## VHDL Introduction and Overview

#### Jing Chen

#### Department of Computing and Software, McMaster University

#### November 18, 2010

▲□▶ ▲□▶ ▲ □▶ ▲ □▶ ▲ □ ● ● ● ●

### What is VHDL?

- Very-High-Speed Integrated Circuit Hardware Description Language
- VHDL appeared in 1983
- VHDL was sponsored by US Department of Defense
- VHDL borrowed heavily from Ada programming language
- Formal definition of VHDL was given by IEEE Standard VHDL Language Reference Manual

▲□▶▲□▶▲□▶▲□▶ □ のQで

## What is the original intent of inventing VHDL?

 Serve as a means of communicating designs from one contractor to another in the very high speed integrated circuit program

■ CDL,ISP,and AHPL have been used since 1970s

## VHDL Design Tools & Products

- Textual Editor
- Schematic Editor
- Simulators
- Checkers and Analyzers
- Optimizers and Synthesizers

▲□▶ ▲□▶ ▲ □▶ ★ □ ▶ □ ● の < @

FPGA

#### **Textual Editor**

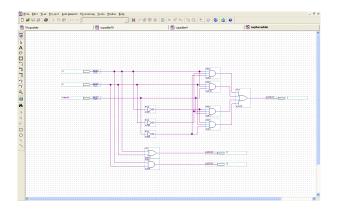
#### An editor can be used to edit VHDL source code.

9	1	library ieee;	1
- 1	2	use ieee.std_logic_1164.all;	
- 46	3	use ieee.std_logic_unsigned.all;	
	4	mentity ALU16 is	
	5	<pre>port(a : in std_logic_vector(15 downto 0);input</pre>	
÷e	6	<pre>b : in std_logic_vector(15 downto 0);input</pre>	
- L	7	<pre>sin: in std_logic_vector(15 downto 0);carry,interrupt:is controled by state register</pre>	
3	в	<pre>m : in std_logic_vector(4 downto 0);mode of calculation</pre>	
X	9	f1: out std_logic_vector(15 downto 0);low output	
2	10	<pre>f2: out std_logic_vector(15 downto 0):high output</pre>	
2	11	sout: out std_logic_vector(15 downto 0) content of16 state register	
	12	);	
	13	end entity ALU16;	
ab/	14	Earchitecture behave of ALU16 is	
	15	signal s16: std_logic_vector(15 downto 0);	
	16	Ebegin	
2	17	process(a,b,sin,m)	
-	18	variable f16: std_logic_vector(15 downto 0);	
_	19	<pre>variable d16: std_logic_vector(15 downto 0);</pre>	
	20	variable int16: integer range 15 downto 0;	
	21	variable a17: std_logic_vector(16 downto 0);	
	22	<pre>variable b17: std_logic_vector(16 downto 0);</pre>	
	23	<pre>variable c17: std_logic_vector(16 downto 0);</pre>	
	24	variable f17: std_logic_vector(16 downto 0);	
	25	variable f32: std_logic_vector(31 downto 0);	
	26	variable q32: std_logic_vector(31 downto 0);	
	27	begin	
3			>

▲□▶ ▲□▶ ▲ □▶ ▲ □▶ ▲ □ ● ● ● ●

### Schematic Editor

An editor which can be used to create and display an interconnected set of graphic tokens.



イロト 不得 とくほ とくほとう

3

#### Simulators(Soft Logic Analyzer)

• A hardware debugger which models the response of a system to input stimuli.

Maxer I in	me Bar:	0 ps Pointer.		14.96 us Interval		14.96 (	a Start	Start. O ps		End: 15.0 us		
	Name	11.45 us	12.0	) us	12.73 us	13	1.37 rus	14. Q1 u	s	14.65 us		
<b>10</b> 0	CLIK	-									m	
1	BESET											
<b>@</b> 2	💌 IBl			0)110()1000)(10001								
19				00)00000001(1010								
<b>€</b> ¥36	■ 181_007	0000000		1111111110110	000000000	001010 (111	111111010 0000	000000000				
<b>₩</b> 53	∎ 182_007	00000000	10000000 )[1	11111111101011	000000000		0000000100	000000000000	000 0111	111111011000	(00	
70	■ 183_007					000000000000						
387		00000000	00001010	000000000000000000000000000000000000000	11 0000		11111110110)000	00000000 (00000000	00001000 0000	000000001001	100	
						00000000000						
121	■ SOUT_OUT	00000000	00010000 X	000000000000000000000000000000000000000	00 ( 0000	000000010000		000)(000	00000000000	00000		
	⊞ 30_007	_				000000000000						
2155	№ 31_007		-10	X			-21				-40	
112	■ 82_007	4	X	-21		X	17		<u> </u>	-40		
2189	■ 33_00T					000000000000				_		
205						00000000000	0000				-	
2223	■ 35_007		10	X		11	X		12		Х	
240		11		10	X	9		X	8			
257	I RT_OUT										-	
214	38_00T     39 00T     39 00T	000000000000										
C 291	★ 39_00T					000000000000000000000000000000000000000		_		_	-	
											-	
🚱 308	H 811 00T					000000000000000000000000000000000000000						



Rule checkers

At the silicon level, design rule checkers are used to insure that the layout implies a circuit that can be fabricated reliably.

At other levels, rule checkers are used to determine if connection rules or fanout rules have been violated.

▲□▶▲□▶▲□▶▲□▶ □ のQで



#### Analyzers

Timing analyzers can be used to check for errors that violate the structure and meaning of the language and also can be used to check for the longest path through a logic circuit or system.

▲□▶ ▲□▶ ▲ □▶ ▲ □▶ ▲ □ ● ● ● ●



#### Optimizers

Optimizers change the form of the design representation to a new form, which is regraded as "improved" in some fashion.

For example, at register level, optimizers are used to determine the best combination of control sequence and data paths.

< □ > < 同 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <

# Synthesizers(Silicon Compiler)

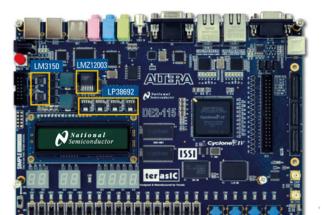
#### Synthesizers

A high-level synthesizer translates a representation of a circuit at a high level of abstraction into a lower-level representation that can be implemented within a particular technology.

▲□▶▲□▶▲□▶▲□▶ □ のQで

#### What is FPGA?

- Field Programmable Gate Array
- FPGAs can be used to implement any logical function specified by a hardware description language



# Language Constructs of VHDL

- Language Constructs Similar to C Programming Language
- Language Constructs Specified for Modeling Hardware Circuits

< □ > < 同 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <

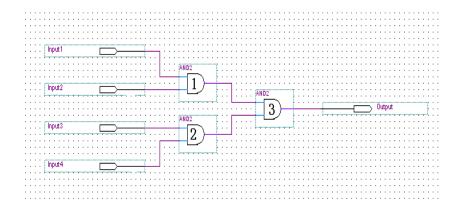
## Features of Digital Circuit

Concurrency

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ─ □ ─ のへぐ

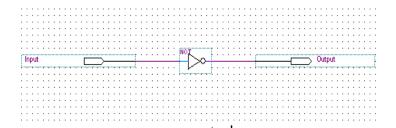
- Delay
- Timing

## Concurrency



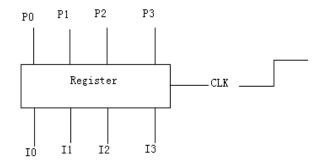
◆□▶ ◆□▶ ◆臣▶ ◆臣▶ ─臣 ─のへで

# Delay

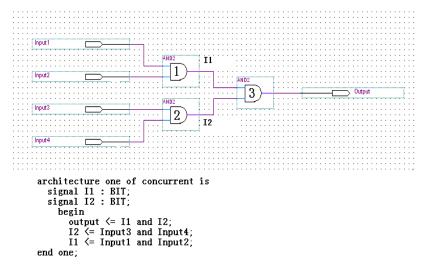




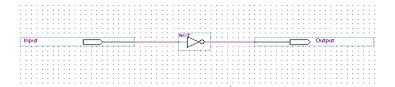
# Timing

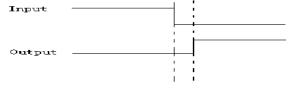


### Concurrent Mechanism in VHDL



### Delay Mechanism in VHDL

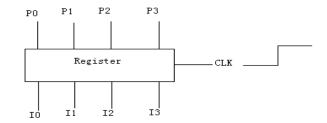




Output <= not Input after 5 ns;

Output <= not Input; after delta time by default

#### Timing Mechanism in VHDL



if (clk'event and clk='1') then
 P[3..0] <= I[3..0];
end if;</pre>

# Signal

- Signals are used to represent data values on actual physical data lines in circuits.
- Logic Value of Signal

```
Type MVL4 is ('X', - unknown
'1', - logic 1
'0', - logic 0
'Z'); -high impedance
Type MVL7 is ('X', - strong X (strong unknown)
'1', - strong 0 (strong low)
'0', - strong1 (strong high)
'Z', - tri-gate X (high impedance)
'W', - weak X (weak unknown)
'L', - weak 0 (weak low)
'H'); - weak 1 (weak high)
```

# 16-bit Multiplier Designed by VHDL

#### VHDL Source Code

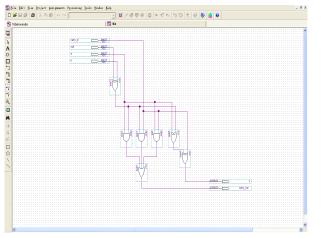
entity Multiplier\_VHDL is
port ( Input1, Input2: in std\_logic\_vector(15 downto 0);
Result: out std\_logic\_vector(31 downto 0) );
end entity Multiplier\_VHDL;
architecture Behavioral of Multiplier\_VHDL is
begin
Result <= std\_logic\_vector(unsigned(Input1) \* unsigned(Input2));</pre>

▲□▶▲□▶▲□▶▲□▶ □ のQで

end architecture Behavioral;

# 16-bit Multiplier Designed by Schematic

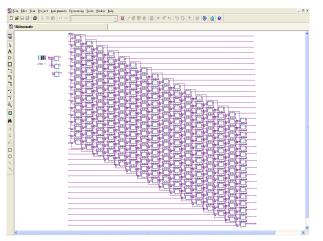
Schematic Representation(Part I)



◆□▶ ◆□▶ ◆三▶ ◆三▶ ・三 のへの

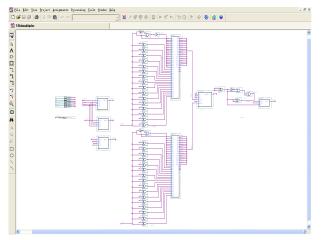
## 16-bit Multiplier Designed by Schematic

Schematic Representation(Part II)



# 16-bit Multiplier Designed by Schematic

Schematic Representation(Part III)



▲ロト▲聞と▲臣と▲臣と 臣 のへの

## Advantages of VHDL on Hardware Design

- Public Design Platform
- Intellectual Property Core
- Logical and Timing errors are basically eliminated before building real hardware

< □ > < 同 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <

Line between hardware and software become unclear

◆□▶ ◆□▶ ◆ □▶ ◆ □▶ ● □ ● ● ● ●



- Long synthesize time
- Synthesizer may make a mistake
- Tool for computing longest path through a circuit is not perfect

▲□▶ ▲□▶ ▲ □▶ ▲ □▶ ▲ □ ● ● ● ●



- A VHDL PRIMER (Third Edition) Author:Jayaram.Bhasker
- VHDL Design Representation and Synthesis (Second Edition) Author:James R.Armstrong, F.Gail Gray

# Thanks!

◆□▶ ◆□▶ ◆目▶ ◆目▶ ● ● ● ●