UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

Implement Embedded Controller using FPGA Chip

Dr. Haresh Pandya[1], Mr. Mahesh Rangapariya[2], Mr. Jitendra Rajput[3]

Professor & Head, Department of Electronics, Saurashtra University, Rajakot, Gujrat, India[1]

Ph.D. Student & Visiting Lecturer, Department of Electronics, Saurashtra University, Rajakot, Gujarat, India[2]

Assistant Professor of Electronics at SSR College of Science, (Affiliated to Pune University), Silvassa, D & NH, India[3]

Abstract: The designer of an FPGA embedded processor system has complete flexibility to select any combination of peripherals and controllers. In fact, the designer can invent new, unique peripherals that can be connected directly to the processor bus. If a designer has a non-standard requirement for a peripheral set, this can be met easily with an FPGA embedded processor system. For example, a designer would not easily find an off-the-shelf processor with ten UARTs. However, in an FPGA, this configuration is very easily accomplished.

Keyword: Embedded Controller, VHDL (VHSIC Hardware Description Language), FPGA (Field Programmable Gate Arrays), Electronics Switch.

I. INTRODUCTION

FPGAs are increasingly used to implement embedded digital systems because of their low time-to-market and low costs compared to integrated circuit design, as well as their superior performance and area over a general purpose microprocessor. However, the hardware design necessary to achieve this superior performance and area is very difficult to perform causing long design times and preventing wide-spread adoption of FPGA technology. The amount of hardware design can be reduced by employing a microprocessor for less-critical computation in the system. Often this microprocessor is implemented using the FPGA reprogrammable fabric as a soft processor which can preserve the benefits of a single-chip FPGA solution without specializing the device with dedicated hard processors. Current soft processors have simple architectures that provide performance adequate for only the least-critical computations. Embedding a processor inside an FPGA has many advantages. Specific peripherals can be chosen based on the application, with unique user-designed peripherals being easily attached. A variety of memory controllers enhance the FPGA embedded processor system. interface capabilities. FPGA embedded processors use general-purpose FPGA logic to construct internal memory, processor busses, internal peripherals, and external peripheral controllers (including external memory controllers). Soft processors are built from general-purpose FPGA logic as well. As more pieces (busses, memory, memory controllers, peripherals, and peripheral controllers) are added to the embedded processor system, the system becomes increasingly more powerful and useful. However, these additions reduce performance and increase the embedded system cost, consuming FPGA resources. Likewise, large banks of external memory can be connected to the FPGA and accessed by the embedded processor system using included memory controllers. Unfortunately, the latency to access this external memory can have a significant, negative impact on performance.

FPGA manufacturers often publish embedded processor performance benchmarks. Like all other companies running benchmarks, these FPGA manufacturers purposely construct and use an FPGA embedded processor system that performs the best for each specific benchmark. This means that the FPGA processor system constructed to run the benchmark has very few peripherals and runs exclusively using internal memory. The embedded designer must understand how a real-world designs. Specific peripheral set and memory architecture will affect the system performance. However, no easy formula or chart exists showing how to compare the performance and cost for different memory strategies and peripheral sets.

Advantages of an FPGA embedded processor

An FPGA embedded processor system offers many exceptional advantages compared to typical microprocessors including:

1) Customization
2) Obsolescence mitigation
3) Component and cost reduction
4) Hardware acceleration

Hardware acceleration

Perhaps the most compelling reason to choose an FPGA embedded processor is the ability to make tradeoffs between hardware and software to maximize efficiency and performance. If an algorithm is identified as a software bottleneck, a

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

custom co-processing engine can be designed in the FPGA specifically for that algorithm. This co-processor can be attached to the FPGA embedded processor through special, low-latency channels, and custom instructions can be defined to exercise the co-processor. With modern FPGA hardware design tools, transitioning software bottlenecks from software to hardware is much easier since the software C code can be readily adapted into hardware with only minor changes to the C code.

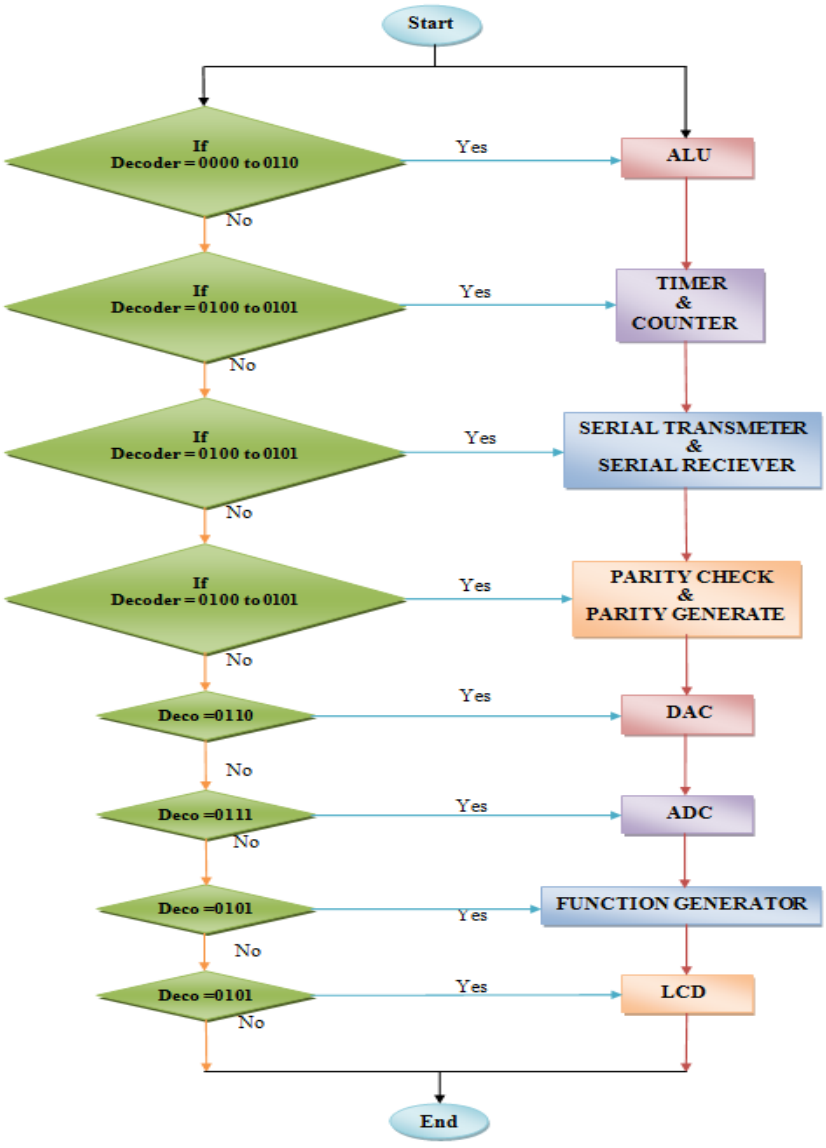
Component and cost reduction

With the versatility of the FPGA, previous systems that required multiple components can be replaced with a single FPGA. Certainly this is the case when an auxiliary I/O chip or a co-processor is required next to an off-the-shelf processor. By reducing the component count in a design, a company can reduce board size and inventory management, both of which will save design time and cost.

Obsolescence mitigation

Some companies, in particular those supporting military contracts, have a design requirement to ensure a product lifespan that is much longer than the lifespan of a standard electronics product. Component obsolescence mitigation is a difficult issue. FPGA soft-processors are an excellent solution in this case since the source HDL for the soft-processor can be purchased Ownership of the processor. VHDL code may fulfill the requirement for product lifespan guarantee.

II. FLOW CHART OF FPGA EMBEDDED SYSTEM



UGC Approved Journal

IARJSET

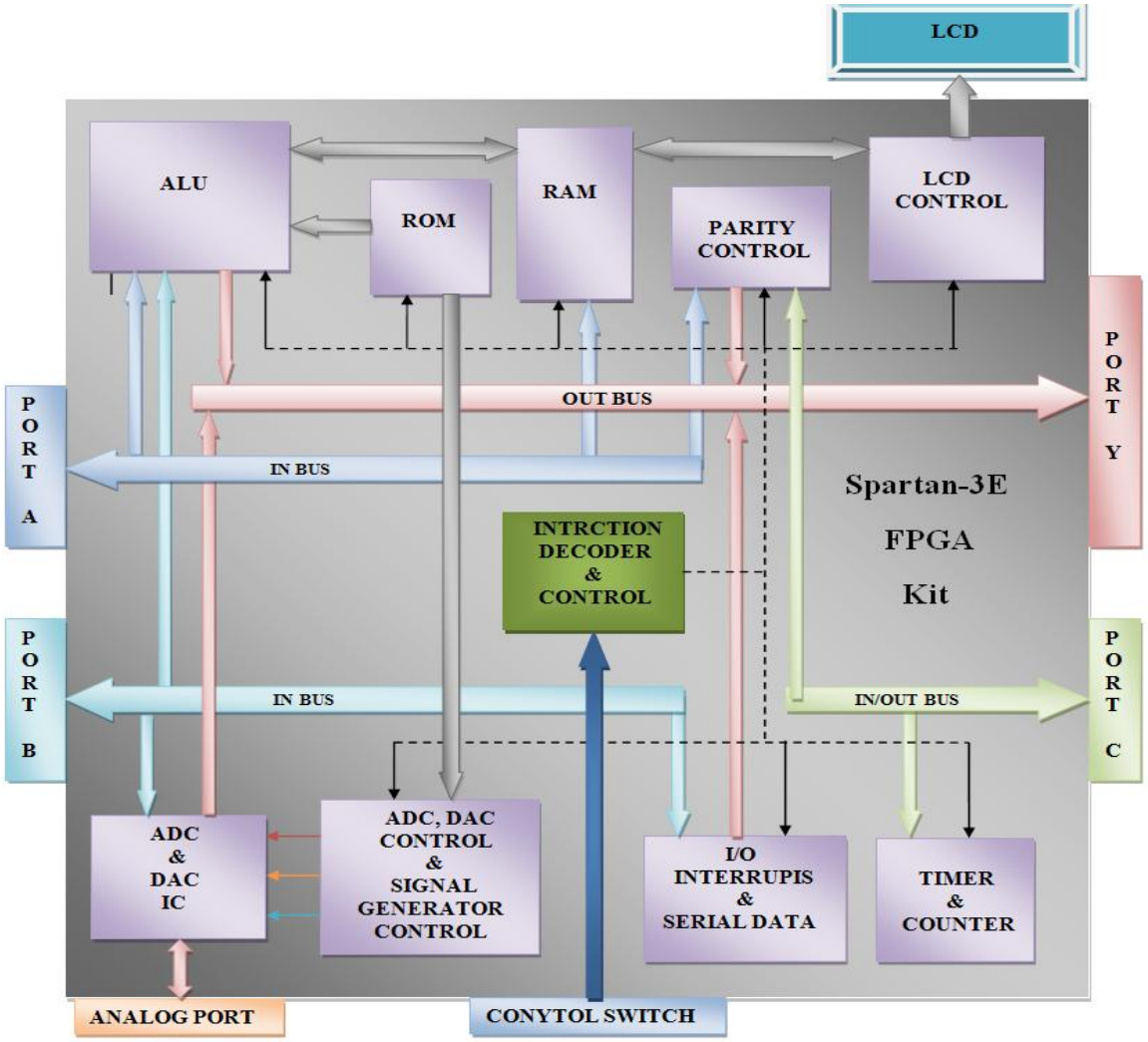
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

III. BLOCK DIAGRAM OF FPGA EMBEDDED SYSTEM



Block Diagram description

The microcontroller has a following main block. Like ALU, Control Unit, Instruction Decoder, in built RAM/ROM, I/O Port, Data bus/Control bus, Timer, Counter, Serial Transmitter/Receiver and Interrupt control. Hardware of the entire block is fix and limited so, Working of block is specific. For Embedded application external ADC, DAC connect with microcontrollers therefore, district Sensor assembly and ADC/DAC externally connection require.

Now a day's FPGA base embedded board is available. Here Xilinx Spartan 3E FPGA board use for design of embedded controller. Xilinx kit has a following facility like, on board two channels serial ADC, four channels serial DAC, 16X2 LCD, 8-Switch and 8-LED. ALU, Control Unit, Instruction Decoder, in built RAM/ ROM, I/O Port, Data bus/Control bus, Timer, Counter, Serial Transmitter/Receiver and Interrupt control block make inside the FPGA using VHDL program. Above all block are reprogrammable so, functionality of all block modify using VHDL program.

Using VHDL program 4 bit/8 bit/16 bit/32 bit/64 bit/128 bit……ALU easily create in side FPGA logic cell. Function of control unit is totally set using VHDL task. Design and modification of instruction set depends VHDL program so, we create over personal mnemonics. Four data bus create they are connected to port A, port B, port C and port Y. All the bus is internally connected to each block of embedded controller.

ALU

ALU give three arithmetic operations and five logical operations. Arithmetic operations are addition, subtraction and multiplication. Logic operations are NOT, OR, AND, EXOR and EXNOR. Here 4-Bit ALU block is design using VHDL program so, operation and bit width of ALU is change using program.

MEMORY

Memory of embedded controller is design inside FPGA chip. RAM/ROM create using VHDL program so, memory size and bit width of both is change using program. VHDL has a three data object and four data type. Data object are

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

variable, signal, constant and data type are single, double, vector and array. Using constant, vector are array make ROM. Using signal, vector and array make a RAM. VHDL support integer data type so, memory address is decimals number. Using decimal address method, memory location is easily access. VHDL base memory single pin use for read and write operation.

LCD CONTROL

Control and DATA signal are use to create character on display. Three main handshake signals are as follow EN, RS, RW and DATA. LCD has a two operation mode one initialization and second display. LCD control block has an initialization and operation mode VHDL program. LCD initialization and operation program store in LCD ROM they execute location by location. First execute initialization program after that execute operation program character bit. As per the applying of character bit character is display on screen.

TIMER/COUNTER

Normally, microcontroller in built timer works as timer or counter. Really, fact that both are not operating simultaneously. In FPGA kit up/down timer and up/down counter is create using VHDL program. FPGA base two Timers and two counters are creating and they are increment/decrement type. All timer and counter work independence simultaneously. VHDL signal object and arithmetic operator is use to make timer and counter. Timer and counter has a control bit they also use to generate soft interrupt. FPGA embedded microcontroller Port C use for timer and counter.

SERIAL TRANSMETER&RECEIVER

Microcontroller 8051 has an only one in built serial transmitter and receiver. FPGA base microcontroller more than one serial transmitter/receiver designed and all work indecently to each author. Serial transmitter consists of parallel to serial conversion program and serial receiver consists of serial to parallel conversion program. Serial clock, Information data and Protocol bit are create using VHDL program so, change using program. Serial transmitter/receiver has a control bit they also use to generate soft interrupt. FPGA embedded microcontroller Port C use for serial transmitter and receiver.

INTERRUPT CONTROL

In FPGA embedded microcontroller Port C use for hardware interrupt. Port C two pin use for timer/counter, two pin use for serial communication and remaining four pin use for hardware interrupt. Priority of all interrupt set using VHDL program and change it, if require. Hardware and software both interrupt work independence simultaneously. For hardware interrupt level sensitive and positive/negative edge (trigger) sensitive both are create. Software control like, timer interrupt, serial communication interrupt are also control using this block.

ADC/DAC CONTROL

ADC/DAC both required handshaking signal to control all operation. Control sequence of ADC and DAC are different so, control block internally divided into two parts. ADC control part generate ADC control signal and DAC control part generate DAC control signal. ADC require five control signal as like, spi_mosi, amp_cs, spi_sck, amp_shdn and ad_conv. DAC also require five control signals like, spi_mosi, dac_cs, spi_sck, dac_clr and spi_miso. Above all signal is produce by using VHDL program with proper sequence. ADC and DAC both are work in serial input mode so, some time input data also allaying using VHDL program.

SIGNAL GENERATOR CONTROL

FPGA base signal generator consists of VHDL ROM and DAC. Different weighted digital data permanent store inside VHDL ROM. Address control program continue and sequentially generate different location of memory. According to generated address, memory location is select and DAC continue read digital data from ROM. DAC produce different analog output according to input digital data. Value of analog output is very with different time function.

IV. VHDL PROGRAMM FOR FPGA EMBEDDED SYSTEM

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
-- Create Date:    19:20:02 01/20/2008
-- Design Name:
-- Module Name:    ECP - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
-- Dependencies:
-- Revision:
-- Revision 0.01 - File Created
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
-- Additional Comments:
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

---- Uncomment the following library declaration if instantiating
---- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ECP is
GENERIC (n : INTEGER := 3 ; m : INTEGER := 2; r : INTEGER := 7 );
        Port ( rst,bclk,pro,exi,cnt_in,dec : in    STD_LOGIC;
          miso,shdn,ampad,ampdo,mosi1 : in    STD_LOGIC;
                       RS,E,RW,clk,clk1,daccs,ampcs,adcon : inout STD_LOGIC;
                ssb,ceo,initb,last : out   STD_LOGIC;
           ashdn, mosi,A_Dclk,DAC_clr : out   STD_LOGIC;
                           a,b,c,alu_data : inout STD_LOGIC_VECTOR (r  downto 0);
                                  deco : inout STD_LOGIC_vector (4  downto 0);
                           digi_in : in    STD_LOGIC_vector (10 downto 0);
                                  y : out   STD_LOGIC_vector (15 downto 0);
                                         DB1 : inout STD_LOGIC_VECTOR (4  downto 0));
end ECP;
architecture Behavioral of sd is

    signal btemp        :  integer range 0 to 500000  :=0;
    signal btemp1       :  integer range 0 to 4800000:=0;
    signal add,count0:  integer range 0 to      31 :=0;

                                                                -----ALU------
                    signal tempa,tempb  : STD_LOGIC_VECTOR(r downto 0);
                    signal add_y,sub_y,not_y,or_y,and_y,xor_y,xnor_y: STD_LOGIC_vector(0 to r);
                    signal mlt_y            : STD_LOGIC_VECTOR(0 to 15);

                                                            ----TIMER/COUNTER----
                    signal ti_dwn_out,ti_up_out,cont_out : STD_LOGIC;
                    constant timer_up_mod  :integer range 0 to 10:=0;  ---mod--0=>50%---mod--1=>25%--mod--2=>10%--mod--3=>5%
                    constant timer_dwn_mod :integer range 0 to 5 :=2;  ---mod--0=>50%---mod--1=>25%--mod--2=>10%--mod--3=>5%
                signal timer0,ti_up_duty,timer1,ti_dwn_duty : integer range 0 to 100;
                    signal cnt_data,cnt1_data :   STD_LOGIC_VECTOR (4 downto 0);

                                                            ----SERIAL TO PARALLEL----
                    signal SE_cs2,SE_cs1,se_in,seri_out: STD_LOGIC ;
                    signal pa_data,pa_in : STD_LOGIC_VECTOR(r downto 0);
                    signal h1           : STD_LOGIC_VECTOR (0 to 9);
                signal h2           : STD_LOGIC_VECTOR (0 to 7);
                    signal temp1      : integer range 0 to 10:= 0;
                signal temp2      : integer range 0 to 7 := 0;
                    signal temp3      : integer range 0 to 12:= 0;
                signal temp4      : integer range 0 to 12:= 0;

                                                                ----PARITY-----
                    signal parity_bit  : STD_LOGIC;
                    signal parity                   : STD_LOGIC_VECTOR(r downto 0);
                    signal prt_data,a1: STD_LOGIC_VECTOR(r downto 0);

                                                                ----DAC-----
    signal DAC_SET,DAC_cs1,DAC_cs2: STD_LOGIC;
    signal h3               : STD_LOGIC_VECTOR (23 downto 0);
    signal h4               : STD_LOGIC_VECTOR (15 downto 0);
    signal DAC_in       : STD_LOGIC_VECTOR (15 downto 0);
    signal DACtemp1   : integer range 0 to 26:=0;
    signal DACtemp2   : integer range 0 to 23:=0;

                                                                ----ADC-----
    signal smiso,AMP_SET,ADC_cs1,ADC_cs2: STD_LOGIC;
    signal ADC_data   : STD_LOGIC_VECTOR(13 downto 0);
    signal h5             : STD_LOGIC_VECTOR (7 downto 0);
    signal h6             : STD_LOGIC_VECTOR (0 to 35);
    signal y1             : STD_LOGIC_VECTOR (0 to 15);
    signal ADCtemp1 : integer range 0 to 10:=0;
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
    signal ADCtemp2 : integer range 0 to 8:= 0;
    signal ADCtemp3 : integer range 0 to 34:=0;
    signal ADCtemp4 : integer range 0 to 32:=0;

                                                    ----FUNCTION GENERATOR------
    signal h7       : STD_LOGIC_VECTOR (23 downto 0);
    signal h8       : STD_LOGIC_VECTOR (15 downto 0);
    signal h9       : STD_LOGIC_VECTOR (15 downto 0);
    signal DAC_SET1,sel: STD_LOGIC;
    type rom_func is array(0 to 20) of STD_LOGIC_VECTOR (15 downto 0);
   signal mem2         : rom_func ;
   signal func         : integer range 0 to 20:=0;
  signal DACtemp3 : integer range 0 to 26:=0;
  signal DACtemp4 : integer range 0 to 23:=0;

                                                    ----LCD CONTROL------
    type rom_array   is array(0 to 8) of STD_LOGIC_VECTOR(4 downto 0);
    signal mem: rom_array ;
    type rom_array1 is array(0 to 25) of STD_LOGIC_VECTOR(4 downto 0);
    signal mem1: rom_array1 ;
    signal lcd1,lcd2 : integer range 0 to 9:= 0;
    signal lcd3      : integer range 0 to 8:= 0;
    signal lcd4      : integer range 0 to 25:=0;
begin

                                                    -------LCD ROM-------
----------higher first------then lower--------
 mem(0)<= "00000" ;
 mem(1)<= "00010" ;--funct
 mem(2)<= "01000" ;
 mem(3)<= "00000" ;--entry mod
 mem(4)<= "00110" ;
 mem(5)<= "00000" ;--display on
 mem(6)<= "01100" ;
 mem(7)<= "00000" ;--clear display
 mem(8)<= "00001" ;
 mem1(0)<= "00000" ;--clear display
 mem1(1)<= "00001" ;
 mem1(2)<= "01000" ;-- GIVE 1-LINE-ADDRESS
 mem1(3)<= "00110" ;
 mem1(4)<= "10100" ;  -- L
 mem1(5)<= "11100" ;
 mem1(6)<= "10100" ;  -- C
 mem1(7)<= "10011" ;
 mem1(8)<= "10100" ;  -- D
 mem1(9)<= "10100" ;
 mem1(10)<= "01100" ;-- GIVE 2-LINE-ADDRESS
 mem1(11)<= "00100" ;
 mem1(12)<= "10100" ; -- C
 mem1(13)<= "10011" ;
 mem1(14)<= "10100" ; -- O
 mem1(15)<= "11111" ;
 mem1(16)<= "10100" ; -- N
 mem1(17)<= "11110" ;
 mem1(18)<= "10101" ; -- T
 mem1(19)<= "10100" ;
 mem1(20)<= "10101" ; -- R
 mem1(21)<= "10010" ;
 mem1(22)<= "10100" ; -- O
 mem1(23)<= "11111" ;
 mem1(24)<= "10100" ; -- L
 mem1(25)<= "11100" ;

                                                    -------function generator RAM--------
 mem2(0) <= "0000000000000000" ;
 mem2(1) <= "0000001000000000" ;
 mem2(2) <= "0000000100000000" ;
 mem2(3) <= "0000000010000000" ;
 mem2(4) <= "0000000001000000" ;
 mem2(5) <= "0000000000100000" ;
 mem2(6) <= "0000000000010000" ;
 mem2(7) <= "0000000000001000" ;
 mem2(8) <= "0000000000000100" ;
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
 mem2(9) <= "0000000000000010" ;
 mem2(10)<= "0000000000000001" ;
 mem2(11)<= "0000000000000010" ;
 mem2(12)<= "0000000000000100" ;
 mem2(13)<= "0000000000001000" ;
 mem2(14)<= "0000000000010000" ;
 mem2(15)<= "0000000000100000" ;
 mem2(16)<= "0000000001000000" ;
 mem2(17)<= "0000000010000000" ;
 mem2(18)<= "0000000100000000" ;
 mem2(19)<= "0000001000000000" ;
 mem2(20)<= "0000010000000000" ;

 b<= "00001111"; a<= "01010101"; deco(4)<=  NOT dec ;
 tempa<= a; pa_in <= a; a1 <= a ; tempb<=b; se_in<= b(0); DAC_in <= a & b ;
  c(7)<= ti_up_out; c(6)<= ti_dwn_out; c(5)<= cont_out; c(4)<= seri_out; c(3)<= parity_bit; c(2)<=smiso;
                                                                                    ----mem----
--process(Clk,rst)
 -- begin
  -- if(rst='1')then
               -- add<=0;
               -- elsif(Clk'event and Clk = '0')then
                --  add <= add+1;
                        --if(add=7)then
                        -- add <=0;
                        -- end if;
                        --end if;
--addl<=conv_std_logic_vector(add,3);
 -- if(pro='1')then
               -- mem(add) <= da_in ;
    -- end if;
  --if(pro='0')then
                --deco <= mem(add) ;
                 --decol <= mem(add) ;
     --end if;
--end process;
-----board ckocl---50M--------
process(bclk)
                begin
                                        if(bclk'event and bclk = '0')then
                    btemp<= btemp+1;
                                                if(btemp>=250000)then
                                clk<='1';
                                                else
                                                    clk<='0'; end if; end if ;
              end process;
process(bclk)
                begin
                                        if(bclk'event and bclk = '0')then
                    btemp1<= btemp1+1;
                                                 if(btemp1>=2400000)then
                                clk1<='1';
                                                else
                                                    clk1<='0'; end if; end if ;
              end process;
process (clk,deco,clk1)----arith-----logic----
 begin
---adder---
  if(deco="00000")then
     add_y <= tempa + tempb ;
                            y <= "00000000" & add_y  ;
  end if;
---sub---
  if(deco="00001")then
     sub_y <= tempa - tempb ;
                y <=  "00000000" & sub_y  ;
  end if;
---multi---
  if(deco="00010")then
     mlt_y <= tempa * tempb ;
                y <= mlt_y ;
  end if;
----NOT----
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
            if(deco="00011")then
      not_y <= not tempa ;
                  y <= "00000000" & not_y   ;
   end if;
----OR----
            if(deco="00100")then
      or_y <= tempa OR tempb ;
                        y <= "00000000" & or_y    ;
   end if;
----AND----
            if(deco="00101")then
      and_y <= tempa AND tempb ;
                  y <=  "00000000" & and_y  ;
   end if;
----XOR----
            if(deco="00110")then
       xor_y <= tempa XOR tempb ;
                        y <=  "00000000" & xor_y ;
   end if;
----XNOR-----
            if(deco="00111")then
                  xnor_y <= tempa XNOR tempb ;
                     y <=  "00000000" & xnor_y ;
   end if;

-------timer-------
----up----
   if(deco="01000")then

     if(timer_up_mod=0)then
                  ti_up_duty<= 50 ;
                        elsif(timer_up_mod= 1 )then
                              ti_up_duty<= 25 ;
                                    elsif(timer_up_mod=2)then
                               ti_up_duty<= 10 ;
                                       else
                                                ti_up_duty<= 5 ;
                end if ;
                if(clk1'event and clk1 = '0')then
                      timer0<= timer0+1;
                                                if(timer0>=ti_up_duty)then
                                    ti_up_out <='1';
                                                else
                                                 ti_up_out <='0' ; end if; end if ;
    end if ;
----down----
   if(deco="01001")then
                if(timer_dwn_mod=0)then
                      ti_dwn_duty<= 50 ;
                            elsif(timer_dwn_mod=1)then
                                  ti_dwn_duty<= 25 ;
                                        elsif(timer_dwn_mod=2)then
                                   ti_dwn_duty<= 10 ;
                                                else
                                                   ti_dwn_duty<= 5 ;
                end if ;
                           if(clk1'event and clk1 = '0')then
                      timer1<= timer1-1;
                                                if(timer1=0)then
                                                            timer1<= 100;
                                                            end if ;
                                    ti_dwn_out <='1';
                                               if(timer1>=ti_dwn_duty)then
                                    ti_dwn_out <='1';
                                              else
                                                 ti_dwn_out <='0' ; end if; end if ;
end if ;

----COUNTER----
if(deco="01010")then
                        ---if(cnt_in'event and cnt_in= '0')then-----count---in-----
                         if(clk1'event and clk1= '0')then-----count---in-----
                      count0<= count0+1;
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
                                        if(count0>=30)then
                                cont_out <='1';
                                        else
                                          cont_out <='0';
                                        end if;-- end if ;
                              cnt_data <= conv_std_logic_vector(count0,5);
                                      cnt1_data <= cnt_data ;
                                                  end if ;
                                    y(4 downto 0) <= cnt1_data ;
end if ;

-----serial transmeter-------
if(deco="01011")then
              if(clk1'event and clk1 = '0')then
                temp1 <= temp1+1;
                          if(temp1=9)then
                           temp1<=0;
                           end if; end if;
    case temp1 is
                 when 0 => SE_cs1<= '0';
                           when 1 => SE_cs1<= '1';
                           when 2 => SE_cs1<= '0';
                           when others =>null ;
                          end case;
  if(SE_cs1='1')then
      h2<= pa_in;----parallel in-----
                 temp2<=0;
              else
                if(clk1'event and clk1='0')then
                            temp2 <= temp2+1;
                            seri_out <= h2(temp2);----serial out----
                            if(temp2=7)then
                              temp2<=0;
                            end if;
                          end if ;
              end if ;
 end if ;

-----serial receiver-------
if(deco="01100")then
              if(clk1'event and clk1 = '0')then
                temp3 <= temp3+1;
                          if(temp3=10)then
                           temp3<=0;
                           end if; end if;

      case temp3 is
                 when 0 => SE_cs2<= '0';
                           when 1 => SE_cs2<= '1';
                           when 2 => SE_cs2<= '0';
                           when others =>null ;
                           end case;
    if(SE_cs2='0')then
      if(clk1'event and clk1='0')then
                            temp4 <= temp4+1;
                            h1(temp4)<= se_in; ----serial_in----
                           if(temp4=7)then
                            temp4<=0;
                            end if;
                                      end if ;
                else
                 pa_data<= h1(1 to 8 ) ;----parallel out
                           temp4<=0;
                          end if ;
       y <= pa_data &"00000000" ;
 end if ;

------PARITY CHECK------
if(deco="01101")then
    parity(0) <= '1';----parity bit
        parity(1) <= parity(0) XOR a1(0);
                          parity(2) <= parity(1) XOR a1(1);
                          parity(3) <= parity(2) XOR a1(2);
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
                        parity(4) <= parity(3) XOR a1(3);
                        parity(5) <= parity(4) XOR a1(4);
                        parity(6) <= parity(5) XOR a1(5);
                        parity(7) <= parity(6) XOR a1(6);
                              parity_bit <= parity(7) XOR a1(7);   ------ evwn=>0  odd=>1

end if ;

-----PARITY BIT GENERAT ------
if(deco="01110")then
    parity(0) <= '1';----parity bit
        parity(1) <= parity(0) XOR a1(0);
                        parity(2) <= parity(1) XOR a1(1);
                        parity(3) <= parity(2) XOR a1(2);
                        parity(4) <= parity(3) XOR a1(3);
                        parity(5) <= parity(4) XOR a1(4);
                        parity(6) <= parity(5) XOR a1(5);
                              parity_bit <= parity(6) XOR a1(6);   ------ evwn=>0  odd=>1
      prt_data <= parity_bit & a1(6 downto 0);
                               y <= prt_data &"00000000" ;
end if ;

------DAC---ADC------
  ssb<='1';ceo<='1';initb<='1';      A_Dclk<=clk;

-----DAC-----
if(deco="01111")then
   DAC_clr<= '1'; ampcs<='1'; adcon<='1';----DAC_clr='0' for reset
   h4<= DAC_in ;----lsb------msb-----
              if(clk'event and clk = '0')then
                DACtemp1 <= DACtemp1+1;
                          if(DACtemp1=26)then
                            DACtemp1<=0;
                          end if; end if;
                         case DACtemp1 is
                  when 0 => DAC_cs1<= '1';
                            when 1 => DAC_cs1<= '1';
                            when 2 => DAC_cs1<= '0';
                            when others =>null ;
                         end case;
                         daccs<=DAC_cs1 ;
   if(DAC_cs1='1')then
    h3 <= h4 & "11111100" ;
                DACtemp2<=0 ;
    elsif(clk'event and clk='0')then
                          DACtemp2 <= DACtemp2+1;
                           DAC_SET <= h3(DACtemp2) ;
                           if(DACtemp2=23)then
                            DACtemp2<=0;
                           end if;  end if ;
                           mosi<= DAC_SET ;
end if ;

-----ADC-----
if(deco="10000")then
       DAC_cs1<='1'; ashdn<=shdn;
   if(ampad='1')then
              if(clk'event and clk = '0')then
                ADCtemp1 <= ADCtemp1+1;
                          if(ADCtemp1=10)then
                            ADCtemp1<=0;
                          end if;
                         end if;
      case ADCtemp1 is
                  when 0 => ADC_cs1<= '1';
                            when 1 => ADC_cs1<= '1';
                            when 2 => ADC_cs1<= '0';
                            when others =>null ;
                            end case;
       ampcs<=ADC_cs1 ;
      if(ADC_cs1='1')then
        h5 <= "10001000" ;
                    ADCtemp2<=0 ;
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
        elsif(clk'event and clk='0')then
                                ADCtemp2 <= ADCtemp2+1;
                                AMP_SET <= h5(ADCtemp2) ;
                                if(ADCtemp2=8)then
                                    ADCtemp2<=0;
                                end if;
                    end if ;
                  end if ;
                              mosi<= AMP_SET ;
   if(ampad='0')then
                 if(clk'event and clk = '0')then
                   ADCtemp3 <= ADCtemp3+1;
                              if(ADCtemp3=34)then
                                ADCtemp3<=0;
                              end if;
                                    end if;
               end if ;
      case ADCtemp3 is
                 when 0 => ADC_cs2<= '0';
                           when 1 => ADC_cs2<= '1';
                           when 2 => ADC_cs2<= '0';
                           when others =>null ;
                           end case;
                          adcon<=ADC_cs2 ;
     smiso<=miso;
  if(ADC_cs2='0')then
    if(clk'event and clk='0')then
                          ADCtemp4 <= ADCtemp4+1;
                          h6(ADCtemp4)<= smiso ;
                           if(ADCtemp4=35)then
                             ADCtemp4<=0;
                           end if; end if ;
               else
                    y1 <= h6(1 to 16 ) ;---msb----lsb-----
                             ADCtemp4<=0;
               end if ;
                y <= y1 ;
end if ;

-----FUNCTION GENERATOR-----
if(deco="10001")then
   DAC_clr<= '1'; ampcs<='1'; adcon<='1';----DAC_clr='0' for reset
     h9<= h8 ;----lsb------msb-----
              if(clk'event and clk = '0')then
                DACtemp3 <= DACtemp3+1;
                          if(DACtemp3=26)then
                            DACtemp3<=0;
                           end if; end if;
                          case DACtemp3 is
                  when 0 => DAC_cs2<= '1';
                            when 1 => DAC_cs2<= '1';
                            when 2 => DAC_cs2<= '0';
                            when others =>null ;
                          end case;
                          daccs<=DAC_cs2 ;
   if(DAC_cs2='1')then
     h7 <= h9 & "11111100" ;
                DACtemp4<=0 ;
     elsif(clk'event and clk='0')then
                          DACtemp4 <= DACtemp4+1;
                           DAC_SET1 <= h7(DACtemp4) ;
                           if(DACtemp4=23)then
                            DACtemp4<=0;
                           end if;  end if ;
                             mosi<= DAC_SET1 ;
end if ;
              if(DAC_cs2'event and DAC_cs2 = '0')then
                   func<= func+1;
                            if(func = 20)then
                                      last <= '1' ;
                                                        func<= 0  ;
                             else
                                      last <= '0' ;
```

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

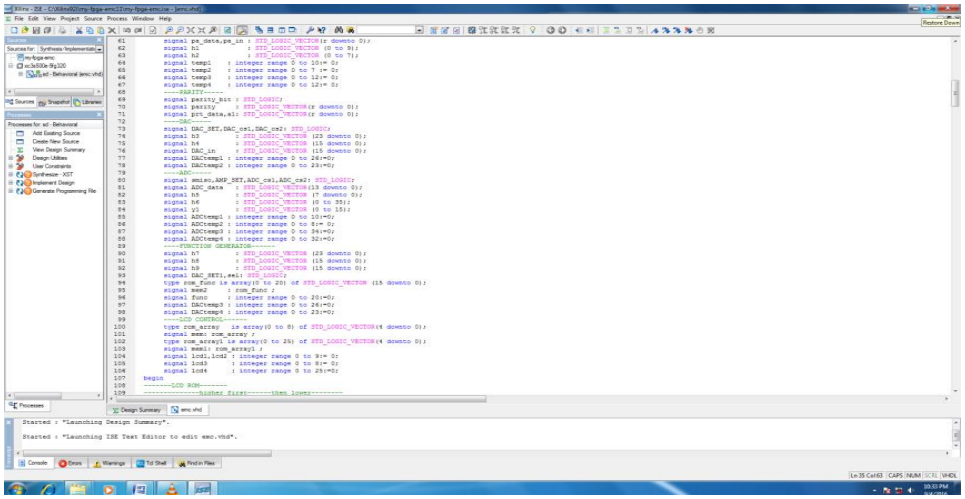
ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017

```
                        end if ;
                        h8<= mem2(func);
                end if ;
end process;

------LCD--CONTROL------
process(clk,deco)
                begin
                        if(clk'event and clk = '0')then
                lcd1<= lcd1+1;
                                if(lcd1>= 2 and lcd1 <= 8)then
                        RW<='0';
                                else
                                        RW<='1' ; end if; end if ;

                if(clk'event and clk = '0')then
                    lcd2<= lcd2+1;
                                if(lcd2>= 4 and lcd2 <= 6)then
                        E<='1';
                                else
                                        E<='0' ; end if; end if ;
            end process;
process(RW,deco)
                begin
if(deco="10010")then      -----DATA DISPLAY-----
                if(RW'event and RW = '0')then
                    lcd4<= lcd4+1;
                                if(lcd4 = 25)then
                                        lcd4 <= 0 ;
                        end if ;
                                DB1<= mem1(lcd4);
      end if ;
  else                  -----LCD SETUP-----
                if(RW'event and RW = '0')then
                    lcd3<= lcd3+1;
                        if(lcd3 = 8)then
                                lcd3 <= 0 ;
                        end if ;
                        DB1<= mem(lcd3);
                end if ;
end if ;
end process;
end Behavioral;
```



V. SIMULATE BEHAVIORAL MODEL

You can now run a functional simulation on the display drive module. With display_drive.vhd highlighted in the **Source** window, the **Process** window will give all the available operations for the particular module. A VHDL file can be synthesized and then implemented through to a bit stream. Normally, a design consists of several lower level modules wired together by a top-level file. In this instance, we are going to simulate only one lower-level module to introduce the functional simulation methodology.

UGC Approved Journal

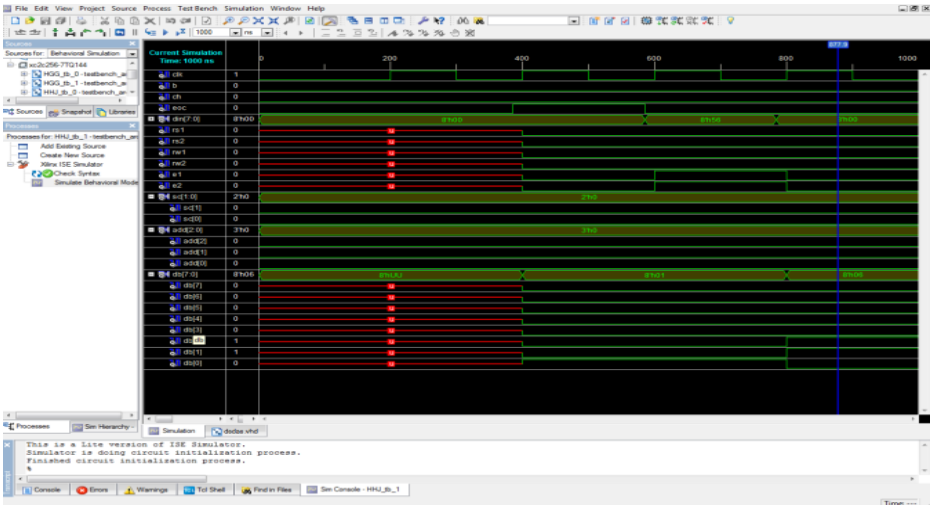
IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 6, June 2017



REFERENCES

[1] VHDL Programming by Example By Douglas L. Perry
[2] Circuit Design with VHDL By Volnei A. Pedroni
[3] DIDITAL DESIGN principles & practices By JOHN F. WAKERLY
[4] Fundamentals of Digital Logic with VHDL Design. By Stephen Brownand Zvonko Vranesic
[5] The Practical Xilinx Designer Lab Book- Prentice Hall. By David Van den Bout
[6] The Practical Xilinx Designer Lab Book- Prentice Hall. By David Van den Bout
[7] VHDL Starter's Guide.-Prentice Hall. By Sudhakar Yalamanchili
[8] Data Acquisition Linear devices Data Book - National Semiconductor.
[9] Digital Designing with Programmable Logic Devices.-Prentice Hall.By John W. Carter
[10] International Journal of Advanced Research in Computer Science and Software Engineering (Volume 3, Issue 8, August 2013)
[11] Programmable Logic Design Quick Start Guide (UG500 (v1.0) May 8, 2008)
[12] V. Jayaprakasan1,M.Madheswaran, "FPGA Implementation of FIR based Decimation Filter Structure for WiMAX Application", International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 7, July 2013
[13] Gaurav Sharma, Arjun Singh Chauhan, Himanshu Joshi, "Delay Comparison of 4 by 4 Vedic Multiplier based on Different Adder Architectures using VHDL", International Journal of IT, Engineering and Applied Sciences Research (IJIEASR) ISSN: 2319-4413 Volume 2, No. 6, June 2013

BIOGRAPHIES

Born in May 1960, **Dr. Haresh N. Pandya** Professor & Head, Department of Electronics, Saurashtra University, Rajkot, Gujarat, India. I obtained B.Sc., M.Sc. and Ph.D. degrees from Saurashtra University-Rajkot. He specialized in the area of Electronic circuits and devices. My Research area in Microprocessor, Microcontroller, Embedded System & Robotics. Published 78 numbers of technical papers in deferent international journals.

Born in December 1986, **Mr. Mahesh Rangapariya** is currently working as a Ph.D. Student & Visiting Lecturer at Department of Electronics, Saurashtra University-Rajkot. I obtained Master's Degree in ELECTRONISC from Department of Electronics Saradar Patel University-Vallabh Vidhaya Nagar. Attended Two-day "WOKRSHOPE ON EMBEDDED SYSTEMS" at Department of Electronics Saradar Patel University-Vallabh Vidhaya Nagar 2009. Attended One-day "WOKRSHOPE ON EMBEDDED SYSTEMS" at The Institution of Electronics and Telecommunication Engineers (IETE)-Rajkot 2010, he obtained B.Sc. degree in electronics from Gujarat University Ahmadabad in 2007.

Born in July 1986, **Mr. Jitendra Rajput** is currently working as Assistant professor of Electronics at SSR College of Science,(Affiliated to Pune University), Silvassa- 396230, D & NH, India. I obtain Master's Degree in ELECTRONISC from Department of Electronics Saradar Patel University-Vallabh Vidhaya Nagar. Attended Two-day "WOKRSHOPE ON EMBEDDED SYSTEMS" at Department of Electronics Saradar Patel University-Vallabh Vidhaya Nagar 2009. Attended Two-day "WOKRSHOPE ON EMBEDDED SYSTEMS" at Microsystems LTD., Chennai. Held during March 23-24, 2010. The workshop included comprehensive module on Embedded Processor, Architecture, Applications and Real Time Operating System (RTOS).