CS2022 Datapath Design Part B

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

# VHDL Code

This section consists of code from the previous assignment, modified slightly in some cases to improve readability or functionality, and new code written to implement a full Datapath design.

## Decoder 3 to 8:

**entity** decoder\_3to8 **is**

**Port** **(** A **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

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 A**(**0**))** and **(**not A**(**1**))** and **(**not A**(**2**)))** **after** 1ns**;**

Q1**<=(**A**(**0**)** and **(**not A**(**1**))** and **(**not A**(**2**)))** **after** 1ns**;**

Q2**<=((**not A**(**0**))** and A**(**1**)** and **(**not A**(**2**)))** **after** 1ns**;**

Q3**<=(**A**(**0**)** and A**(**1**)** and **(**not A**(**2**)))** **after** 1ns**;**

Q4**<=((**not A**(**0**))** and **(**not A**(**1**))** and A**(**2**))** **after** 1ns**;**

Q5**<=(**A**(**0**)** and **(**not A**(**1**))** and A**(**2**))** **after** 1ns**;**

Q6**<=((**not A**(**0**))** and A**(**1**)** and A**(**2**))** **after** 1ns**;**

Q7**<=(**A**(**0**)** and A**(**1**)** and A**(**2**))** **after** 1ns**;**

**end** Behavioral**;**

## Mux 2 16 bit:

**entity** mux2\_16bit **is**

**Port** **(** IN0 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC**;**

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

**end** mux2\_16bit**;**

**architecture** Behavioral **of** mux2\_16bit **is**

**begin**

Z **<=** IN0 **after** 1ns **when** S **=** '0' **else**

IN1 **after** 1ns **when** S **=** '1' **else**

x"0000" **after** 1ns**;**

**end** Behavioral**;**

## Mux 3 1 bit:

**entity** mux3\_1bit **is**

**Port** **(** IN0 **:** **in** STD\_LOGIC**;**

IN1 **:** **in** STD\_LOGIC**;**

IN2 **:** **in** STD\_LOGIC**;**

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

Z **:** **out** STD\_LOGIC**);**

**end** mux3\_1bit**;**

**architecture** Behavioral **of** mux3\_1bit **is**

**begin**

Z **<=** IN0 **after** 1ns **when** S**(**0**)** **=** '0' and S**(**1**)** **=** '0' **else**

IN1 **after** 1ns **when** S**(**0**)** **=** '1' and S**(**1**)** **=** '0' **else**

IN2 **after** 1ns **when** S**(**0**)** **=** '0' and S**(**1**)** **=** '1' **else**

'0' **after** 1ns**;**

**end** Behavioral**;**

## Mux 4 16 bit:

**entity** mux4\_16bit **is**

**Port** **(** IN0 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN2 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN3 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

**end** mux4\_16bit**;**

**architecture** Behavioral **of** mux4\_16bit **is**

**begin**

Z **<=** IN0 **after** 1ns **when** S**(**0**)=**'0' and S**(**1**)=**'0' **else**

IN1 **after** 1ns **when** S**(**0**)=**'1' and S**(**1**)=**'0' **else**

IN2 **after** 1ns **when** S**(**0**)=**'0' and S**(**1**)=**'1' **else**

IN3 **after** 1ns **when** S**(**0**)=**'1' and S**(**1**)=**'1' **else**

x"0000" **after** 1ns**;**

**end** Behavioral**;**

## Mux 8 16 bit:

**entity** mux8\_16bit **is**

**Port** **(** IN0 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN2 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN3 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN4 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN5 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN6 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN7 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

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

**end** mux8\_16bit**;**

**architecture** Behavioral **of** mux8\_16bit **is**

**begin**

Z **<=** IN0 **after** 1ns **when** S**(**0**)=**'0' and S**(**1**)=**'0' and S**(**2**)=**'0' **else**

IN1 **after** 1ns **when** S**(**0**)=**'1' and S**(**1**)=**'0' and S**(**2**)=**'0' **else**

IN2 **after** 1ns **when** S**(**0**)=**'0' and S**(**1**)=**'1' and S**(**2**)=**'0' **else**

IN3 **after** 1ns **when** S**(**0**)=**'1' and S**(**1**)=**'1' and S**(**2**)=**'0' **else**

IN4 **after** 1ns **when** S**(**0**)=**'0' and S**(**1**)=**'0' and S**(**2**)=**'1' **else**

IN5 **after** 1ns **when** S**(**0**)=**'1' and S**(**1**)=**'0' and S**(**2**)=**'1' **else**

IN6 **after** 1ns **when** S**(**0**)=**'0' and S**(**1**)=**'1' and S**(**2**)=**'1' **else**

IN7 **after** 1ns **when** S**(**0**)=**'1' and S**(**1**)=**'1' and S**(**2**)=**'1' **else**

"0000000000000000" **after** 1ns**;**

**end** Behavioral**;**

## Register 16 bit:

**entity** reg16 **is**

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

LOAD **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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**(**0**)** **=**'1' and LOAD**(**1**)** **=**'1' **then**

Q**<=**D **after** 1 ns**;**

**end** **if;**

**end** **if;**

**end** **process;**

**end** Behavioral**;**

## Register File with 8 16 bit registers and A and B buses:

**entity** regfile\_16bit **is**

**Port** **(** A\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

B\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

D\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

A\_Data **:** **out** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

B\_Data **:** **out** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

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

Load\_En **:** **in** STD\_LOGIC**;**

CLK **:** **in** STD\_LOGIC**);**

**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\_VECTOR**(**1 **downto** 0**);**

CLK **:** **in** STD\_LOGIC**;**

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

**);**

**END** **COMPONENT;**

-- 3 to 8 Decoder

**COMPONENT** decoder\_3to8

**PORT(** A **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

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;**

-- MUX8 16 bit

**COMPONENT** mux8\_16bit

**PORT(**IN0 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN2 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN3 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN4 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN5 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN6 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN7 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

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**);**

**begin**

-- Port Maps

-- Registers

reg00**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg0**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg0\_q

**);**

reg01**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg1**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg1\_q

**);**

reg02**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg2**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg2\_q

**);**

reg03**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg3**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg3\_q

**);**

reg04**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg4**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg4\_q

**);**

reg05**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg5**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg5\_q

**);**

reg06**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg6**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg6\_q

**);**

reg07**:** reg16 **PORT** **MAP(**

D **=>** D\_DATA**,**

LOAD**(**0**)** **=>** load\_reg7**,**

LOAD**(**1**)** **=>** Load\_En**,**

CLK **=>** CLK**,**

Q **=>** reg7\_q

**);**

-------------------------------------------

-- Destination Register Decoder

dest\_decoder\_3to8**:** decoder\_3to8 **PORT** **MAP(**

A**(**0**)** **=>** D\_Sel**(**0**),**

A**(**1**)** **=>** D\_Sel**(**1**),**

A**(**2**)** **=>** D\_Sel**(**2**),**

Q0 **=>** load\_reg0**,**

Q1 **=>** load\_reg1**,**

Q2 **=>** load\_reg2**,**

Q3 **=>** load\_reg3**,**

Q4 **=>** load\_reg4**,**

Q5 **=>** load\_reg5**,**

Q6 **=>** load\_reg6**,**

Q7 **=>** load\_reg7

**);**

---------------------------------------------

-- MUX A

muxA**:** mux8\_16bit **PORT** **MAP(**

IN0 **=>** reg0\_q**,**

IN1 **=>** reg1\_q**,**

IN2 **=>** reg2\_q**,**

IN3 **=>** reg3\_q**,**

IN4 **=>** reg4\_q**,**

IN5 **=>** reg5\_q**,**

IN6 **=>** reg6\_q**,**

IN7 **=>** reg7\_q**,**

S**(**0**)** **=>** A\_Sel**(**0**),**

S**(**1**)** **=>** A\_Sel**(**1**),**

S**(**2**)** **=>** A\_Sel**(**2**),**

Z **=>** A\_Data

**);**

-- MUX B

muxB**:** mux8\_16bit **PORT** **MAP(**

IN0 **=>** reg0\_q**,**

IN1 **=>** reg1\_q**,**

IN2 **=>** reg2\_q**,**

IN3 **=>** reg3\_q**,**

IN4 **=>** reg4\_q**,**

IN5 **=>** reg5\_q**,**

IN6 **=>** reg6\_q**,**

IN7 **=>** reg7\_q**,**

S**(**0**)** **=>** B\_Sel**(**0**),**

S**(**1**)** **=>** B\_Sel**(**1**),**

S**(**2**)** **=>** B\_Sel**(**2**),**

Z **=>** B\_Data

**);**

---------------------------------------------

**end** Behavioral**;**

## Full Adder:

**entity** full\_adder **is**

**Port** **(** X **:** **in** STD\_LOGIC**;**

Y **:** **in** STD\_LOGIC**;**

Cin **:** **in** STD\_LOGIC**;**

Sum **:** **out** STD\_LOGIC**;**

Cout **:** **out** STD\_LOGIC**);**

**end** full\_adder**;**

**architecture** Behavioral **of** full\_adder **is**

**begin**

Sum **<=** X xor Y xor Cin **after** 1ns**;**

Cout **<=** **(**X and Y**)** or **(**Cin and **(**X xor Y**))** **after** 1ns**;**

**end** Behavioral**;**

## Ripple Adder 16 bit:

**entity** ripple\_adder\_16bit **is**

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

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

Cin **:** **in** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

Cout **:** **out** STD\_LOGIC**;**

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

**end** ripple\_adder\_16bit**;**

**architecture** Behavioral **of** ripple\_adder\_16bit **is**

-- Components

-- Full Adder

**COMPONENT** full\_adder

**PORT** **(** X **:** **in** STD\_LOGIC**;**

Y **:** **in** STD\_LOGIC**;**

Cin **:** **in** STD\_LOGIC**;**

Sum **:** **out** STD\_LOGIC**;**

Cout **:** **out** STD\_LOGIC

**);**

**END** **COMPONENT;**

**signal** C1**,** C2**,** C3**,** C4**,** C5**,** C6**,** C7**,** C8**,** C9**,** C10**,** C11**,** C12**,** C13**,** C14**,** C15**,** C16**:** STD\_LOGIC**;**

**begin**

-- Port Maps

-- Full Adders

FA0**:** full\_adder **PORT** **MAP(**

X **=>** A**(**0**),**

Y **=>** B**(**0**),**

Cin **=>** Cin**,**

Sum **=>** G**(**0**),**

Cout **=>** C1

**);**

FA1**:** full\_adder **PORT** **MAP(**

X **=>** A**(**1**),**

Y **=>** B**(**1**),**

Cin **=>** C1**,**

Sum **=>** G**(**1**),**

Cout **=>** C2

**);**

FA2**:** full\_adder **PORT** **MAP(**

X **=>** A**(**2**),**

Y **=>** B**(**2**),**

Cin **=>** C2**,**

Sum **=>** G**(**2**),**

Cout **=>** C3

**);**

FA3**:** full\_adder **PORT** **MAP(**

X **=>** A**(**3**),**

Y **=>** B**(**3**),**

Cin **=>** C3**,**

Sum **=>** G**(**3**),**

Cout **=>** C4

**);**

FA4**:** full\_adder **PORT** **MAP(**

X **=>** A**(**4**),**

Y **=>** B**(**4**),**

Cin **=>** C4**,**

Sum **=>** G**(**4**),**

Cout **=>** C5

**);**

FA5**:** full\_adder **PORT** **MAP(**

X **=>** A**(**5**),**

Y **=>** B**(**5**),**

Cin **=>** C5**,**

Sum **=>** G**(**5**),**

Cout **=>** C6

**);**

FA6**:** full\_adder **PORT** **MAP(**

X **=>** A**(**6**),**

Y **=>** B**(**6**),**

Cin **=>** C6**,**

Sum **=>** G**(**6**),**

Cout **=>** C7

**);**

FA7**:** full\_adder **PORT** **MAP(**

X **=>** A**(**7**),**

Y **=>** B**(**7**),**

Cin **=>** C7**,**

Sum **=>** G**(**7**),**

Cout **=>** C8

**);**

FA8**:** full\_adder **PORT** **MAP(**

X **=>** A**(**8**),**

Y **=>** B**(**8**),**

Cin **=>** C8**,**

Sum **=>** G**(**8**),**

Cout **=>** C9

**);**

FA9**:** full\_adder **PORT** **MAP(**

X **=>** A**(**9**),**

Y **=>** B**(**9**),**

Cin **=>** C9**,**

Sum **=>** G**(**9**),**

Cout **=>** C10

**);**

FA10**:** full\_adder **PORT** **MAP(**

X **=>** A**(**10**),**

Y **=>** B**(**10**),**

Cin **=>** C10**,**

Sum **=>** G**(**10**),**

Cout **=>** C11

**);**

FA11**:** full\_adder **PORT** **MAP(**

X **=>** A**(**11**),**

Y **=>** B**(**11**),**

Cin **=>** C11**,**

Sum **=>** G**(**11**),**

Cout **=>** C12

**);**

FA12**:** full\_adder **PORT** **MAP(**

X **=>** A**(**12**),**

Y **=>** B**(**12**),**

Cin **=>** C12**,**

Sum **=>** G**(**12**),**

Cout **=>** C13

**);**

FA13**:** full\_adder **PORT** **MAP(**

X **=>** A**(**13**),**

Y **=>** B**(**13**),**

Cin **=>** C13**,**

Sum **=>** G**(**13**),**

Cout **=>** C14

**);**

FA14**:** full\_adder **PORT** **MAP(**

X **=>** A**(**14**),**

Y **=>** B**(**14**),**

Cin **=>** C14**,**

Sum **=>** G**(**14**),**

Cout **=>** C15

**);**

FA15**:** full\_adder **PORT** **MAP(**

X **=>** A**(**15**),**

Y **=>** B**(**15**),**

Cin **=>** C15**,**

Sum **=>** G**(**15**),**

Cout **=>** C16

**);**

Cout **<=** C16**;**

V **<=** **(**C16 xor C15**);**

**end** Behavioral**;**

## B Input Logic:

**entity** b\_input\_logic **is**

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

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

**end** b\_input\_logic**;**

**architecture** Behavioral **of** b\_input\_logic **is**

**begin**

Y**(**0**)** **<=** **(**B**(**0**)** and S**(**0**))** or **(**not B**(**0**)** and S**(**1**))** **after** 1ns**;**

Y**(**1**)** **<=** **(**B**(**1**)** and S**(**0**))** or **(**not B**(**1**)** and S**(**1**))** **after** 1ns**;**

Y**(**2**)** **<=** **(**B**(**2**)** and S**(**0**))** or **(**not B**(**2**)** and S**(**1**))** **after** 1ns**;**

Y**(**3**)** **<=** **(**B**(**3**)** and S**(**0**))** or **(**not B**(**3**)** and S**(**1**))** **after** 1ns**;**

Y**(**4**)** **<=** **(**B**(**4**)** and S**(**0**))** or **(**not B**(**4**)** and S**(**1**))** **after** 1ns**;**

Y**(**5**)** **<=** **(**B**(**5**)** and S**(**0**))** or **(**not B**(**5**)** and S**(**1**))** **after** 1ns**;**

Y**(**6**)** **<=** **(**B**(**6**)** and S**(**0**))** or **(**not B**(**6**)** and S**(**1**))** **after** 1ns**;**

Y**(**7**)** **<=** **(**B**(**7**)** and S**(**0**))** or **(**not B**(**7**)** and S**(**1**))** **after** 1ns**;**

Y**(**8**)** **<=** **(**B**(**8**)** and S**(**0**))** or **(**not B**(**8**)** and S**(**1**))** **after** 1ns**;**

Y**(**9**)** **<=** **(**B**(**9**)** and S**(**0**))** or **(**not B**(**9**)** and S**(**1**))** **after** 1ns**;**

Y**(**10**)** **<=** **(**B**(**10**)** and S**(**0**))** or **(**not B**(**10**)** and S**(**1**))** **after** 1ns**;**

Y**(**11**)** **<=** **(**B**(**11**)** and S**(**0**))** or **(**not B**(**11**)** and S**(**1**))** **after** 1ns**;**

Y**(**12**)** **<=** **(**B**(**12**)** and S**(**0**))** or **(**not B**(**12**)** and S**(**1**))** **after** 1ns**;**

Y**(**13**)** **<=** **(**B**(**13**)** and S**(**0**))** or **(**not B**(**13**)** and S**(**1**))** **after** 1ns**;**

Y**(**14**)** **<=** **(**B**(**14**)** and S**(**0**))** or **(**not B**(**14**)** and S**(**1**))** **after** 1ns**;**

Y**(**15**)** **<=** **(**B**(**15**)** and S**(**0**))** or **(**not B**(**15**)** and S**(**1**))** **after** 1ns**;**

**end** Behavioral**;**

## Arithmetic Unit:

**entity** arithmetic\_unit **is**

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

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

Cin **:** **in** STD\_LOGIC**;**

SelB **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

Cout **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**);**

**end** arithmetic\_unit**;**

**architecture** Behavioral **of** arithmetic\_unit **is**

-- Components

-- B Input Logic

**COMPONENT** b\_input\_logic

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

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

**END** **COMPONENT;**

-- 16 bit Ripple Adder

**COMPONENT** ripple\_adder\_16bit

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

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

Cin **:** **in** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

Cout **:** **out** STD\_LOGIC**;**

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

**END** **COMPONENT;**

**signal** blogic\_to\_adder**,** adder\_to\_G\_output**:** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

**begin**

-- Port Maps

-- B Logic

BLOGIC**:** b\_input\_logic **PORT** **MAP(**

B **=>** B**,**

S **=>** SelB**,**

Y **=>** blogic\_to\_adder

**);**

-- Ripple Adder

RA**:** ripple\_adder\_16bit **PORT** **MAP(**

A **=>** A**,**

B **=>** blogic\_to\_adder**,**

Cin **=>** Cin**,**

V **=>** V**,**

Cout **=>** Cout**,**

G **=>** adder\_to\_G\_output

**);**

G **<=** adder\_to\_G\_output **after** 2ns**;**

**end** Behavioral**;**

## Logic Unit:

**entity** logic\_unit **is**

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

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

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

**end** logic\_unit**;**

**architecture** Behavioral **of** logic\_unit **is**

-- Components

-- MUX 4 to 1

**COMPONENT** mux4\_16bit

**PORT(** IN0 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN2 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN3 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

**END** **COMPONENT;**

**signal** AandB**,** AorB**,** AxorB**,** notA**:** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

**begin**

-- Port maps

-- signals

AandB **<=** A and B**;**

AorB **<=** A or B**;**

AxorB **<=** A xor B**;**

notA **<=** not A**;**

-- MUX

MUX**:** mux4\_16bit **PORT** **MAP(**

IN0 **=>** AandB**,**

IN1 **=>** AorB**,**

IN2 **=>** AxorB**,**

IN3 **=>** notA**,**

S **=>** S**,**

Z **=>** G

**);**

**end** Behavioral**;**

## ALU:

**entity** alu\_16bit **is**

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

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

GSel **:** **in** STD\_LOGIC\_VECTOR**(**3 **downto** 0**);**

C **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

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

**end** alu\_16bit**;**

**architecture** Behavioral **of** alu\_16bit **is**

-- Components

-- Arithmetic Circuit

**COMPONENT** arithmetic\_unit

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

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

Cin **:** **in** STD\_LOGIC**;**

SelB **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

Cout **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**);**

**END** **COMPONENT;**

-- Logic Circuit

**COMPONENT** logic\_unit

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

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

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

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

**END** **COMPONENT;**

-- Select Arithmetic or Logic MUX

**COMPONENT** mux2\_16bit

**PORT(**S **:** **in** STD\_LOGIC**;**

IN0 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR **(**15 **downto** 0**);**

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

**END** **COMPONENT;**

**signal** Cin**,** C\_out**,** V\_out**:** STD\_LOGIC**;**

**signal** FSel**:** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

**signal** AUtoMUX**,** LUtoMUX**:** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

**begin**

-- Port Maps

-- Signals

Cin **<=** GSel**(**0**);**

FSel **<=** GSel**(**3 **downto** 1**);**

-- Arithmetic Unit

AU**:** arithmetic\_unit **PORT** **MAP(**

A **=>** A**,**

B **=>** B**,**

Cin **=>** Cin**,**

SelB**(**0**)** **=>** FSel**(**0**),**

SelB**(**1**)** **=>** FSel**(**1**),**

G **=>** AUtoMUX**,**

Cout **=>** C\_out**,**

V **=>** V\_out**);**

-- Logic Unit

LUL**:** logic\_unit **PORT** **MAP(**

A **=>** A**,**

B **=>** B**,**

S**(**0**)** **=>** FSel**(**0**),**

S**(**1**)** **=>** FSel**(**1**),**

G **=>** LUtoMUX **);**

-- MUX

ALUMUX**:** mux2\_16bit **PORT** **MAP(**

S **=>** FSel**(**2**),**

IN0 **=>** AUtoMUX**,**

IN1 **=>** LUtoMUX**,**

Z **=>** G **);**

-- Control Flags

C **<=** C\_out **when** FSel**(**2**)** **=** '0' **else**

'0'**;**

V **<=** V\_out **when** FSel**(**2**)** **=** '0' **else**

'0'**;**

**end** Behavioral**;**

## Shifter:

**entity** shifter\_16bit **is**

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

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

IR **:** **in** STD\_LOGIC**;**

IL **:** **in** STD\_LOGIC**;**

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

**end** shifter\_16bit**;**

**architecture** Behavioral **of** shifter\_16bit **is**

-- Components

-- MUX3

**COMPONENT** mux3\_1bit

**PORT(**IN0 **:** **in** STD\_LOGIC**;**

IN1 **:** **in** STD\_LOGIC**;**

IN2 **:** **in** STD\_LOGIC**;**

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

Z **:** **out** STD\_LOGIC**);**

**END** **COMPONENT;**

**begin**

-- Port Maps

--MUX0

MUX0**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**0**),**

IN1 **=>** B**(**1**),**

IN2 **=>** IL**,**

S **=>** S**,**

Z **=>** H**(**0**));**

--MUX1

MUX1**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**1**),**

IN1 **=>** B**(**2**),**

IN2 **=>** B**(**0**),**

S **=>** S**,**

Z **=>** H**(**1**));**

--MUX2

MUX2**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**2**),**

IN1 **=>** B**(**3**),**

IN2 **=>** B**(**1**),**

S **=>** S**,**

Z **=>** H**(**2**));**

--MUX3

MUX3**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**3**),**

IN1 **=>** B**(**4**),**

IN2 **=>** B**(**2**),**

S **=>** S**,**

Z **=>** H**(**3**));**

--MUX4

MUX4**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**4**),**

IN1 **=>** B**(**5**),**

IN2 **=>** B**(**3**),**

S **=>** S**,**

Z **=>** H**(**4**));**

--MUX5

MUX5**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**5**),**

IN1 **=>** B**(**6**),**

IN2 **=>** B**(**4**),**

S **=>** S**,**

Z **=>** H**(**5**));**

--MUX6

MUX6**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**6**),**

IN1 **=>** B**(**7**),**

IN2 **=>** B**(**5**),**

S **=>** S**,**

Z **=>** H**(**6**));**

--MUX7

MUX7**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**7**),**

IN1 **=>** B**(**8**),**

IN2 **=>** B**(**6**),**

S **=>** S**,**

Z **=>** H**(**7**));**

--MUX8

MUX8**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**8**),**

IN1 **=>** B**(**9**),**

IN2 **=>** B**(**7**),**

S **=>** S**,**

Z **=>** H**(**8**));**

--MUX9

MUX9**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**9**),**

IN1 **=>** B**(**10**),**

IN2 **=>** B**(**8**),**

S **=>** S**,**

Z **=>** H**(**9**));**

--MUX10

MUX10**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**10**),**

IN1 **=>** B**(**11**),**

IN2 **=>** B**(**9**),**

S **=>** S**,**

Z **=>** H**(**10**));**

--MUX11

MUX11**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**11**),**

IN1 **=>** B**(**12**),**

IN2 **=>** B**(**10**),**

S **=>** S**,**

Z **=>** H**(**11**));**

--MUX12

MUX12**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**12**),**

IN1 **=>** B**(**13**),**

IN2 **=>** B**(**11**),**

S **=>** S**,**

Z **=>** H**(**12**));**

--MUX13

MUX13**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**13**),**

IN1 **=>** B**(**14**),**

IN2 **=>** B**(**12**),**

S **=>** S**,**

Z **=>** H**(**13**));**

--MUX14

MUX14**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**14**),**

IN1 **=>** B**(**15**),**

IN2 **=>** B**(**13**),**

S **=>** S**,**

Z **=>** H**(**14**));**

--MUX15

MUX15**:** mux3\_1bit

**PORT** **MAP(**IN0 **=>** B**(**15**),**

IN1 **=>** IR**,**

IN2 **=>** B**(**14**),**

S **=>** S**,**

Z **=>** H**(**15**));**

**end** Behavioral**;**

## Function Unit:

**entity** function\_unit **is**

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

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

FSel **:** **in** STD\_LOGIC\_VECTOR**(**4 **downto** 0**);**

C **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

N **:** **out** STD\_LOGIC**;**

Z **:** **out** STD\_LOGIC**;**

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

**end** function\_unit**;**

**architecture** Behavioral **of** function\_unit **is**

-- Components

-- ALU

**COMPONENT** alu\_16bit

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

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

GSel **:** **in** STD\_LOGIC\_VECTOR**(**3 **downto** 0**);**

C **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

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

**END** **COMPONENT;**

-- Shifter

**COMPONENT** shifter\_16bit

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

S **:** **in** STD\_LOGIC\_VECTOR**(**1 **downto** 0**);**

IR **:** **in** STD\_LOGIC**;**

IL **:** **in** STD\_LOGIC**;**

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

**END** **COMPONENT;**

-- MUX2

**COMPONENT** mux2\_16bit

**PORT(**IN0 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC**;**

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

**END** **COMPONENT;**

**signal** outputALU**,** outputShifter**,** muxFOUT**:** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

**begin**

-- Port Maps

-- ALU

ALU**:** alu\_16bit

**PORT** **MAP(**A **=>** A**,**

B **=>** B**,**

GSel **=>** FSel**(**3 **downto** 0**),**

C **=>** C**,**

V **=>** V**,**

G **=>** outputALU**);**

-- Shifter

SHIFTER**:** shifter\_16bit

**PORT** **MAP(**B **=>** B**,**

S **=>** FSel**(**3 **downto** 2**),**

IR **=>** '0'**,**

IL **=>** '0'**,**

H **=>** outputShifter**);**

-- MUXF

MUXF**:** mux2\_16bit

**PORT** **MAP(**IN0 **=>** outputALU**,**

IN1 **=>** outputShifter**,**

S **=>** FSel**(**4**),**

Z **=>** muxFOUT**);**

Z **<=** '1' **when** muxFOUT **=** x"0000" **else**

'0'**;**

N **<=** '1' **when** muxFOUT**(**15**)** **=** '1' **else**

'0'**;**

F **<=** muxFOUT**;**

**end** Behavioral**;**

## Datapath:

**entity** datapath\_16bit **is**

**Port** **(** CONTROL\_WORD **:** **in** STD\_LOGIC\_VECTOR**(**16 **downto** 0**);**

CONST\_IN **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

DATA\_IN **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

CLK **:** **in** STD\_LOGIC**;**

ADDR\_OUT **:** **out** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

DATA\_OUT **:** **out** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

C **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

N **:** **out** STD\_LOGIC**;**

Z **:** **out** STD\_LOGIC**);**

**end** datapath\_16bit**;**

**architecture** Behavioral **of** datapath\_16bit **is**

-- Components

-- Regfile

**COMPONENT** regfile\_16bit

**PORT(**A\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

B\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

D\_Sel **:** **in** STD\_LOGIC\_VECTOR**(**2 **downto** 0**);**

A\_Data **:** **out** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

B\_Data **:** **out** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

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

Load\_En **:** **in** STD\_LOGIC**;**

CLK **:** **in** STD\_LOGIC**);**

**END** **COMPONENT;**

-- Function Unit

**COMPONENT** function\_unit

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

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

FSel **:** **in** STD\_LOGIC\_VECTOR**(**4 **downto** 0**);**

C **:** **out** STD\_LOGIC**;**

V **:** **out** STD\_LOGIC**;**

N **:** **out** STD\_LOGIC**;**

Z **:** **out** STD\_LOGIC**;**

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

**END** **COMPONENT;**

-- Mux2

**COMPONENT** mux2\_16bit

**PORT(**IN0 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

IN1 **:** **in** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

S **:** **in** STD\_LOGIC**;**

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

**END** **COMPONENT;**

**signal** B\_regfile\_muxB**,** muxB\_out**,** A\_regfile\_function\_unit**,** function\_unit\_out**,** muxD\_out**:** STD\_LOGIC\_VECTOR**(**15 **downto** 0**);**

**begin**

-- Port Maps

-- MUXB

MUXB**:** mux2\_16bit

**PORT** **MAP(**IN0 **=>** B\_regfile\_muxB**,**

IN1 **=>** CONST\_IN**,**

S **=>** CONTROL\_WORD**(**7**),**

Z **=>** muxB\_out**);**

--MUXD

MUXD**:** mux2\_16bit

**PORT** **MAP(**IN0 **=>** function\_unit\_out**,**

IN1 **=>** DATA\_IN**,**

S **=>** CONTROL\_WORD**(**1**),**

Z **=>** muxD\_out**);**

-- Regfile

REGFILE**:** regfile\_16bit

**PORT** **MAP(**A\_Sel **=>** CONTROL\_WORD**(**13 **downto** 11**),**

B\_Sel **=>** CONTROL\_WORD**(**10 **downto** 8**),**

D\_Sel **=>** CONTROL\_WORD**(**16 **downto** 14**),**

A\_Data **=>** A\_regfile\_function\_unit**,**

B\_Data **=>** B\_regfile\_muxB**,**

D\_Data **=>** muxD\_out**,**

Load\_En **=>** CONTROL\_WORD**(**0**),**

CLK **=>** CLK**);**

-- Function Unit

FUNC\_UNIT**:** function\_unit

**PORT** **MAP(**A **=>** A\_regfile\_function\_unit**,**

B **=>** muxB\_out**,**

FSel **=>** CONTROL\_WORD**(**6 **downto** 2**),**

C **=>** C**,**

V **=>** V**,**

N **=>** N**,**

Z **=>** Z**,**

F **=>** function\_unit\_out**);**

-- Signals

ADDR\_OUT **<=** A\_regfile\_function\_unit**;**

DATA\_OUT **<=** muxB\_out**;**

**end** Behavioral**;**

# VHDL Test Benches:

This section shows the code used to test the functionality of all of the components of the datapath including the datapath itself.

## Decoder 3 to 8:

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;**

## Mux 2 16 bit:

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;

## Mux 3 1 bit:

Cycles through ‘1’, ‘0’, ‘1’ on the output.

**ENTITY** mux3\_1bit\_tb **IS**

**END** mux3\_1bit\_tb**;**

**ARCHITECTURE** behavior **OF** mux3\_1bit\_tb **IS**

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

**COMPONENT** mux3\_1bit

**PORT(**

IN0 **:** **IN** std\_logic**;**

IN1 **:** **IN** std\_logic**;**

IN2 **:** **IN** std\_logic**;**

S **:** **IN** std\_logic\_vector**(**1 **downto** 0**);**

Z **:** **OUT** std\_logic

**);**

**END** **COMPONENT;**

--Inputs

**signal** IN0 **:** std\_logic **:=** '0'**;**

**signal** IN1 **:** std\_logic **:=** '0'**;**

**signal** IN2 **:** std\_logic **:=** '0'**;**

**signal** S **:** std\_logic\_vector**(**1 **downto** 0**)** **:=** **(others** **=>** '0'**);**

--Outputs

**signal** Z **:** std\_logic**;**

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** mux3\_1bit **PORT** **MAP** **(**

IN0 **=>** IN0**,**

IN1 **=>** IN1**,**

IN2 **=>** IN2**,**

S **=>** S**,**

Z **=>** Z

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

IN0 **<=** '1'**;**

IN1 **<=** '0'**;**

IN2 **<=** '1'**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

**end** **process;**

**END;**

## Mux 8 16 bit:

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 16 bit:

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 with 8 16 bit registers:

Loads 0xFFFF into R0 with the load enable signal set to high. The load enable signal is set to low and we try to load 0xAAAA into R0 and fail.

**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(**

A\_Sel **:** **IN** std\_logic\_vector**(**2 **downto** 0**);**

B\_Sel **:** **IN** std\_logic\_vector**(**2 **downto** 0**);**

D\_Sel **:** **IN** std\_logic\_vector**(**2 **downto** 0**);**

A\_Data **:** **OUT** std\_logic\_vector**(**15 **downto** 0**);**

B\_Data **:** **OUT** std\_logic\_vector**(**15 **downto** 0**);**

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

Load\_En **:** **IN** std\_logic**;**

CLK **:** **IN** std\_logic

**);**

**END** **COMPONENT;**

--Inputs

**signal** A\_Sel **:** std\_logic\_vector**(**2 **downto** 0**)** **:=** **(others** **=>** '0'**);**

**signal** B\_Sel **:** std\_logic\_vector**(**2 **downto** 0**)** **:=** **(others** **=>** '0'**);**

**signal** D\_Sel **:** std\_logic\_vector**(**2 **downto** 0**)** **:=** **(others** **=>** '0'**);**

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

**signal** Load\_En **:** std\_logic **:=** '0'**;**

**signal** CLK **:** std\_logic **:=** '0'**;**

--Outputs

**signal** A\_Data **:** std\_logic\_vector**(**15 **downto** 0**);**

**signal** B\_Data **:** 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** **(**

A\_Sel **=>** A\_Sel**,**

B\_Sel **=>** B\_Sel**,**

D\_Sel **=>** D\_Sel**,**

A\_Data **=>** A\_Data**,**

B\_Data **=>** B\_Data**,**

D\_Data **=>** D\_Data**,**

Load\_En **=>** Load\_En**,**

CLK **=>** CLK

**);**

-- 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**

D\_Data **<=** x"FFFF"**;**

Load\_En **<=** '1'**;**

D\_Sel**(**0**)** **<=** '0'**;**

D\_Sel**(**1**)** **<=** '0'**;**

D\_Sel**(**2**)** **<=** '0'**;**

**wait** **for** 10ns**;**

D\_data **<=** x"AAAA"**;**

Load\_En **<=** '0'**;**

D\_Sel**(**0**)** **<=** '0'**;**

D\_Sel**(**1**)** **<=** '0'**;**

D\_Sel**(**2**)** **<=** '0'**;**

**wait** **for** 10ns**;**

**end** **process;**

**END;**

## Full Adder:

Cycles through addition of all binary combinations of Cin, X and Y.

**ENTITY** full\_adder\_tb **IS**

**END** full\_adder\_tb**;**

**ARCHITECTURE** behavior **OF** full\_adder\_tb **IS**

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

**COMPONENT** full\_adder

**PORT(**

X **:** **IN** std\_logic**;**

Y **:** **IN** std\_logic**;**

Cin **:** **IN** std\_logic**;**

Sum **:** **OUT** std\_logic**;**

Cout **:** **OUT** std\_logic

**);**

**END** **COMPONENT;**

--Inputs

**signal** X **:** std\_logic **:=** '0'**;**

**signal** Y **:** std\_logic **:=** '0'**;**

**signal** Cin **:** std\_logic **:=** '0'**;**

--Outputs

**signal** Sum **:** std\_logic**;**

**signal** Cout **:** std\_logic**;**

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** full\_adder **PORT** **MAP** **(**

X **=>** X**,**

Y **=>** Y**,**

Cin **=>** Cin**,**

Sum **=>** Sum**,**

Cout **=>** Cout

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

Cin **<=** '0'**;**

Y **<=** '0'**;**

X **<=** '0'**;**

**wait** **for** 10ns**;**

Cin **<=** '1'**;**

Y **<=** '0'**;**

X **<=** '0'**;**

**wait** **for** 10ns**;**

Cin **<=** '0'**;**

Y **<=** '1'**;**

X **<=** '0'**;**

**wait** **for** 10ns**;**

Cin **<=** '1'**;**

Y **<=** '1'**;**

X **<=** '0'**;**

**wait** **for** 10ns**;**

Cin **<=** '0'**;**

Y **<=** '0'**;**

X **<=** '1'**;**

**wait** **for** 10ns**;**

Cin **<=** '1'**;**

Y **<=** '0'**;**

X **<=** '1'**;**

**wait** **for** 10ns**;**

Cin **<=** '0'**;**

Y **<=** '1'**;**

X **<=** '1'**;**

**wait** **for** 10ns**;**

Cin **<=** '1'**;**

Y **<=** '1'**;**

X **<=** '1'**;**

**wait** **for** 10ns**;**

**end** **process;**

**END;**

## Ripple Adder 16 bit:

Adds two positive numbers and then shows the operation of the overflow flag by adding two, two’s complement negative numbers to get a positive number.

**ENTITY** ripple\_adder\_16bit\_tb **IS**

**END** ripple\_adder\_16bit\_tb**;**

**ARCHITECTURE** behavior **OF** ripple\_adder\_16bit\_tb **IS**

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

**COMPONENT** ripple\_adder\_16bit

**PORT(**

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

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

Cin **:** **IN** std\_logic**;**

V **:** **OUT** std\_logic**;**

Cout **:** **OUT** std\_logic**;**

G **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

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

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

**signal** Cin **:** std\_logic **:=** '0'**;**

--Outputs

**signal** V **:** std\_logic**;**

**signal** Cout **:** std\_logic**;**

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

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** ripple\_adder\_16bit **PORT** **MAP** **(**

A **=>** A**,**

B **=>** B**,**

Cin **=>** Cin**,**

V **=>** V**,**

Cout **=>** Cout**,**

G **=>** G

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

A **<=** x"0001"**;**

B **<=** x"0001"**;**

Cin **<=** '0'**;**

**wait** **for** 10ns**;**

A **<=** x"8001"**;**

B **<=** x"8001"**;**

Cin **<=** '0'**;**

**wait** **for** 10ns**;**

**end** **process;**

**END;**

## B Input Logic:

Cycles through all the binary combinations of the select line.

**ENTITY** b\_input\_logic\_tb **IS**

**END** b\_input\_logic\_tb**;**

**ARCHITECTURE** behavior **OF** b\_input\_logic\_tb **IS**

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

**COMPONENT** b\_input\_logic

**PORT(**

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

S **:** **IN** std\_logic\_vector**(**1 **downto** 0**);**

Y **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

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

**signal** S **:** std\_logic\_vector**(**1 **downto** 0**)** **:=** **(others** **=>** '0'**);**

--Outputs

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

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** b\_input\_logic **PORT** **MAP** **(**

B **=>** B**,**

S **=>** S**,**

Y **=>** Y

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

B **<=** x"AAAA"**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

**end** **process;**

**END;**

## Arithmetic Unit:

Cycles through all of the arithmetic operations that can be performed on the two operands A and B.

**ENTITY** arithmetic\_unit\_tb **IS**

**END** arithmetic\_unit\_tb**;**

**ARCHITECTURE** behavior **OF** arithmetic\_unit\_tb **IS**

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

**COMPONENT** arithmetic\_unit

**PORT(**

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

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

Cin **:** **IN** std\_logic**;**

SelB **:** **IN** std\_logic\_vector**(**1 **downto** 0**);**

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

Cout **:** **OUT** std\_logic**;**

V **:** **OUT** std\_logic

**);**

**END** **COMPONENT;**

--Inputs

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

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

**signal** Cin **:** std\_logic **:=** '0'**;**

**signal** SelB **:** std\_logic\_vector**(**1 **downto** 0**)** **:=** **(others** **=>** '0'**);**

--Outputs

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

**signal** Cout **:** std\_logic**;**

**signal** V **:** std\_logic**;**

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** arithmetic\_unit **PORT** **MAP** **(**

A **=>** A**,**

B **=>** B**,**

Cin **=>** Cin**,**

SelB **=>** SelB**,**

G **=>** G**,**

Cout **=>** Cout**,**

V **=>** V

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

A **<=** x"0002"**;**

B **<=** x"0001"**;**

**wait** **for** 10ns**;**

-- G = A (Transfer)

SelB**(**0**)** **<=** '0'**;**

SelB**(**1**)** **<=** '0'**;**

Cin **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A + 1 (Increment)

SelB**(**0**)** **<=** '0'**;**

SelB**(**1**)** **<=** '0'**;**

Cin **<=** '1'**;**

**wait** **for** 100ns**;**

-- G = A + B (Add)

SelB**(**0**)** **<=** '1'**;**

SelB**(**1**)** **<=** '0'**;**

Cin **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A + B + 1

SelB**(**0**)** **<=** '1'**;**

SelB**(**1**)** **<=** '0'**;**

Cin **<=** '1'**;**

**wait** **for** 100ns**;**

-- G = A + notB

SelB**(**0**)** **<=** '0'**;**

SelB**(**1**)** **<=** '1'**;**

Cin **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A + notB + 1 (Subtract)

SelB**(**0**)** **<=** '0'**;**

SelB**(**1**)** **<=** '1'**;**

Cin **<=** '1'**;**

**wait** **for** 100ns**;**

-- G = A - 1 (Decrement)

SelB**(**0**)** **<=** '1'**;**

SelB**(**1**)** **<=** '1'**;**

Cin **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A (Transfer)

SelB**(**0**)** **<=** '1'**;**

SelB**(**1**)** **<=** '1'**;**

Cin **<=** '1'**;**

**wait** **for** 100ns**;**

**end** **process;**

**END;**

## Logic Unit:

Performs the 4 operations AND, OR, XOR and NOT on the operands A and B.

**ENTITY** logic\_unit\_tb **IS**

**END** logic\_unit\_tb**;**

**ARCHITECTURE** behavior **OF** logic\_unit\_tb **IS**

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

**COMPONENT** logic\_unit

**PORT(**

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

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

S **:** **IN** std\_logic\_vector**(**1 **downto** 0**);**

G **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

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

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

**signal** S **:** std\_logic\_vector**(**1 **downto** 0**)** **:=** **(others** **=>** '0'**);**

--Outputs

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

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** logic\_unit **PORT** **MAP** **(**

A **=>** A**,**

B **=>** B**,**

S **=>** S**,**

G **=>** G

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

A **<=** x"1111"**;**

B **<=** x"0101"**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

**end** **process;**

**END;**

## ALU:

Performs all of the arithmetic and logical operations on operands A and B.

**ENTITY** alu\_16bit\_tb **IS**

**END** alu\_16bit\_tb**;**

**ARCHITECTURE** behavior **OF** alu\_16bit\_tb **IS**

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

**COMPONENT** alu\_16bit

**PORT(**

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

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

GSel **:** **IN** std\_logic\_vector**(**3 **downto** 0**);**

C **:** **OUT** std\_logic**;**

V **:** **OUT** std\_logic**;**

G **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

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

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

**signal** GSel **:** std\_logic\_vector**(**3 **downto** 0**)** **:=** **(others** **=>** '0'**);**

--Outputs

**signal** C **:** std\_logic**;**

**signal** V **:** std\_logic**;**

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

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** alu\_16bit **PORT** **MAP** **(**

A **=>** A**,**

B **=>** B**,**

GSel **=>** GSel**,**

C **=>** C**,**

V **=>** V**,**

G **=>** G

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

------Arithmetic Ops-----------------------------------------

-- G = A

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '0'**;**

GSel**(**1**)** **<=** '0'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A + 1

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '0'**;**

GSel**(**1**)** **<=** '0'**;**

GSel**(**0**)** **<=** '1'**;**

**wait** **for** 100ns**;**

-- G = A + B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '0'**;**

GSel**(**1**)** **<=** '1'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A + B + 1

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '0'**;**

GSel**(**1**)** **<=** '1'**;**

GSel**(**0**)** **<=** '1'**;**

**wait** **for** 100ns**;**

-- G = A + notB

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '1'**;**

GSel**(**1**)** **<=** '0'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A - B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '1'**;**

GSel**(**1**)** **<=** '0'**;**

GSel**(**0**)** **<=** '1'**;**

**wait** **for** 100ns**;**

-- G = A - 1

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '1'**;**

GSel**(**1**)** **<=** '1'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '0'**;**

GSel**(**2**)** **<=** '1'**;**

GSel**(**1**)** **<=** '1'**;**

GSel**(**0**)** **<=** '1'**;**

**wait** **for** 100ns**;**

------Logical Ops--------------------------------------------

-- G = A and B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '1'**;**

GSel**(**2**)** **<=** '0'**;** -- S1 is don't care for all logical ops

GSel**(**1**)** **<=** '0'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A or B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '1'**;**

GSel**(**2**)** **<=** '0'**;**

GSel**(**1**)** **<=** '1'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = A xor B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '1'**;**

GSel**(**2**)** **<=** '1'**;**

GSel**(**1**)** **<=** '0'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

-- G = notA

A **<=** x"0001"**;**

B **<=** x"0002"**;**

GSel**(**3**)** **<=** '1'**;**

GSel**(**2**)** **<=** '1'**;**

GSel**(**1**)** **<=** '1'**;**

GSel**(**0**)** **<=** '0'**;**

**wait** **for** 100ns**;**

**end** **process;**

**END;**

## Shifter:

Performs a pass-through, left shift and right shift on B with also a demonstration of the IR and IL inputs being shifted into B.

**ENTITY** shifter\_16bit\_tb **IS**

**END** shifter\_16bit\_tb**;**

**ARCHITECTURE** behavior **OF** shifter\_16bit\_tb **IS**

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

**COMPONENT** shifter\_16bit

**PORT(**

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

S **:** **IN** std\_logic\_vector**(**1 **downto** 0**);**

IR **:** **IN** std\_logic**;**

IL **:** **IN** std\_logic**;**

H **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

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

**signal** S **:** std\_logic\_vector**(**1 **downto** 0**)** **:=** **(others** **=>** '0'**);**

**signal** IR **:** std\_logic **:=** '0'**;**

**signal** IL **:** std\_logic **:=** '0'**;**

--Outputs

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

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** shifter\_16bit **PORT** **MAP** **(**

B **=>** B**,**

S **=>** S**,**

IR **=>** IR**,**

IL **=>** IL**,**

H **=>** H

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

-- B pass through

B **<=** x"0001"**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

-- B right shift

B **<=** x"0002"**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '0'**;**

**wait** **for** 10ns**;**

-- B left shift

B **<=** x"0002"**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '1'**;**

**wait** **for** 10ns**;**

-- B right shift with input on IR

B **<=** x"0002"**;**

S**(**0**)** **<=** '1'**;**

S**(**1**)** **<=** '0'**;**

IL **<=** '0'**;**

IR **<=** '1'**;**

**wait** **for** 10ns**;**

-- B left shift with input on IL

B **<=** x"0002"**;**

S**(**0**)** **<=** '0'**;**

S**(**1**)** **<=** '1'**;**

IL **<=** '1'**;**

IR **<=** '0'**;**

**wait** **for** 10ns**;**

IL **<=** '0'**;**

IR **<=** '0'**;**

**end** **process;**

**END;**

## Function Unit:

Performs the arithmetic and logical operations as the previous test benches have with the addition of updating the Carry, Overflow, Zero and Negative control flags.

**ENTITY** function\_unit\_tb **IS**

**END** function\_unit\_tb**;**

**ARCHITECTURE** behavior **OF** function\_unit\_tb **IS**

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

**COMPONENT** function\_unit

**PORT(**

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

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

FSel **:** **IN** std\_logic\_vector**(**4 **downto** 0**);**

C **:** **OUT** std\_logic**;**

V **:** **OUT** std\_logic**;**

N **:** **OUT** std\_logic**;**

Z **:** **OUT** std\_logic**;**

F **:** **OUT** std\_logic\_vector**(**15 **downto** 0**)**

**);**

**END** **COMPONENT;**

--Inputs

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

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

**signal** FSel **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** **(others** **=>** '0'**);**

--Outputs

**signal** C **:** std\_logic**;**

**signal** V **:** std\_logic**;**

**signal** N **:** std\_logic**;**

**signal** Z **:** std\_logic**;**

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

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** function\_unit **PORT** **MAP** **(**

A **=>** A**,**

B **=>** B**,**

FSel **=>** FSel**,**

C **=>** C**,**

V **=>** V**,**

N **=>** N**,**

Z **=>** Z**,**

F **=>** F

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

-- TRANSFER A

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00000"**;**

**wait** **for** 100ns**;** -- F = 0000 0001

-- INCREMENT

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00001"**;**

**wait** **for** 100ns**;** -- F = 0000 0010

-- ADD

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00010"**;**

**wait** **for** 100ns**;** -- F = 0000 0011

-- ADD WITH CARRY

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00011"**;**

**wait** **for** 100ns**;** -- F = 0000 0100

-- ADD WITH NOT B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00100"**;**

**wait** **for** 100ns**;** -- F = 1111 1110

-- SUBTRACT

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00101"**;**

**wait** **for** 100ns**;** -- F = 1111 1111

-- DECREMENT

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00110"**;**

**wait** **for** 100ns**;** -- F = 0000 0000

-- TRANSFER A

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "00111"**;**

**wait** **for** 100ns**;** -- F = 0000 0001

-- AND

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "01000"**;**

**wait** **for** 100ns**;** -- F = 0000 0000

-- OR

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "01010"**;**

**wait** **for** 100ns**;** -- F = 0000 0011

-- XOR

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "01100"**;**

**wait** **for** 100ns**;** -- F = 0000 0011

-- NOT

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "01110"**;**

**wait** **for** 100ns**;** -- F = 1111 1110

-- TRANSFER B

A **<=** x"0001"**;**

B **<=** x"0002"**;**

FSel **<=** "10000"**;**

**wait** **for** 100ns**;** -- F = 0000 0010

-- SHIFT RIGHT (also displays carry)

A **<=** x"0001"**;**

B **<=** x"F002"**;**

FSel **<=** "10100"**;**

**wait** **for** 100ns**;** -- F = 0000 0001

-- SHIFT LEFT (also displays carry)

A **<=** x"0001"**;**

B **<=** x"F002"**;**

FSel **<=** "11000"**;**

**wait** **for** 100ns**;** -- F = 0000 0100

**end** **process;**

**END;**

## Datapath:

Shows the use of the Control Word to perform tow load operations on R0 and R1, and an arithmetic addition between R0 and R1 into R2. We can use the ADDR\_OUT to view R2.

**ENTITY** datapath\_16bit\_tb **IS**

**END** datapath\_16bit\_tb**;**

**ARCHITECTURE** behavior **OF** datapath\_16bit\_tb **IS**

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

**COMPONENT** datapath\_16bit

**PORT(**

CONTROL\_WORD **:** **IN** std\_logic\_vector**(**16 **downto** 0**);**

CONST\_IN **:** **IN** std\_logic\_vector**(**15 **downto** 0**);**

DATA\_IN **:** **IN** std\_logic\_vector**(**15 **downto** 0**);**

CLK **:** **IN** std\_logic**;**

ADDR\_OUT **:** **OUT** std\_logic\_vector**(**15 **downto** 0**);**

DATA\_OUT **:** **OUT** std\_logic\_vector**(**15 **downto** 0**);**

C **:** **OUT** std\_logic**;**

V **:** **OUT** std\_logic**;**

N **:** **OUT** std\_logic**;**

Z **:** **OUT** std\_logic

**);**

**END** **COMPONENT;**

--Inputs

**signal** CONTROL\_WORD **:** std\_logic\_vector**(**16 **downto** 0**)** **:=** **(others** **=>** '0'**);**

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

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

**signal** CLK **:** std\_logic **:=** '0'**;**

--Outputs

**signal** ADDR\_OUT **:** std\_logic\_vector**(**15 **downto** 0**);**

**signal** DATA\_OUT **:** std\_logic\_vector**(**15 **downto** 0**);**

**signal** C **:** std\_logic**;**

**signal** V **:** std\_logic**;**

**signal** N **:** std\_logic**;**

**signal** Z **:** std\_logic**;**

-- Clock period definitions

**constant** CLK\_period **:** time **:=** 10 ns**;**

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

uut**:** datapath\_16bit **PORT** **MAP** **(**

CONTROL\_WORD **=>** CONTROL\_WORD**,**

CONST\_IN **=>** CONST\_IN**,**

DATA\_IN **=>** DATA\_IN**,**

CLK **=>** CLK**,**

ADDR\_OUT **=>** ADDR\_OUT**,**

DATA\_OUT **=>** DATA\_OUT**,**

C **=>** C**,**

V **=>** V**,**

N **=>** N**,**

Z **=>** Z

**);**

-- 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**

-- R0 = 0x0001

CONTROL\_WORD **<=** "00000000010000011"**;**

DATA\_IN **<=** x"0001"**;**

**wait** **for** 20ns**;**

-- R1 = 0x0002

CONTROL\_WORD **<=** "00100100010000011"**;**

DATA\_IN **<=** x"0002"**;**

**wait** **for** 20ns**;**

-- R2 = R0 + R1

CONTROL\_WORD **<=** "01000000100001001"**;**

**wait** **for** 20ns**;**

-- Observe R2 on ADDR\_OUT

CONTROL\_WORD **<=** "01001000000000001"**;**

**wait** **for** 40ns**;**

**end** **process;**

**END;**

# VHDL Test Bench Images:

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

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

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\mux3.pngMUX3 1 bit:

## 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\INetCache\Content.Word\regfile.pngRegister File:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\ripple adder.pngRipple Adder:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\b input logic.pngB Input Logic:

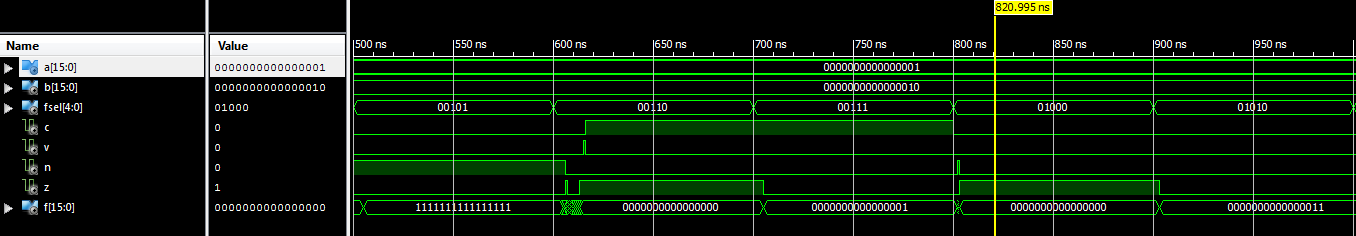
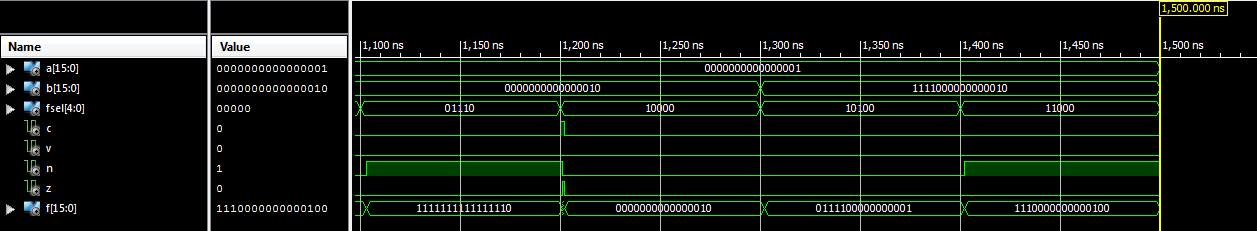
## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\arithmetic unit 2.pngC:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\arithmetic unit 1.pngArithmetic Unit:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\logic unit.pngLogic Unit:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\alu 3.pngC:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\alu 1.pngALU:C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\alu 2.png

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\shifter.pngShifter:

## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\function unit 1.pngFunction Unit:



## C:\Users\JackCassidy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\datapath.pngDatapath: