Generics & Configurations (Pg. 159)

* Equivalent to MACRO of 'C' language.

* Example : N i/p AND gate

```
entity AND_N_ip is
    generic (N: Natural);
    port (A: in bit_vector (1 to N);
          Z: out bit);
end AND_N_ip;
ar —————————————
begin
    process (A)
        variable AND_OUT: bit;
    begin
        AND_OUT := '1';
        for K in 1 to N
        loop
            AND_OUT: AND_OUT and A(k);
            exit when AND_OUT = '0';
        end loop;
        Z <= AND_OUT;
    end process;
end arch;
```

* A generic declares a constant object of mode IN (i.e. read only) & can be used in the entity declaration & its corresponding arch-bodies.
Generic's value can be specified in
1) Entity declaration
2) Component declaration
3) Configuration specification
4) Configuration declaration.

* Ex:

① entity NAND_G is
generic (M: integer := 2)
:
:
end NAND_G;

} Entity declaration

② Component declaration
archi

component AND_N_ip
generic (N: natural := 5)
:
:
end component;
:
end arch;

③ Component instantiation
archi

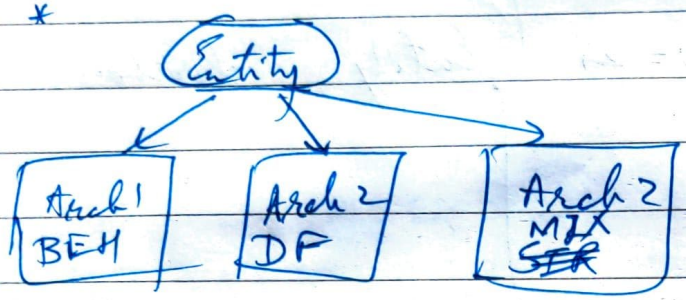
component NAND_G
generic (M: integer)
port( );
end NAND_G;

begin
N1 : NAND_G generic map (6) port map( ); 1st way 2nd way
N2 : NAND_G generic map (M => 10) port map( );
A1 : AND_N_ip generic map (N => 2) port map( );

N = 2 overrides N = 5

✡ Configuration (Pg. 163)

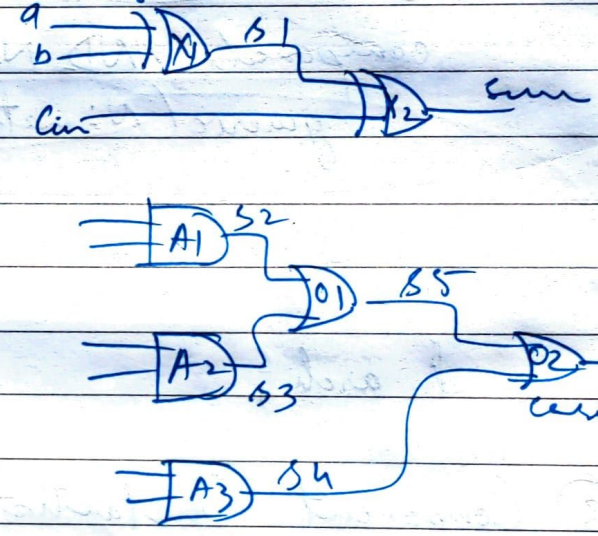
*



* Configuration can be used to bind an architecture body to its entity declaration

✡ Configuration Specification

* Example:




```
library MS_LIB;
library CMOS_LIB;

entity fa is
    port (a, b, cin
          sum, carry);
end

or ——— FA_STR
    component XOR2

    Compo AND2

    compo OR2
--Now binding
for X1, X2: XOR
    use entity work. XOR2 (XOR2BEH);
for A3: AND2
    use entity MS_LIB. AND2MS(AND2STR)
    port map (MS_B => A1, MS_Z => Z, MS_A => A0);
```

```
for all: OR2
    use entity CMOS_LJB.OR2_CMOS(OR2_STR);

for others: AND2
    use entity work.A_Gate(A_Gate_Body);
    port map(A0, A1, Z);

signal s1, s2, s3, s4, s5: bit;
begin
  X1: XOR2 port map(A, B, S1);
  X2:
  A1:
  A2:
  A3:
  O1:
  O2:
end FA_STR;
```

✱ Configuration Declaration (Pg. 171)

Why to use this?

→ Configuration specification have to appear in the architecture body. Therefore to change the binding, it is necessary to change the architecture.

Configuration Declaration is a ~~spc~~ separate design unit written outside the architecture body & does the same job as configuration specification.

Doubtful

Configuration Declaration is written ~~outs~~ after the ~~ar~~ end of architecture body.

→ I think its a separate file.

✱ Example:

```
library ...
    .
    .
entity full_adder is
    .
    .
arch--. FA_STR
    .
    .
    .
end ~~arch~~ FA_STR;
```

config. name — main entity name

```
configuration FA_CON of full_adder is
    for FA_STR                          -> main architecture name
        use work.all;
        for A1, A2, A3 : AND2
            use entity CMOS_LIB.BIGAND2 (AND2STR);
        end for;
        for others : OR2 -> use default OR2.vhd from work lib.
        end for;
        for all : XOR2
            use configuration work.XOR2CON;
        end for;                        another config. file that exists in work lib.
    end for;
end FA_CON;
```

✡ Practice Examples