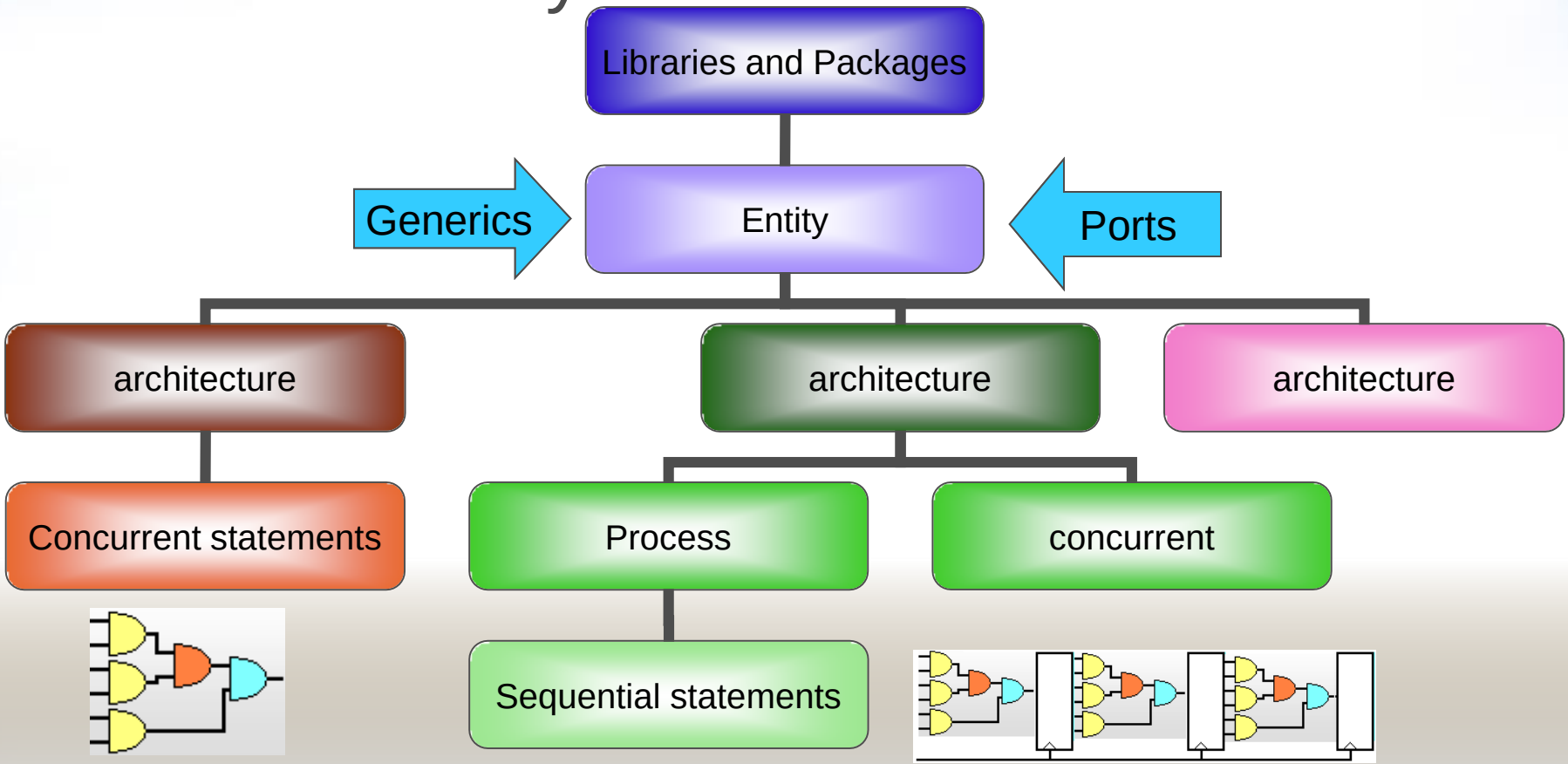
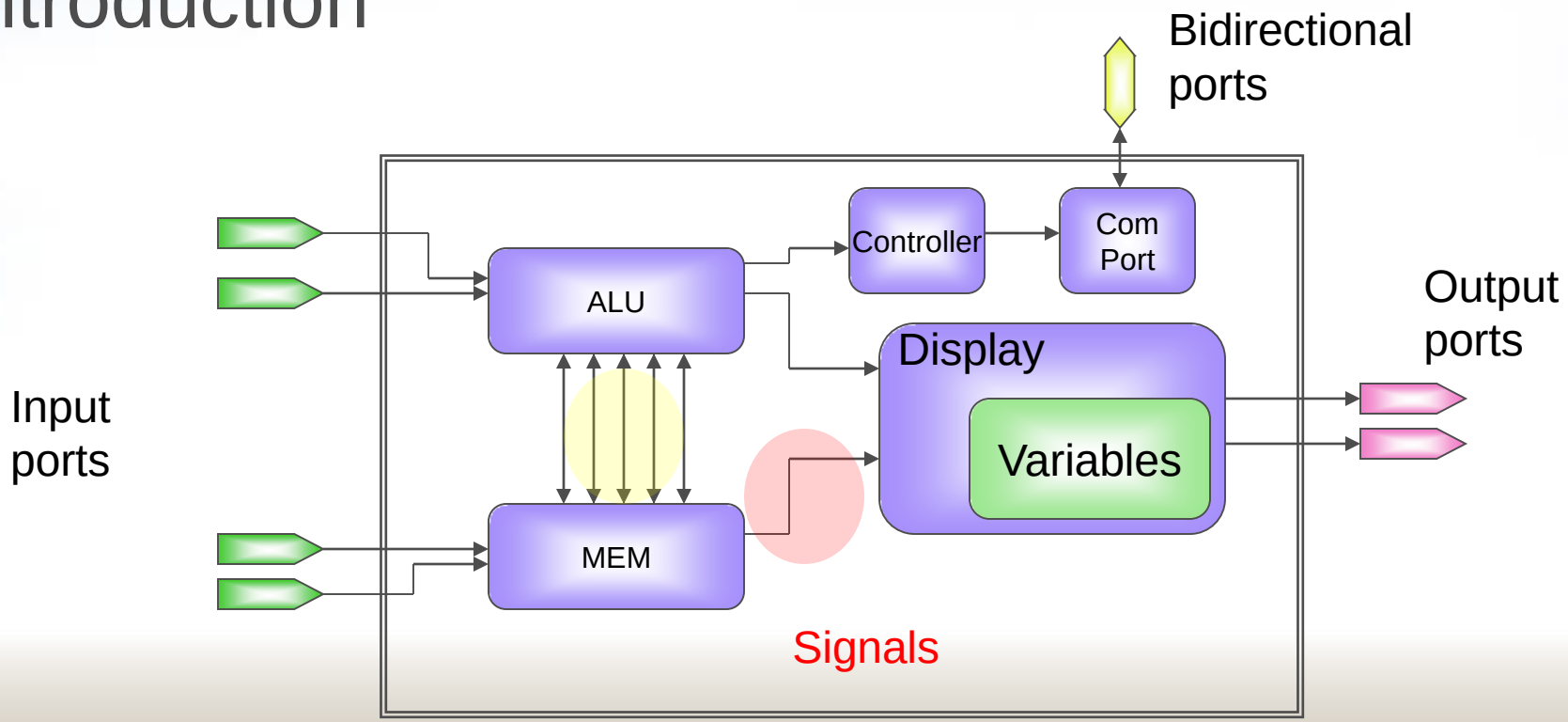
DATA TYPES

By: M. SHIPLE

Previous summary

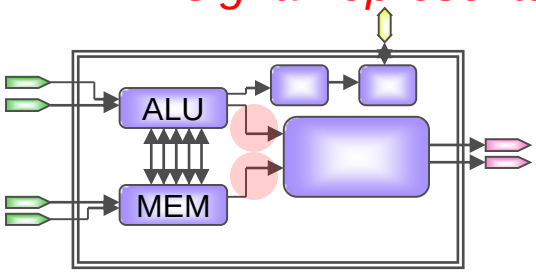


Introduction



Signals

“signal represents circuit interconnects (wires)”

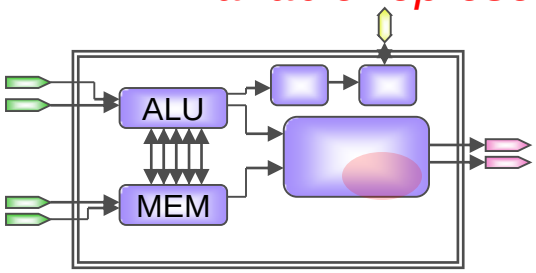


SIGNAL name : Signal_type [range] [:= initial_value];

```
SIGNAL control: STD_LOGIC := '0';
SIGNAL count: INTEGER RANGE 0 TO 100;
SIGNAL y: STD_LOGIC_VECTOR (7 DOWNTO 0);
```

Variables

"Variable represents storage element"



VARIABLE name : Signal_type [range] [:= initial_value];

```
VARIABLE  control:    STD_LOGIC := '0';
VARIABLE  count:    INTEGER RANGE 0 TO 100;
VARIABLE   y: STD_LOGIC_VECTOR (7 DOWNTO 0);
```

Variables vs Signals

Represent

SIGNAL control: STD_LOGIC := '0';

Assignment

VARIABLE Control: STD_LOGIC := '0';

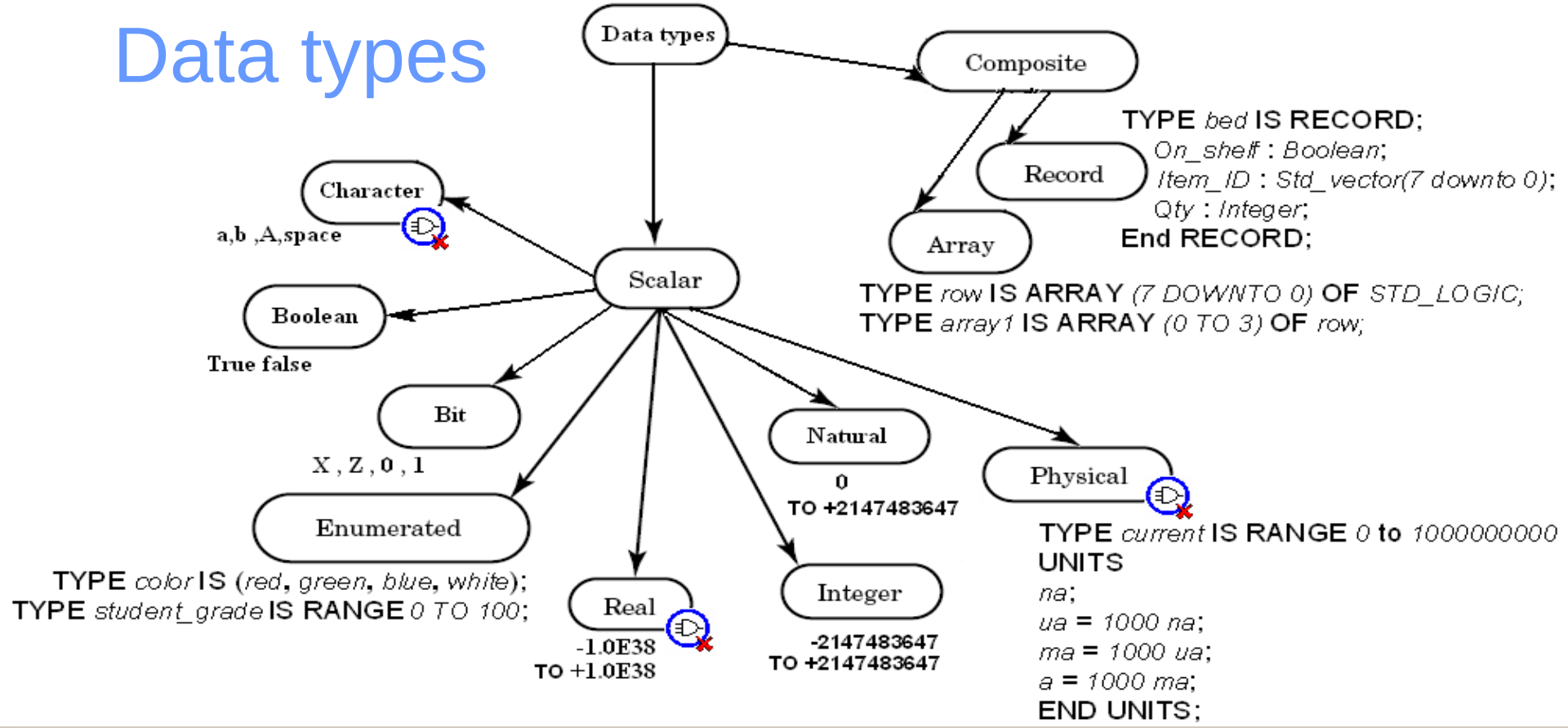
Scope

immediate

not immediate

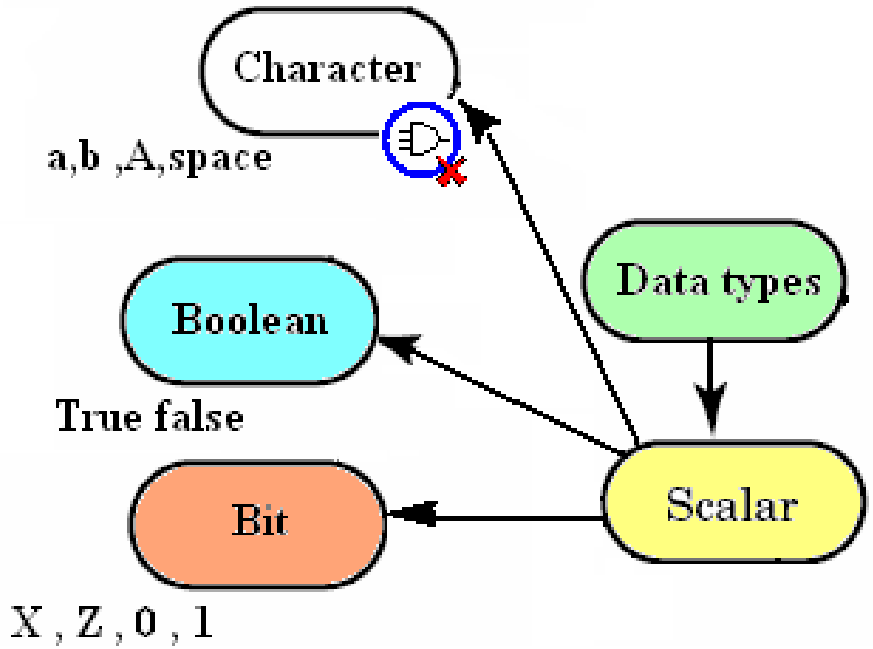
Behavior

Data types


Data types
Composite
Record
Array
Scalar
Character
a,b ,A,space
Boolean
True false
Bit
X , Z , 0 , 1
Enumerated
Real
-1.0E38
TO +1.0E38
Integer
-2147483647
TO +2147483647
Natural
0
TO +2147483647
Physical
TYPE bed IS RECORD;
On_shelf : Boolean;
Item_ID : Std_vector(7 downto 0);
Qty : Integer;
End RECORD;
TYPE row IS ARRAY (7 DOWNTO 0) OF STD_LOGIC;
TYPE array1 IS ARRAY (0 TO 3) OF row;
TYPE color IS (red, green, blue, white);
TYPE student_grade IS RANGE 0 TO 100;
TYPE current IS RANGE 0 to 1000000000
UNITS
na;
ua = 1000 na;
ma = 1000 ua;
a = 1000 ma;
END UNITS;

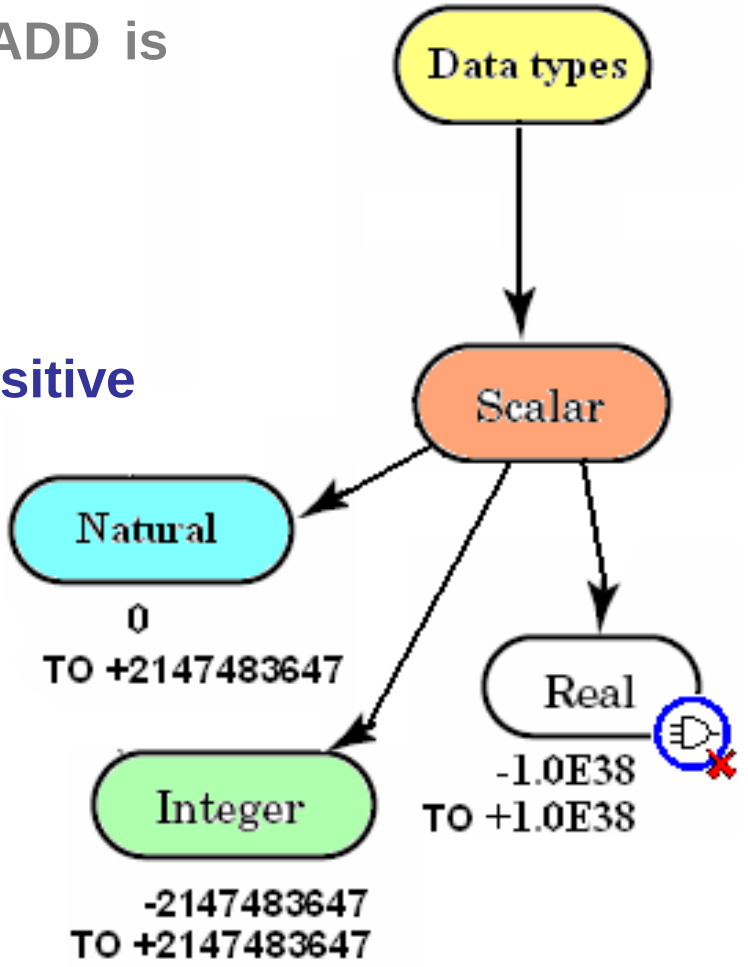
Simple Data types

```
architecture struct of  HALFADD is
Signal sumA: std_logic;
Signal sumAA: std_logic_vector(3 downto 0);
Signal carry : Boolean;
Signal Sumb : character;
begin
     sumA   <= '1';
     sumAA <= " 1011";
     carry    <= (A>B );
     SUMb  <= "Ahmed";
end  struct ;
```



Numerical Data types

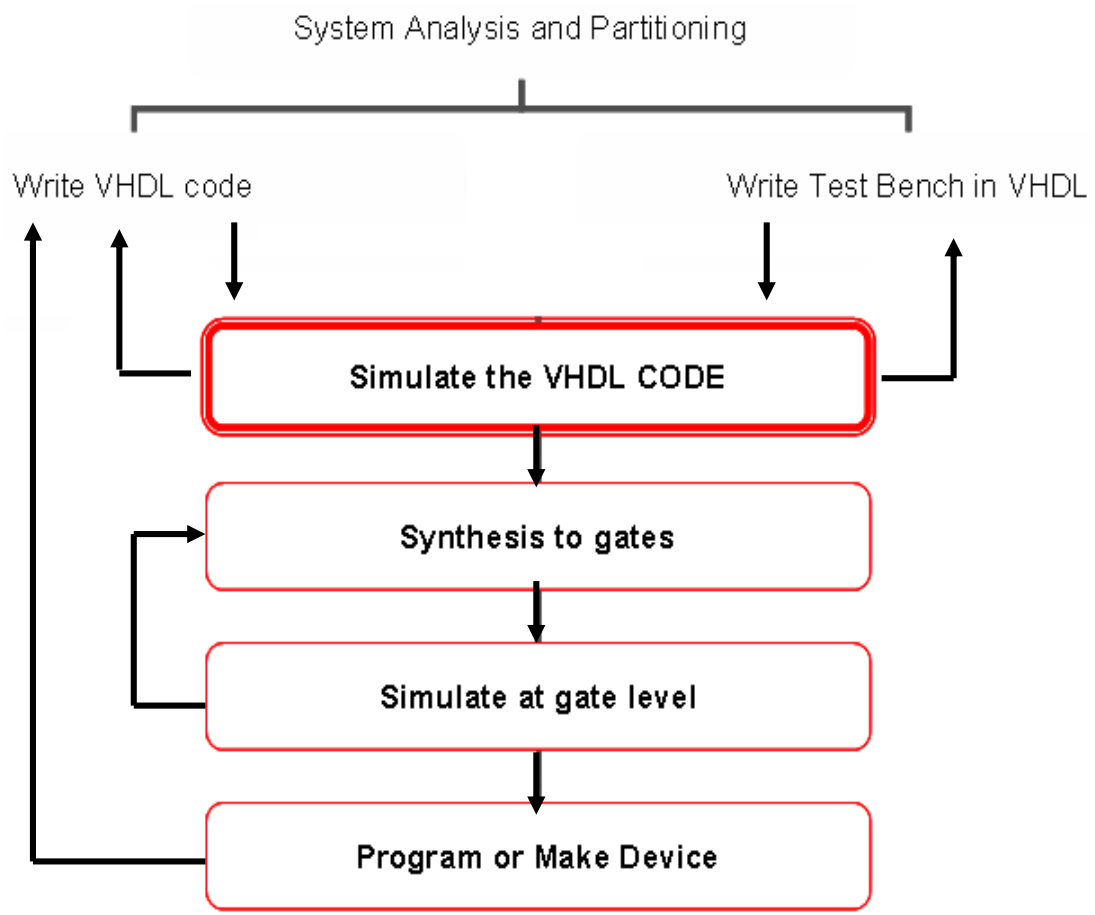
```
architecture struct of HALFADD is

Signal sum: Real;

Signal carry : Integer;

Signal SumA : Natural; --positive

begin

        SUM   <= 1.5 + 2.3;
      CARRY   <= 3 - 5;
       sumA   <= 2** 2

end struct ;
```



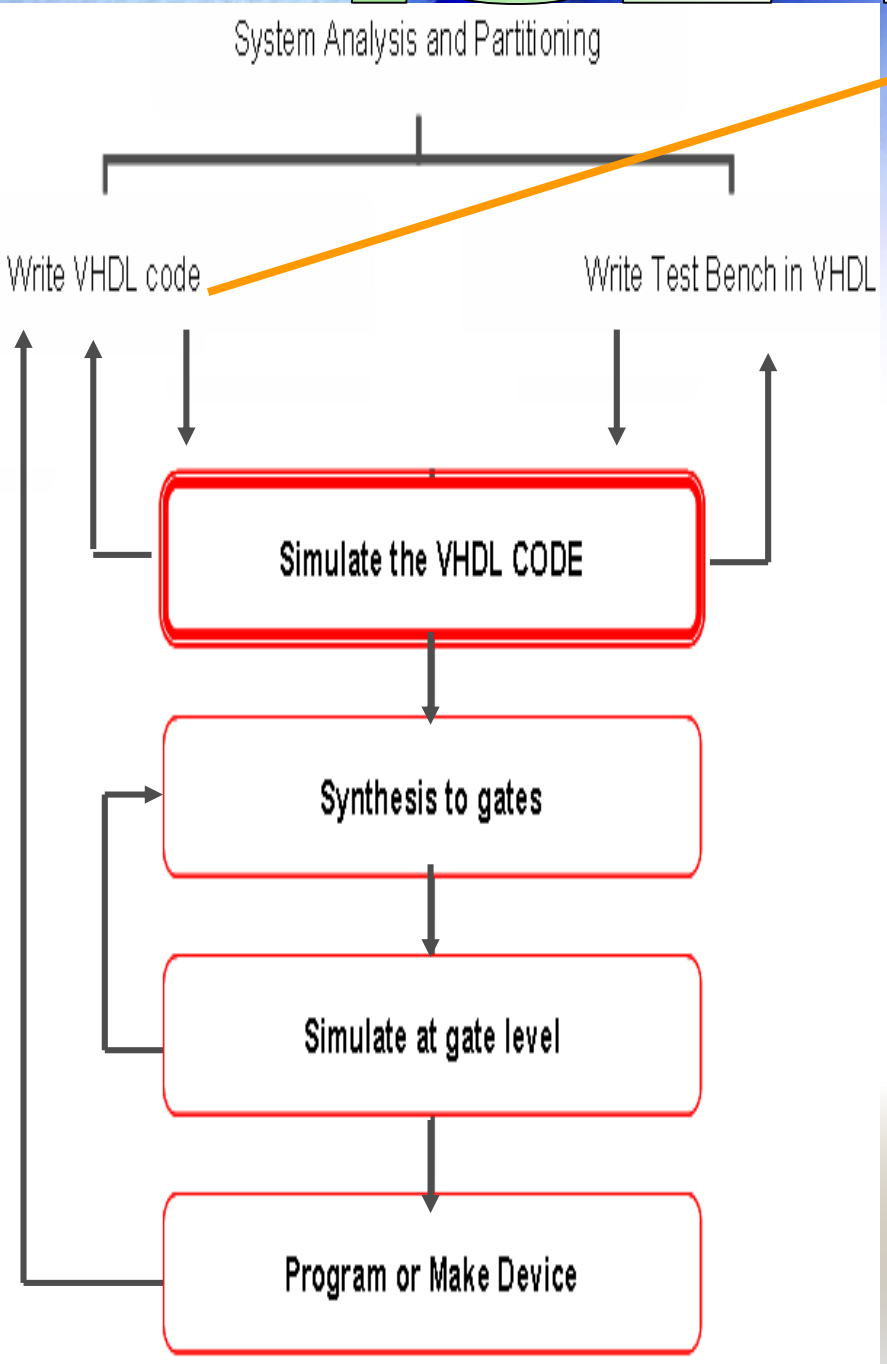
Exercise

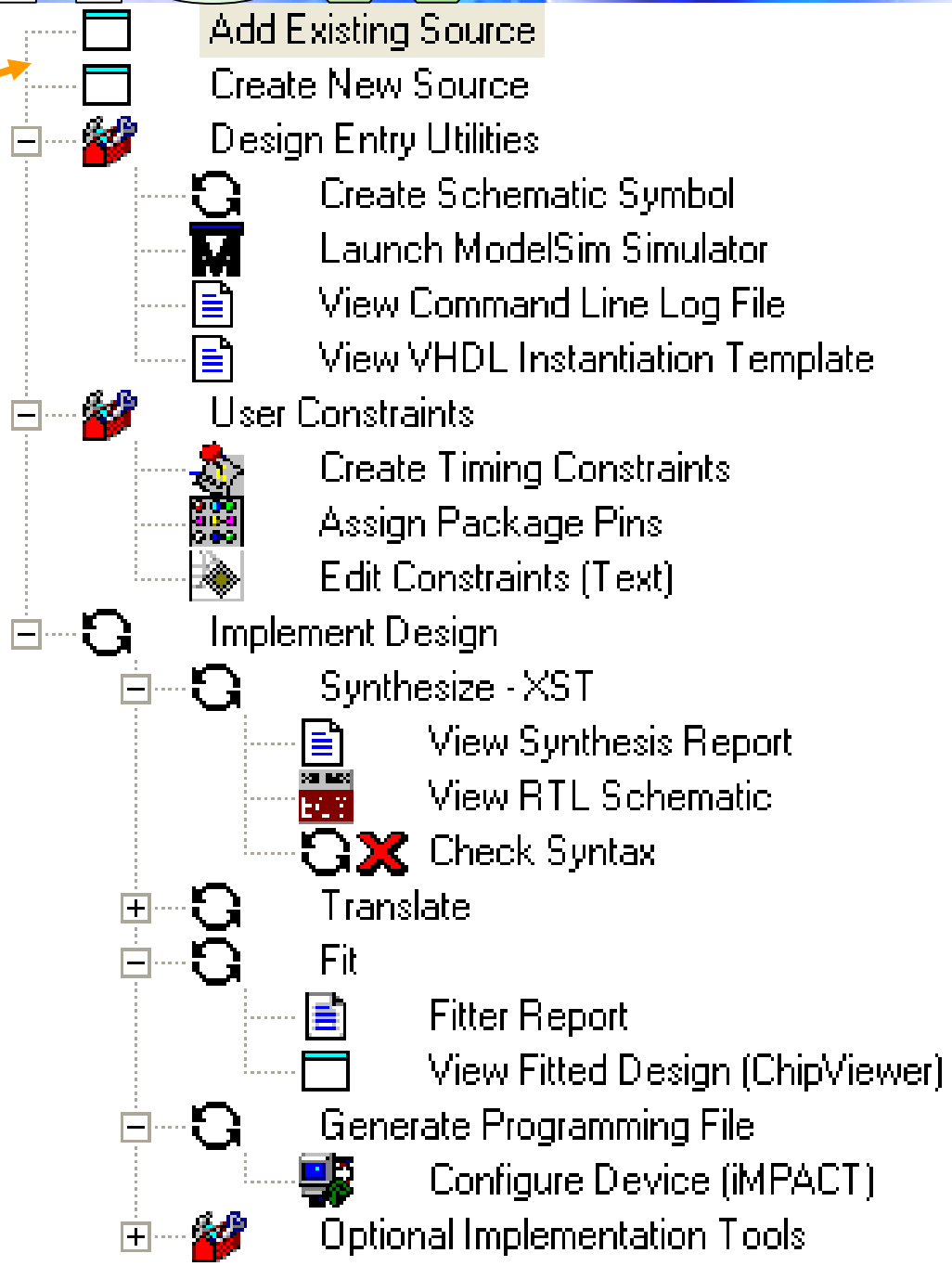
```
a       <= '0';
b       <= '1';
c       <= 'Z';
avec    <= "00111010";
bvec    <= X"3A";
cvec    <= X"3" & X"A";
```

Lab1

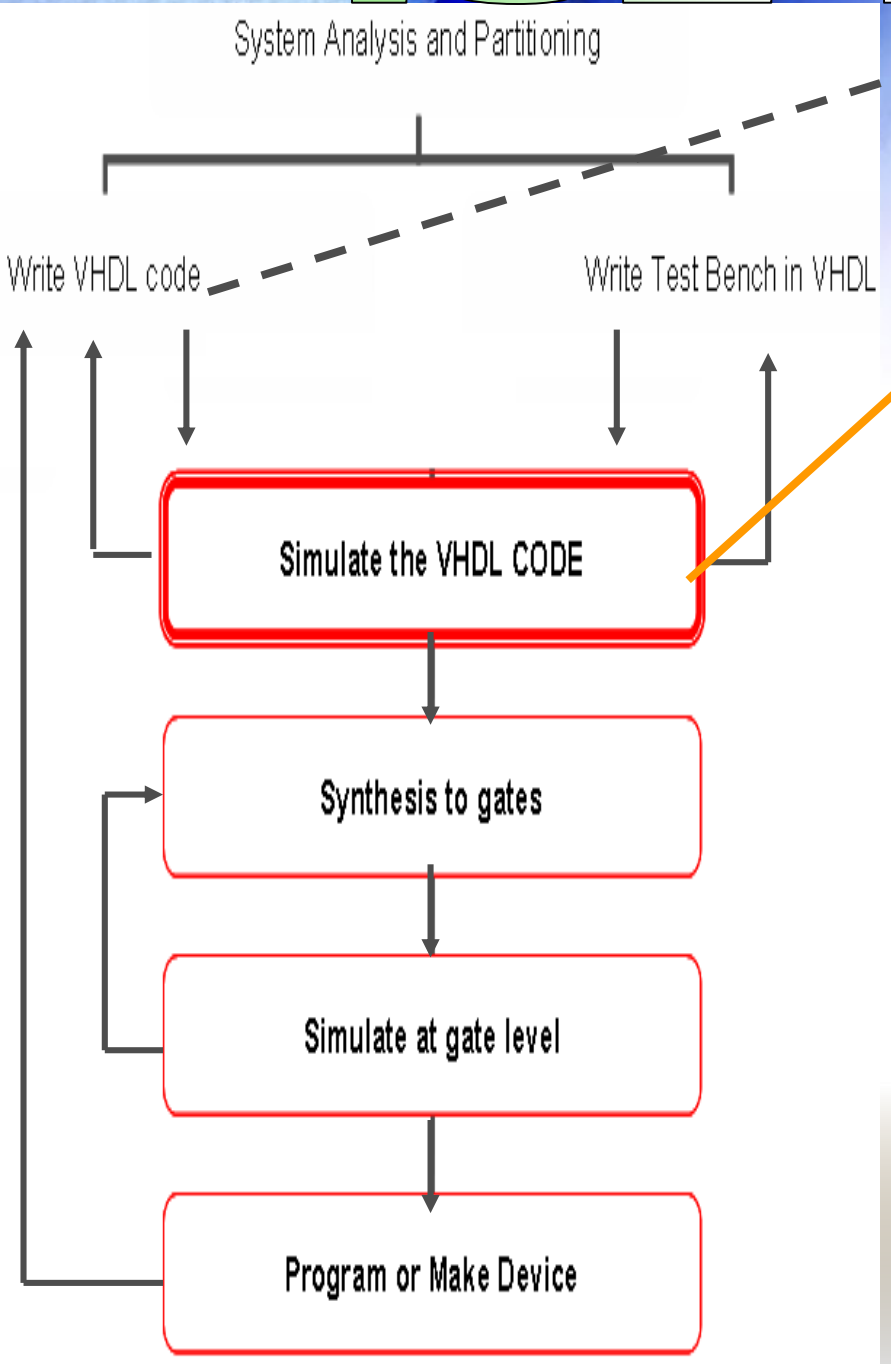

System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device

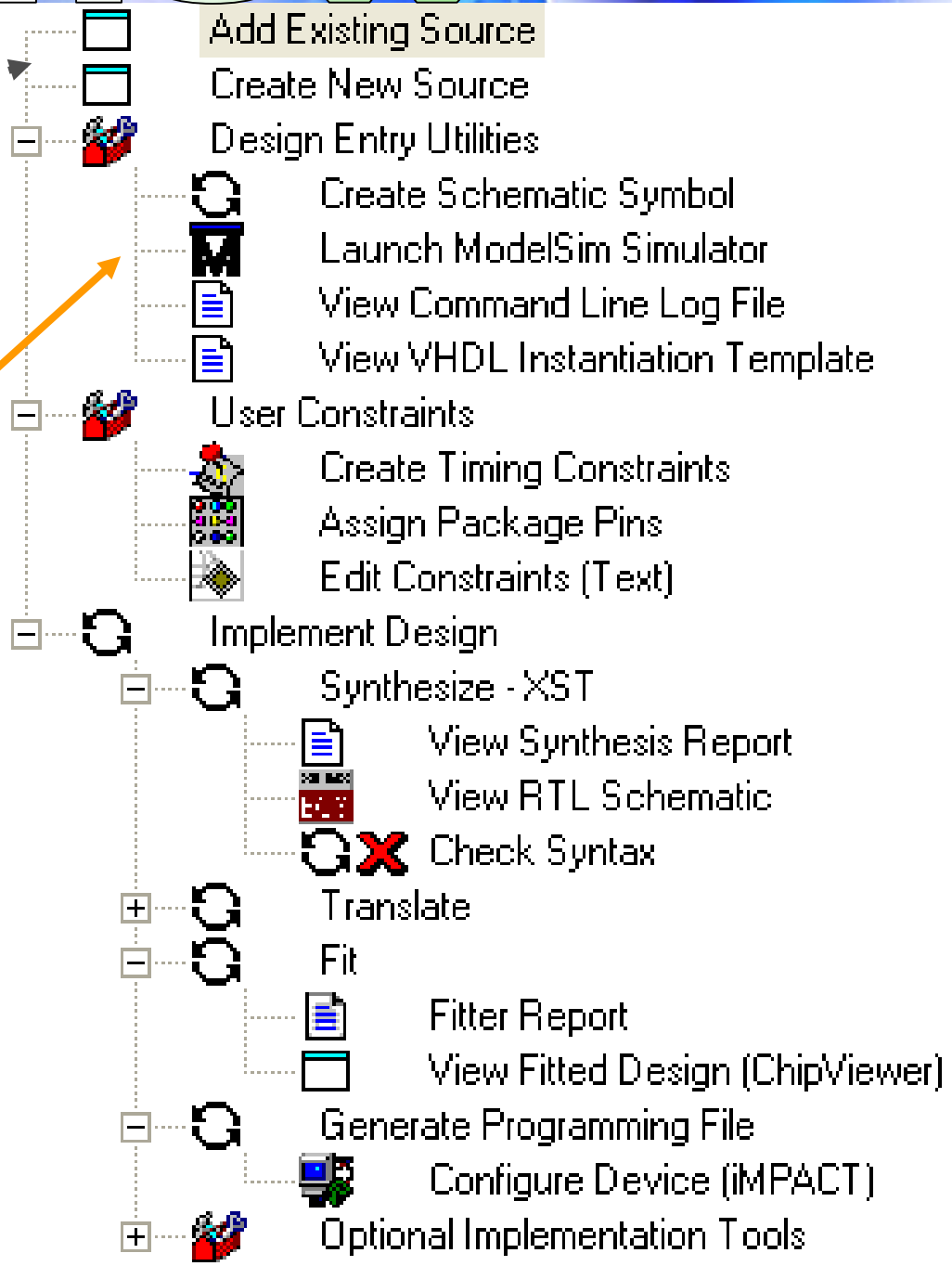
ISE flow


System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device

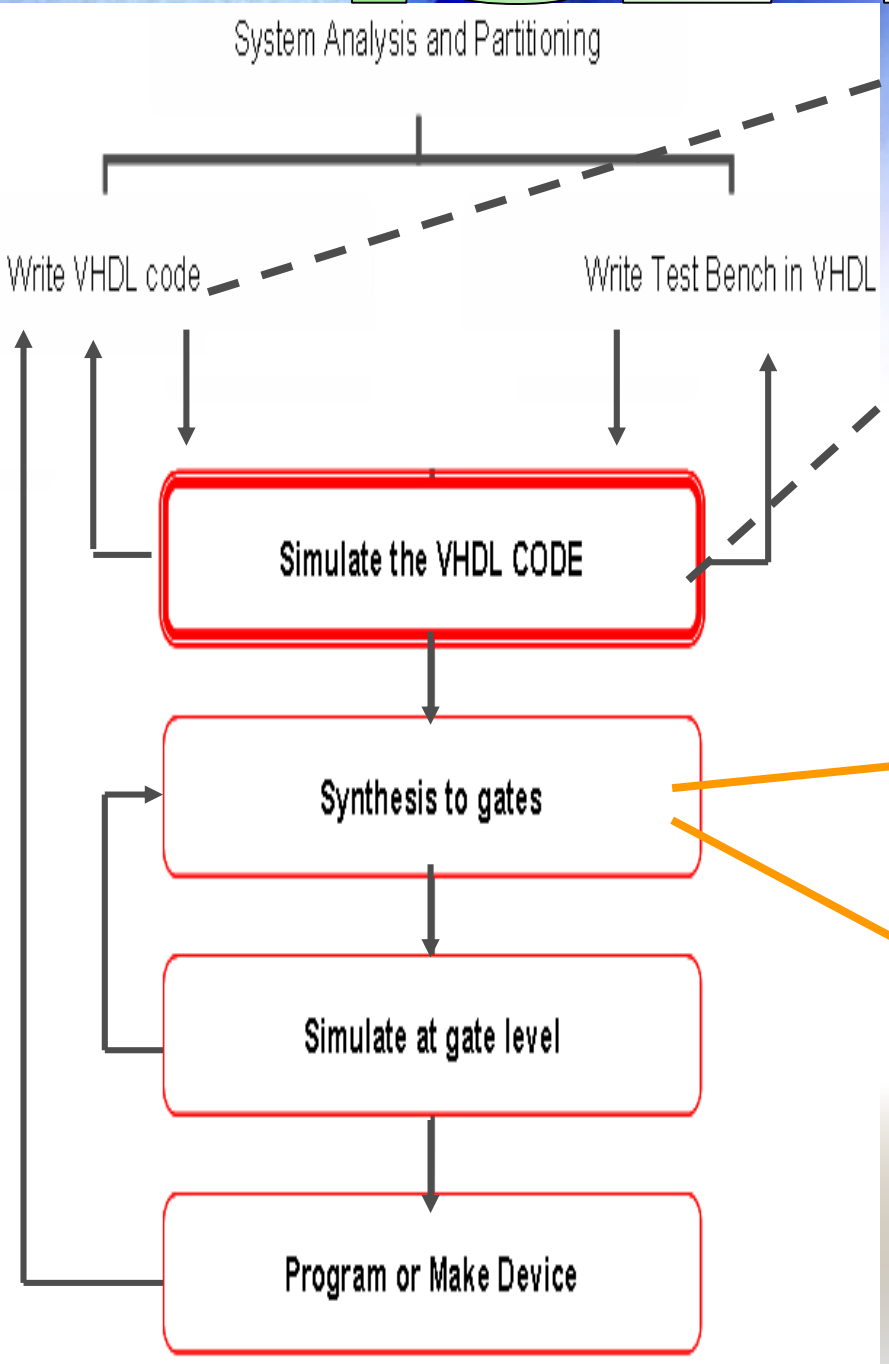

Add Existing Source
Create New Source
Design Entry Utilities
Create Schematic Symbol
Launch ModelSim Simulator
View Command Line Log File
View VHDL Instantiation Template
User Constraints
Create Timing Constraints
Assign Package Pins
Edit Constraints (Text)
Implement Design
Synthesize - XST
View Synthesis Report
View RTL Schematic
Check Syntax
Translate
Fit
Fitter Report
View Fitted Design (ChipViewer)
Generate Programming File
Configure Device (iMPACT)
Optional Implementation Tools

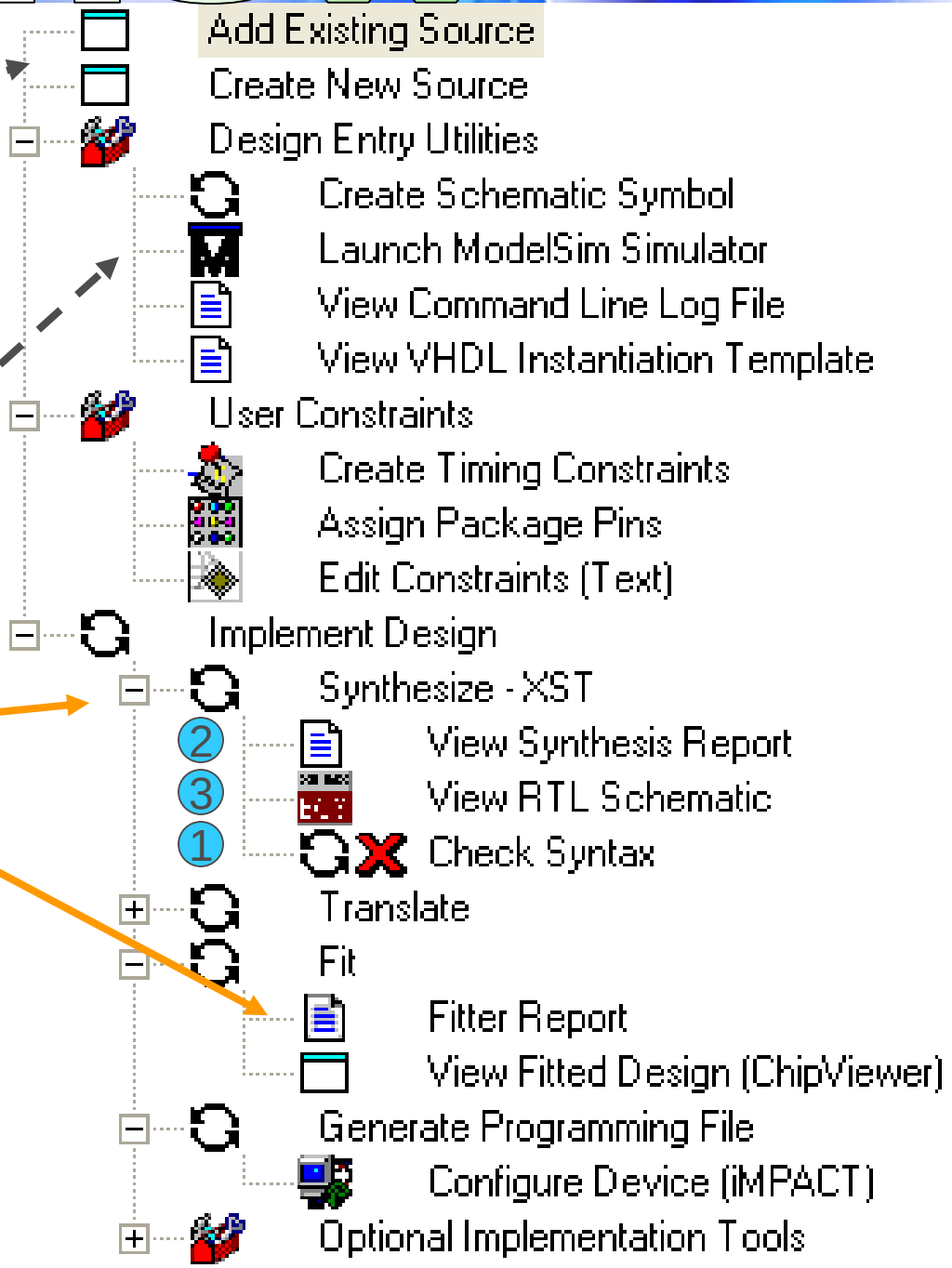
ISE flow


System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device

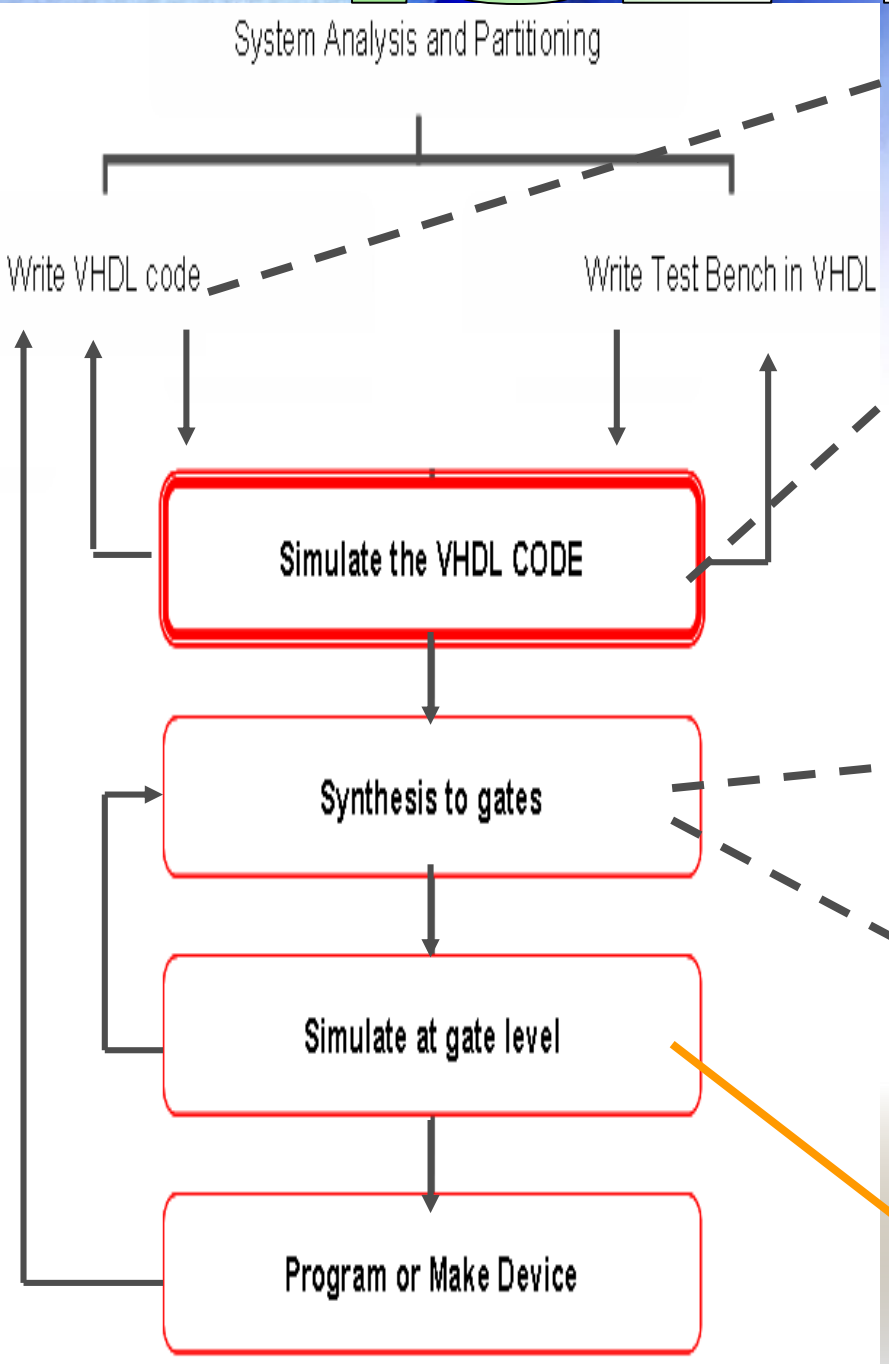

Add Existing Source
Create New Source
Design Entry Utilities
Create Schematic Symbol
Launch ModelSim Simulator
View Command Line Log File
View VHDL Instantiation Template
User Constraints
Create Timing Constraints
Assign Package Pins
Edit Constraints (Text)
Implement Design
Synthesize - XST
View Synthesis Report
View RTL Schematic
Check Syntax
Translate
Fit
Fitter Report
View Fitted Design (ChipViewer)
Generate Programming File
Configure Device (iMPACT)
Optional Implementation Tools

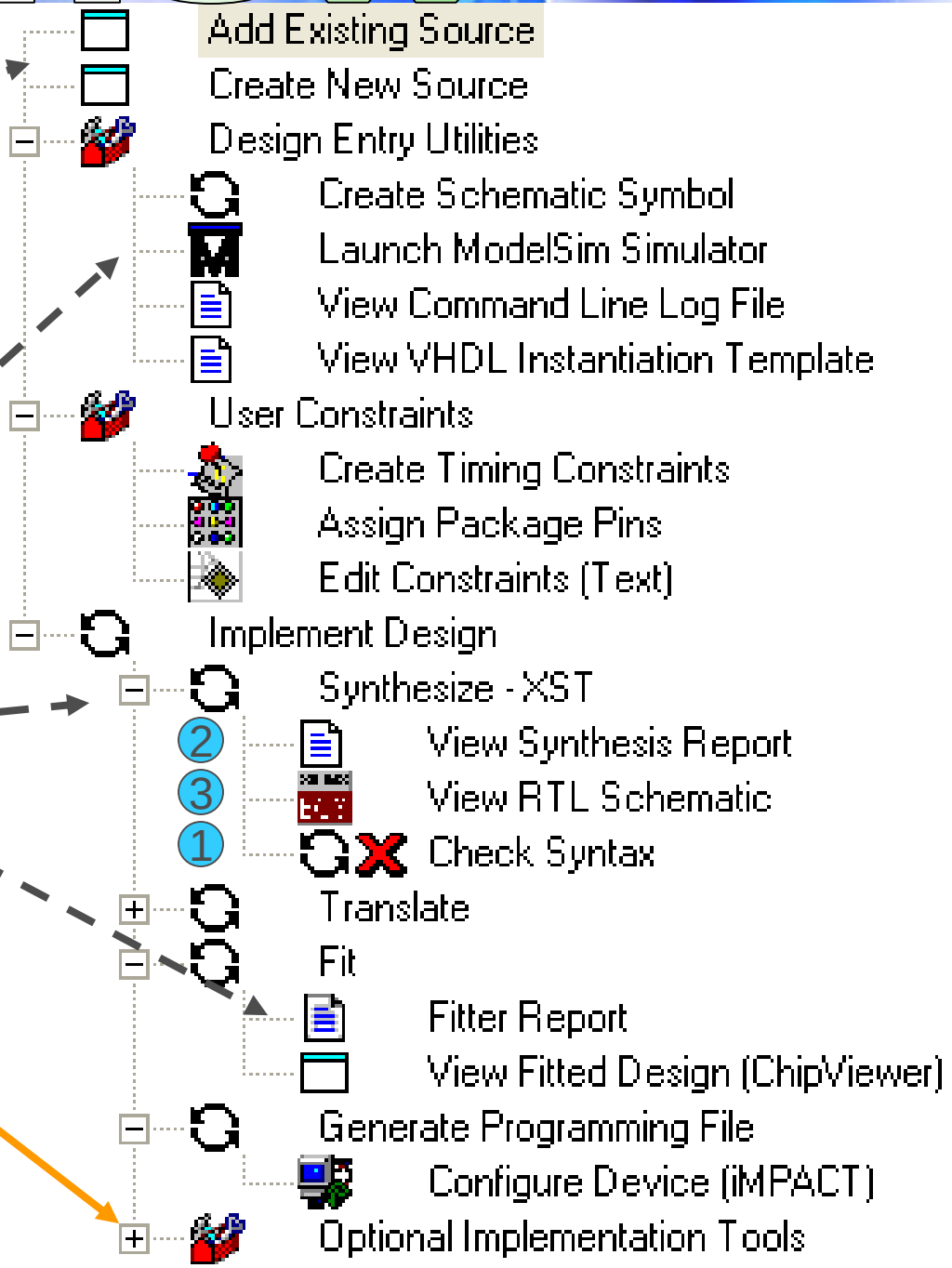
ISE flow


System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device

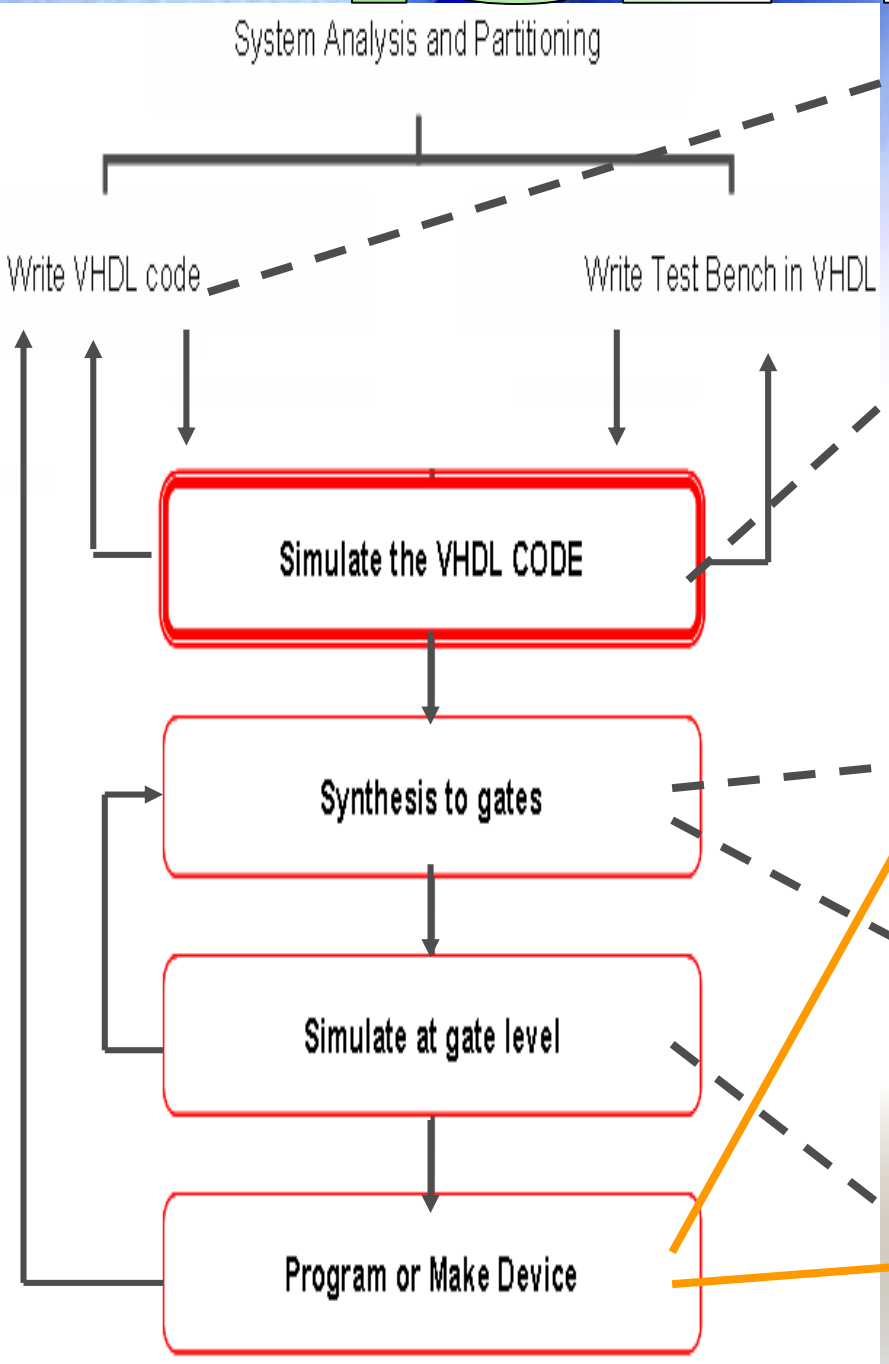

Add Existing Source
Create New Source
Design Entry Utilities
Create Schematic Symbol
Launch ModelSim Simulator
View Command Line Log File
View VHDL Instantiation Template
User Constraints
Create Timing Constraints
Assign Package Pins
Edit Constraints (Text)
Implement Design
Synthesize - XST
2
View Synthesis Report
3
View RTL Schematic
1
Check Syntax
Translate
Fit
Fitter Report
View Fitted Design (ChipViewer)
Generate Programming File
Configure Device (iMPACT)
Optional Implementation Tools

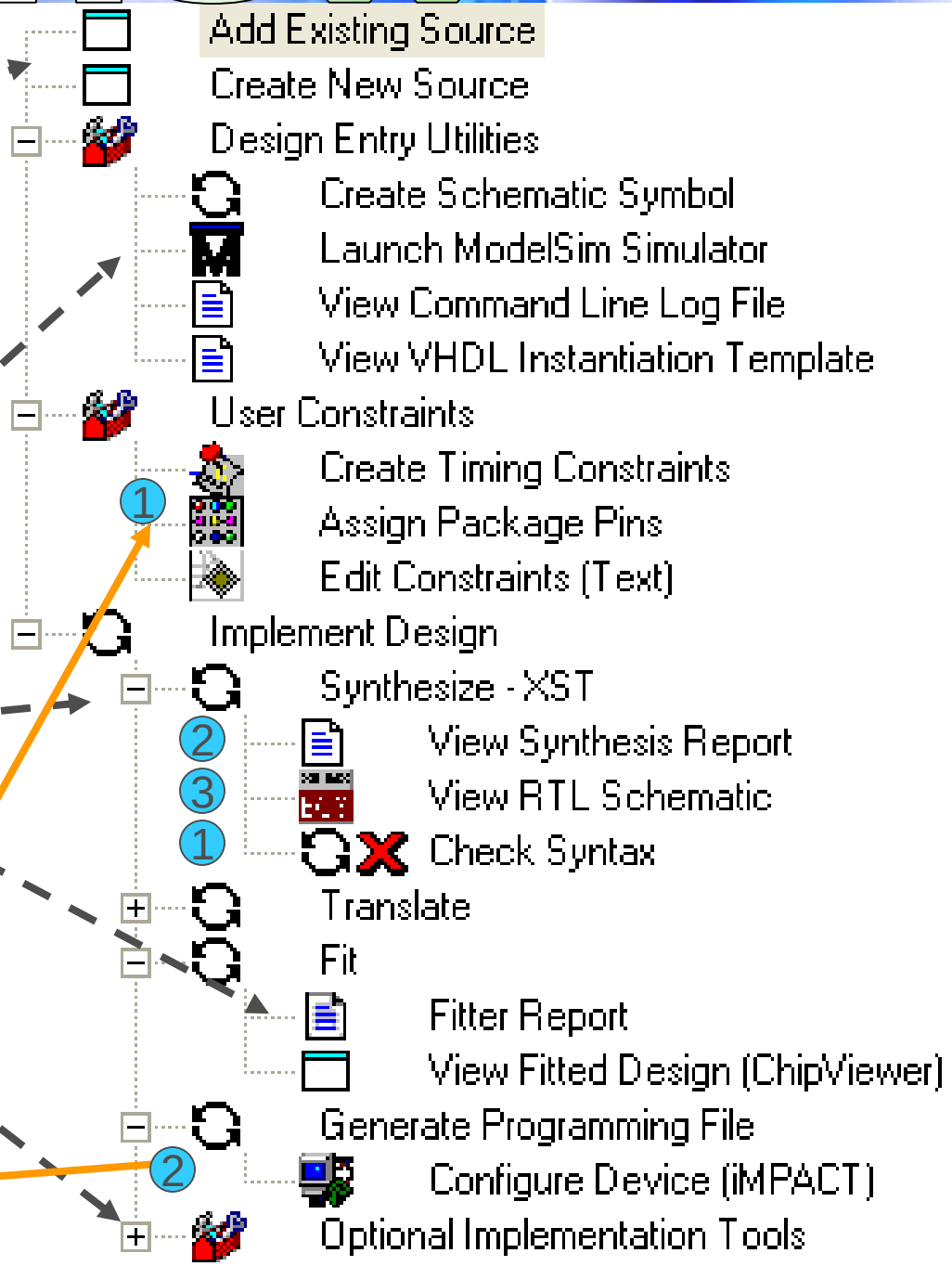
ISE flow


System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device

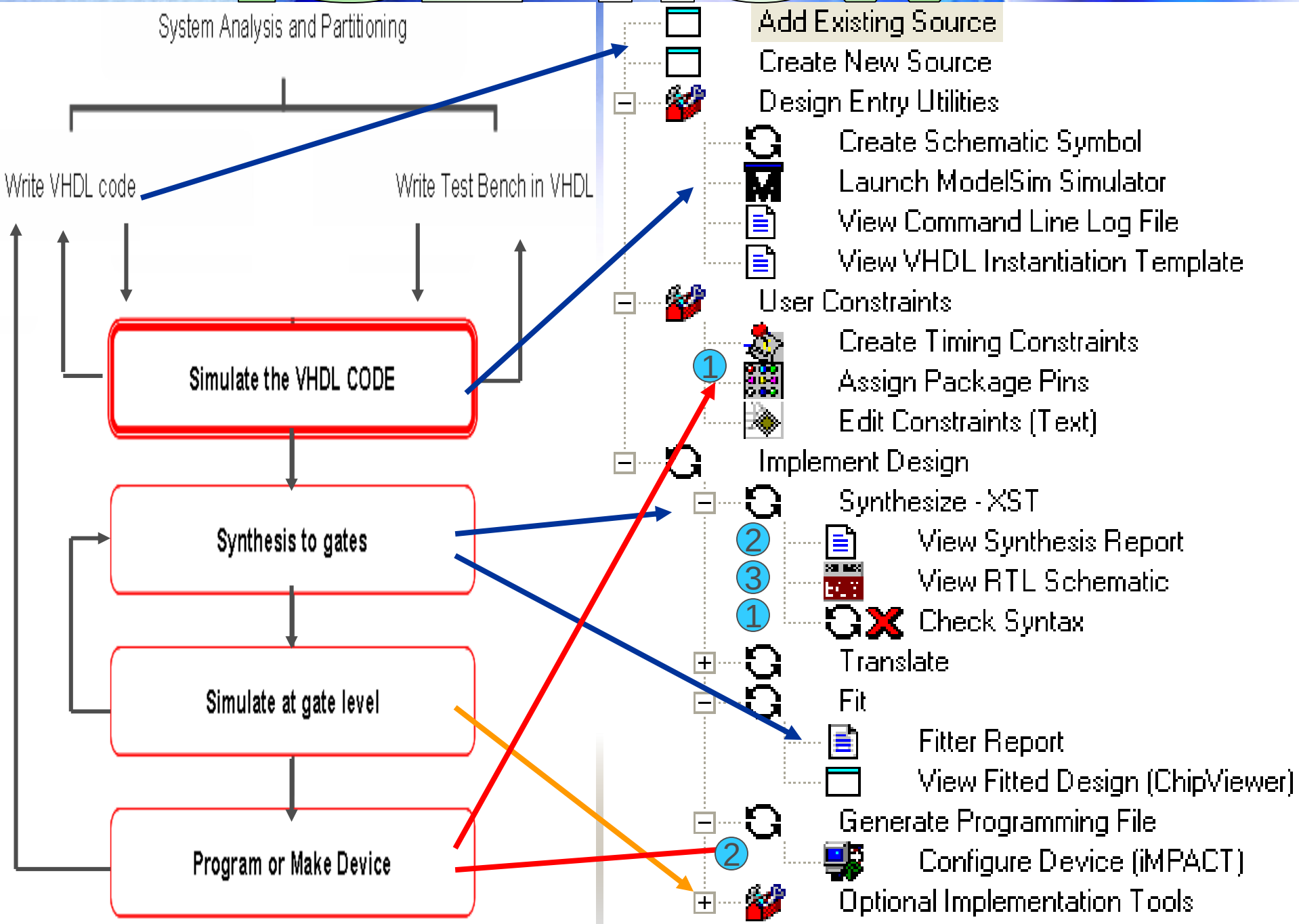

Add Existing Source
Create New Source
Design Entry Utilities
Create Schematic Symbol
Launch ModelSim Simulator
View Command Line Log File
View VHDL Instantiation Template
User Constraints
Create Timing Constraints
Assign Package Pins
Edit Constraints (Text)
Implement Design
Synthesize - XST
2
View Synthesis Report
3
View RTL Schematic
1
Check Syntax
Translate
Fit
Fitter Report
View Fitted Design (ChipViewer)
Generate Programming File
Configure Device (iMPACT)
Optional Implementation Tools

ISE flow


System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device


Add Existing Source
Create New Source
Design Entry Utilities
Create Schematic Symbol
Launch ModelSim Simulator
View Command Line Log File
View VHDL Instantiation Template
User Constraints
Create Timing Constraints
Assign Package Pins
Edit Constraints (Text)
Implement Design
Synthesize - XST
View Synthesis Report
View RTL Schematic
Check Syntax
Translate
Fit
Fitter Report
View Fitted Design (ChipViewer)
Generate Programming File
Configure Device (iMPACT)
Optional Implementation Tools
1
2
3
1
2

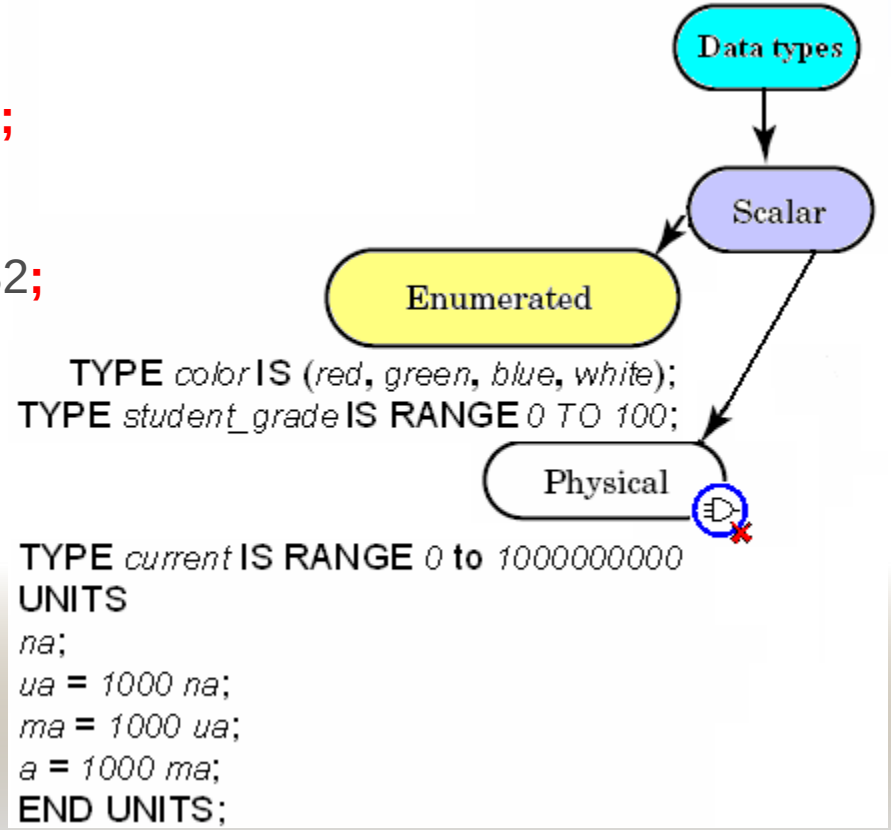
ISE flow


System Analysis and Partitioning
Write VHDL code
Write Test Bench in VHDL
Simulate the VHDL CODE
Synthesis to gates
Simulate at gate level
Program or Make Device
Add Existing Source
Create New Source
Design Entry Utilities
Create Schematic Symbol
Launch ModelSim Simulator
View Command Line Log File
View VHDL Instantiation Template
User Constraints
Create Timing Constraints
Assign Package Pins
Edit Constraints (Text)
Implement Design
Synthesize - XST
View Synthesis Report
View RTL Schematic
Check Syntax
Translate
Fit
Fitter Report
View Fitted Design (ChipViewer)
Generate Programming File
Configure Device (iMPACT)
Optional Implementation Tools
1
2
3
1
2

Enumerated Data types

TYPE color **IS (**red, green, blue, white**);**
TYPE color **IS (**“00”, “01”, “10”, “11”**);**

TYPE my_integer **IS RANGE** -32 **TO** 32**;**



Subtypes

- SUBTYPE name IS datatype RANGE range;

Examples

- SUBTYPE natural IS INTEGER RANGE 0 TO INTEGER'HIGH;
- SUBTYPE positive IS INTEGER RANGE ..?.. TO INTEGER'HIGH;
- SUBTYPE my_logic IS bit RANGE '0' TO 'Z';
- ……. bit =('X','0','1','Z','W','L','H','-').

Subtypes vs Enumerated

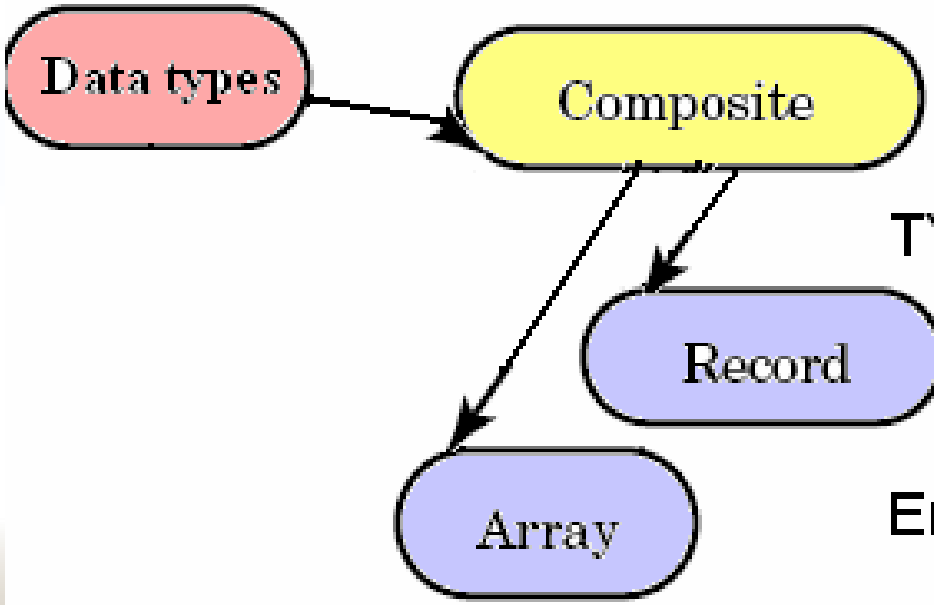
Comparison

- Save and reduce the width of registers and buses.

Contrast

- Operations between different data types are  .
  Subtypes have a privilege over than enumerated style.

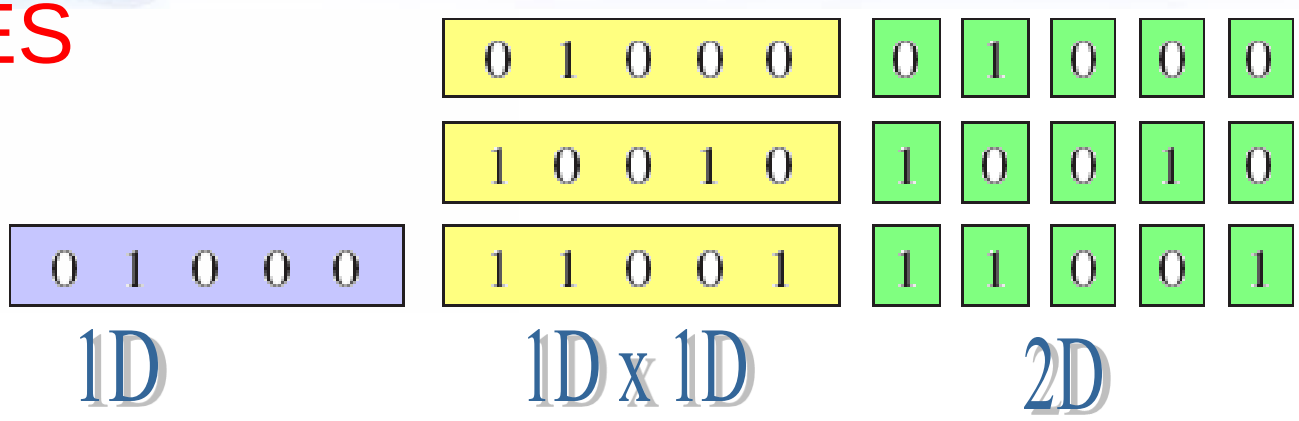
Composite Data types




```
TYPE bed IS RECORD;
    On_shelf : Boolean;
    Item_ID : Std_vector(7 downto 0);
    Qty : Integer;
End RECORD;
```

```
TYPE row IS ARRAY (7 DOWNTO 0) OF STD_LOGIC;
TYPE array1 IS ARRAY (0 TO 3) OF row;
```

ARRAIES




```
TYPE OneD        IS ARRAY (7 DOWNTO 0) OF STD_LOGIC;
TYPE OneDxOneD   IS ARRAY (3 DOWNTO 0)  OF OneD;
TYPE twoD        IS ARRAY (0 to 3,0 to 7)    OF STD_LOGIC;
```

```
OneD(3) <= '0';

OneDxOneD (3)(5) <= '1';        OneDxOneD (3) <="10110010";

twoD(3,5) <= '0';
```

Array examples

Signal y: std_logic_vector(7 downto 0);

- y <= "11111110";
- y <= ('1','1','1','1','1','1','0','Z');
- y <= "11111" & "000";
- y <= (OTHERS => '1');
- y <= (7 =>'0', 1 =>'0', OTHERS => '1');
- y(2 DOWNTO 0) <= z(6 DOWNTO 4);

Array examples

Type OneD is array (3 downto 0) of std_logic_vector (7)

Type twoD is array (0 to 3, 0 to 7) of std_logic;

- OneD(0)(7 DOWNTO 0) <= "11110000";
- OneD <= ("11111100", ('0','0','0','0','Z','Z','Z','Z'),
  (OTHERS=>'0'), (OTHERS=>'0'));
- OneD(0)(7) <= ‘1’;

- twoD(0,0) <= ‘1’;
- twoD <= ("11110000", x“3" ,"11110000", (OTHERS=>'0'));

Records

Records are similar to arrays, with objects of different types.

```
TYPE record_name IS
RECORD
        member_name1 : datatype;
        member_name2 : datatype;
END RECORD;
```

Examples

```
TYPE instruction IS
RECORD
        opcode : std_logic;
            src : INTEGER;
            dst : INTEGER;
END RECORD;
```

inst.src <= 5;

inst := (‘1’, 5, 2);

Solve page 10 and 11

Basic VHDL code

$$Y = (\bar{A} \oplus B) + (\overline{CD})$$

- Write the source code
- Compile and synthesis
- Simulate
- Download
- Test
- Write the results.

E

N

D