library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity imem is

port( inst\_addr : in std\_logic\_vector(7 downto 0); -- input from PC

clk : in std\_logic;

instruction : out std\_logic\_vector(15 downto 0) -- output to go to decoder

);

end imem;

architecture behv of imem is

begin

process (clk)

type memory\_bank is array ( 255 downto 0) of STD\_LOGIC\_VECTOR (15 downto 0);

variable IMEM: memory\_bank;

begin

-- created instructions to be sent to the decoder

if clk'event and clk='1' then

IMEM(0):="0000000000000000";

IMEM(1):="0001000000010001"; --17

IMEM(2):="0001000100100010"; -- 34

IMEM(3):="0001001000110011"; --51

IMEM(4):="0001001101000100"; --68

IMEM(5):="0001010001010101";--85

IMEM(6):="0001010101100110";--102

IMEM(7):="0001011001110111";--119

IMEM(8):="0001011110001000";--136

IMEM(9):="0001100010011001"; --153

IMEM(10):="0001100110101010"; --170

IMEM(11):="0001101010111011"; --187

IMEM(12):="0001101111001100"; --204

IMEM(13):="0001110011011101";--221

IMEM(14):="0001110111011110"; --238

IMEM(15):="0001111011111111"; --255

IMEM(16):="0001111100000000"; --0

IMEM(17):="0010000000100000"; --Add

IMEM(18):="0010000100110010"; --subtract

IMEM(19):="0011000000100000"; --Increment

IMEM(20):="0011000101000000"; --Decrement

IMEM(21):="0100000000000001"; --Shift Left

IMEM(22):="0100000100110001"; --Shift Right

IMEM(23):="0101000000000000"; --Logical NOT

IMEM(24):="0101000100000001"; -- Logical NOR

IMEM(25):="0101001000010000";--Logical NAND

IMEM(26):="0101001100000011";--Logical XOR

IMEM(27):="0101010000110001";--Logical AND

IMEM(28):="0101010100010000";--Logical OR

IMEM(29):="0101011000000000";--Clear

IMEM(30):="0101011100010000";--Set

IMEM(31):="0101111100110001";--Set if less then

IMEM(32):="0101100000000000";--Move

IMEM(33):="1000000011111111"; -- load indirect

IMEM(34):="1001000000000000"; -- store indirect

IMEM(35):="1010000111111111"; -- load register

IMEM(36):="1011000000000000"; -- store register

IMEM(37):="1100000001000000";--Jump

IMEM(64):="0010000000010001";--add op

IMEM(65):="1101000000000010"; --branch if zero

IMEM(66):="0000000000000000"; --NoOP

IMEM(67):="1110000000000010"; -- branch if not zero

-- IMEM(68):="0000000000000000";

IMEM(68):="1100000000000001"; --jump

end if;

instruction <= IMEM(CONV\_INTEGER(inst\_addr));

end process;

end behv;