ARBITRARY BASE CONVERTER

ELEC 204 Digital System Design

Laboratory

Lab Project Report

Instructor: Mehmet Cengiz Onbaşlı

Designed by:

İrem Arpag: Department of Computer Engineering

Ayşenur Torun: Department of Electrical and Electronics Engineering

Koç University

Fall 2018

CONTENTS

1.AIM OF THE PROJECT

By choosing this project, we would like to understand programming an arbitrary base converter by using VHDL and get familiar with sequential logic design by using if statements, flip flops and clock cycles.

2.PROCESS

Before the start project, simple schematic was designed. Basic design was given below.

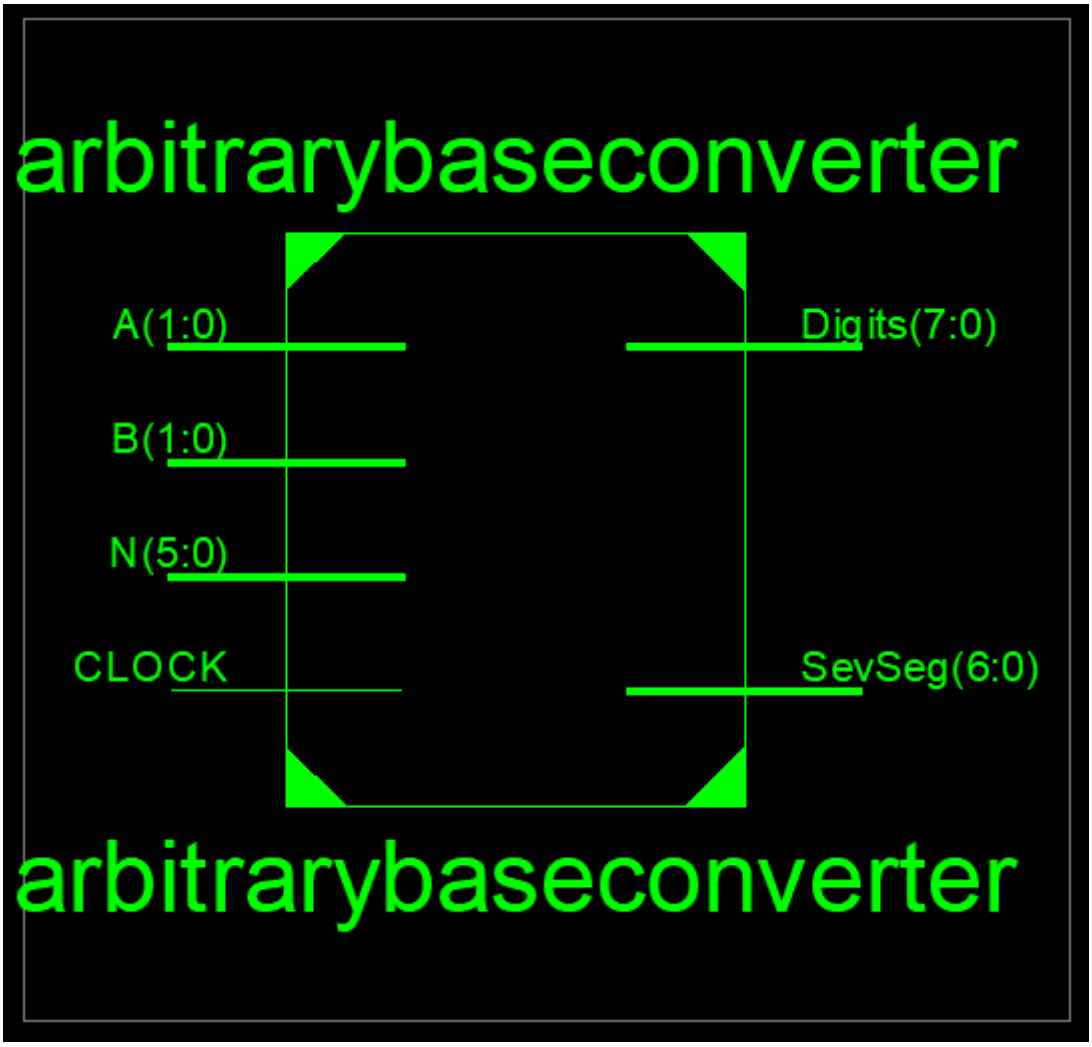

Figure.1 RTL Schematic of the arbitrary base converter

A controls the base for the number which is shown by last four anodes. Similarly, B controls the base for the number which is shown by first four anodes. For the “00” case of A input, the base is selected as 3. Then base is 4 in the case of “01”, 5 for “10” and 6 for “11”.

This selection is valid for B input. N is a 6-bit input which is the number to be converted. ‘Digits’ stands for the anodes. ‘SevSeg’ is the abbreviation of seven segment.

To be able to determine the possibilities for each anode for each base (which is explained in part 3), we used sequential logic and comparators.

```
Macro Statistics
# Adders/Subtractors                                   : 1
 16-bit adder                                          : 1
# Counters                                             : 1
 17-bit up counter                                     : 1
# Registers                                            : 15
 Flip-Flops                                            : 15
# Comparators                                          : 72
 6-bit comparator greatequal                           : 30
 6-bit comparator greater                              : 8
 6-bit comparator less                                 : 3
 6-bit comparator lessequal                            : 31
```

Figure.2 Advanced HDL Synthesis Report

As a working mechanism, a binary number is selected by N switches. It is converted to the base from two to the base which is determined by A. Also, it is converted to the base from two to the base which is determined by B. As a result, you see two different base on FPGA. It seems that you convert the base which is determined by A to the base which is determined by B.At the last stage, the design is simulated on FPGA board with Prometheus.

3. ANALYZING THE CODE

```
entity arbitrarybaseconverter is
    generic (X : INTEGER:=50*10**6;                          -- 50*10^6 Hertz Clock
             Y: INTEGER:=65536);                             -- 2^13=8192
    Port (CLOCK : in  STD_LOGIC;
          B: in  STD_LOGIC_VECTOR (1 downto 0);              --base of the second number
          A : in  STD_LOGIC_VECTOR (1 downto 0);             --base of the first number
          N  : in  STD_LOGIC_VECTOR (5 downto 0);            --the number which is in binary
          SevSeg : out  STD_LOGIC_VECTOR (6 downto 0);       --seven segment
          Digits: out STD_LOGIC_VECTOR(7 downto 0));         --anodes
end arbitrarybaseconverter;
```

Figure.3 Port declaration

A part of the procedure for the number whose base is controlled by 'A' switches is shown below. If we assign the “11111011” to the digits, we choose the third anode. For instance, if we assign “01” to the A, we work in base 4. Firstly, we write the all possibilities which makes the third anode 3 in base 4. The same procedure is repeated for the numbers 2, 1 and 0. He appropriate values of ‘SevSeg’ is assigned to show those numbers in FPGA.

```
Digits <= "11111011";
if(A="00") then              --'00' case -> base:3
   if((N>="101101" and N<"110110") or (N>="010010" and N<"011011")) then SevSeg <=  "0010010"; --all the possibilities which makes
   elsif(N="111111" or (N>="100100" and N<="101100") or (N>="001001" and N<="010001")) then SevSeg <= "1001111";  --all the possibi
   elsif((N<"111111" and N>="110110") or (N<="011011" and N>="100011") or (N>="000000" and N<="001000"))then SevSeg <=  "0000001";
   end if;
elsif(A="01") then           --'01' case -> base:4
   if(N>"101111") then SevSeg <=  "0000110"; --all the possibilities which makes the third anode 3 in base 4
   elsif(N>"011111") then SevSeg <=  "0010010"; --all the possibilities which makes the third anode 2 in base 4
   elsif(N>"001111") then SevSeg <= "1001111";  --all the possibilities which makes the third anode 1 in base 4
   else SevSeg <=  "0000001"; -- 0
   end if;
```

Figure.4 The main part of the code

IOB PROPERTIES

	IOB Name	Type	Direction	IO Standard	Diff Term	Drive Strength	Slew Rate	Reg (s)	Resistor	IBUF/IFD Delay	Suspend
1	A<0>	IBUF	INPUT	LVCMO...						0 / 0	
2	A<1>	IBUF	INPUT	LVCMO...						0 / 0	
3	B<0>	IBUF	INPUT	LVCMO...						0 / 0	
4	B<1>	IBUF	INPUT	LVCMO...						0 / 0	
5	CLOCK	IBUF	INPUT	LVCMO...						0 / 0	
6	Digits<0>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
7	Digits<1>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
8	Digits<2>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
9	Digits<3>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
10	Digits<4>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
11	Digits<5>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
12	Digits<6>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
13	Digits<7>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
14	N<0>	IBUF	INPUT	LVCMO...						0 / 0	
15	N<1>	IBUF	INPUT	LVCMO...						0 / 0	
16	N<2>	IBUF	INPUT	LVCMO...						0 / 0	
17	N<3>	IBUF	INPUT	LVCMO...						0 / 0	
18	N<4>	IBUF	INPUT	LVCMO...						0 / 0	
19	N<5>	IBUF	INPUT	LVCMO...						0 / 0	
20	SevSeg<0>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
21	SevSeg<1>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
22	SevSeg<2>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
23	SevSeg<3>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
24	SevSeg<4>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
25	SevSeg<5>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE
26	SevSeg<6>	IOB	OUTPUT	LVCMO...		12	SL...			0 / 0	3STATE

Figure.5 Summary of Implementation Constraints File

4. FPGA BOARD

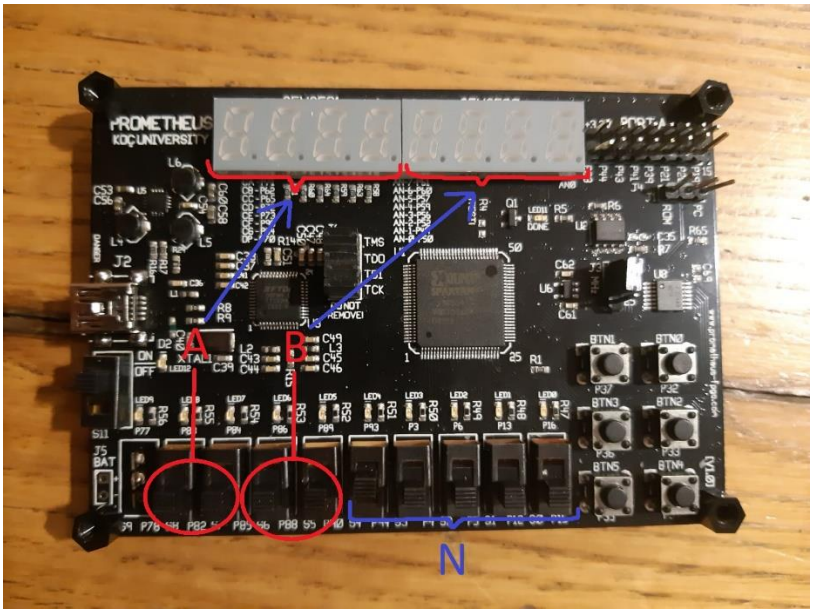

Figure.6 FPGA Board

EXPERIMENTAL RESULTS

As an example, we choose “00” for the A input (base:3), “01” for the input B (base:4)

We determined the N (6-bit binary number) as “001011” which is 11 in decimal.

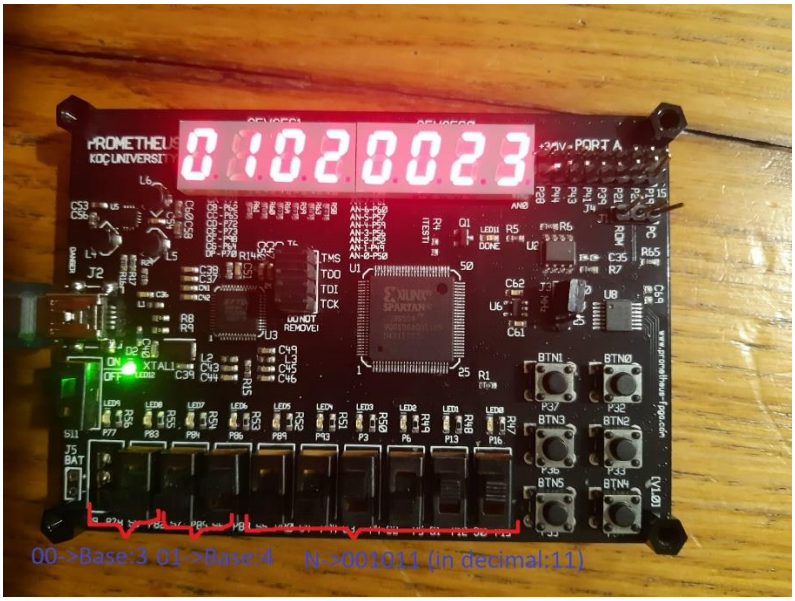

Figure.7 FPGA Board Example

$(0102)_3 = 0x3^3 + 1x3^2 + 0x3^1 + 2x3^0 = (001011)_2 = 11$

$(0023)_4 = 0x4^3 + 0x4^2 + 2x4^1 + 3x4^0 = (001011)_2 = 11$

5.CONCLUSION

In this project, we tried to understand programming an arbitrary base converter by using VHDL and get familiar with sequential logic design by using if statements, flip flops and clock cycles. So we successfully completed our objective and completed our first fully functional digital clock. The most important thing we learnt is that in order to accomplish to complete such complicated project we first need to break it into small pieces and learn how to unite them. For base conversion VHDL Code is written and design is assigned to FPGA Board's pins through Implementation constraint file. Result on the FPGA board verifies the design.