Architecture behavior of upcount is
begin
process (clock, reset)
begin
if reset = '1' then
Q <= 0;
elsif clk'event and clk = '1' then
if E = '1' then
Q <= In;
else
Q <= Q + 1;
end if;
end if;

if E = 1 then the flip flops in the Counter are loded parallely with the value of input In on positive clock edge. But if E = 0, the Counter state will be incremented.

VHDL Code for 3-bit Synchronous up Counter

```
library ieee;
use ieee. std_logic_1164. all;
entity Counter3 is
Port ( clk, reset : in std_logic;
       Cout : inout integer range 0 to 7);
end Counter3;
Architecture behaviour of Counter3 is
 begin
Process (clk)
 begin
if clk'event and clk = '1' then
  if (reset = '1' or Count >= 7) then
    Cout <= 0;
    else
    Count <= Count + 1;
    end if;
    end if;
    end process;
    end behaviour;
```

Designing of such kind of Synchronous Counter is a long process. It is done in various steps following finding the no. of flip flops, States diagrams, choice of flip flop and excitation tables and then finding the boolean expression for each flip flop o/p using k-maps.

without going in the depth of design procedure of Synchronous Counter, VHDL Code for 3-bit Synchronous up Counter is shown below which Counts from 0 to 7.

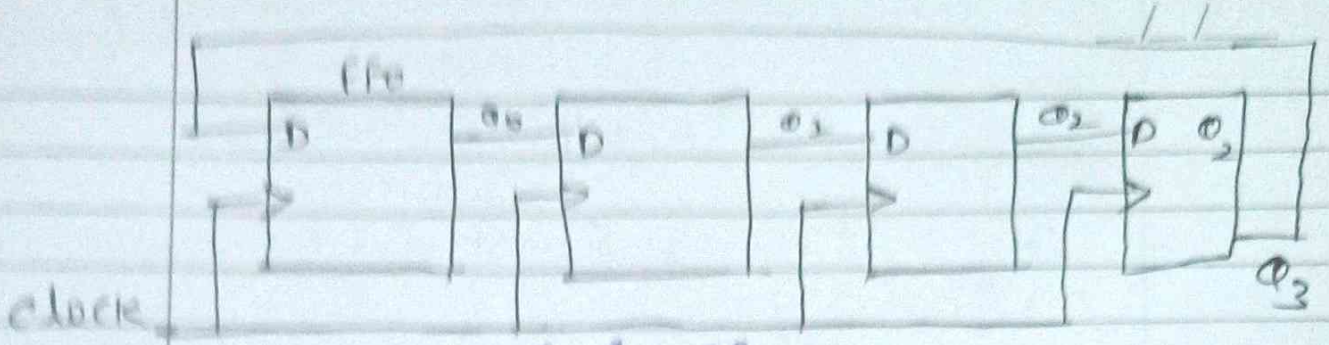
The Code describe a 3-bit Synchronous up Counter. it has two input rst & clk and one o/p Count. Since it is a Synchronous Counter priority is given to clock and only clock signal is present in Sensitivity list. in this Counter if Reset = '1' or Count >= 7 then on the next the clk edge 0 is assigned to Count. But if reset = 0 Count is incremented on each positive clk pulse.

Shift Register Counter →

1. Ring Counter
2. Johnson Counter

In a Johnson Counter, the Complement of the output of last flip flop (FF3). is Connected back to the input D of first flip flop (FF0)

This feed back circuit produces a special Sequence of States. Notice that the 4-bit Sequence has total of eight States. that is, a Johnson Counter produces a Modulo of 2n.



a) Circuit Diagram

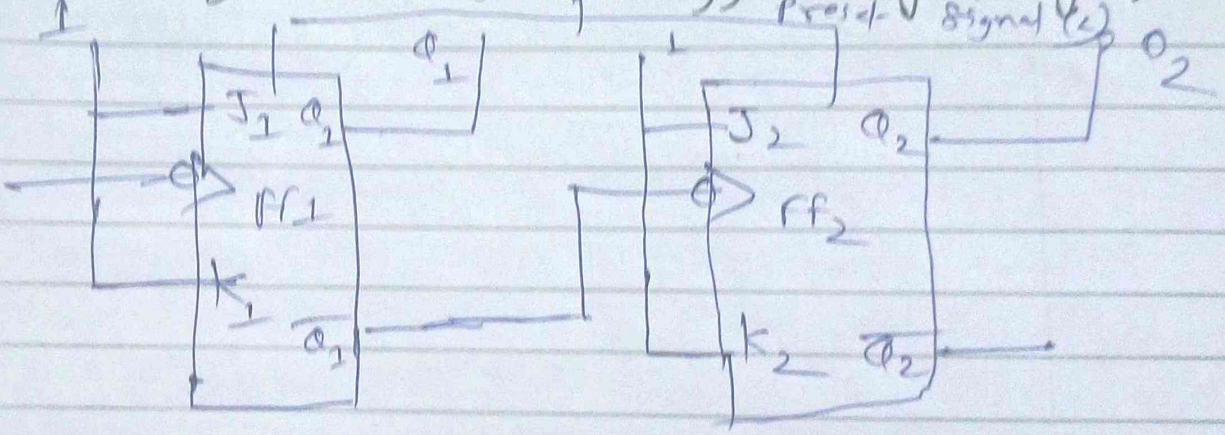
clock	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

b) 4 bit Johnson Sequence.

```
entity Johnson is
generic (size : integer := 4);
Port ( clk, reset : in Std_logic;
       Q : out Std_logic_Vector
           (size-1 downto 0));
end Johnson;
Architecture behavior of Johnson is
Signal Qout : Std_logic_Vector (size-1
                downto 0);
begin
Process (clk)
```

```
begin
if (reset = '1') then
  Qout <= (other => '0');
elsif clk'event and clk = '1' then
  Qout(0) <= not Qout(size-1);
  for i in 1 to size-1 loop
    Qout(i) <= Qout(i-1);
  end loop;
 end if;
 end Process;
 Q <= Qout;
 end behaviour;
```

Asynchronous 2-bit down Counter using negative edge - triggered flip-flops



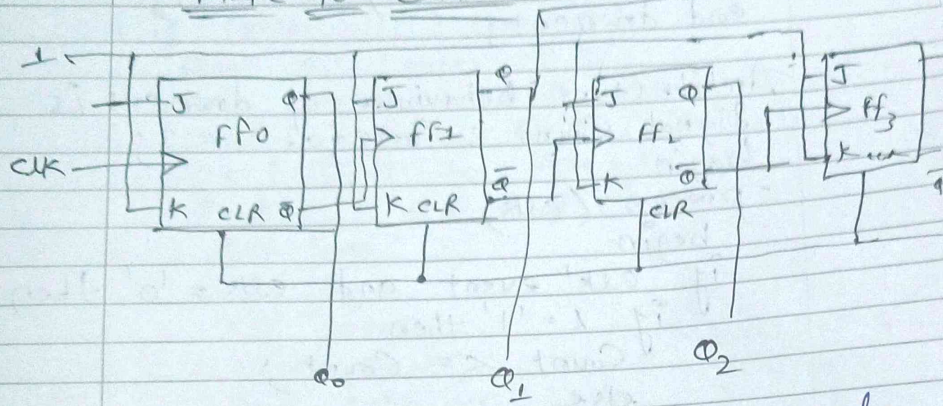
Logic diagram

```
library ieee;
use ieee.std_logic_1164.all;
 entity dowment is

port ( clk, L : in std_logic;
```

```
Q : out integer range 0 to 3);
end downcnt;

Architecture behaviour of downcnt is
Signal Count : integer := 3;
begin
Process (clk)
  begin
  if clk' event and clk = '0' then
    if L = '1' then
      Count <= Count;
      else
      Count <= Count -1;
      end if;
      end if;
      end process;
      Q <= Count;
      end behaviour;
```

The Counter describes a 4-bit ripple down Counters. it has two input clk and L and one output Q. The Signal Count is declared in the architecture body for decrement operation. On the +ve clock edge if L=1, the Counter is ~~reload~~ loaded with the value Count (i.e. 3 in this case) and if L=0 the Signal Count is decremented. The value of Count is assigned to the output Q at the end of Code.

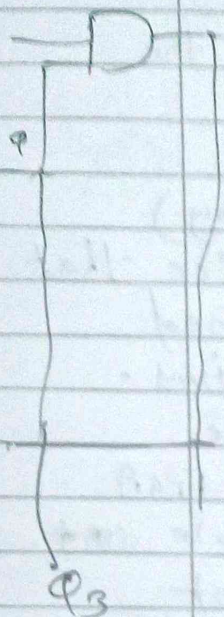
Mode 10 Counter



VHDL Code for Asynchronous Decade (mod 10) Counter

```
library ieee;
use ieee.std_logic_1164.all;
entity decade is
Port ( clk, clr : in std_logic;
       q : out integer range 0 to 9);
end decade;

Architecture behaviour of decade is

signal temp : integer range 0 to 9;
  begin
Process (clk, clr)
  begin
if (clr = '1' or temp >= 9) then
```

$(10)_{10} = (1010)_2$




```
temp <= 0;
elsif clk'event and clk = '1' then
    temp <= temp + 1;
  end if;
end process;
Q <= temp;
end behavior;
```

1011

Q. What is meant by FPGA ? Describe the FPGA Architecture ?

Ans FPGA (Field Programmable gate array)
An FPGA is a Semiconductor device that can be Programmed and Reprogrammed to Perform Specific logic functions. unlike microprocessors which execute Software instructions Sequentially, FPGA Allow Parallel execution of multiple tasks. making them highly efficient for applications requiring high speed data Processing.

The FPGA Consists of Several key Compments.

① Configurable logic blocks (CLB's)

These blocks Contains logic elements like loop up tables, multiplexer and flip flops they Can be programmed to perform Various logic function.

② Inter Connects → A Network of Programmable routing channels that Connect CLB's.

Allow the designers to customize data flow b/w logic elements.

(3) Input / output blocks → Provide an interface between FPGA and external devices.
Support various Voltage levels and signaling standards.
These Interface the FPGA with the outside world, allowing it to receive input signals & send o/p signals.

(4) Specialized Blocks → FPGA's often include Specialized block like digital Signal Processing (DSP) blocks, blocks RAM's & embeded Processors, which can be used for specific applications.

(5) Global Switch matrix (GSM) → This is a N/W of interConnects that allows signals to be routed between different parts of the FPGA, including CLB's, DSP blocks, & RAM blocks.

(6) Routing channels → These are the paths that signals take to travel between different blocks within the FPGA.

(7) Logic elements → These are the basic building blocks of logic functions, ranging from simple Combinational logic to Complex embeded Processors.

(8) Configuration memory → FPGA's use Configuration memory to store the user's design. which is then used to reprogram the logic elements & Inter Connects.

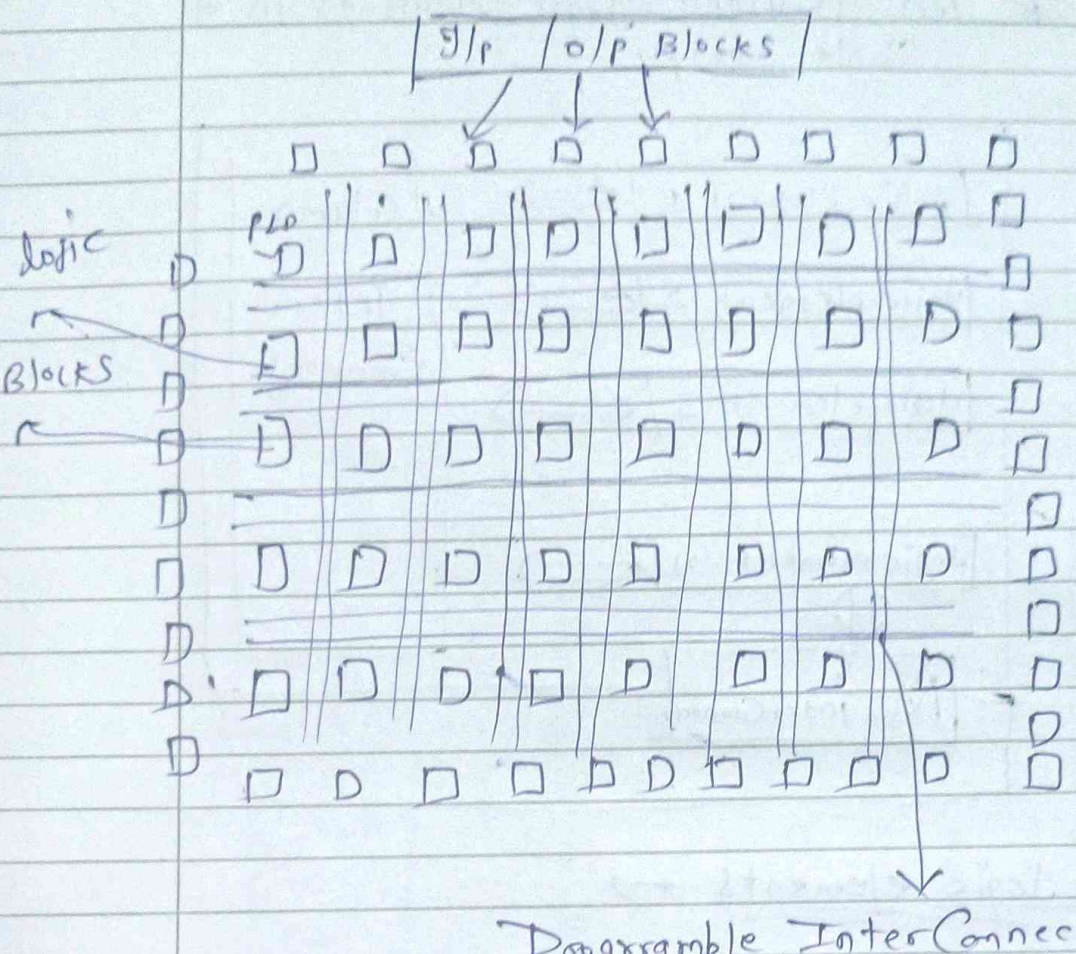
Types of FPGA's

FPGA's Can be based on different technologies such as SRAM (static Random Access memory), flash Memory or anti-fuse technology.

FPGA → field : ability of the gate arrange to be programmed for a specific function by the user.

Array → indicate a series of Columns & rows of gates that Can be programmed by the end users.

. larger & Complex devices.



* LUT → (look up table)

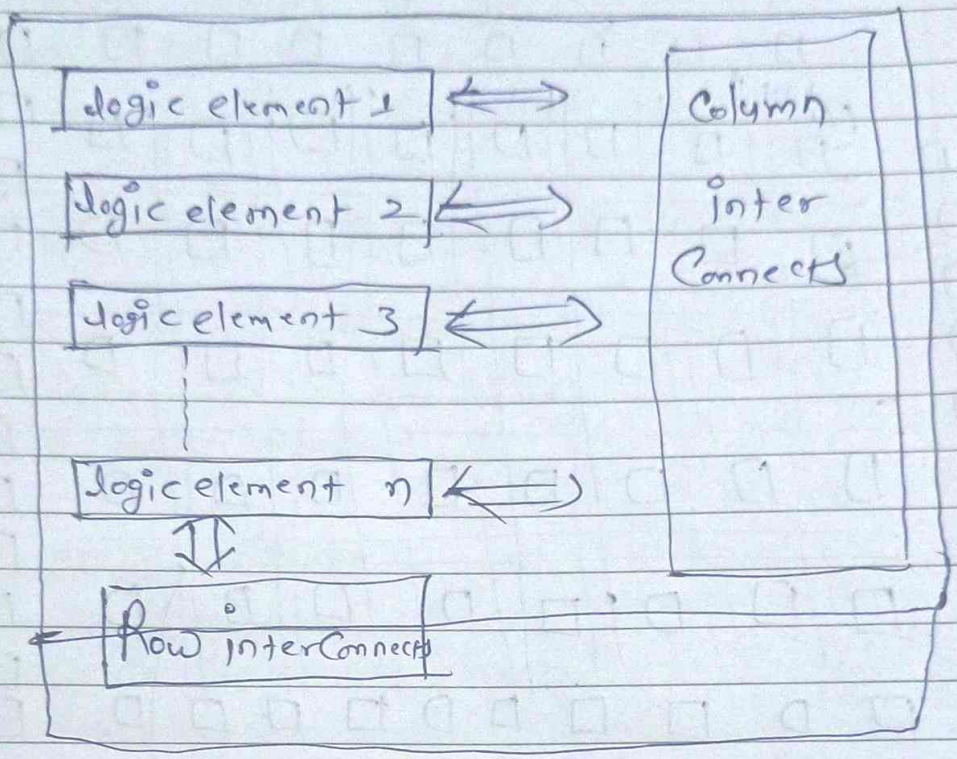
* memory devices that can be programmed to perform logic function.

* it replaces the AND/OR array logic in a CPLD.

Logic Blocks →

(1) it Contains Several logic elements.

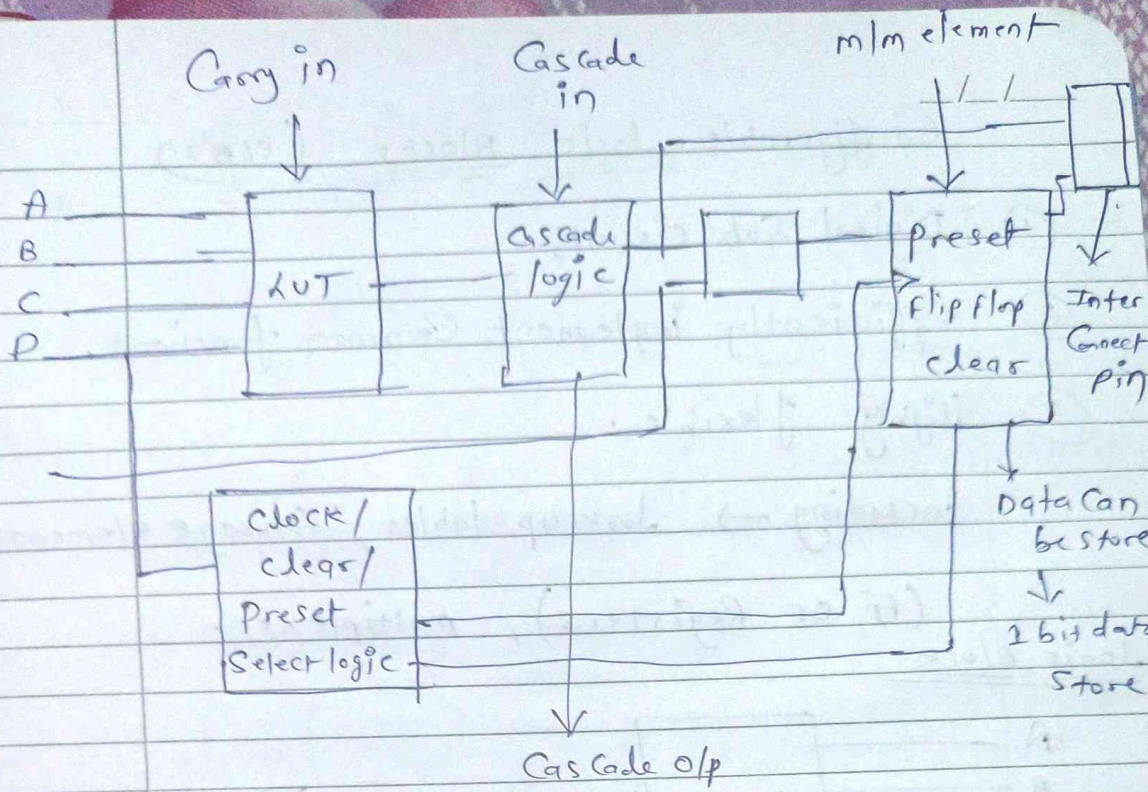
* Ten thousand logic elements in a single chip.



Logic elements →

* Each logic element Contains a 4-I/p LUT that Can be Programmed as logic function generators.

* Using Cascade logic, an LUT Can be expanded by Cascading with LUT's in other logic elements.



FPGA Advantages & Disadvantages

Advantages	Disadvantages
1. Do Anything (micro Processor & µC)	1) Expensive
2. Super fast	2) High Power
3. field Programmable	3) Voltatile / Boot time
4. Massively Parallel	4) High PIN Count
5. High Ilo Count	5) Complicated
	6) Many traps
	7) Complex tools
	8) Hard to choose/ Compare
	9) programming is not easy.