Report BTP

Team Size :2

Jashwanth Reddy

Kurra Lokesh

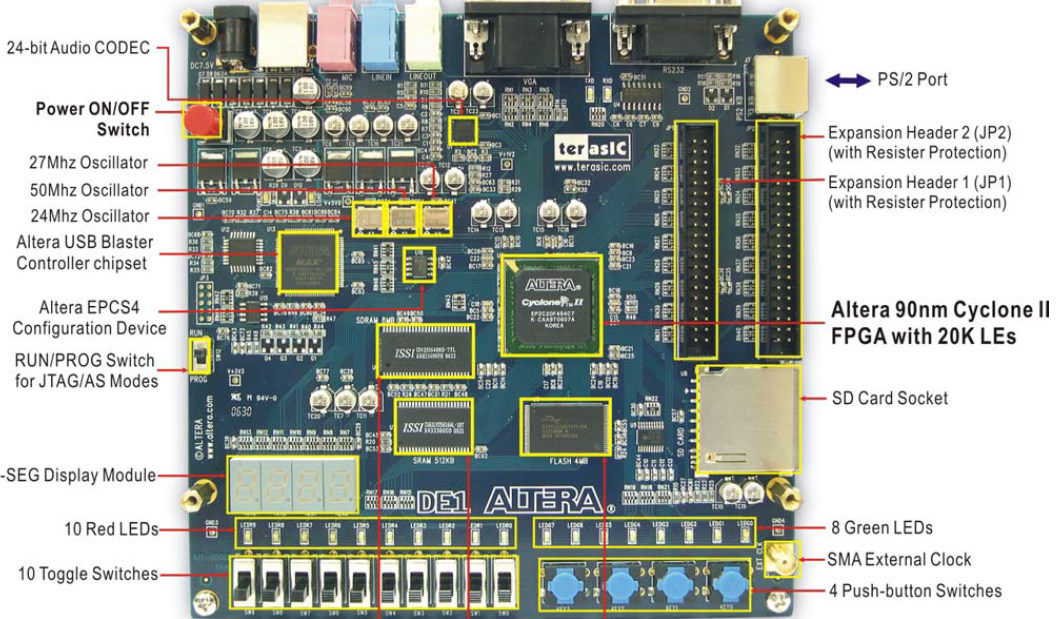
Guide : shubhajit roy chowdhury

Project : Computer on FPGA

Aim : The aim of the project is to use FPGA as a mother board to interface with I/O devices and memory management to develop some multimedia applications if possible.

**Start to BTP eval-1:**

During this part we are busy in understanding state of art of Altera Cyclone II 90nm FPGA.



The following hardware is provided on the DE1 board:

• USB Blaster (on board) for programming and user API control; both JTAG and Active Serial (AS) programming modes are supported

• 512-Kbyte SRAM

• 8-Mbyte SDRAM

• 4-Mbyte Flash memory

• SD Card socket

• 4 pushbutton switches

• 10 toggle switches

• 10 red user LEDs

• 8 green user LEDs

• 50-MHz oscillator, 27-MHz oscillator and 24-MHz oscillator for clock sources

• 24-bit CD-quality audio CODEC with line-in, line-out, and microphone-in jacks

• VGA DAC (4-bit resistor network) with VGA-out connector

• RS-232 transceiver and 9-pin connector

• PS/2 mouse/keyboard connector

• Two 40-pin Expansion Headers with resistor protection

• Powered by either a 7.5V DC adapter or a USB cable .

In order to use the DE1 board , the user has to be familiar with quartus software.

First we tried with the installation of altera USB blaster driver for communication between host computer and DE1 board

The default file loaded from flash memory helps to blink LED’s , 7 Segment Displays start counting and outputs in Hexadecimal format.

We can also connect a micro phone and also VGA monitor to display default images.

The supporting software are

Quartus II web Edition

Nios II Software Build Tools

SOPC builder

Nios II Embedded Design Suite

Modelsim Altera Starter Edition 6.5b

Coming to our work we developed some simple processor of 4 bit which can perform adder, substractor , comparator etc… first in ModelSim

And then in Quartus II web edition. The code and simulation result of a simple binary counter code in Modelsim looks like

**library ieee;**

**Use ieee.std\_logic\_1164.all;**

**entity counter is**

**port( clk: in std\_logic; reset: in std\_logic;**

**enable: in std\_logic;**

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

**end counter;**

**architecture behav of counter is**

**signal pre\_count:**

**std logic\_vector (3 downto 0);**

**begin**

**process( clk , enable, reset)**

**begin**

**if reset = '1' then**

**pre\_count <= "0000";**

**Elsif (clk='1' and clk'event) then**

**if enable = '1' then**

**pre\_count <= pre\_count + "1";**

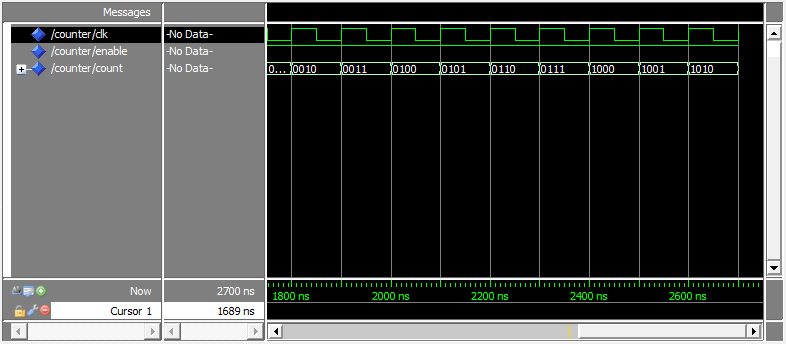
**end if;**

**end if;**

**end process;**

**count <= pre\_count;**

**end behav;**



There are pins availability for mapping these functions on FPGA .

This is done using IO-Assignment block editor in the software.The pin mappings are as follows:

|  |  |
| --- | --- |
| 27 MHz Clock | PIN\_D12 and PIN\_E12 |
| 50 MHz Clock | PIN\_L1 |
| 24 MHz Clock | PIN\_A12 and PIN\_B12 |
| *Green LEDs* | |
| Green\_LED\_0 | PIN\_U22 |
| Green\_LED\_1 | PIN\_U21 |
| Green\_LED\_2 | PIN\_V22 |
| Green\_LED\_3 | PIN\_V21 |
| Green\_LED\_4 | PIN\_W22 |
| Green\_LED\_5 | PIN\_W21 |
| Green\_LED\_6 | PIN\_Y22 |
| Green\_LED\_7 | PIN\_Y21 |
| *Seven-segment Displays* | |
| Hex\_0[0] | PIN\_J2 |
| Hex\_0[1] | PIN\_J1 |
| Hex\_0[2] | PIN\_H2 |
| Hex\_0[3] | PIN\_H1 |
| Hex\_0[4] | PIN\_F2 |
| Hex\_0[5] | PIN\_F1 |
| Hex\_0[6] | PIN\_E2 |
| Hex\_0, Decimal Point | No Connection |
| Hex\_1[0] | PIN\_E1 |
| Hex\_1[1] | PIN\_H6 |
| Hex\_1[2] | PIN\_H5 |
| Hex\_1[3] | PIN\_H4 |
| Hex\_1[4] | PIN\_G3 |
| Hex\_1[5] | PIN\_D2 |
| Hex\_1[6] | PIN\_D1 |
| Hex\_1, Decimal Point | No Connection |
| Hex\_2[0] | PIN\_G5 |
| Hex\_2[1] | PIN\_G6 |
| Hex\_2[2] | PIN\_C2 |
| Hex\_2[3] | PIN\_C1 |
| Hex\_2[4] | PIN\_E3 |
| Hex\_2[5] | PIN\_E4 |
| Hex\_2[6] | PIN\_D3 |
| Hex\_2, Decimal Point | No Connection |
| Hex\_3[0] | PIN\_F4 |
| Hex\_3[1] | PIN\_D5 |
| Hex\_3[2] | PIN\_D6 |
| Hex\_3[3] | PIN\_J4 |
| Hex\_3[4] | PIN\_L8 |
| Hex\_3[5] | PIN\_F3 |
| Hex\_3[6] | PIN\_D4 |
| Hex\_3, Decimal Point | No Connection |
| *Push Buttons* | |
| Key\_0 | PIN\_R22 |
| Key\_1 | PIN\_R21 |
| Key\_2 | PIN\_T22 |
| Key\_3 | PIN\_T21 |
| *Red LEDs* | |
| Red\_LED\_0 | PIN\_R20 |
| Red\_LED\_1 | PIN\_R19 |
| Red\_LED\_2 | PIN\_U19 |
| Red\_LED\_3 | PIN\_Y19 |
| Red\_LED\_4 | PIN\_T18 |
| Red\_LED\_5 | PIN\_V19 |
| Red\_LED\_6 | PIN\_Y18 |
| Red\_LED\_7 | PIN\_U18 |
| Red\_LED\_8 | PIN\_R18 |
| Red\_LED\_9 | PIN\_R17 |
| *Slide Switches* | |
| Switch\_0 | PIN\_L22 |
| Switch\_1 | PIN\_L21 |
| Switch\_2 | PIN\_M22 |
| Switch\_3 | PIN\_V12 |
| Switch\_4 | PIN\_W12 |
| Switch\_5 | PIN\_U12 |
| Switch\_6 | PIN\_U11 |
| Switch\_7 | PIN\_M2 |
| Switch\_8 | PIN\_M1 |
| Switch\_9 | PIN\_L2 |

We achieved control of FPGA using the DE1 control panel user interface developed through which we can control LED’s , SRAM , SDRAM , Flash memory on the chip.

We also got acquainted with SOPC builder . One can use the SOPC Builder to implement a desired system simply by choosing the required components and specifying the parameters needed to make each component fit the overall requirements of the system.

the overall requirements of the system.

This ended the first part of evaluation.

**SECOND PART OF BTP:**

In this part of BTP we provided VGA interface to FPGA .The VGA connector looks like



The main signals are VGA horizontal sync ,Vertical Sync

The pin connections and code are given below. VGA RGB data which were controlled in the following code using clock of 25 MHZ generated by FPGA.

module monitor{

CLOCK\_24 , CLOCK\_27,

CLOCK\_50, EXT\_CLOCK,

KEY, SW, VGA\_HS,

VGA\_VS, VGA\_R, VGA\_G,

VGA\_B,

);

input [1:0] CLOCK\_24;

input [1:0] CLOCK\_27;

input CLOCK\_50;

input EXT\_CLOCK;

input [3:0] KEY;

input [9:0] SW;

output VGA\_HS; VGA\_VS;VGA V\_SYNC

output [3:0] VGA\_R;

output [3:0] VGA\_G;

output [3:0] VGA\_B;

wire VGA\_CTRL\_CLK;

wire AUD\_CTRL\_CLK;

wire [9:0] mVGA\_X ; wire [9:0] mVGA\_Y;wire [9:0] mVGA\_R;

wire [9:0] mVGA\_G; wire [9:0] mVGA\_B; wire [9:0] mPAR\_R;

wire [9:0] mPAR\_G; wire [9:0] mPAR\_B; wire [9:0] mPAR1\_R;

wire [9:0] mPAR1\_G; wire [9:0] mPAR1\_B; wire [9:0] mPAR2\_R;

wire [9:0] mPAR2\_G; wire [9:0] mPAR2\_B; wire [9:0] mPAR3\_R;

wire [9:0] mPAR3\_G; wire [9:0] mPAR3\_B; wire [9:0] mPAR4\_R;

wire [9:0] mPAR4\_G; wire [9:0] mPAR4\_B; wire [9:0] mPAR5\_R;

wire [9:0] mPAR5\_G; wire [9:0] mPAR5\_B; wire [9:0] mOSD\_R;

wire [9:0] mOSD\_G; wire [9:0] mOSD\_B; wire [9:0] oVGA\_R;

wire [9:0] oVGA\_G; wire [9:0] oVGA\_B;

wire [19:0] mVGA\_ADDR;

reg [27:0] Cont; reg ST;

always@(posedge CLOCK\_50) Cont <= Cont+1'b1;

// VGA Data 10-bit to 4-bit

assign VGA\_R = oVGA\_R[9:6];

assign VGA\_G = oVGA\_G[9:6];

assign VGA\_B = oVGA\_B[9:6];

// VGA Source pattern assignment of colors

assign mVGA\_R = ( SW[0] & !SW[1] & !SW[2] ) ? mPAR1\_R : ( !SW[0] & SW[1] & !SW[2] ) ? mPAR2\_R : ( !SW[0] & !SW[1] & SW[2] ) ? mPAR3\_R : mPAR4\_R ;

assign mVGA\_G = ( SW[0] & !SW[1] & !SW[2] ) ? mPAR1\_G : ( !SW[0] & SW[1] & !SW[2] ) ? mPAR2\_G : ( !SW[0] & !SW[1] & SW[2] ) ? mPAR3\_G : mPAR4\_G ;

assign mVGA\_B = ( SW[0] & !SW[1] & !SW[2] ) ? mPAR1\_B : ( !SW[0] & SW[1] & !SW[2] ) ? mPAR2\_B : ( !SW[0] & !SW[1] & SW[2] ) ? mPAR3\_B : mPAR4\_B ;

VGA\_Audio\_PLL u1(.inclk0(CLOCK\_27[0]),.c0(VGA\_CTRL\_CLK),.c1(AUD\_CTRL\_CLK) );

VGA\_Pattern u6(.oRed(mPAR4\_R), .oGreen(mPAR4\_G) , .oBlue(mPAR4\_B), .iVGA\_X(mVGA\_X), .iVGA\_Y(mVGA\_Y), .iVGA\_CLK(VGA\_CTRL\_CLK), .iRST\_N(KEY[0]) );

endmodule

/ \*…………….. The VGA pattern code looks as follows …….. \*/

module

VGA\_Pattern2 ( oRed , oGreen , oBlue , iVGA\_X , iVGA\_Y ,

iVGA\_CLK, iRST\_N );

output reg [9:0] oRed;

output reg [9:0] oGreen;

output reg [9:0] oBlue;

input [9:0] iVGA\_X;

input [9:0] iVGA\_Y;

input iVGA\_CLK;

// Control Signals

input iRST\_N;

always@(posedge iVGA\_CLK or negedge iRST\_N)

begin

if(!iRST\_N)

begin

oRed <= 0;

oGreen <= 0;

oBlue <= 0;

end

else

begin

oRed <= 0;

oGreen <= 512 ;

oBlue <= 0 ;

end

end

endmodule

The results of VGA patterns were found satisfactory to the panel members and recommended further changes like character printing and also other interfaces to the board which we are looking forward to.