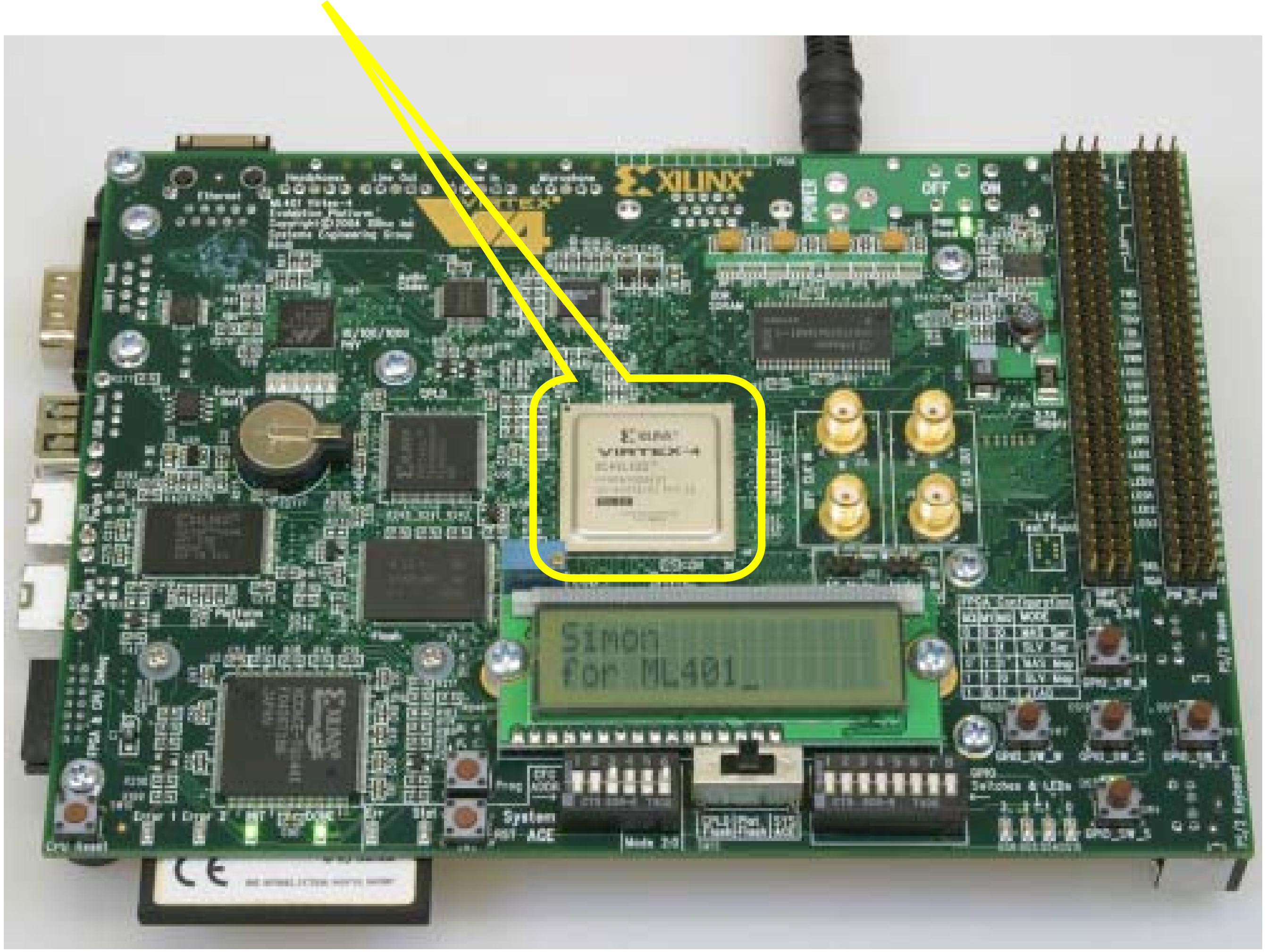
Field Programmable Gate Array

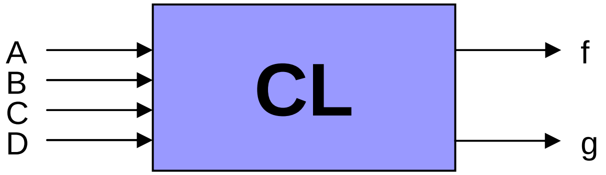
What is FPGA?



FPGA

- Programmable (= reconfigurable) Digital System
- Component
  - Basic components
    - Combinational logics
    - Flip Flops
  - Macro components
    - Multiplier (large combinational logic)
    - Random Access Memory (Large Density)
    - Read Only memory (Large Density)
    - CPU
  - Programmable Interconnection
  - Programmable Input/Output circuit
  - Programmable Clock Generator

What is Combinational Logic?

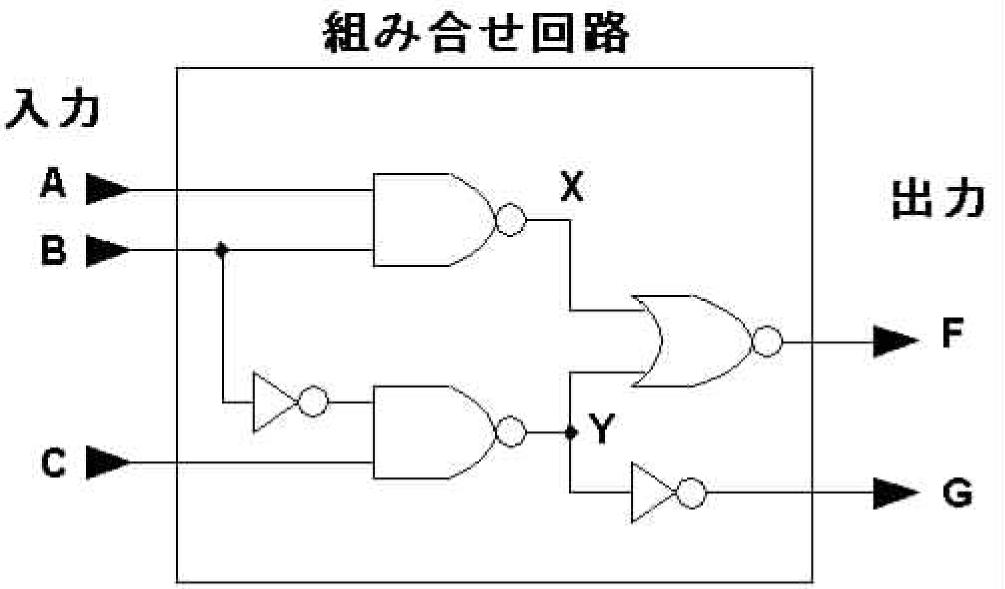


A, B, C, D, f, g are all binary signal.

- If output f, g are function of only inputs (A, B, C, D) then the circuit is combinational circuit.
- In another word, output signal is determined by only the combination of input signals.
  - f = func1(A, B, C, D)
  - g = func2(A, B, C, D)

- Combinational logic does NOT include memories such as Flip-Flops.
- Combinational logic can be constructed by just primitive gates such as NOT, NAND, NOR, etc. (But no feedback loop)

Combinational Logic realization - gates -

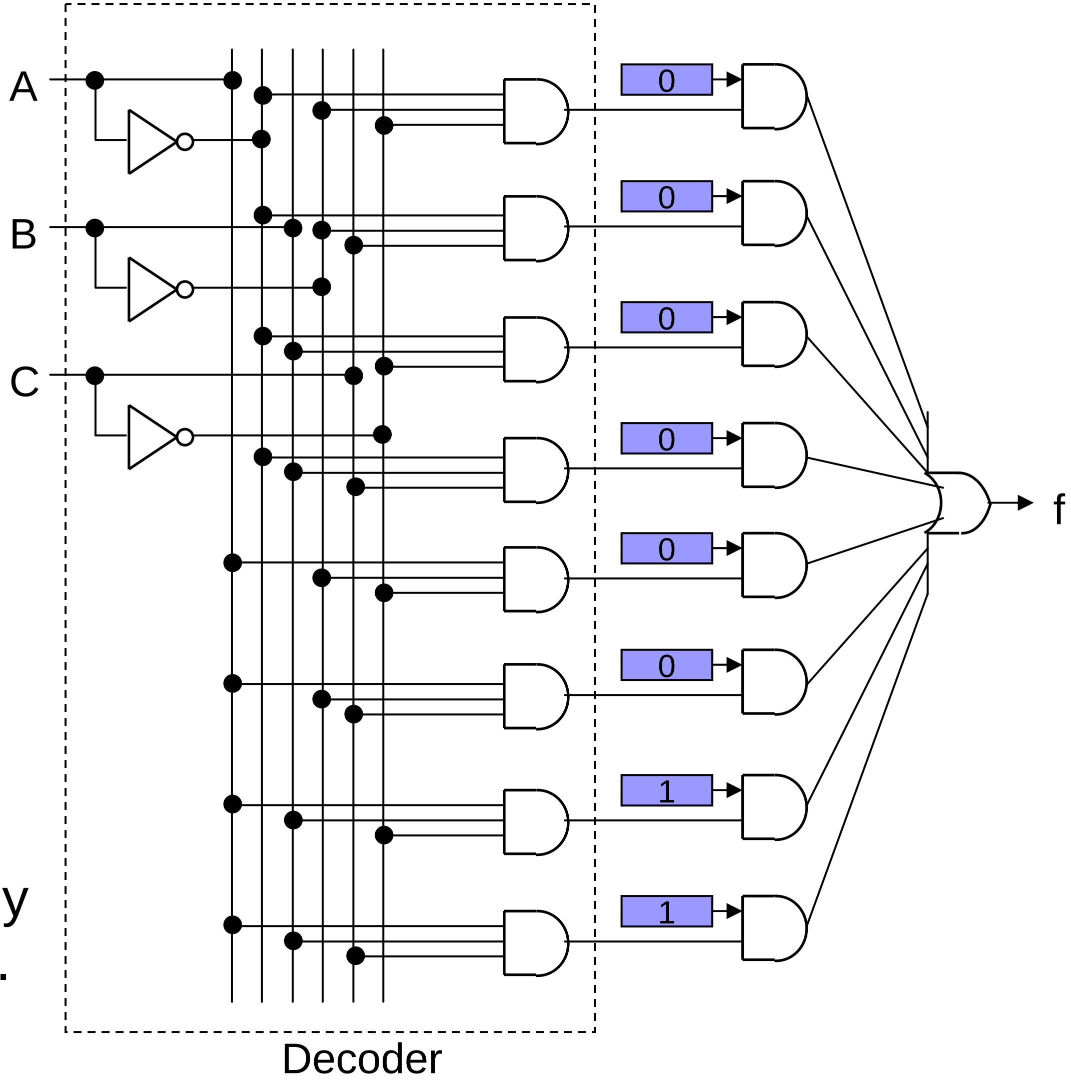


- There is no signal loop in the circuit.
- In combinational logic, signal loop is prohibited since the loop makes states (Memory).
- Function is not configurable.

Combinational Logic realization - Table -

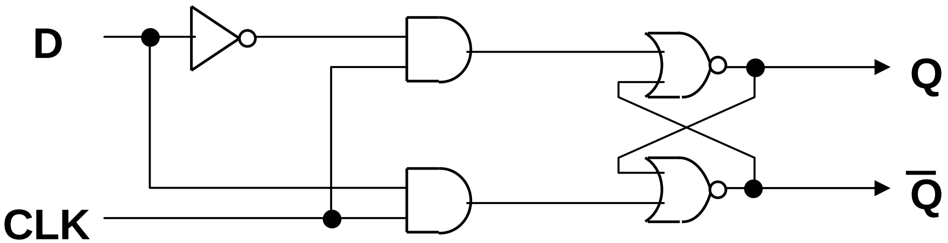
TRUTH TABLE

A	B	C	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



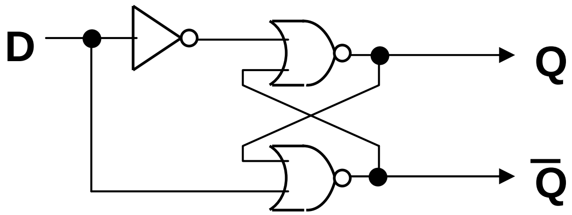
- Function is configurable by storing the TABLE values.

Clocked D LATCH

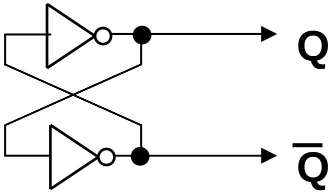


- 1 bit memory by NOR cross-loop
- When CLK=1, Q = D, /Q=not(D)
- When CLK=0, Q holds previous data.

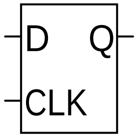
When CLK=‘1’



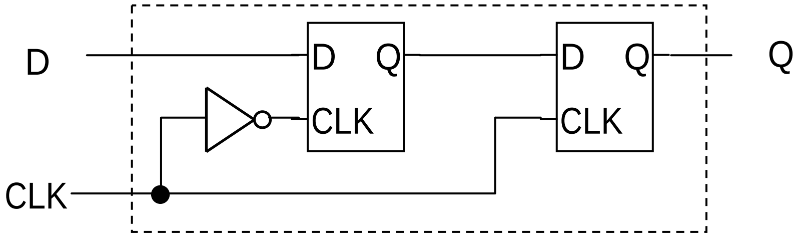
When CLK=‘0’

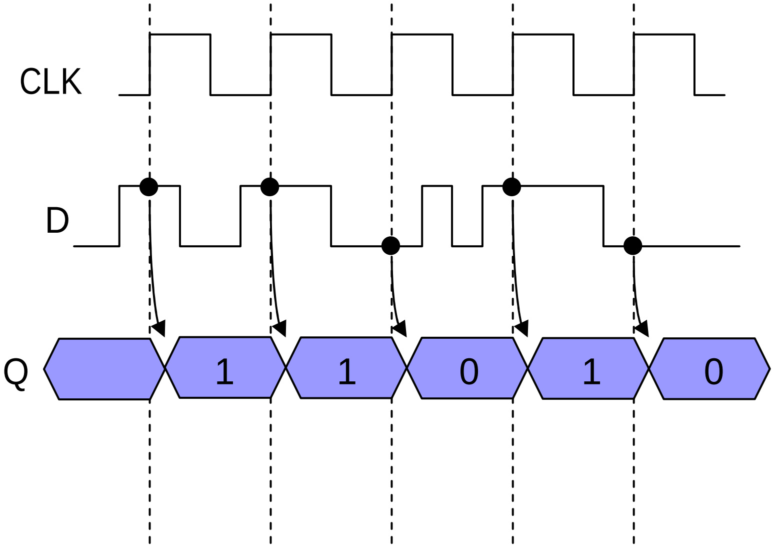


CIRCUIT SYMBOL:



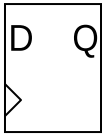
Master-Slave D Flip-Flop



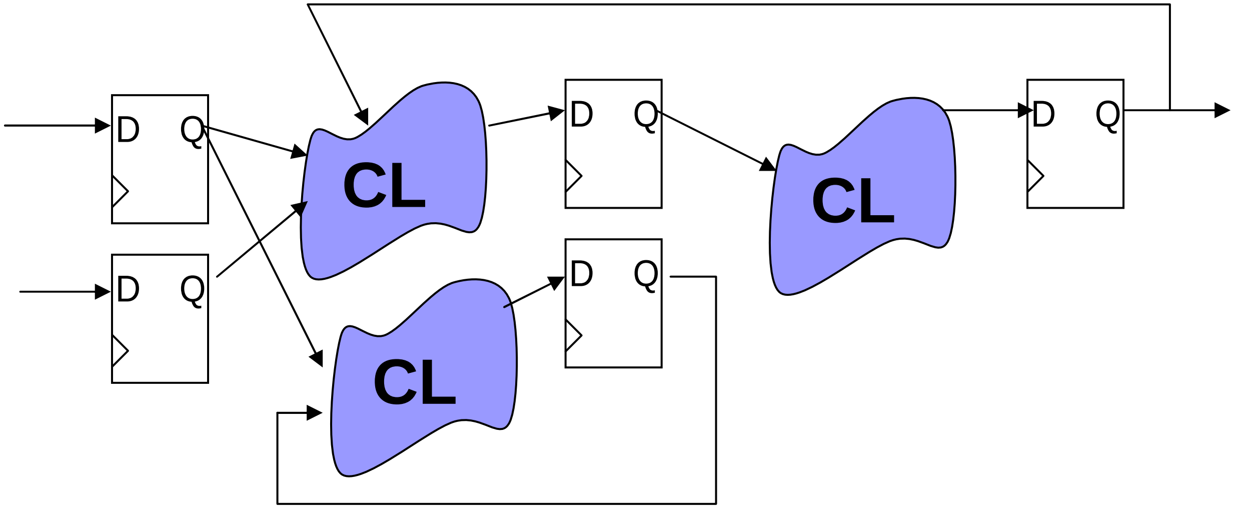


- 2 LATCHES in series
- Still work as 1 bit memory
- CLK edge Trigger Operation
- Most commonly used memory element in the state-of-the-art synchronous Digital Design.
- Q only changes CLK edge (once in one cycle).

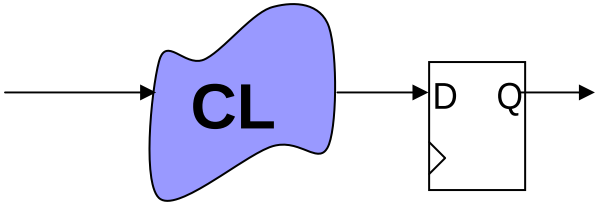
CIRCUIT SYMBOL:



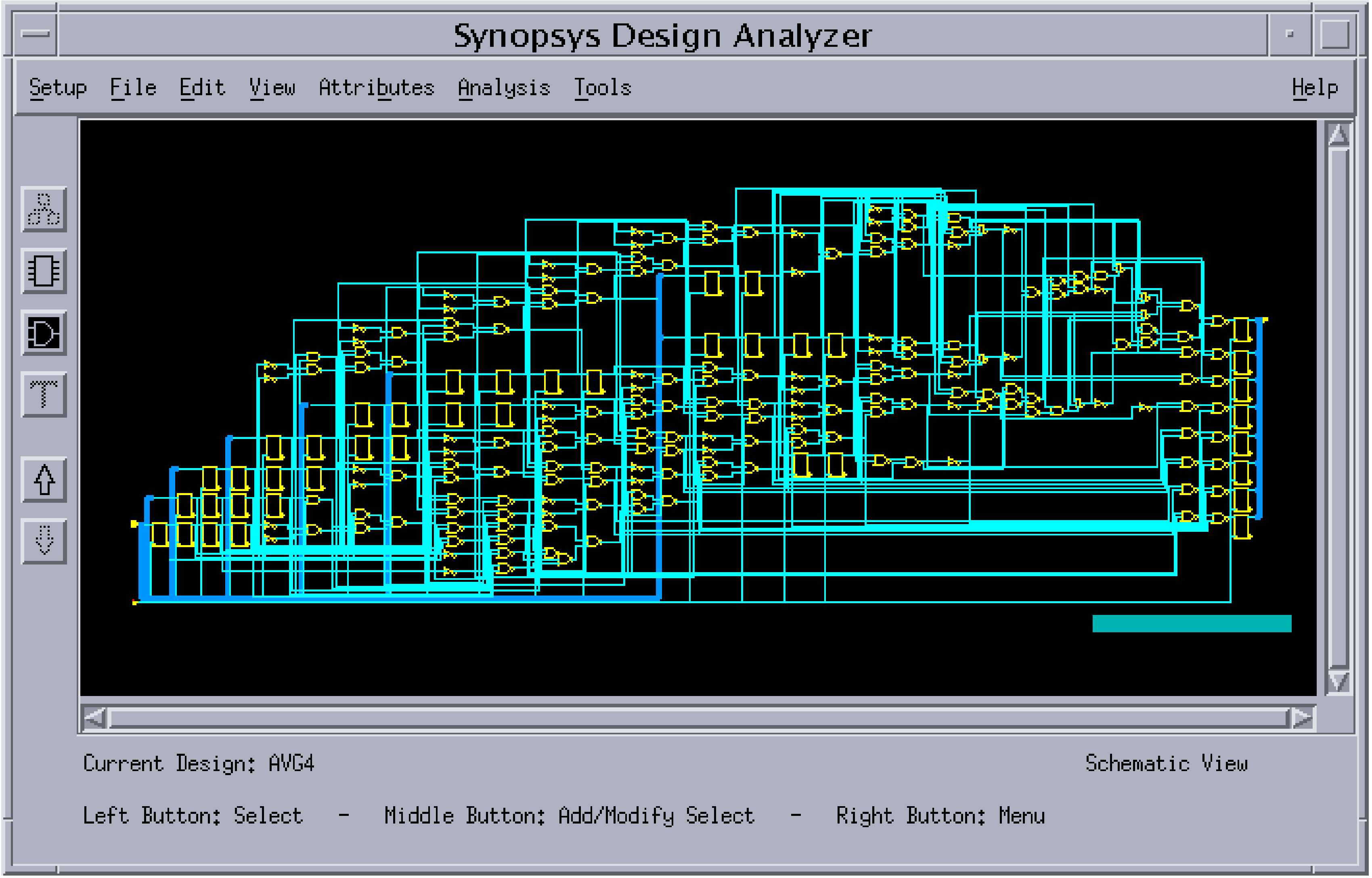
Digital System is just FF + CLs



- FPGA supports such digital circuit with configurability.
- FPGA's basic element



Example of Circuit Synthesis



Current Design: AVG4

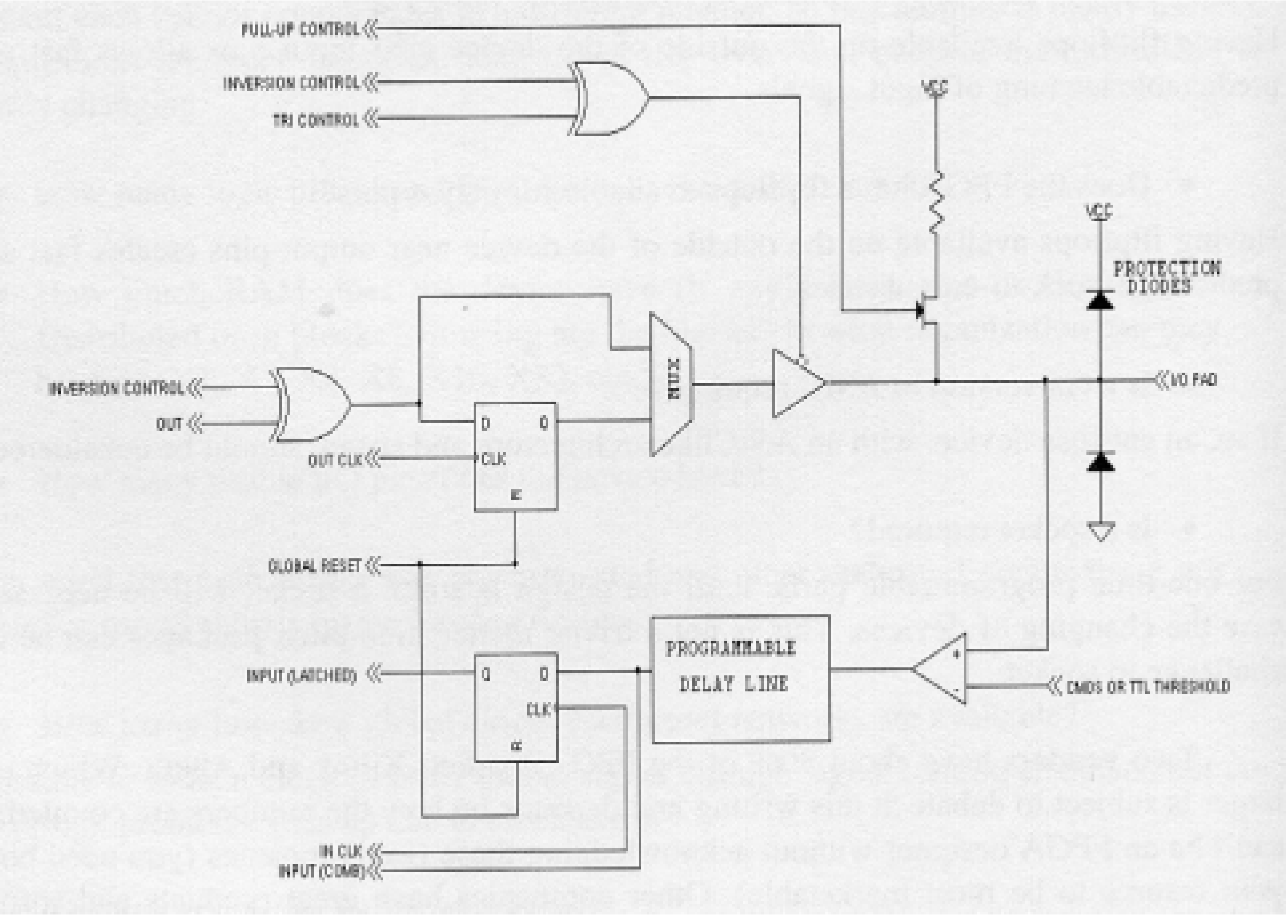
Schematic View

Left Button: Select - Middle Button: Add/Modify Select - Right Button: Menu

XILINX FPGA

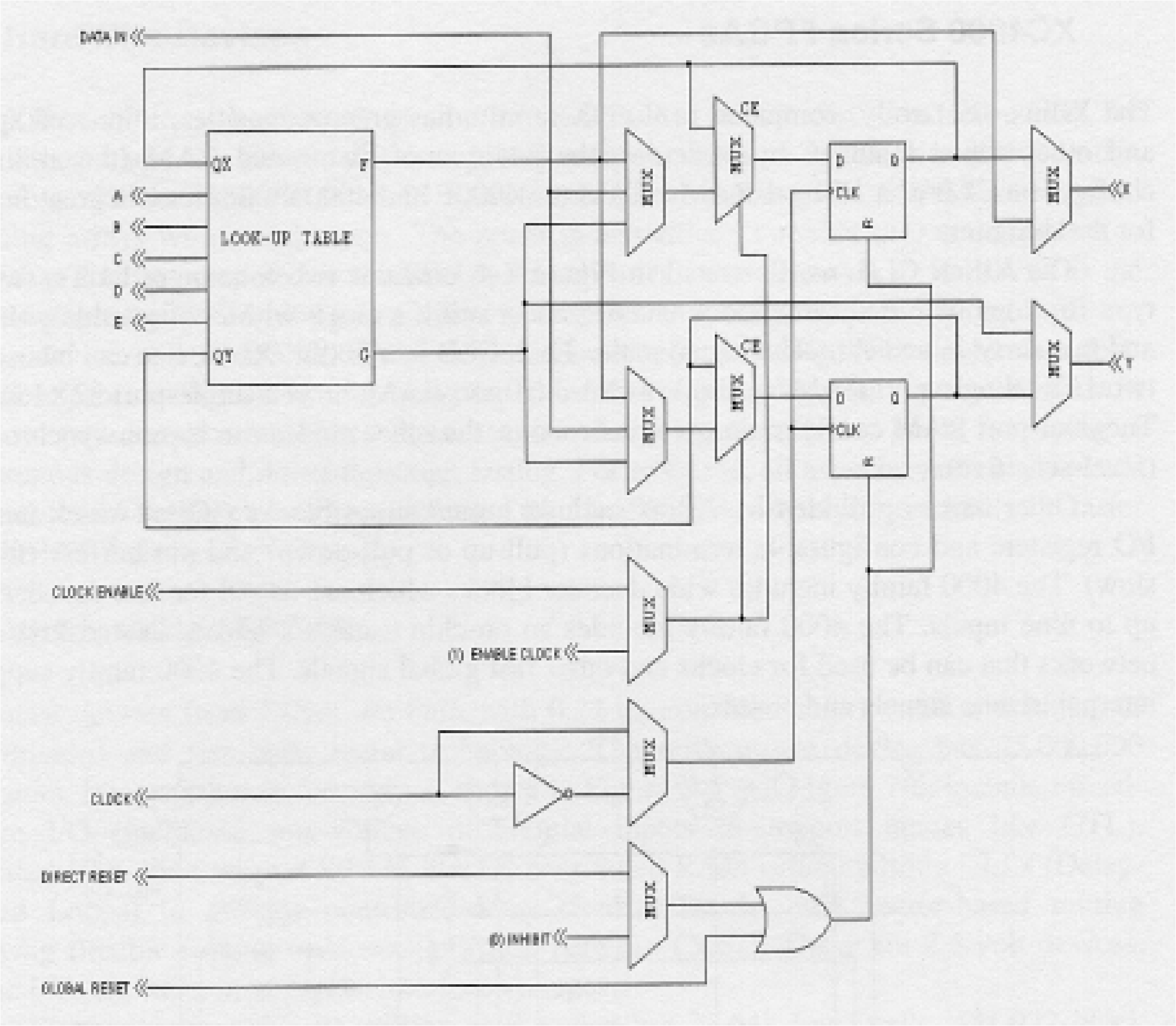
- Field Programmable Gate Array

XILINX XC3000 Family I/O



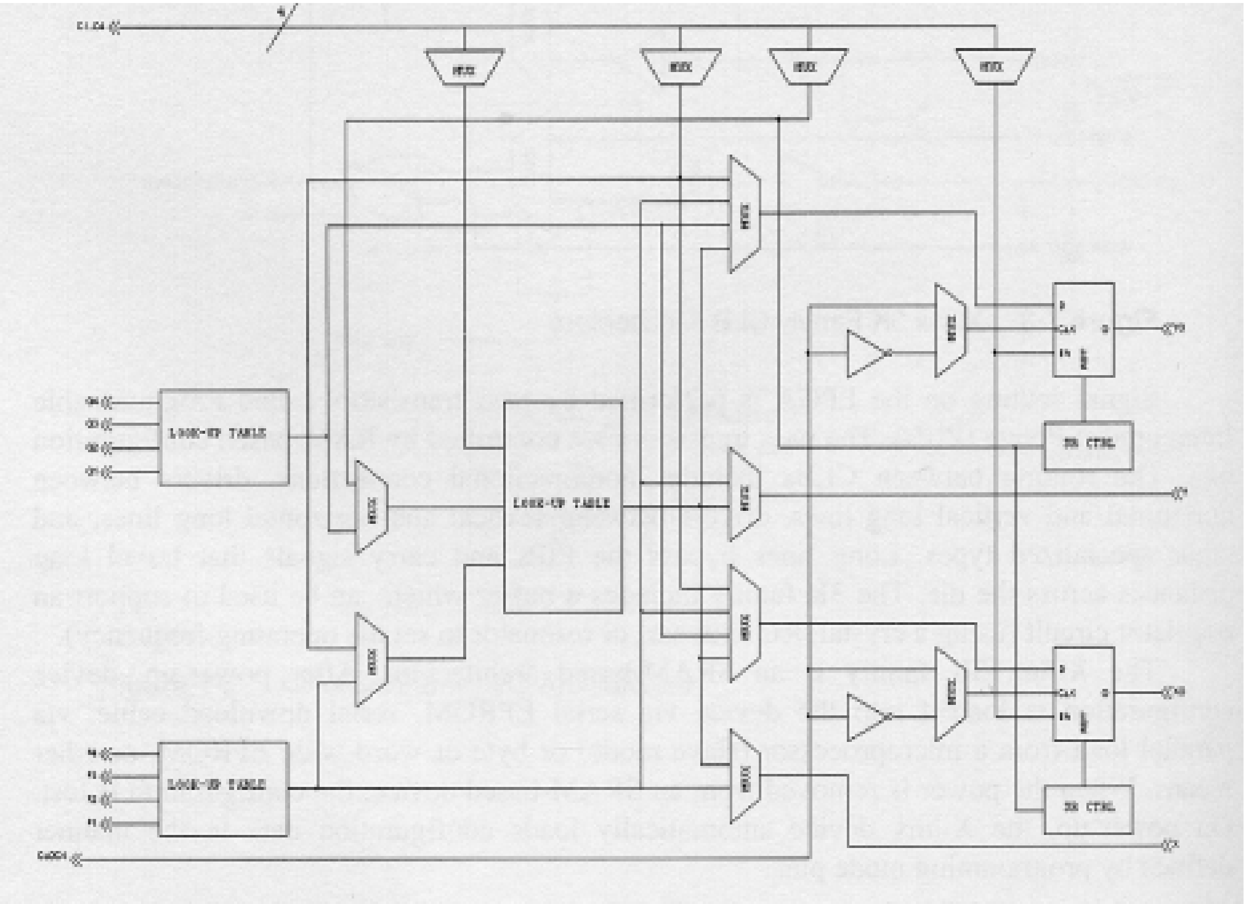
- Electronic Static Discharge Protection
- CMOS, TTL input
- Registered /Non Registered I/O

XILINX XC3000 Family CLB



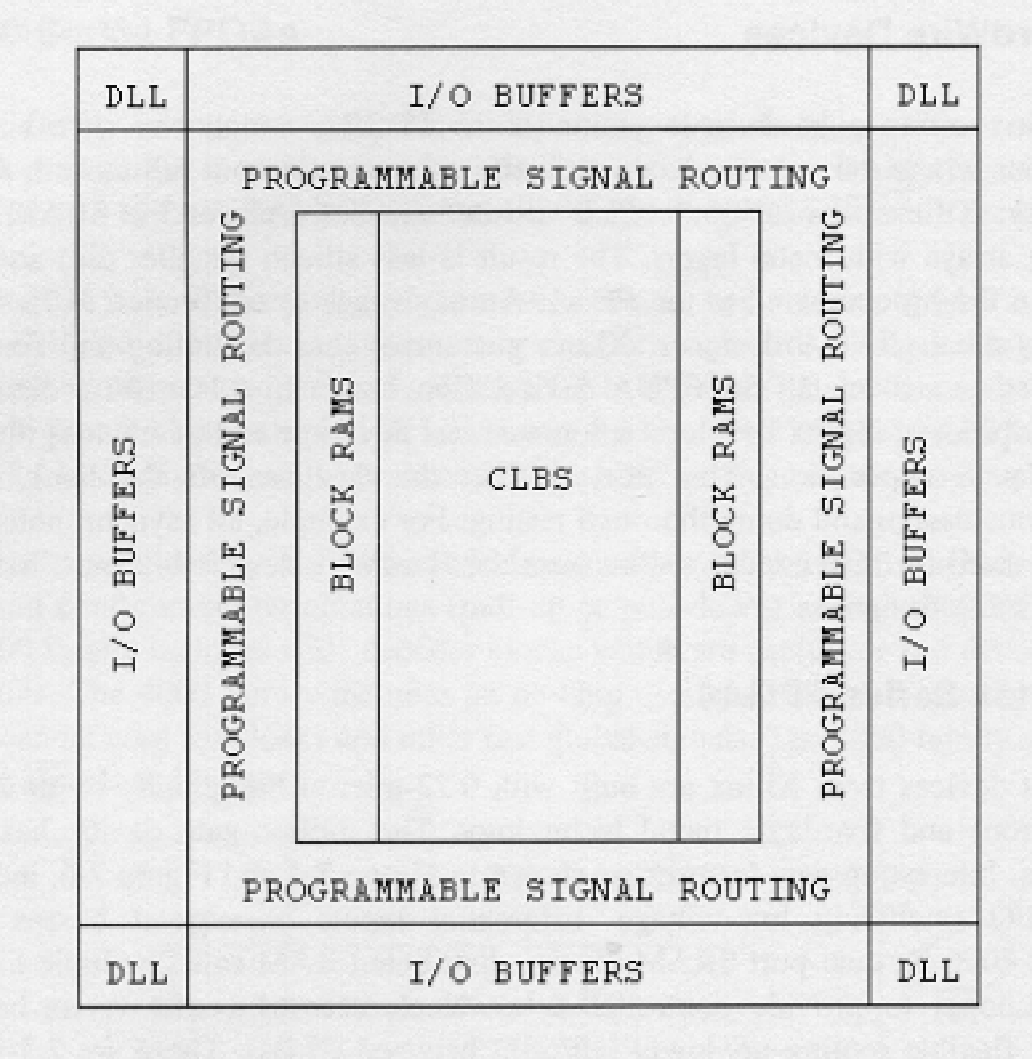
- CLB: Configurable Logic Block
- Look-up table for combinational logic
- D-Flip-Flops
- Look-up Table = RAM

XILINX XC4000 Family CLB



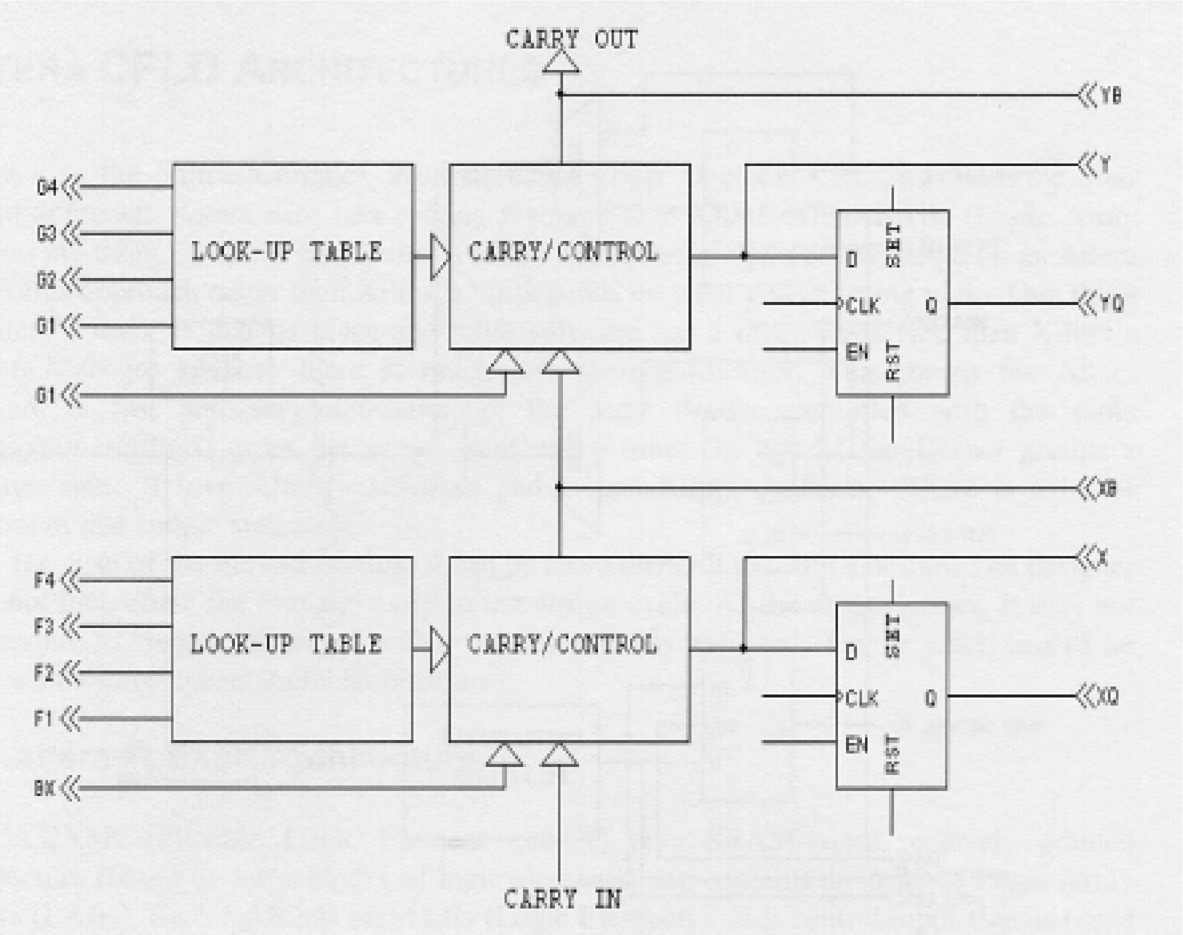
- Two Stage Look-up Table

XILINX VIRTEX FAMILY ARCHITECTURE


DLL
I/O BUFFERS
DLL
PROGRAMMABLE SIGNAL ROUTING
I/O BUFFERS
PROGRAMMABLE SIGNAL ROUTING
BLOCK RAMS
CLBS
BLOCK RAMS
PROGRAMMABLE SIGNAL ROUTING
I/O BUFFERS
PROGRAMMABLE SIGNAL ROUTING
DLL
I/O BUFFERS
DLL

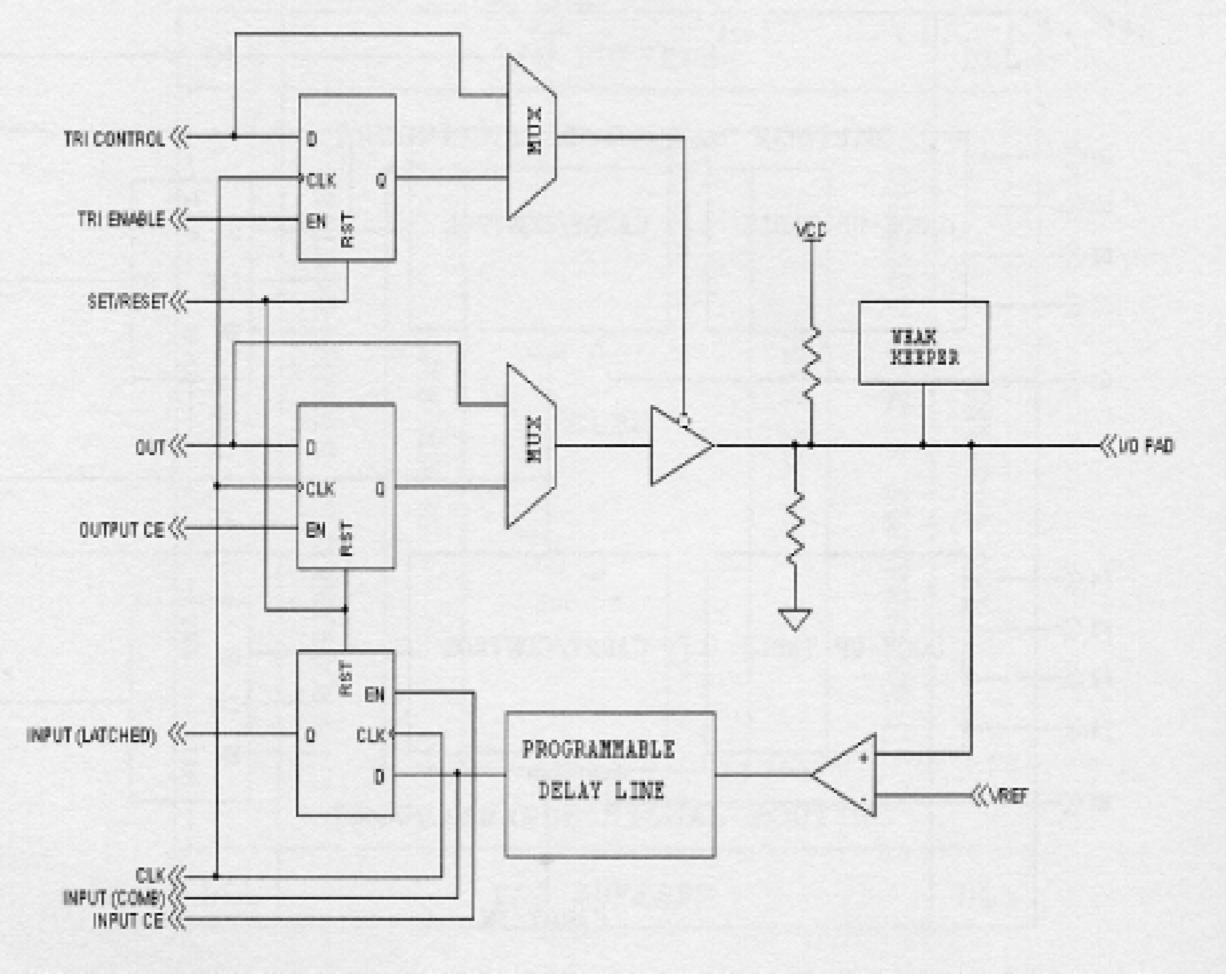
- CLB: Configurable Logic Block
- Many 4Kbit RAM BLOCK RAM
- DLL (Delay-Locked Loops) to provide controlled-delay clock networks
- Multiplier (18b x 18b) Macro also supported (not in figure)

XILINX VIRTEX FAMILY CLB



- CLB: Configurable Logic Block
- Many 4Kbit RAM BLOCK RAM
- DLL (Delay-Locked Loops) to provide controlled-delay clock networks

XILINX VIRTEX FAMILY I/O

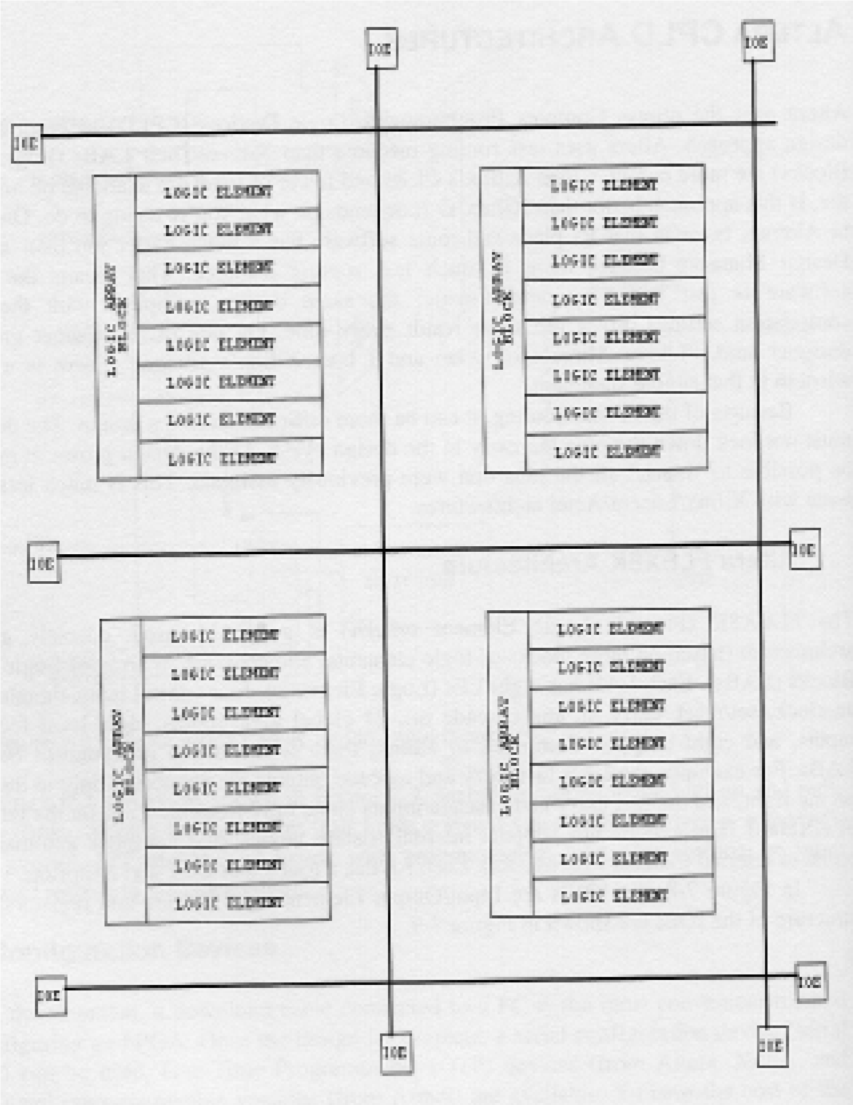


- Electronic Static Discharge Protection
- CMOS, TTL input
- Registered /Non Registered I/O

ALTERA CPLD

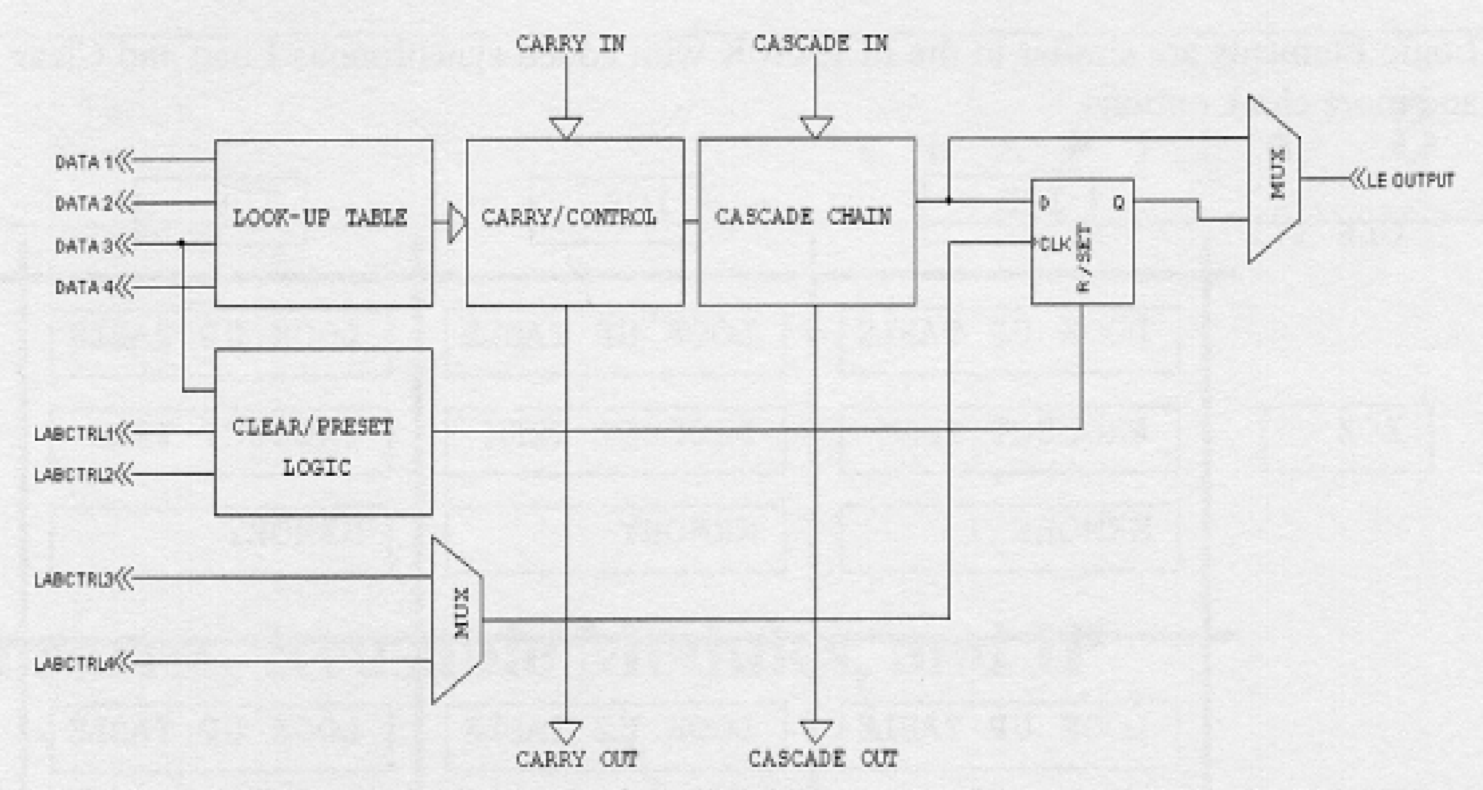
- Complex Programmable Logic Devices
- Altera uses less routing resource than Xilinx
- Altera's Logic Array Block (LAB) is more complex than Xilinx's CLBs. Then fewer LABs in on chip than Xilinx's CLBs.

ALTERA FLEX8000 ARCHITECURE



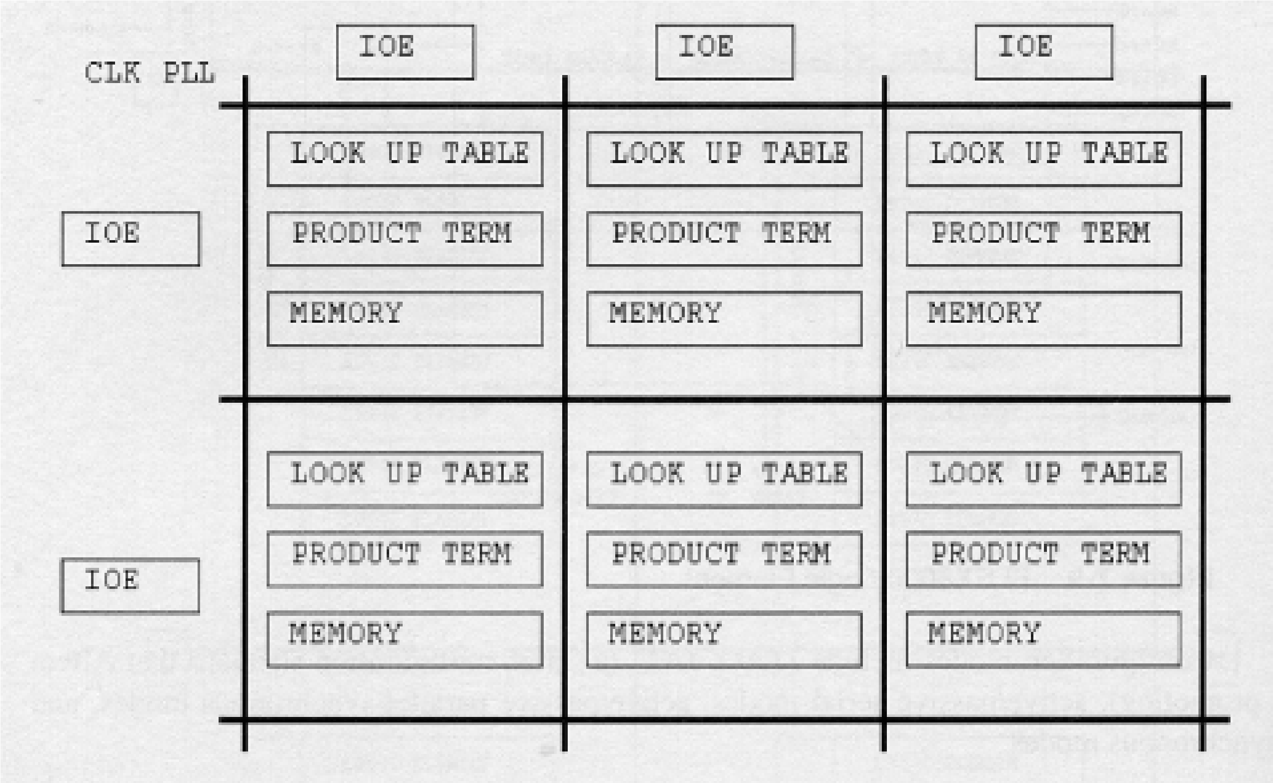
- Each LAB has eight LEs (Logic Elements) .

ALTERA FLEX8000 Logic Element (LE)



- CARRY, CASCADE signals

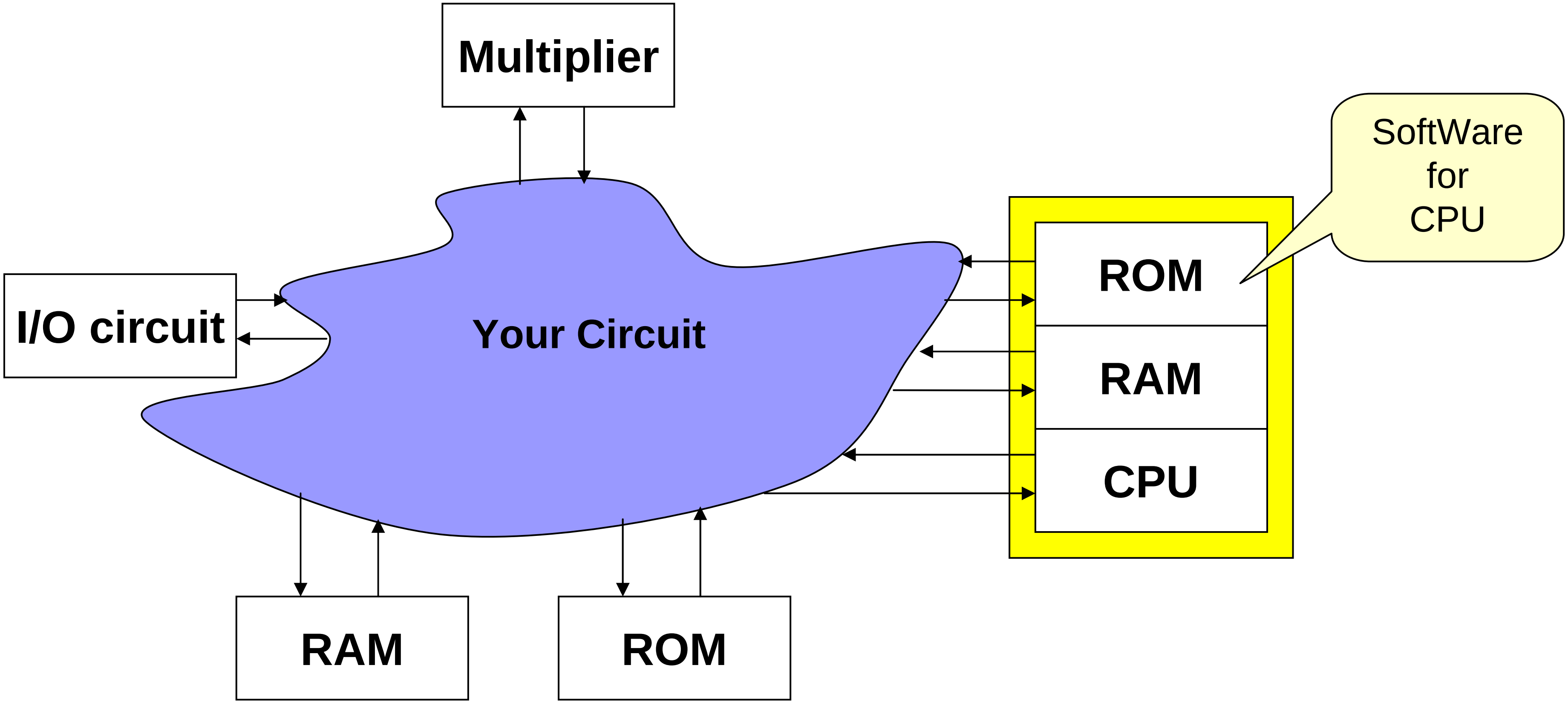
ALTERA APEX 20K ARCHITECTURE



- MANY RAMs
- Large Number Input combinational logic such as Multiplier
- Phase Locked Loop for Advanced Clock generation

How to Design your Digital System using Hard-Macro Blocks

- **White Blocks might be available (Hardware pre-designed Blocks)**



Hardware Description Languages (HDLs)

- HDL is a software programming language used to model the intended operation of a piece of hardware.
- Two level of modeling
  - Abstract behavior modeling
  - Hardware structure modeling:
    Input to Circuit Synthesis
- Two kinds of Language
  - VHDL: Very High Speed Integrated Circuit hardware description language
    - Similar to Pascal Programming language
  - Verilog HDL:
    - Similar to C Programming language

HALF_ADDER example

VHDL

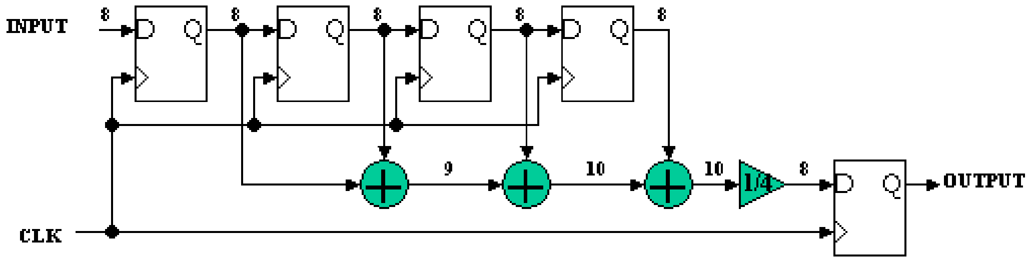
```
library IEEE;
use IEEE.std_logic_1164.all;

entity HALF_ADDER is
             port ( A, B : in std_logic;
                    S, C : out std_logic );
end HALF_ADDER;

architecture STRUCTURE of HALF_ADDER is
  begin
    S <= A xor B;
    C <= A and B;
end STRUCTURE;
```

Verilog HDL

```
module HALF_ADDER (
  A, B,
  S, C
);

  input A, B;
  output S, C;

  assign S = A ^ B;
  assign C = A & B;
endmodule
```

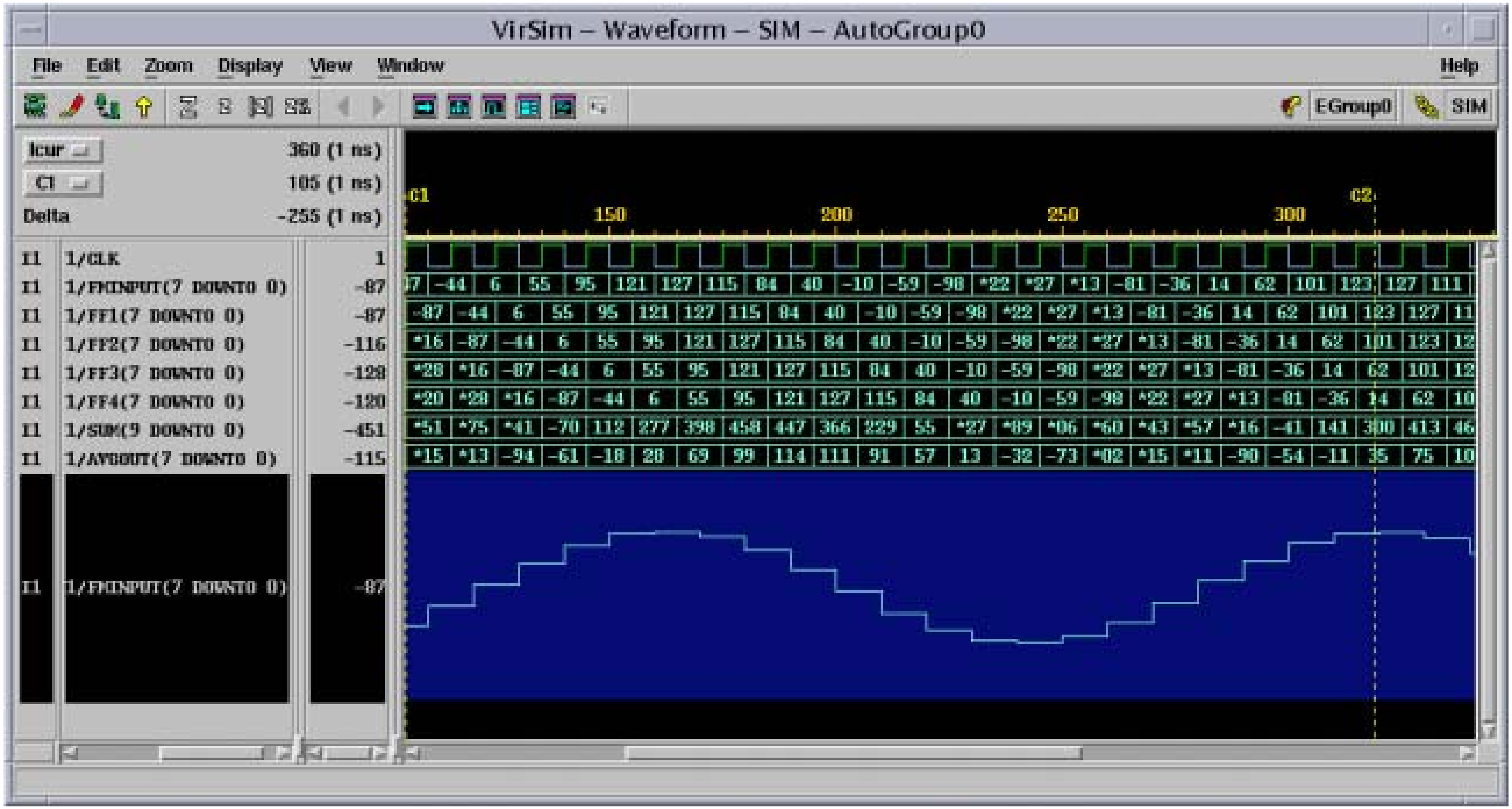
Moving Average Filter by VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_ARITH.all;

entity AVG4 is
  port(CLK      : in  std_logic;
       FMINPUT : in  std_logic_vector(7 downto 0);
       AVGOUT  : out std_logic_vector(7 downto 0));
end AVG4;

architecture RTL of AVG4 is

signal FF1, FF2, FF3, FF4 : std_logic_vector(7 downto 0);
signal SUM     : std_logic_vector(9 downto 0);

begin

-- SHIFT REGISTER
  process(CLK) begin
    if (CLK'event and CLK = '1') then
       FF1 <= FMINPUT;
       FF2 <= FF1;
       FF3 <= FF2;
       FF4 <= FF3;
    end if;
  end process;
```



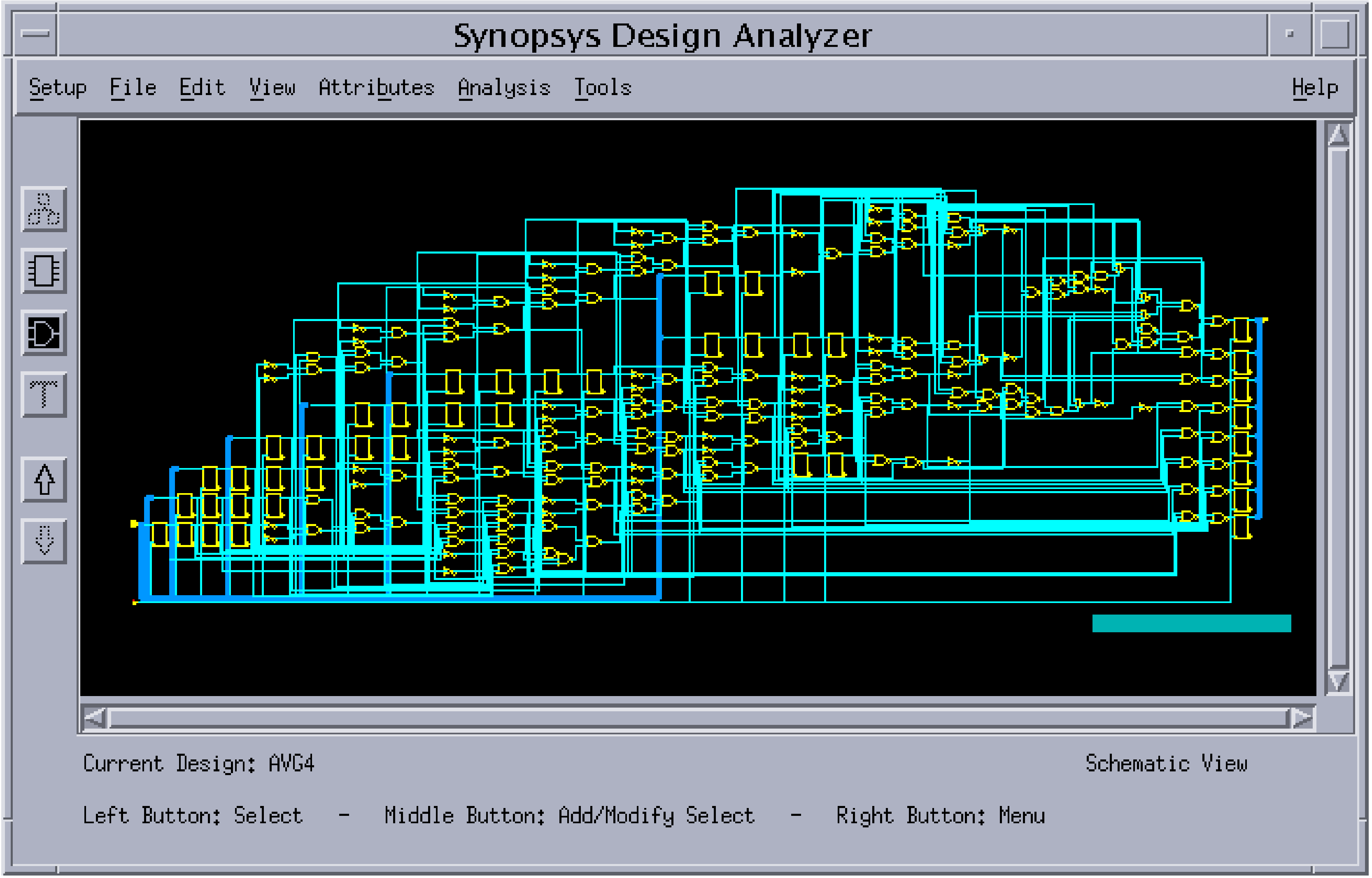
```
-- SUM
  SUM <=signed(FF1(7)&FF1(7)&FF1)+signed(FF2(7)&FF2(7)&FF2)
        +signed(FF3(7)&FF3(7)&FF3)+signed(FF4(7)&FF4(7)&FF4);

-- DIVIDE BY 4 (SHIFT 2 bit), OUTPUT REGISTER
  process(CLK) begin
    if (CLK'event and CLK='1') then
        AVGOUT <= SUM(9 downto 2);
    end if;
  end process;

end RTL;
```

Simulated Waveform



```
Icur          360 (1 ns)
C1            105 (1 ns)
Delta        -255 (1 ns)

I1  1/CLK                      1
I1  1/FMINPUT(7 DOWNTO 0)    -87   7 -44 6 55 95 121 127 115 84 40 -10 -59 -98 *22 *27 *13 -81 -36 14 62 101 123 127 111
I1  1/FF1(7 DOWNTO 0)        -87   -87 -44 6 55 95 121 127 115 84 40 -10 -59 -98 *22 *27 *13 -81 -36 14 62 101 123 127 11
I1  1/FF2(7 DOWNTO 0)       -116   *16 -87 -44 6 55 95 121 127 115 84 40 -10 -59 -98 *22 *27 *13 -81 -36 14 62 101 123 12
I1  1/FF3(7 DOWNTO 0)       -128   *28 *16 -87 -44 6 55 95 121 127 115 84 40 -10 -59 -98 *22 *27 *13 -81 -36 14 62 101 12
I1  1/FF4(7 DOWNTO 0)       -120   *20 *28 *16 -87 -44 6 55 95 121 127 115 84 40 -10 -59 -98 *22 *27 *13 -81 -36 14 62 10
I1  1/SUM(9 DOWNTO 0)       -451   *51 *75 *41 -70 112 277 398 458 447 366 229 55 *27 *89 *06 *60 *43 *57 *16 -41 141 300 413 46
I1  1/AVOUT(7 DOWNTO 0)     -115   *15 *13 -94 -61 -18 20 69 99 114 111 91 57 13 -32 -73 *02 *15 *11 -90 -54 -11 35 75 10

I1  1/FMINPUT(7 DOWNTO 0)    -87
```

Synthesized Circuit

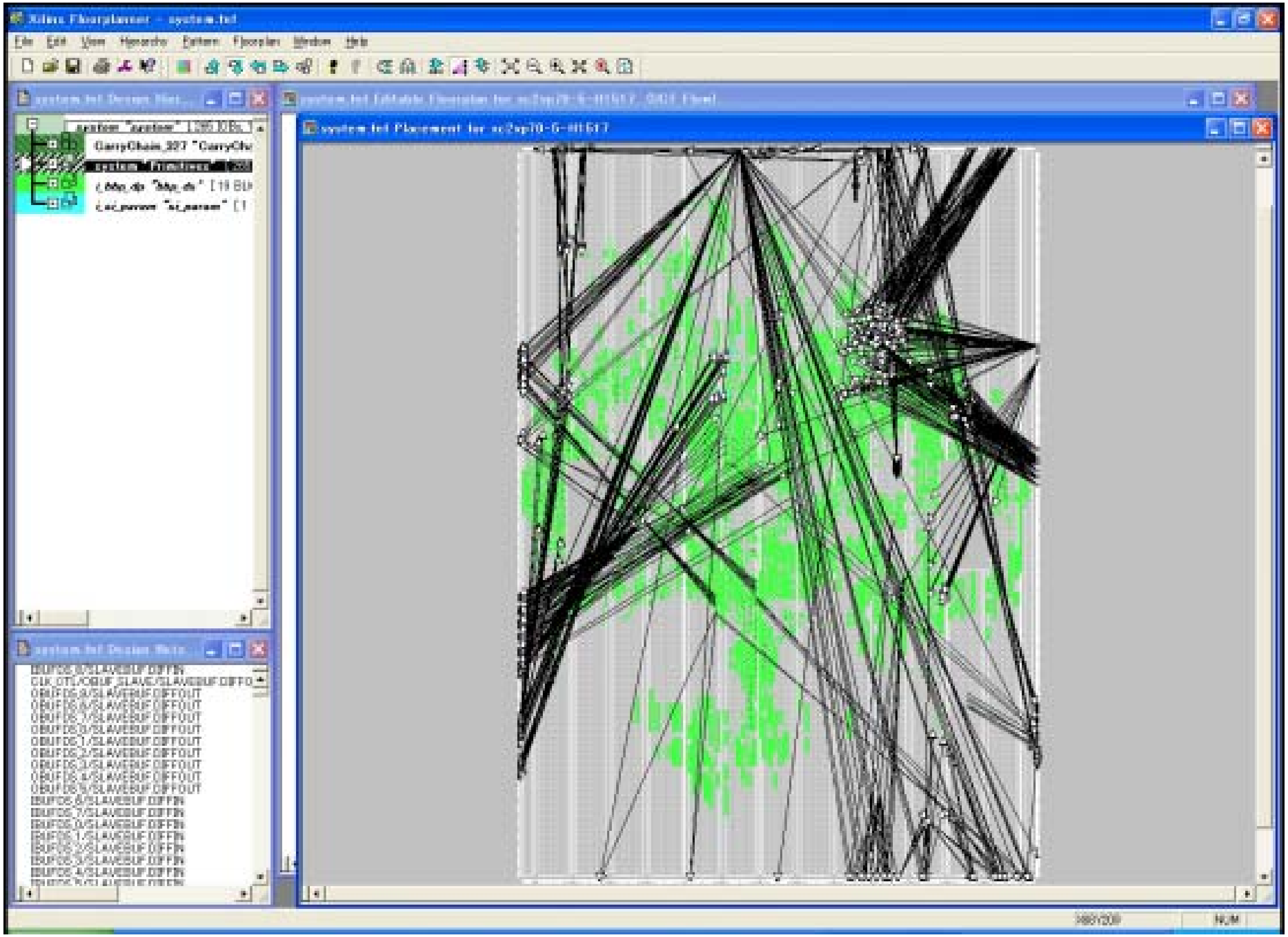


Current Design: AVG4

Schematic View

Left Button: Select - Middle Button: Add/Modify Select - Right Button: Menu

XILINX VP70 FLOORPLAN


Xilinx Floorplanner - system.fnf
File Edit View Hierarchy Pattern Floorplan Window Help
system.fnf Design Hier...
system.fnf Editable Floorplan for xc2vp70-6-ff1517 [illegible]
system.fnf Placement for xc2vp70-6-ff1517
system.fnf Design Nets
NUM

system "system" [385 ELBs]
CarryChain_327 "CarryCh
system "Primitives"
i_bbp_ds "bbp_ds" [19 Bl
i_sc_param "sc_param" [1

IBUFDS_8/SLAVEBUF.DIFFIN
CLK_DTL/OBUF_SLAVE/SLAVEBUF.DIFFO
OBUFDS_9/SLAVEBUF.DIFFOUT
OBUFDS_8/SLAVEBUF.DIFFOUT
OBUFDS_7/SLAVEBUF.DIFFOUT
OBUFDS_0/SLAVEBUF.DIFFOUT
OBUFDS_1/SLAVEBUF.DIFFOUT
OBUFDS_2/SLAVEBUF.DIFFOUT
OBUFDS_3/SLAVEBUF.DIFFOUT
OBUFDS_4/SLAVEBUF.DIFFOUT
OBUFDS_5/SLAVEBUF.DIFFOUT
IBUFDS_6/SLAVEBUF.DIFFIN
IBUFDS_7/SLAVEBUF.DIFFIN
IBUFDS_0/SLAVEBUF.DIFFIN
IBUFDS_1/SLAVEBUF.DIFFIN
IBUFDS_2/SLAVEBUF.DIFFIN
IBUFDS_3/SLAVEBUF.DIFFIN
IBUFDS_4/SLAVEBUF.DIFFIN
IBUFDS_5/SLAVEBUF.DIFFIN