Introduction to Verilog Programming on Icoboard FPGA

Hemanth Kumar Desineedi and G V V Sharma*

Contents

***Abstract*—This manual provides an introduction to Verilog programming using the Icoboard-Lattice FPGA. This is done by implementing a decade counter using verilog. The process is likely to be similar for other FPGA boards as well.**

1 Componets

Component	Value	Quantity
Breadboard		1
Resistor	≥ 220Ω	1
FPGA	Icoboard	1
Seven Segment Display	Common Anode	1
Jumper Wires		20

TABLE I

2 Software Setup

1) Open a console window and execute the following commands for installing wiringPi, IcoProg, IcoStorm tools, Arachne-pnr and Yosys.

*The authors are with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

```
cd $HOME
git clone git://git.drogon.net/
    wiringPi
cd wiringPi && ./build

cd $HOME
sudo apt-get install subversion
svn co http://svn.clifford.at/
    handicraft/2015/icoprog
cd icoprog && make install

sudo apt-get install build-
    essential clang bison flex
    libreadline-dev
sudo apt-get install gawk tcl-
    dev libffi-dev git mercurial
     graphviz
sudo apt-get install xdot pkg-
    config python python3
    libftdi-dev

cd $HOME
git clone https://github.com/
    cliffordwolf/icestorm.git
    icestorm
cd icestorm && make && sudo
    make install

cd $HOME
git clone https://github.com/
    cseed/arachne-pnr.git
    arachne-pnr
cd arachne-pnr && make && sudo
    make install

cd $HOME
git clone https://github.com/
    cliffordwolf/yosys.git yosys
```

```
cd yosys && make && sudo make
    install
```

2) Open a text editor and type the following code. Save it as **Makefile**.

```
.PHONY: default
default: prog_sram

$(v_fname).blif: $(v_fname).v
        yosys -p 'synth_ice40 -
            blif $(v_fname).blif
            ' $(v_fname).v

$(v_fname).asc: $(v_fname).blif
    $(v_fname).pcf
        arachne-pnr -d 8k -p $(
            v_fname).pcf -o $(
            v_fname).asc $(
            v_fname).blif

$(v_fname).bin: $(v_fname).asc
        icetime -d hx8k -c 25 $
            (v_fname).asc
        icepack $(v_fname).asc
            $(v_fname).bin

prog_sram: $(v_fname).bin
        icoprog -p < $(v_fname)
            .bin
```

3 Example Codes

3.1 Blink LED

1) In the same directory, open a text editor and type the following verilog code and save it as **blink.v**

```
//Program for blinking LED
module example(input wire clk,
    output reg A);

reg[26:0] delay;

always@(posedge clk) begin
delay = delay+1;
if(delay==27'
    b101111101011110000100000000
    )
begin
delay=27'b0;
A=!A;
end
end
endmodule
```

2) In addition to the verilog file, we need to indicate to which FPGA pin we want to connect the A output.
3) This mapping is done in the file .pcf (pcf = Physical Constraint file).
4) Open a text editor and type the following and save it as **blink.pcf**

```
set_io clk R9
set_io A  B7
```

5) Setup the seven segment display on the breadboard.
6) Make pin connections according to Table II.
7) Now open terminal and go to the Directory where all files are saved and type the following command.

```
make v_fname=blink
```

8) What do you observe?
9) Find out how the delay is obtained in the above code, given that the clock frequency of the FPGA is 100 MHz.
10) Modify **blink.v** to turn the LED on and Off

Icoboard	**Display**
Vcc	Resistor
B7	Dot

TABLE II: Pin connections using bank P2 of Icoboard.

3.2 Display

1) Make pin connections from the Icoboard to the seven segment display in Fig. 1 according to Table III.
2) Execute the following verilog code

```
module sevenseg(output reg a,
    output reg b, output reg c,
    output reg d,output reg e,
    output reg f,output reg g);
     initial
     begin
     a=0;
```

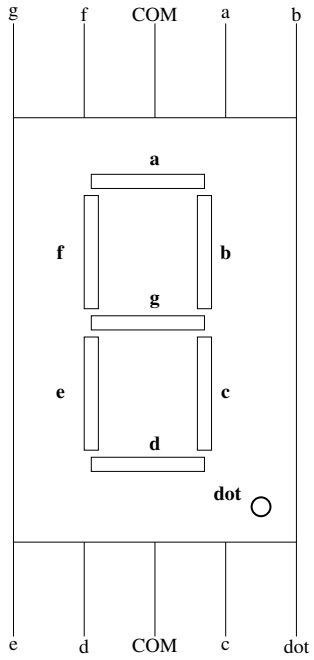

Fig. 1

Pin	Segment
A5	a
A2	b
C3	c
B4	d
B7	e
B6	f
B3	g
Vcc	COM

TABLE III: Pin Connections

```
    b=0;
    c=0;
    d=0;
    e=0;
    f=0;
    g=1;
    end
endmodule
```

with pcf

```
set_io a A5
set_io b A2
set_io c C3
set_io d B4
set_io e B7
set_io f B6
set_io g B3
```

What do you observe?

3) Use the above codes to generate numbers from 0-9 on the display. Comment.

3.3 Decade Counter

1) Execute the following verilog code

```
module display_decoder(
    input  wire clk,

    output reg a,//1-bit
       variable register // a,
       b, c, d, e, f, g are the
        final outputs.
    output reg b,
    output reg c,
    output reg d,
    output reg e,
    output reg f,
    output reg g
);
reg A;
reg B;
reg C;
reg D;
reg W;
reg X;
reg Y;
reg Z;

  reg [26:0] delay;//for delay
     of 1 second

  initial begin
    W<=0;
    X<=0;
    Y<=0;
    Z<=0;
    end
  always @(posedge clk) begin

  a=(!D&!C&!B&A)|(!D&C&!B&!A);
  b=(!D&C&!B&A)|(!D&C&B&!A);
  c=(!D&!C&B&!A);
  d=(!D&!C&!B&A)|(!D&C&!B&!A)
      |(!D&C&B&A);
```

```
   e=(!D&!C&!B&A)|(!D&!C&B&A)|(!
      D&C&!B&!A)|(!D&C&!B&A)|(!D
      &C&B&A)|(D&!C&!B&A);
   f=(!D&!C&!B&A)|(!D&!C&B&!A)
      |(!D&!C&B&A)|(!D&C&B&A);
   g=(!D&!C&!B&!A)|(!D&!C&!B&A)
      |(!D&C&B&A);

   D<=(W&X&Y&!Z)|(!W&!X&!Y&Z);
   C<=(Y&!X)|(Y&!W)|(!Y&X&W);
   B<=(!W&X)|(!Z&!X&W);
   A<=!W;




       delay = delay+1;

  if( delay == 27'
     b10111110101111000010000000
     )
begin

delay=27'b0;
   W<=A;
   X<=B;
   Y<=C;
   Z<=D;

  end

  end
endmodule
```

with pcf

```
set_io clk R9
set_io a A5
set_io b A2
set_io c C3
set_io d B4
set_io e B7
set_io f B6
set_io g B3
```

and observe the output on the display.