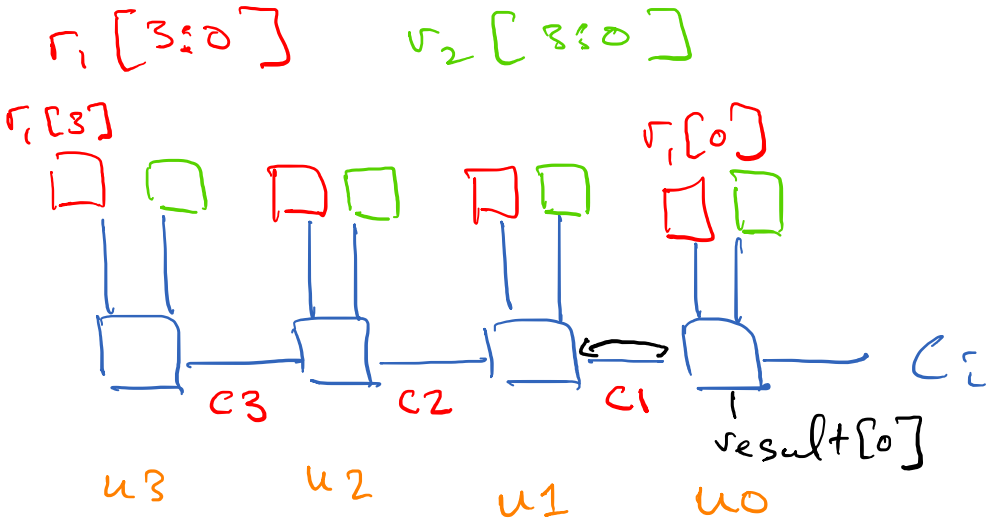
FULL ADDER




MODULE
INSTANCE
WIRE
REG
WIRE
WIRE
REG
WIRE

```
module adder(
 result,
 carry,
 r1,
 r2,
 ci
);

 input [3:0] r1;
 input [3:0] r2;
 input ci;

 output [3:0] result;
 output carry;

 wire [3:0] r1;
 wire [3:0] r2;
 wire ci;
 wire [3:0] result;
 wire carry;

 wire c1;
 wire c2;
 wire c3;

 addbit u0(r1[0],r2[0],ci,result[0],c1);
 addbit u1(r1[1],r2[1],c1,result[1],c2);
 addbit u2(r1[2],r2[2],c2,result[2],c3);
 addbit u3(r1[3],r2[3],c3,result[3],carry);

endmodule
```





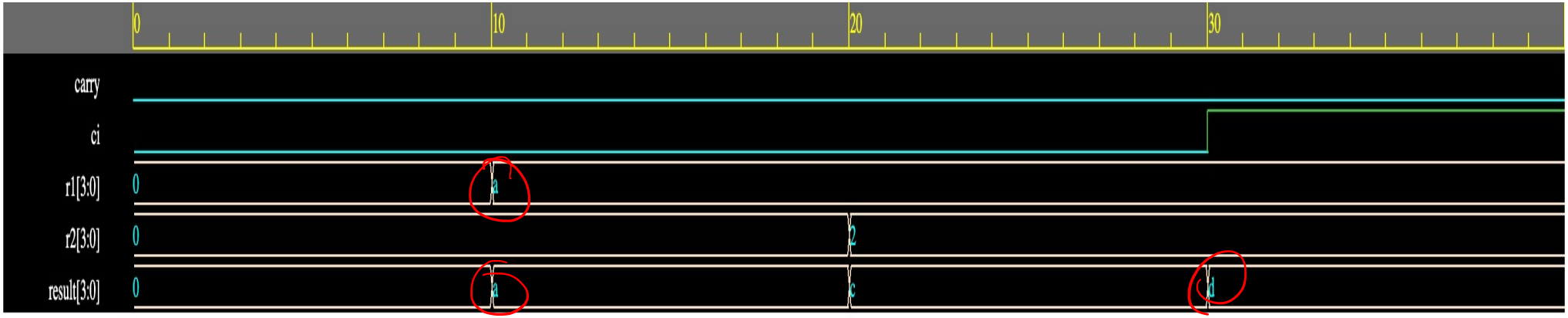
```
module tb();

 reg [3:0] r1, r2;
 reg ci;
 wire [3:0] result;
 wire carry;

 initial begin

    // Dump waves
  $dumpfile("dump.vcd");
  $dumpvars(1);

  $display("time\t r1 r2 ci result carry");

  $monitor("%g %b %b %b %b %b",
        $time, r1, r2, ci, result, carry);

  r1 = 0;
  r2 = 0;
  ci = 0;
  #10 r1 = 10;
  #10 r2 = 2;
  #10 ci = 1;
  #10
  $finish;
 end


 adder U(result,carry,r1,r2,ci);

endmodule
```

TESTBENCH

edaplayground.com

#10 1010

#10 0010



time r1 r2 ci result carry

0 0000 0000 0 0000 0
10 1010 0000 0 1010 0
20 1010 0010 0 1100 0
30 1010 0010 1 1101 0

$r_1 \rightarrow 10$ "a"

$r_2 = 0$

$r_1 + r_2 = a$

$r_1 = a$

$r_2 = 2$

$a + 2 = c$

$a + 2 + 1 = d$

time r1 r2 ci result carry

0 0000 0000 0 0000 0
10 1010 0000 0 1010 0
20 1010 0110 0 0000 1
30 1010 0110 1 0001 1