Design Units

- Elements that make large scale design modelling easier
- Nominal propagation delays, min-max delays, setup and hold time constraints and spike detection can be described naturally.
- Generic Design
- Back Annotation
- Capability of defining new datatypes

Hardware Abstratction

- External View: Interface of device through which it communicate
- Internal View:
  - Functionality

This is Called Entity

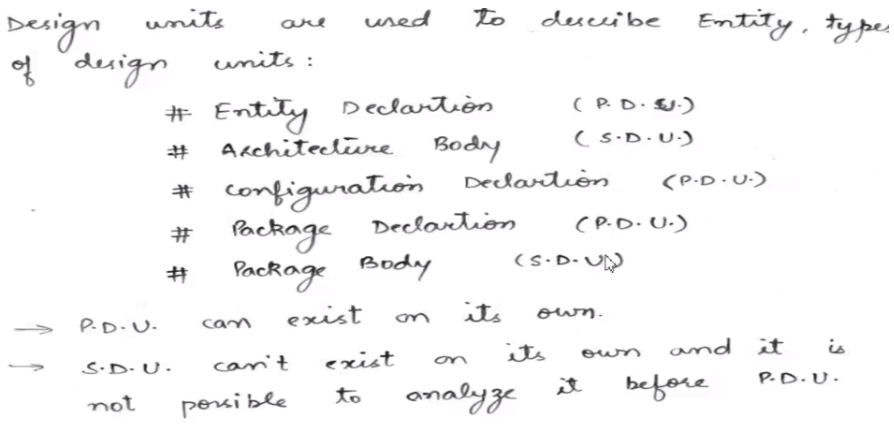


Figure 1: Design Units

Entity Modelling

Declaration [External Value]

+

Architecture Body [Internal View]

- Entity has one external view and one or more internal views
- Hardware Device may be represented by on or more entites

Entity Declaration

- Entity Consists of only ports
- Specifies name of entity being modelled and list of set inerface ports

Entity Declaration

Specifies name of entity being modelled and list of set of interface ports.

Ports are signals with which entity communicates with other models in its external enviornment.

```
entity HALF_ADDER is
    port ( A, B : in BIT;
           SUM, CARRY: out BIT);
end HALF_ADDER;
```

'Bit' can Take value '0' or '1'

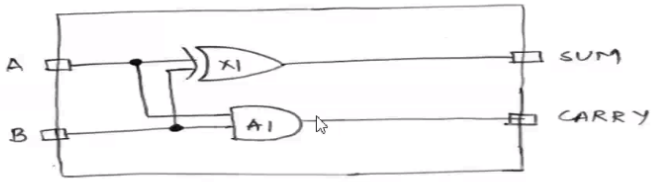


Figure 2: EntityDeclaration

Decoder Example

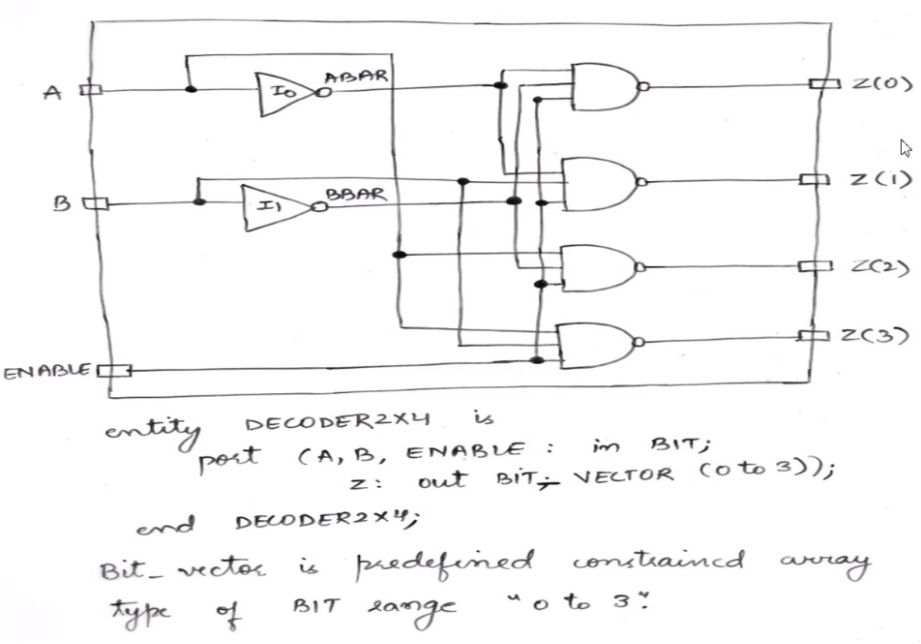


Figure 3: Decoder Example

Encoder Example

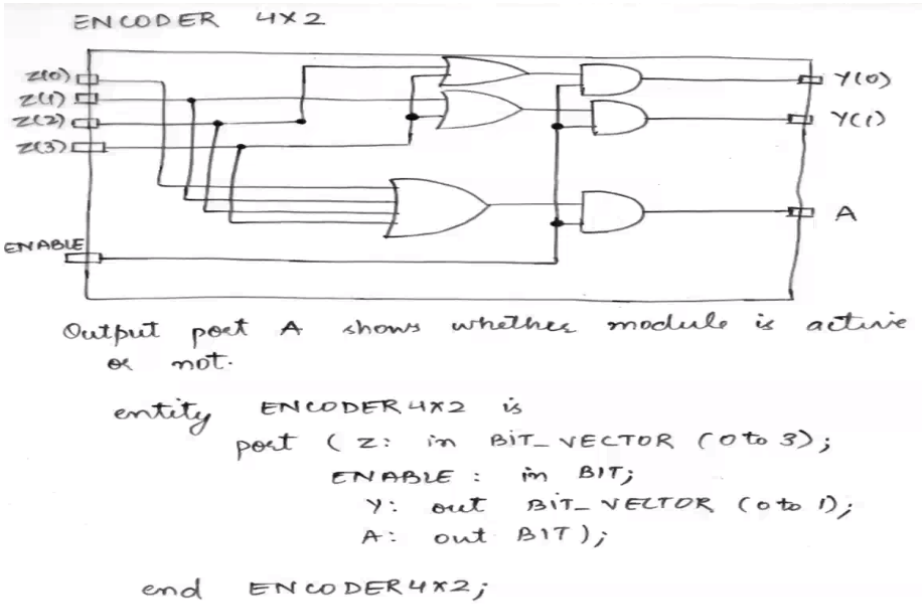


Figure 4: Encoder 4x2

- z(0) in MSB, z(3) is LSB
- y(0) is MSB

another keyword downto

eg.

bit_vector(3 down_to 0):

`- z(3) will be MSB, z(0) will be LSB`

In entity only I/O ports will be declared

- Internal Signals will not be declared in entity block

Architecture Body

REPRESTATION STYLES
– As interconnection of components
STRUCTURAL
– As concurrent assignment statments.
DATAFLOW
– As sequential assignment statments
BEHAVIORAL
– combination of above three.

STRUCTURAL STYLE OF MODELLING

```
architecture HA_STRUCT of HALF_ADDER is
    component XOR2
        port (X, Y: in BIT; Z: out BIT);
    end component;
    component AND2
        port (L, M: in BIT; N: out BIT);
    end component;
    begin
        X1: XOR2 portmap (A, B, SUM);
        A1: AND2 portmap (A, B, CARRY);
    end HA_STRUCT;
```

Figure 5: Architecture Body

Styles of repr

- Structural:
  - interconnection of components
- DataFlow:
  - concurrent assignment statements
- Behavioral:
  - sequential assignment
- Combination of all

Structural Style Of Modelling

Half Adder

```
architecture Half_adder of HALF_ADDER is
    component XOR2
        port (x, y: in BIT; z: out BIT);
    end component;

    component AND2
        port(L, M: in BIT; N: out BIT);
    end component;

    begin
        X1: XOR2 portmap(A, B, SUM);
        A1: AND2 portmap(A, B, CARRY);
    end Half_adder;
```

Internal Signal

Encode and Decoder Example

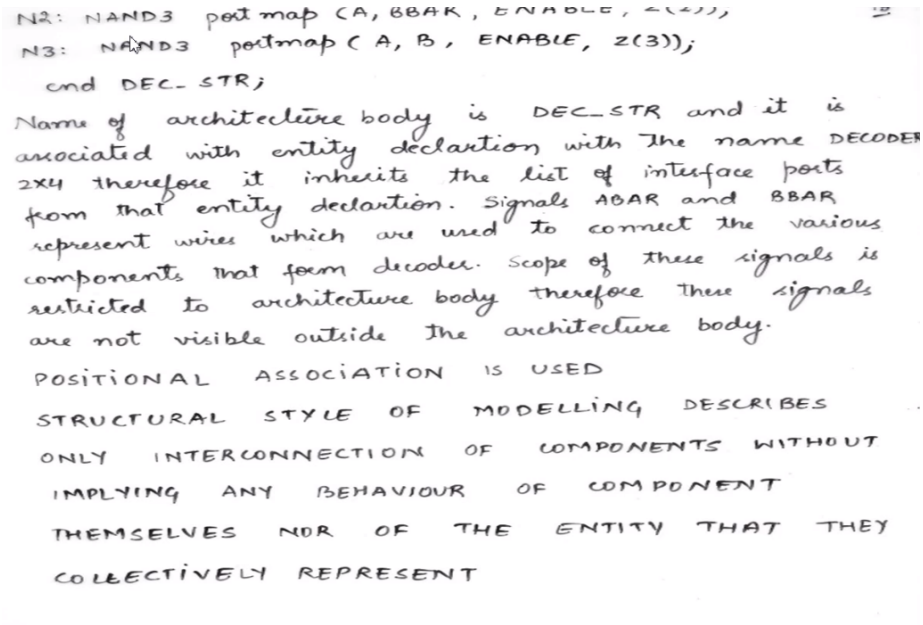
Architecture body has 2 parts 11

- Declartive
- Statment
- XOR2 & AND2 may be predefined or they will later be bound to components in Library.
- X1 & A1 and Labels for component instantiation
- Seperate Entity models are / may be required for XOR2 & AND2.

```
architecture EN_ARC of ENCODER4X2 is
  component OR2 is
     port (L1, L2: in BIT;  L3 : out BIT);
   end component;
  component AND2 is
     port ( N1, N2: in BIT;  N3: out BIT);
   end component;
   component OR4 is
     port ( M1, M2, M3, M4 : in BIT;
                    M5: out BIT);
    end component;
```

12

```
signal  A1, A2, A3 :  BIT;
  begin
   P0:   OR2   portmap ( z(1), z(3), A1);
   P1:   OR2   portmap (z(2), z(3), A2);
   P2:   AND2  portmap (A1, ENABLE, Y(1));
   P3:   AND2  portmap ( A2, ENABLE, Y(0));
   P4:   OR4   portmap (z(1), z(2), z(3), z(4), A3);
   P5:   AND2  portmap (A3, ENABLE, A);
   end  EN_ARC;
architecture  DEC_STR is DECODER2X4 is
   component INV
      port (PIN : in BIT;  POUT : out BIT);
    end component;
   component NAND3
     port ( D0, D1, D2 : in BIT;  D2 : out BIT);
    end component;
    signal   ABAR, BBAR; BIT;
    begin
      V0:  INV   port map (A, ABAR);
      V1:  INV   port map (B, BBAR);
      N0:  NAND3  port map (ABAR, BBAR, ENABLE, Z(0));
      N1:  NAND3  port map (ABAR, B, ENABLE, Z(1));
```



DataFlow Style of Modelling

- Entity is expressed using concurrent signal assignment statments
- Structure of entity is not specified explicityly byt can be deduced implicitly

```
architecture HA_concurrent of HALF_ADDER is
  begin
      SUM <= A XOR B after 8ns;
      Carry <= A AND after 4ns;
  end HA_concurrent
```

- concurrent signal assignment are executed when there is ann envent on any signal on RHS
- To include time delay 'after' clause is used. architecture HS_concurrent of HALF_SUBTRACT is :

```
architecture HS_concurrent of HALF_SUBTRACT is
signal BBAR;
begin
    DIFF <= A xor B after 8ns;
    BORROW <= A and BBAR;
    BBAR <= not B;
end HS_concurrent;
```

- order of writing doesn't matter, since all events are executed

Behavioral Style

- Statements are executed sequentially in a specified order

Syntax

```
process (sensitivity list)
    begin
    --
    --
    --
    end process
```

Mixed Style

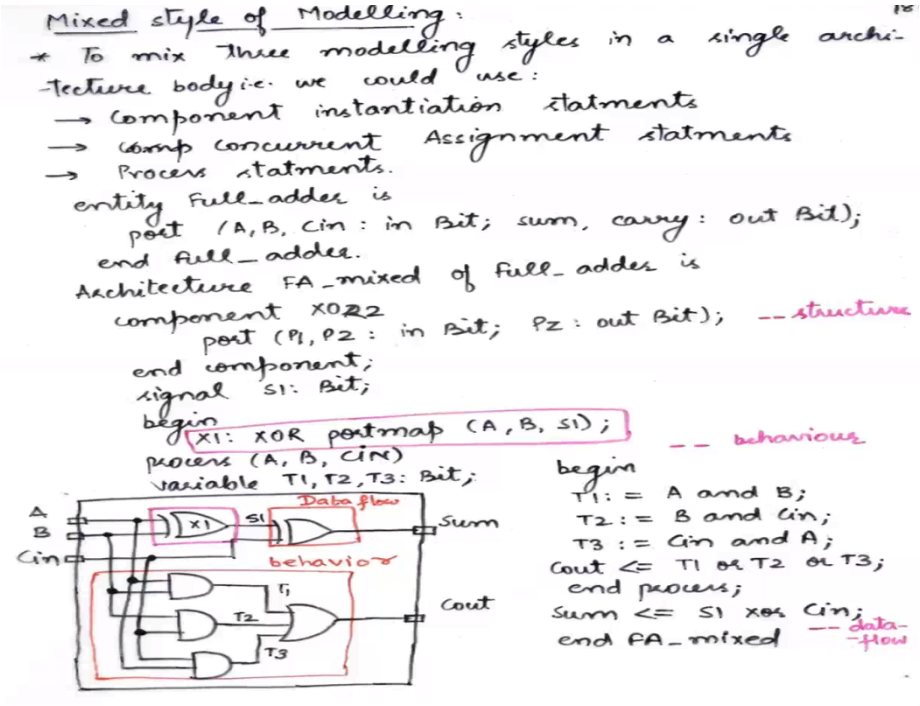


Figure 6: Mixed Style of Modelling

Configuration Declaration

- Used to select one of possibly many architecture bodies:
  - change archtecture within same code
- To bind components used in the architecture body:
  - components is the xerox copy of component
  - used to bind component to entity

Package Declaration

- used to sotre a set of common declaration such as components, tupes, procedures and functions.
- **use* keyword

Package body

- used with package declaration
- used to store the definition of functions in the corressponding package declaration