library ieee;

use ieee.std\_logic\_1164.all;

use work.all;

entity decoder is

port( instructions: in std\_logic\_vector( 15 downto 0); -- input from the instruction memory.

--Opperation Codes-----------------

opcode : out std\_logic\_vector(3 downto 0);

sel : out std\_logic\_vector(3 downto 0);

imm\_val : out std\_logic\_vector(7 downto 0); -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr: out std\_logic\_vector( 7 downto 0); --

---to decoder

rst : out std\_logic; -- reset for the Register bank

rdx\_en: out std\_logic; -- enables the read process for the Register bank

rdy\_en: out std\_logic; -- enables the read process for the Register bank

rdx : out std\_logic\_vector(3 downto 0); -- Rx output from Decoder to the register bank

rdy : out std\_logic\_vector(3 downto 0); -- Ry output from Decoder to the register bank

wr\_en : out std\_logic; -- enable for the right process to the Register bank

wrd : out std\_logic\_vector(3 downto 0); -- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux : out std\_logic;

sel\_rlsmux: out std\_logic;

sel\_wbmux : out std\_logic;

--PC counter lines

offset: out std\_logic\_vector(7 downto 0);

pcjump: out std\_logic\_vector(7 downto 0);

interupt\_line: out std\_logic

);

end decoder;

architecture logic of decoder is

begin

process (instructions)

begin

case (instructions(15 downto 12)) is

--============NO OPERATION============================================================================

when "0000" => null;

--============ADD IMMEDIATE===========================================================================

when "0001" => opcode <= instructions(15 downto 12);

sel <= instructions(11 downto 8);

imm\_val <= instructions(7 downto 0);-- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000";--Not being used so set to 0

---to decoder

rst <= '0'; -- reset for the Register bank

rdx\_en<= '1', '0' after 51 ps; -- enables the read process for the Register bank

rdy\_en<= '1', '0' after 51 ps; -- not used. set to zero

rdx <= instructions(11 downto 8); -- Rx output from Decoder to the register bank

rdy <= "0000"; -- not used. set to zero

wr\_en <= '0', '1' after 51 ps; --'0' after 125 ps;-- enable for the right process to the Register bank

wrd <= instructions(11 downto 8); -- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '1'; -- pushes immediate value through instead of RY value from Regbank

sel\_rlsmux<= '0'; -- not used. set to zero

sel\_wbmux <= '0'; -- pushes output value of ALU through for writeback '1', '0' after 75 ps;

--PC counter lines

offset <="00000000"; -- not used. set to zero

pcjump <="00000000"; -- not used. set to zero

interupt\_line <= '0';

--==============ADD OPERATION OR SUBTRACT==============================================================

when "0010" => opcode <= instructions(15 downto 12);

sel <= instructions(11 downto 8);

imm\_val <= "00000000";-- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000";--Not being used so set to 0

---to decoder

rst <= '0'; -- reset for the Register bank

rdx\_en<= '1', '0' after 25 ps; -- enables the read process for the Register bank

rdy\_en<= '1', '0' after 25 ps; -- enables the read process for the Register bank

rdx <= instructions(7 downto 4);-- Rx output from Decoder to the register bank

rdy <= instructions(3 downto 0);-- Ry output from Decoder to the register bank

wr\_en <= '0', '1' after 51 ps;

wrd <= instructions(7 downto 4); -- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; -- pushes ry value through instead of immediate value from Regbank

sel\_rlsmux<= '0'; -- not used. set to zero

sel\_wbmux <= '0'; -- pushes output value of ALU through for writeback

--PC counter lines

offset <= "00000000"; -- not used. set to zero

pcjump <= "00000000"; -- not used. set to zero

interupt\_line <= '0';

--==============INCREMENT OR DECREMENT===========================================================

when "0011" => opcode <= instructions(15 downto 12);

sel <= instructions(11 downto 8);

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000"; --

---to decoder

rst <= '0'; -- reset for the Register bank

rdx\_en<= '1', '0' after 51 ps; -- enables the read process for the Register bank

rdy\_en<= '0'; -- not used. set to zero and disabled

rdx <= instructions(7 downto 4);-- Rx output from Decoder to the register bank

rdy <= "0000"; -- not used. set to zero

wr\_en <= '0', '1' after 51 ps; -- enable for the right process to the Register bank

wrd <= instructions(7 downto 4);-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; --RY value not used for this operation.

sel\_rlsmux<= '0'; -- not used. set to zero

sel\_wbmux <= '0'; -- pushes output value of ALU through for writeback

--PC counter lines

offset <= "00000000"; -- not used. set to zero

pcjump <= "00000000"; -- not used. set to zero

interupt\_line <= '0';

--==============SHIFT LEFT OR RIGHT OPERATION===========================================================

when "0100" => opcode <= instructions(15 downto 12);

sel <= instructions(11 downto 8);

imm\_val <= "00000000";-- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000"; --Not being used so set to 0

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1', '0' after 51 ps;-- enables the read process for the Register bank

rdy\_en<= '1', '0' after 51 ps; -- enables the read process for the Register bank

rdx <= instructions(7 downto 4);-- Rx output from Decoder to the register bank

rdy <= instructions(3 downto 0);-- Ry output from Decoder to the register bank

wr\_en <= '0', '1' after 51 ps; -- enable for the right process to the Register bank

wrd <= instructions(7 downto 4);-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; -- pushes ry value through instead of immediate value from Regbank

sel\_rlsmux<= '0'; -- not used. set to zero

sel\_wbmux <= '0'; --selects the output from the ALU for writback

--PC counter lines

offset <= "00000000"; -- not used. set to zero

pcjump <= "00000000"; -- not used. set to zero

interupt\_line <= '0';

--==============Logical Operations===========================================================

when "0101" => opcode <= instructions(15 downto 12);

sel <= instructions(11 downto 8);

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000";

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1', '0' after 51 ps;-- enables the read process for the Register bank

rdy\_en<= '1', '0' after 51 ps;-- enables the read process for the Register bank

rdx <= instructions(7 downto 4);-- Rx output from Decoder to the register bank

rdy <= instructions(3 downto 0);-- Ry output from Decoder to the register bank

wr\_en <= '0', '1' after 51 ps; -- enable for the right process to the Register bank

wrd <= instructions(7 downto 4);-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; -- pushes ry value through instead of immediate value from Regbank

sel\_rlsmux<= '0'; -- not used. set to zero

sel\_wbmux <= '0'; -- selects the output from the ALU for writeback

--PC counter lines

offset <= "00000000"; -- not used set to zero

pcjump <= "00000000"; -- not used set to zero

interupt\_line <= '0';

-- --==============INTERUPTS===========================================================

-- when "0111" => opcode <= instructions(15 downto 12);

-- sel <= instructions(11 downto 8);

-- --imm\_val <= instructions(7 downto 0)-- immediate value being added to RX. Goes to RY Mux

--

-- --Load and Store line

-- --rl\_addr <= instructions(7 downto 0);--

--

-- ---to decoder

-- rst <= '0';-- reset for the Register bank

-- rdx\_en<= '1';-- enables the read process for the Register bank

-- rdy\_en<= '1';-- enables the read process for the Register bank

-- rdx <= instruction(7 downto 4);-- Rx output from Decoder to the register bank

-- rdy <= instruction(3 downto 0);-- Ry output from Decoder to the register bank

-- wr\_en <= '1';-- enable for the right process to the Register bank

-- wrd <= instruction(7 downto 4)-- Write address to the Register bank

--

-- -- Select line for the MUXs

-- sel\_rymux <= '0';

-- --sel\_rlsmux<=

-- sel\_wbmux <= '0';

--

-- --PC counter lines

-- --offset <=

-- --pcjump <=

--==============LOAD INDIRECT===========================================================

when "1000" => opcode <= instructions(15 downto 12);

sel <= "0000";

imm\_val <= "00000000";-- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= instructions(7 downto 0);

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1';-- enables the read process for the Register bank

rdy\_en<= '1';-- enables the read process for the Register bank

rdx <= instructions(7 downto 4);-- Rx output from Decoder to the register bank

rdy <= instructions(3 downto 0);-- Ry output from Decoder to the register bank

wr\_en <= '1'; -- enable for the right process to the Register bank

wrd <= instructions(7 downto 4);-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; -- RYMux not used for this operation.

sel\_rlsmux<= '1'; -- pushes RY value from regbank to Datamem

sel\_wbmux <= '1'; -- Pushes output from Datamem for writeback

--PC counter lines

offset <= "00000000"; -- not used set to zero

pcjump <= "00000000"; -- not used set to zero

interupt\_line <= '0';

--==============STORE INDIRECT===========================================================

when "1001" => opcode <= instructions(15 downto 12);

sel <= instructions(11 downto 8);

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= instructions(7 downto 0);--

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1';-- enables the read process for the Register bank

rdy\_en<= '1';-- enables the read process for the Register bank

rdx <= instructions(7 downto 4);-- Rx output from Decoder to the register bank

rdy <= instructions(3 downto 0);-- Ry output from Decoder to the register bank

wr\_en <= '1'; -- enable for the right process to the Register bank

wrd <= instructions(7 downto 4);-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; -- RYMux not used for this operation.

sel\_rlsmux<= '1'; -- pushes RY value from regbank to Datamem

sel\_wbmux <= '1'; -- Pushes output from Datamem for writeback

--PC counter lines

offset <= "00000000"; -- not used set to zero

pcjump <= "00000000"; -- not used set to zero

interupt\_line <= '0';

--==============LOAD REGISTER===========================================================

when "1010" => opcode <= instructions(15 downto 12);

sel <= "0000";

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= instructions(7 downto 0);--

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1';-- enables the read process for the Register bank

rdy\_en<= '0';-- not used. set to zero and disabled

rdx <= instructions(11 downto 8);-- Rx output from Decoder to the register bank

rdy <= "0000"; -- not used. set to zero

wr\_en <= '1'; -- enable for the right process to the Register bank

wrd <= instructions(11 downto 8);-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0'; -- Ry mux not used in this operation.

sel\_rlsmux<= '0'; -- pushes the Rl address from the decoder

sel\_wbmux <= '1'; -- Pushes output from Datamem for writeback

--PC counter lines

offset <= "00000000"; -- not used set to zero

pcjump <= "00000000"; -- not used set to zero

interupt\_line <= '0';

--==============STORE OPERATION===========================================================

when "1011" => opcode <= instructions(15 downto 12);

sel <= "0000";

imm\_val <= "00000000";-- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= instructions(7 downto 0);--

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1';-- enables the read process for the Register bank

rdy\_en<= '0';-- not used. set to zero and disabled

rdx <= instructions(11 downto 8);-- Rx output from Decoder to the register bank

rdy <= "0000"; -- not used. set to zero

wr\_en <= '0'; -- enable for the right process to the Register bank

wrd <= "0000"; --Writback not used so set to zero

-- Select line for the MUXs

sel\_rymux <= '0';

sel\_rlsmux<= '0';

sel\_wbmux <= '0';

--PC counter lines

offset <= "00000000"; -- not used set to zero

pcjump <= "00000000"; -- not used set to zero

interupt\_line <= '0';

--==============JUMP OPERATION===========================================================

when "1100" => opcode <= instructions(15 downto 12);

sel <= "0000";

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000"; --

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '0';-- enables the read process for the Register bank

rdy\_en<= '0';-- enables the read process for the Register bank

rdx <= "0000"; -- Rx output from Decoder to the register bank

rdy <= "0000";-- Ry output from Decoder to the register bank

wr\_en <= '0'; -- enable for the right process to the Register bank

wrd <= "0000";-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0';

sel\_rlsmux<= '0';

sel\_wbmux <= '0';

--PC counter lines

offset <= "00000000"; -- not used. set to zero

pcjump <= instructions(7 downto 0); --"00100100",

interupt\_line <= '0';

--==============BRANCH IF ZERO OPERTATION===========================================================

when "1101" => opcode <= instructions(15 downto 12);

sel <= "0000";

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000";

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1', '0' after 51 ps;-- enables the read process for the Register bank

rdy\_en<= '0';-- enables the read process for the Register bank

rdx <= instructions(11 downto 8);-- Rx output from Decoder to the register bank

rdy <= "0000";-- Ry output from Decoder to the register bank

wr\_en <= '0', '1' after 51 ps; -- enable for the right process to the Register bank

wrd <= "0000";-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0';

sel\_rlsmux<= '0';

sel\_wbmux <= '0';

--PC counter lines

offset <= instructions(7 downto 0) AFTER 51 ps;

pcjump <= "00000000"; -- not used. set to zero

interupt\_line <= '0';

--==============BRANCH NOT ZERO===========================================================

when "1110" => opcode <= instructions(15 downto 12);

sel <= "0000";

imm\_val <= "00000000"; -- immediate value being added to RX. Goes to RY Mux

--Load and Store line

rl\_addr <= "00000000";

---to decoder

rst <= '0';-- reset for the Register bank

rdx\_en<= '1', '0' after 51 ps;-- enables the read process for the Register bank

rdy\_en<= '0';-- enables the read process for the Register bank

rdx <= instructions(11 downto 8);-- Rx output from Decoder to the register bank

rdy <= "0000";-- Ry output from Decoder to the register bank

wr\_en <= '0', '1' after 51 ps; -- enable for the right process to the Register bank

wrd <= "0000";-- Write address to the Register bank

-- Select line for the MUXs

sel\_rymux <= '0';

sel\_rlsmux<= '0';

sel\_wbmux <= '0';

--PC counter lines

offset <= instructions(7 downto 0);

pcjump <= "00000000"; -- not used. set to zero

interupt\_line <= '0';

-- --==============RETURN FROM INTERUPT===========================================================

-- when "1111" => opcode <= instructions(15 downto 12);

-- sel <= instructions(11 downto 8);

-- imm\_val <= instructions(7 downto 0)-- immediate value being added to RX. Goes to RY Mux

--

-- --Load and Store line

-- rl\_addr <= instructions(7 downto 0)--

--

-- ---to decoder

-- rst <= '0';-- reset for the Register bank

-- rdx\_en<= -- enables the read process for the Register bank

-- rdy\_en<= -- enables the read process for the Register bank

-- rdx <= -- Rx output from Decoder to the register bank

-- rdy <= -- Ry output from Decoder to the register bank

-- wr\_en <= -- enable for the right process to the Register bank

-- wrd <= -- Write address to the Register bank

--

-- -- Select line for the MUXs

-- sel\_rymux <=

-- sel\_rlsmux<=

-- sel\_wbmux <=

--

-- --PC counter lines

-- offset <=

-- pcjump <=

when others => NULL;

end case;

end process;

end logic;