CS2022 Datapath Design Part A

Jack Cassidy | Student No. 1432 0816 | jacassid@tcd.ie

# VHDL Code – Components

## Decoder 3to 8 16bit:

entity decoder\_3to8 is

Port ( A0 : in STD\_LOGIC;

A1 : in STD\_LOGIC;

A2 : in STD\_LOGIC;

Q0 : out STD\_LOGIC;

Q1 : out STD\_LOGIC;

Q2 : out STD\_LOGIC;

Q3 : out STD\_LOGIC;

Q4 : out STD\_LOGIC;

Q5 : out STD\_LOGIC;

Q6 : out STD\_LOGIC;

Q7 : out STD\_LOGIC);

end decoder\_3to8;

architecture Behavioral of decoder\_3to8 is

begin

Q0<=((not A0) and (not A1) and (not A2)) after 5ns;

Q1<=(A0 and (not A1) and (not A2)) after 5ns;

Q2<=((not A0) and A1 and (not A2)) after 5ns;

Q3<=(A0 and A1 and (not A2)) after 5ns;

Q4<=((not A0) and (not A1) and A2) after 5ns;

Q5<=(A0 and (not A1) and A2) after 5ns;

Q6<=((not A0) and A1 and A2) after 5ns;

Q7<=(A0 and A1 and A2) after 5ns;

end Behavioral;

## MUX2 16bit:

entity mux2\_16bit is

Port ( s : in STD\_LOGIC;

ln0 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln1 : in STD\_LOGIC\_VECTOR (15 downto 0);

Z : out STD\_LOGIC\_VECTOR (15 downto 0));

end mux2\_16bit;

architecture Behavioral of mux2\_16bit is

begin

Z <= ln0 after 5ns when s='0' else

ln1 after 5ns when s='1' else

"0000000000000000" after 5ns;

end Behavioral;

## MUX8 16bit:

entity mux8\_16bit is

Port ( ln0 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln1 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln2 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln3 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln4 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln5 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln6 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln7 : in STD\_LOGIC\_VECTOR (15 downto 0);

S0 : in STD\_LOGIC;

S1 : in STD\_LOGIC;

S2 : in STD\_LOGIC;

Z : out STD\_LOGIC\_VECTOR (15 downto 0));

end mux8\_16bit;

architecture Behavioral of mux8\_16bit is

begin

Z <= ln0 after 5ns when S0='0' and S1='0' and S2='0' else

ln1 after 5ns when S0='1' and S1='0' and S2='0' else

ln2 after 5ns when S0='0' and S1='1' and S2='0' else

ln3 after 5ns when S0='1' and S1='1' and S2='0' else

ln4 after 5ns when S0='0' and S1='0' and S2='1' else

ln5 after 5ns when S0='1' and S1='0' and S2='1' else

ln6 after 5ns when S0='0' and S1='1' and S2='1' else

ln7 after 5ns when S0='1' and S1='1' and S2='1' else

"0000000000000000" after 5ns;

end Behavioral;

## Register 16bit:

entity reg16 is

Port ( D : in STD\_LOGIC\_VECTOR (15 downto 0);

load : in STD\_LOGIC;

Clk : in STD\_LOGIC;

Q : out STD\_LOGIC\_VECTOR(15 downto 0)

);

end reg16;

architecture Behavioral of reg16 is

begin process(Clk)

begin

if(rising\_edge(Clk)) then

if load ='1' then

Q<=D after 5 ns;

end if;

end if;

end process;

end Behavioral;

## Register File with 8 16bit Registers:

entity regfile\_16bit is

Port ( src\_s0 : in STD\_LOGIC;

src\_s1 : in STD\_LOGIC;

src\_s2 : in STD\_LOGIC;

des\_A0 : in STD\_LOGIC;

des\_A1 : in STD\_LOGIC;

des\_A2 : in STD\_LOGIC;

Clk : in STD\_LOGIC;

data\_src : in STD\_LOGIC;

data : in STD\_LOGIC\_VECTOR(15 downto 0);

reg0 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg1 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg2 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg3 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg4 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg5 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg6 : out STD\_LOGIC\_VECTOR(15 downto 0);

reg7 : out STD\_LOGIC\_VECTOR(15 downto 0));

end regfile\_16bit;

architecture Behavioral of regfile\_16bit is

-- Components

-- 16-bit Register

COMPONENT reg16

PORT( D : in STD\_LOGIC\_VECTOR (15 downto 0);

load : in STD\_LOGIC;

Clk : in STD\_LOGIC;

Q : out STD\_LOGIC\_VECTOR(15 downto 0)

);

END COMPONENT;

-- 3 to 8 Decoder

COMPONENT decoder\_3to8

PORT( A0 : in STD\_LOGIC;

A1 : in STD\_LOGIC;

A2 : in STD\_LOGIC;

Q0 : out STD\_LOGIC;

Q1 : out STD\_LOGIC;

Q2 : out STD\_LOGIC;

Q3 : out STD\_LOGIC;

Q4 : out STD\_LOGIC;

Q5 : out STD\_LOGIC;

Q6 : out STD\_LOGIC;

Q7 : out STD\_LOGIC

);

END COMPONENT;

-- MUX2 16 bit

COMPONENT mux2\_16bit

PORT( s : in STD\_LOGIC;

ln0 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln1 : in STD\_LOGIC\_VECTOR (15 downto 0);

Z : out STD\_LOGIC\_VECTOR (15 downto 0)

);

END COMPONENT;

-- MUX8 16 bit

COMPONENT mux8\_16bit

PORT( ln0 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln1 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln2 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln3 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln4 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln5 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln6 : in STD\_LOGIC\_VECTOR (15 downto 0);

ln7 : in STD\_LOGIC\_VECTOR (15 downto 0);

S0 : in STD\_LOGIC;

S1 : in STD\_LOGIC;

S2 : in STD\_LOGIC;

Z : out STD\_LOGIC\_VECTOR (15 downto 0)

);

END COMPONENT;

signal load\_reg0, load\_reg1, load\_reg2, load\_reg3, load\_reg4, load\_reg5, load\_reg6, load\_reg7: STD\_LOGIC;

signal reg0\_q, reg1\_q, reg2\_q, reg3\_q, reg4\_q, reg5\_q, reg6\_q, reg7\_q: STD\_LOGIC\_VECTOR(15 downto 0);

signal data\_src\_mux\_out, src\_reg: STD\_LOGIC\_VECTOR(15 downto 0);

begin

-- Port Maps

-- Register 0

reg00: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg0,

Clk => Clk,

Q => reg0\_q

);

reg01: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg1,

Clk => Clk,

Q => reg1\_q

);

reg02: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg2,

Clk => Clk,

Q => reg2\_q

);

reg03: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg3,

Clk => Clk,

Q => reg3\_q

);

reg04: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg4,

Clk => Clk,

Q => reg4\_q

);

reg05: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg5,

Clk => Clk,

Q => reg5\_q

);

reg06: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg6,

Clk => Clk,

Q => reg6\_q

);

reg07: reg16 PORT MAP(

D => data\_src\_mux\_out,

load => load\_reg7,

Clk => Clk,

Q => reg7\_q

);

-- Destination Register Decoder

des\_decoder\_3to8: decoder\_3to8 PORT MAP(

A0 => des\_A0,

A1 => des\_A1,

A2 => des\_A2,

Q0 => load\_reg0,

Q1 => load\_reg1,

Q2 => load\_reg2,

Q3 => load\_reg3,

Q4 => load\_reg4,

Q5 => load\_reg5,

Q6 => load\_reg6,

Q7 => load\_reg7

);

-- 2 to 1 Data Source MUX

data\_src\_mux2\_16bit: mux2\_16bit PORT MAP(

ln0 => data,

ln1 => src\_reg,

s => data\_src,

Z => data\_src\_mux\_out

);

-- 8 to 1 Source Register MUX

lnst\_mux8\_16bit: mux8\_16bit PORT MAP(

ln0 => reg0\_q,

ln1 => reg1\_q,

ln2 => reg2\_q,

ln3 => reg3\_q,

ln4 => reg4\_q,

ln5 => reg5\_q,

ln6 => reg6\_q,

ln7 => reg7\_q,

S0 => src\_s0,

S1 => src\_s1,

S2 => src\_s2,

Z => src\_reg

);

reg0 <= reg0\_q;

reg1 <= reg1\_q;

reg2 <= reg2\_q;

reg3 <= reg3\_q;

reg4 <= reg4\_q;

reg5 <= reg5\_q;

reg6 <= reg6\_q;

reg7 <= reg7\_q;

end Behavioral;

# VHDL Code – Test Benches

## Decoder 3to 8 16bit Test Bench:

Cycles through all the possible input permutations of the 3to8 decoder.

ENTITY decoder\_3to8\_tb IS

END decoder\_3to8\_tb;

ARCHITECTURE behavior OF decoder\_3to8\_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT decoder\_3to8

PORT(

A0 : IN std\_logic;

A1 : IN std\_logic;

A2 : IN std\_logic;

Q0 : OUT std\_logic;

Q1 : OUT std\_logic;

Q2 : OUT std\_logic;

Q3 : OUT std\_logic;

Q4 : OUT std\_logic;

Q5 : OUT std\_logic;

Q6 : OUT std\_logic;

Q7 : OUT std\_logic

);

END COMPONENT;

--Inputs

signal A0 : std\_logic := '0';

signal A1 : std\_logic := '0';

signal A2 : std\_logic := '0';

--Outputs

signal Q0 : std\_logic;

signal Q1 : std\_logic;

signal Q2 : std\_logic;

signal Q3 : std\_logic;

signal Q4 : std\_logic;

signal Q5 : std\_logic;

signal Q6 : std\_logic;

signal Q7 : std\_logic;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: decoder\_3to8 PORT MAP (

A0 => A0,

A1 => A1,

A2 => A2,

Q0 => Q0,

Q1 => Q1,

Q2 => Q2,

Q3 => Q3,

Q4 => Q4,

Q5 => Q5,

Q6 => Q6,

Q7 => Q7

);

-- Not a clocked component

-- Stimulus process

stim\_proc: process

begin

wait for 10ns;

A0 <= '0';

A1 <= '0';

A2 <= '0';

wait for 10ns;

A0 <= '1';

A1 <= '0';

A2 <= '0';

wait for 10ns;

A0 <= '0';

A1 <= '1';

A2 <= '0';

wait for 10ns;

A0 <= '1';

A1 <= '1';

A2 <= '0';

wait for 10ns;

A0 <= '0';

A1 <= '0';

A2 <= '1';

wait for 10ns;

A0 <= '1';

A1 <= '0';

A2 <= '1';

wait for 10ns;

A0 <= '0';

A1 <= '1';

A2 <= '1';

wait for 10ns;

A0 <= '1';

A1 <= '1';

A2 <= '1';

end process;

END;

## MUX2 16bit Test Bench:

Cycles through putting 0xFFFF and 0xAAAA on the output.

ENTITY mux2\_16bit\_tb IS

END mux2\_16bit\_tb;

ARCHITECTURE behavior OF mux2\_16bit\_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT mux2\_16bit

PORT(

s : IN std\_logic;

ln0 : IN std\_logic\_vector(15 downto 0);

ln1 : IN std\_logic\_vector(15 downto 0);

Z : OUT std\_logic\_vector(15 downto 0)

);

END COMPONENT;

--Inputs

signal s : std\_logic := '0';

signal ln0 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln1 : std\_logic\_vector(15 downto 0) := (others => '0');

--Outputs

signal Z : std\_logic\_vector(15 downto 0);

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: mux2\_16bit PORT MAP (

s => s,

ln0 => ln0,

ln1 => ln1,

Z => Z

);

-- Stimulus process

stim\_proc: process

begin

wait for 10ns;

ln0 <= x"FFFF";

ln1 <= x"AAAA";

wait for 10ns;

s <= '0';

wait for 10ns;

s <= '1';

end process;

END;

## MUX8 16bit Test Bench:

Cycles through putting 0xFFFF, 0xEEEE down to 0x8888 on the output.

ENTITY mux8\_16bit\_tb IS

END mux8\_16bit\_tb;

ARCHITECTURE behavior OF mux8\_16bit\_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT mux8\_16bit

PORT(

ln0 : IN std\_logic\_vector(15 downto 0);

ln1 : IN std\_logic\_vector(15 downto 0);

ln2 : IN std\_logic\_vector(15 downto 0);

ln3 : IN std\_logic\_vector(15 downto 0);

ln4 : IN std\_logic\_vector(15 downto 0);

ln5 : IN std\_logic\_vector(15 downto 0);

ln6 : IN std\_logic\_vector(15 downto 0);

ln7 : IN std\_logic\_vector(15 downto 0);

S0 : IN std\_logic;

S1 : IN std\_logic;

S2 : IN std\_logic;

Z : OUT std\_logic\_vector(15 downto 0)

);

END COMPONENT;

--Inputs

signal ln0 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln1 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln2 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln3 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln4 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln5 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln6 : std\_logic\_vector(15 downto 0) := (others => '0');

signal ln7 : std\_logic\_vector(15 downto 0) := (others => '0');

signal S0 : std\_logic := '0';

signal S1 : std\_logic := '0';

signal S2 : std\_logic := '0';

--Outputs

signal Z : std\_logic\_vector(15 downto 0);

-- No clocks detected in port list. Replace <clock> below with

-- appropriate port name

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: mux8\_16bit PORT MAP (

ln0 => ln0,

ln1 => ln1,

ln2 => ln2,

ln3 => ln3,

ln4 => ln4,

ln5 => ln5,

ln6 => ln6,

ln7 => ln7,

S0 => S0,

S1 => S1,

S2 => S2,

Z => Z

);

-- Stimulus process

stim\_proc: process

begin

ln0 <= x"FFFF";

ln1 <= x"EEEE";

ln2 <= x"DDDD";

ln3 <= x"CCCC";

ln4 <= x"BBBB";

ln5 <= x"AAAA";

ln6 <= x"9999";

ln7 <= x"8888";

wait for 10ns;

S0 <= '0';

S1 <= '0';

S2 <= '0';

wait for 10ns;

S0 <= '1';

S1 <= '0';

S2 <= '0';

wait for 10ns;

S0 <= '0';

S1 <= '1';

S2 <= '0';

wait for 10ns;

S0 <= '1';

S1 <= '1';

S2 <= '0';

wait for 10ns;

S0 <= '0';

S1 <= '0';

S2 <= '1';

wait for 10ns;

S0 <= '1';

S1 <= '0';

S2 <= '1';

wait for 10ns;

S0 <= '0';

S1 <= '1';

S2 <= '1';

wait for 10ns;

S0 <= '1';

S1 <= '1';

S2 <= '1';

end process;

END;

## Register 16bit Test Bench:

Turns load on, puts 0xFFFF into the register, turns load off, turns load on, puts 0xAAAA into the register.

ENTITY reg16\_tb IS

END reg16\_tb;

ARCHITECTURE behavior OF reg16\_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT reg16

PORT(

D : IN std\_logic\_vector(15 downto 0);

load : IN std\_logic;

Clk : IN std\_logic;

Q : OUT std\_logic\_vector(15 downto 0)

);

END COMPONENT;

--Inputs

signal D : std\_logic\_vector(15 downto 0) := (others => '0');

signal load : std\_logic := '0';

signal Clk : std\_logic := '0';

--Outputs

signal Q : std\_logic\_vector(15 downto 0);

-- Clock period definitions

constant Clk\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: reg16 PORT MAP (

D => D,

load => load,

Clk => Clk,

Q => Q

);

-- Clock process definitions

Clk\_process :process

begin

Clk <= '0';

wait for Clk\_period/2;

Clk <= '1';

wait for Clk\_period/2;

end process;

-- Stimulus process

stim\_proc: process

begin

wait for 10ns;

D <= x"FFFF";

load <= '1';

wait for 10ns;

load <= '0';

wait for 10ns;

D <= x"AAAA";

load <= '1';

wait for 10ns;

load <= '0';

end process;

END;

## Register File 8 16bit Registers Test Bench:

Scenario 1: Loads the value 0xFFFF into R0, 0xEEEE into R1 and so on to 0x8888 into R7.

ENTITY regfile\_16bit\_tb IS

END regfile\_16bit\_tb;

ARCHITECTURE behavior OF regfile\_16bit\_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT regfile\_16bit

PORT(

src\_s0 : IN std\_logic;

src\_s1 : IN std\_logic;

src\_s2 : IN std\_logic;

des\_A0 : IN std\_logic;

des\_A1 : IN std\_logic;

des\_A2 : IN std\_logic;

Clk : IN std\_logic;

data\_src : IN std\_logic;

data : IN std\_logic\_vector(15 downto 0);

reg0 : OUT std\_logic\_vector(15 downto 0);

reg1 : OUT std\_logic\_vector(15 downto 0);

reg2 : OUT std\_logic\_vector(15 downto 0);

reg3 : OUT std\_logic\_vector(15 downto 0);

reg4 : OUT std\_logic\_vector(15 downto 0);

reg5 : OUT std\_logic\_vector(15 downto 0);

reg6 : OUT std\_logic\_vector(15 downto 0);

reg7 : OUT std\_logic\_vector(15 downto 0)

);

END COMPONENT;

--Inputs

signal src\_s0 : std\_logic := '0';

signal src\_s1 : std\_logic := '0';

signal src\_s2 : std\_logic := '0';

signal des\_A0 : std\_logic := '0';

signal des\_A1 : std\_logic := '0';

signal des\_A2 : std\_logic := '0';

signal Clk : std\_logic := '0';

signal data\_src : std\_logic := '0';

signal data : std\_logic\_vector(15 downto 0) := (others => '0');

--Outputs

signal reg0 : std\_logic\_vector(15 downto 0);

signal reg1 : std\_logic\_vector(15 downto 0);

signal reg2 : std\_logic\_vector(15 downto 0);

signal reg3 : std\_logic\_vector(15 downto 0);

signal reg4 : std\_logic\_vector(15 downto 0);

signal reg5 : std\_logic\_vector(15 downto 0);

signal reg6 : std\_logic\_vector(15 downto 0);

signal reg7 : std\_logic\_vector(15 downto 0);

-- Clock period definitions

constant Clk\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: regfile\_16bit PORT MAP (

src\_s0 => src\_s0,

src\_s1 => src\_s1,

src\_s2 => src\_s2,

des\_A0 => des\_A0,

des\_A1 => des\_A1,

des\_A2 => des\_A2,

Clk => Clk,

data\_src => data\_src,

data => data,

reg0 => reg0,

reg1 => reg1,

reg2 => reg2,

reg3 => reg3,

reg4 => reg4,

reg5 => reg5,

reg6 => reg6,

reg7 => reg7

);

-- Clock process definitions

Clk\_process :process

begin

Clk <= '0';

wait for Clk\_period/2;

Clk <= '1';

wait for Clk\_period/2;

end process;

-- Stimulus process

stim\_proc: process

begin

wait for 10ns;

des\_A0 <= '0';

des\_A1 <= '0';

des\_A2 <= '0';

data <= x"FFFF";

wait for 10ns;

des\_A0 <= '1';

des\_A1 <= '0';

des\_A2 <= '0';

data <= x"EEEE";

wait for 10ns;

des\_A0 <= '0';

des\_A1 <= '1';

des\_A2 <= '0';

data <= x"DDDD";

wait for 10ns;

des\_A0 <= '1';

des\_A1 <= '1';

des\_A2 <= '0';

data <= x"CCCC";

wait for 10ns;

des\_A0 <= '0';

des\_A1 <= '0';

des\_A2 <= '1';

data <= x"BBBB";

wait for 10ns;

des\_A0 <= '1';

des\_A1 <= '0';

des\_A2 <= '1';

data <= x"AAAA";

wait for 10ns;

des\_A0 <= '0';

des\_A1 <= '1';

des\_A2 <= '1';

data <= x"9999";

wait for 10ns;

des\_A0 <= '1';

des\_A1 <= '1';

des\_A2 <= '1';

data <= x"8888";

end process;

END;

Scenario 2: Copies the value of R0 into the other seven registers.

ENTITY regfile\_16bit\_tb IS

END regfile\_16bit\_tb;

ARCHITECTURE behavior OF regfile\_16bit\_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT regfile\_16bit

PORT(

src\_s0 : IN std\_logic;

src\_s1 : IN std\_logic;

src\_s2 : IN std\_logic;

des\_A0 : IN std\_logic;

des\_A1 : IN std\_logic;

des\_A2 : IN std\_logic;

Clk : IN std\_logic;

data\_src : IN std\_logic;

data : IN std\_logic\_vector(15 downto 0);

reg0 : OUT std\_logic\_vector(15 downto 0);

reg1 : OUT std\_logic\_vector(15 downto 0);

reg2 : OUT std\_logic\_vector(15 downto 0);

reg3 : OUT std\_logic\_vector(15 downto 0);

reg4 : OUT std\_logic\_vector(15 downto 0);

reg5 : OUT std\_logic\_vector(15 downto 0);

reg6 : OUT std\_logic\_vector(15 downto 0);

reg7 : OUT std\_logic\_vector(15 downto 0)

);

END COMPONENT;

--Inputs

signal src\_s0 : std\_logic := '0';

signal src\_s1 : std\_logic := '0';

signal src\_s2 : std\_logic := '0';

signal des\_A0 : std\_logic := '0';

signal des\_A1 : std\_logic := '0';

signal des\_A2 : std\_logic := '0';

signal Clk : std\_logic := '0';

signal data\_src : std\_logic := '0';

signal data : std\_logic\_vector(15 downto 0) := (others => '0');

--Outputs

signal reg0 : std\_logic\_vector(15 downto 0);

signal reg1 : std\_logic\_vector(15 downto 0);

signal reg2 : std\_logic\_vector(15 downto 0);

signal reg3 : std\_logic\_vector(15 downto 0);

signal reg4 : std\_logic\_vector(15 downto 0);

signal reg5 : std\_logic\_vector(15 downto 0);

signal reg6 : std\_logic\_vector(15 downto 0);

signal reg7 : std\_logic\_vector(15 downto 0);

-- Clock period definitions

constant Clk\_period : time := 10 ns;

BEGIN

-- Instantiate the Unit Under Test (UUT)

uut: regfile\_16bit PORT MAP (

src\_s0 => src\_s0,

src\_s1 => src\_s1,

src\_s2 => src\_s2,

des\_A0 => des\_A0,

des\_A1 => des\_A1,

des\_A2 => des\_A2,

Clk => Clk,

data\_src => data\_src,

data => data,

reg0 => reg0,

reg1 => reg1,

reg2 => reg2,

reg3 => reg3,

reg4 => reg4,

reg5 => reg5,

reg6 => reg6,

reg7 => reg7

);

-- Clock process definitions

Clk\_process :process

begin

Clk <= '0';

wait for Clk\_period/2;

Clk <= '1';

wait for Clk\_period/2;

end process;

-- Stimulus process

stim\_proc: process

begin

data <= x"FFFF";

wait for 10ns;

des\_A0 <= '0';

des\_A1 <= '0';

des\_A2 <= '0';

wait for 10ns;

data\_src <= '1';

wait for 10ns; -- R1

des\_A0 <= '1';

des\_A1 <= '0';

des\_A2 <= '0';

wait for 10ns; -- R2

des\_A0 <= '0';

des\_A1 <= '1';

des\_A2 <= '0';

wait for 10ns; -- R3

des\_A0 <= '1';

des\_A1 <= '1';

des\_A2 <= '0';

wait for 10ns; -- R4

des\_A0 <= '0';

des\_A1 <= '0';

des\_A2 <= '1';

wait for 10ns; -- R5

des\_A0 <= '1';

des\_A1 <= '0';

des\_A2 <= '1';

wait for 10ns; -- R6

des\_A0 <= '0';

des\_A1 <= '1';

des\_A2 <= '1';

wait for 10ns; -- R7

des\_A0 <= '1';

des\_A1 <= '1';

des\_A2 <= '1';

end process;

END;

# Screenshots of Test Bench Simulations

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCacheContent.Word\decodertb.pngDecoder 3to 8 16bit:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCacheContent.Word\mux2_16bit_tb.pngMUX2 16bit:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCacheContent.Word\mux8_16bit_tb.pngMUX8 16bit:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCacheContent.Word\reg16tb.pngRegister 16bit:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCacheContent.Word\regfile16bit_tb_reg2reg.pngC:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCacheContent.Word\regfile16bit_tb.pngRegister File with 8 16bit Registers:

Scenario 2

Scenario