VHDL

Imtiaz Ahammad

Department of Electronics
Dinabandhu Andrews College
University of Calcutta

What is VHDL?

- **HDL** stands for

  ‘**Hardware Description Language**’

- **VHDL** is **VHSIC HDL**
  - **VHSIC** stands for *Very High Speed Integrated Circuits*

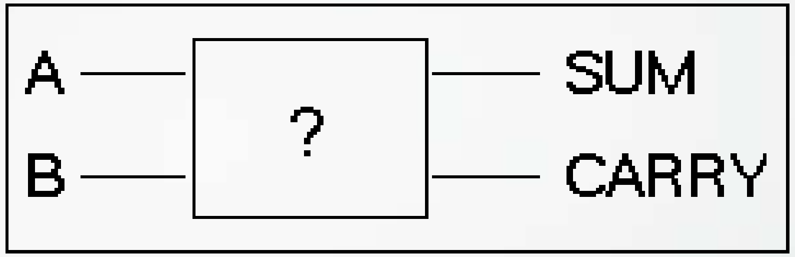
What is *Hardware Description* and why do we need a special language for that?

VHDL DESIGN ELEMENT

- Entity
- Architecture

ENTITY

- Interface description
- No behavioral definition
- Declares name of the entity and list of the interfacing ports



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

Entity declaration example

```
Entity AOI is
   port (A,B,C,D: in BIT;
        Z: out BIT);
end AOI;
```

Note the semicolons

A, B, C, D, Z and AOI are user defined symbols

- Describes a circuit with 4 input and 1 output port
- Input/output signals are of type BIT

- Circuit may be And-Or-Invert
- Alternatively, it could be 16 X 1 bit memory
- or almost anything else !

ARCHITECTURE

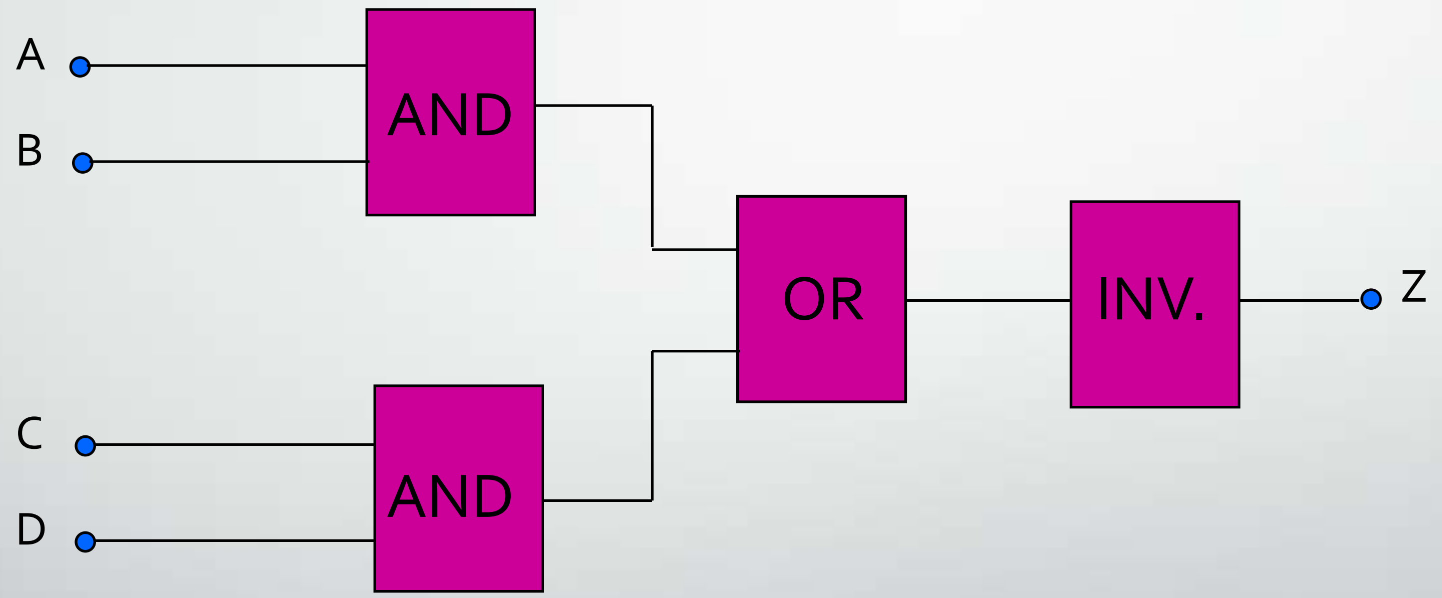
- Gives the internal description of the circuit.
- Three types of modeling style available
    - Dataflow modeling
    - Behavior Modeling
    - Structural Modeling
- Mixed Modeling

ARCHITECTURE

- **Dataflow Modeling:** Flow of data through the entity is expressed using a concurrent signal assignment statement.
- **Behavioral Modeling:** The behavior of an entity is described as a set of statement that are executed sequentially in specific order.
- **Structural Modeling:** Described by the interconnection of components.

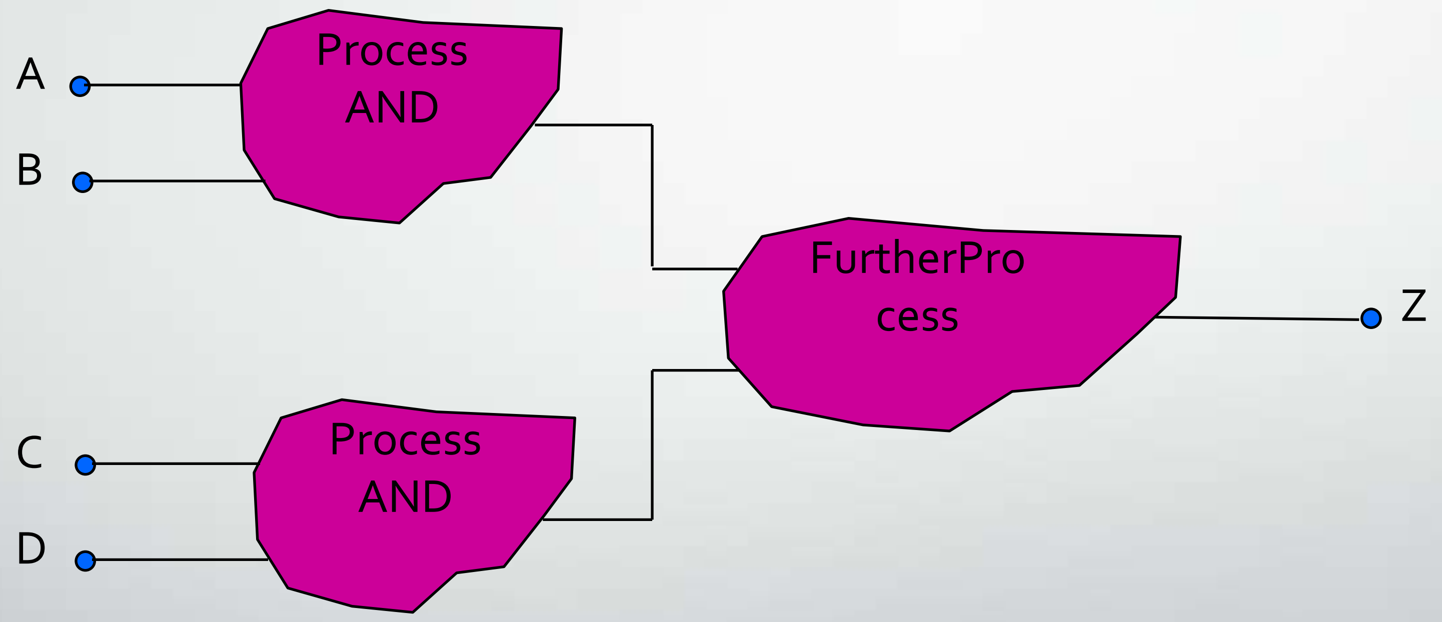
Structural

1. Specify Terminals
2. Put the building blocks
3. Connect Wires



Dataflow

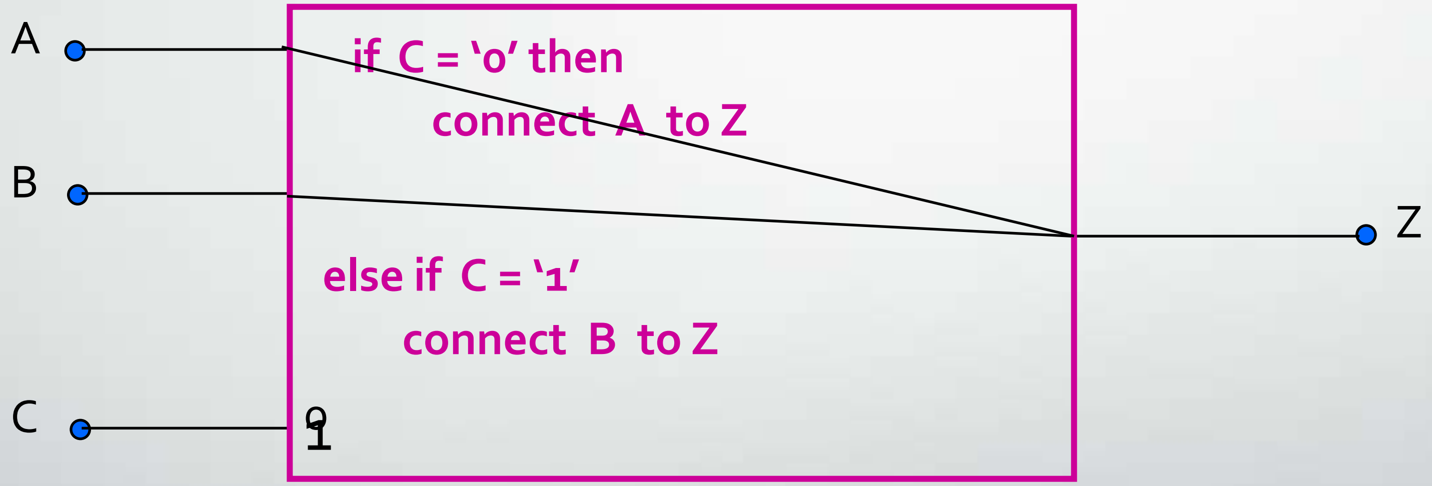
1. Specify Input Data set
2. Process data following Boolean Logic
3. Send to next process and continue



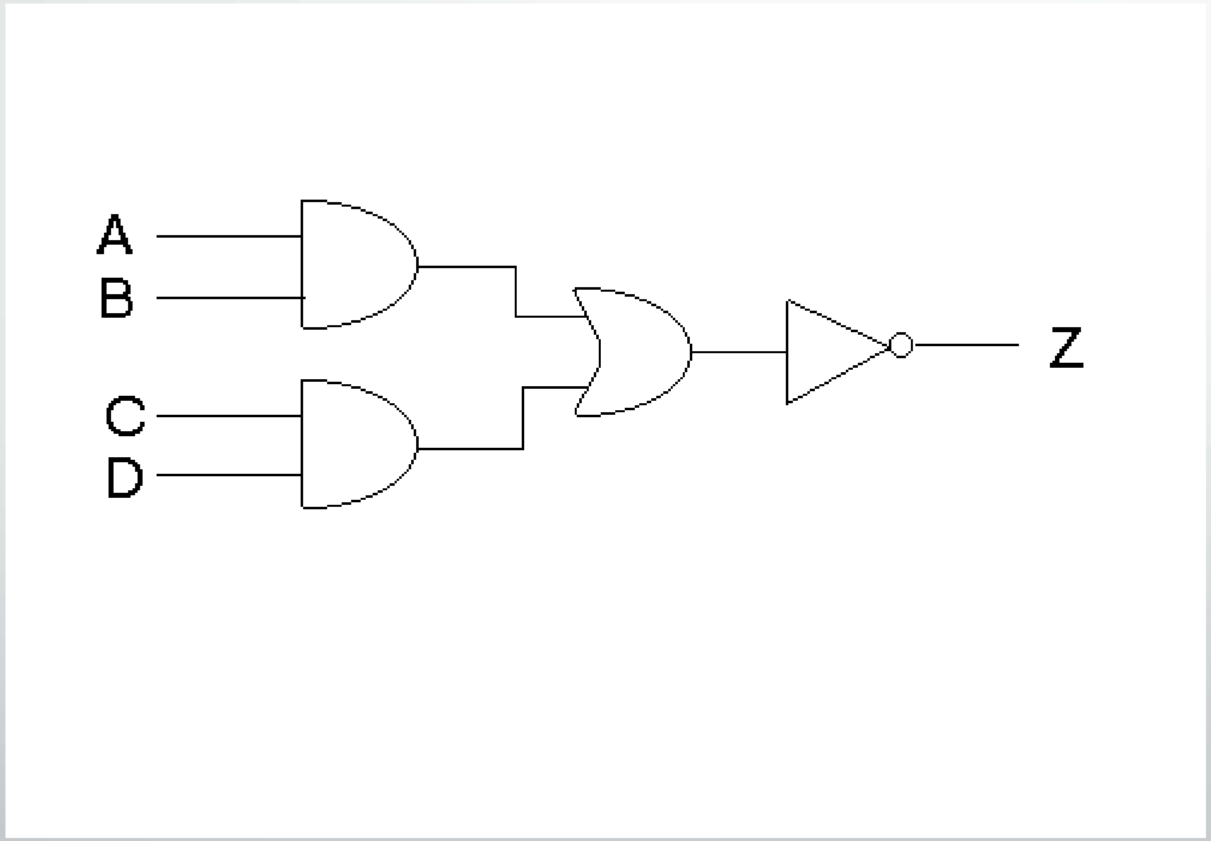
Behavioral

1. Specify Input/Output Data set

2. State what you want eg.



Consider the *And-Or-Invert* circuit



Structural Modeling

```
architecture struct_view of AOI is
    component AND2
      port (X,Y:in BIT; Z:out BIT);
    end component;
    component OR2
      port (X,Y:in BIT; Z:out BIT);
    end component;
    component INVERT
      port (X:in BIT; Y:out BIT);
    end component;
    signal S1, S2, S3: BIT;
```

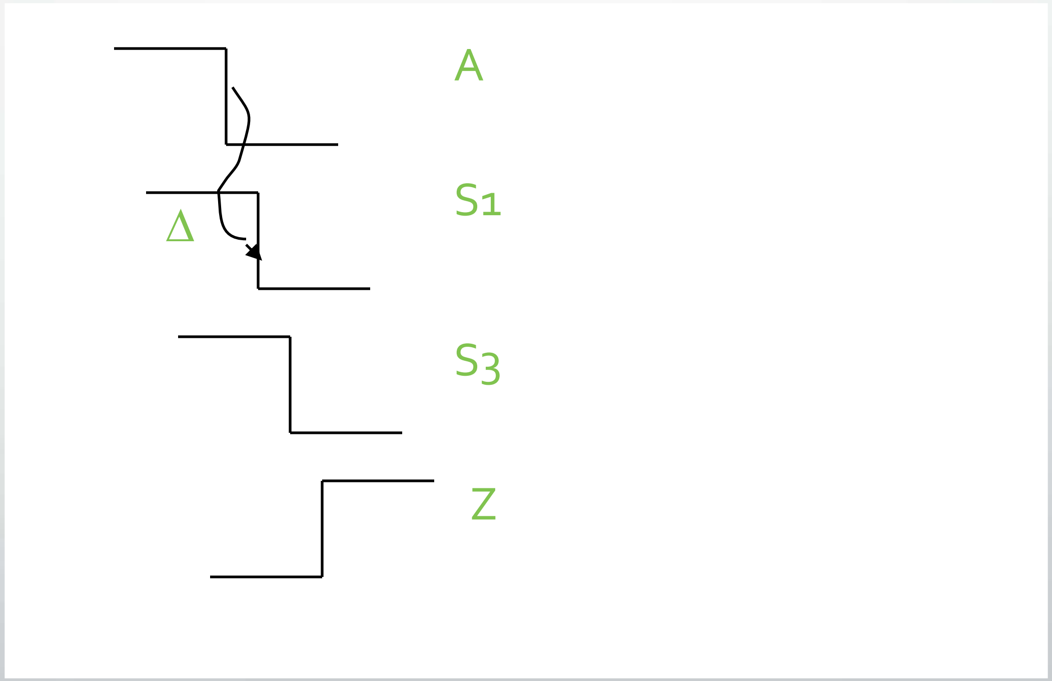
declarations

```
begin
    A1: AND2 port map (A,B,S1);
    A2: AND2 port map (C,D,S2);
    O1: OR2 port map (S1,S2,S3);
    I1 : INVERT port map (S3,Z);
end struct_view;
```

Component instantiation

Dataflow modeling

```
Architecture aoi_dataflow of aoi is
    signal s1, s2, s3 : BIT;
begin
    s1 <= A and B;
    s2 <= C and D;
    s3 <= s1 or s2;
    z <= not s3;
end aoi_dataflow;
```



Behavioral Model

- Set of statements that are executed sequentially in the specified order
- The set is specified inside a process statement
- A process statement is a concurrent statement that appears within the architecture body

Similar to programming language

Behavioral AOI

```
Architecture aoi_behavior of AOI is
begin
    process (A,B,C,D)
    variable    S1,S2: bit;
    begin
      S1 <= A AND B;
      S2 <= C AND D;
      Z <= NOT (S1 OR S2);
    end process;
end aoi_behavior;
```

VHDL Language and Syntax

Imtiaz Ahammad

VHDL Language and Syntax

- General
- Identifiers
- Naming Convention

General

- Case insensitive
- Comments: '--' until end of line
- Statements are terminated by ';'
  (may span multiple lines)
- List delimiter: ','
- Signal assignment: '<='
- User defined names:
  - letters, numbers, underscores
  - start with a letter

IDENTIFIER

Two type of Identifier.

- Basic Identifier
- Extended Identifier

Basic Identifier

- Sequence of character

❖ Upper Case Character: A……..Z

❖ Lower Case Character: a……z

❖ Digit: 0…….9

❖ Under Score: _

❖ In basic identifier first character must be a letter and last character may not be a underscore.

❖ Lowercase and Uppercase letters are equivalent

❖ No double underscore

❖ Eg. CouNT, count, COUNT, r2d2,Ram_address

Extended Identifier

- Sequence of character written between two backslash
- Character includes: ., !,@, and $
- Within extended identifier lower and uppercase characters are unique.
- Eg. \TEST\, \2for$\

Data objects

- Data objects holds value of a specific type
  - Constant
  - Signal
  - Variable
  - File

Signal

- ❖An object of signal class holds a list of values;
- ❖Values are current values of signal and future values of signal

```
signal mySignal: bit;        -- an example signal

signal databus: bit_vector (7 downto 0);

signal gatedelay: Time:=10 ns;
```

Variable

- ❖An object of variable class holds a different values;
- ❖Different values are assigned at different time;

```
variable myvariable: bit;        -- an example variable

variable databus: bit_vector (7 downto 0);
```

Constant

- An object of constant class holds a single value of a given type;
- Value is assigned before simulation .
- Value can not be changed during the course of simulation

- **constant** bus_width: **integer :=**8;

File

❖ An object of file class is declared using file declaration statement;

file *filename* **:** *file-type-name* **[[open** mode **] is** string-expression**];**

Example:

file *results* : *text* **[[open** read **] is** ".\results.txt"**];**

Data Types

- Every data object can hold a value that belongs to a set of following data types.

- **Subtype**

  A type with constraint.

  The type is the base type of the subtype

Example:

subtype MYINT **is** INTEGER **range** 10 **to** 40;

Data Types

- **Scalar Type**
  - Enumeration Type
  - Integer Type
  - Floating point Type
  - Physical Type

Enumeration Type

- A type that has a set of user-defined values consisting of identifiers and character literals.
- Relational operators can be used on these values

- type MVL is (‘U’, ‘0’, ‘1’, ‘Z’);

- signal control_A: MVL;

Type STD_ULOGIC

```
Type STD_ULOGIC is (
    'U'         -- Uninitialized
    'X'         -- Forcing unknown
    '0'         -- Forcing 0
    '1'         -- Forcing 1
    'Z'         -- High impedance
    'W'   -- Weak unkown
    'L'         -- Weak 0
    'H'         -- Weak 1
    '-'         -- Don't care
  );
```

Imtiaz Ahammad

Other Enumeration Type

- BIT ➔ ‘0’, ‘1’;
- BOOLEAN ➔ ‘FALSE’ , ‘TRUE’;
- SEVERITY_LEVEL ➔ ‘NOTE’, ‘WARNING’, ‘ERROR’, ‘FAILURE’;

OPERATORS

- LOGICAL
- RELATIONAL
- SHIFT
- ADDING

LOGICAL OPERATORS

- and, or, nand, nor, xor, xnor, not

RELATIONAL OPERATORS

- =, /=, <, <=, >, >=

SHIFT OPERATOR

- sll, srl, sla, sra, rol, ror
- Data type **BIT** or **BOOLEAN**

- “1001010” sll 2 ➔ “0101000”;
- “1001010” srl 3 ➔ “0001001” ;
- “1001010” sla 2➔ “0101000” ;
- “1001010” sra 3➔ “1111001” ;
- “1001010” rol 2➔ “0101010”;
- “1001010” ror 3➔ “0101001” ;

ADDING OPERATOR

- +,-,&
- ‘0’&’1’→ “01”;
- Many separate signals can be grouped into single bus.
- Yout<= a & b;

MULTIPLYING OPERATOR

- *,   /,   mod,   rem
- mod, rem operate on integer data type
- A rem B→ A-(A/B)*B;
- A mod B→ A-B*N ; -- for some integer N
- 7 mod 4→ 3;
- (-7) rem 4→ -3
- 7 mod (-4) → -1;
- (-7) rem (-4)→ -3

Advantages and Disadvantages

- Data flow Modeling Advantages
  - The flow of data in the circuit can be determined by examining the VHDL code.
  - Allow to make an intelligent guess as to how the actual logic will appear during synthesize of the circuit.
  - Influence on the synthesized hardware.

❑ The dataflow style works fine for small and primitive circuits. But as circuits become more complicated, it is usually advantageous to switch to behavioral style models.

Behavioral Modeling

- No details as to how the design is implemented in actual hardware.
- Reflects "how the circuit outputs will react to (or behave) the circuit inputs". It is the VHDL synthesizer tool that decides the actual circuit description.
- Behavioral modeling is the ultimate "black box" approach to circuit designing.