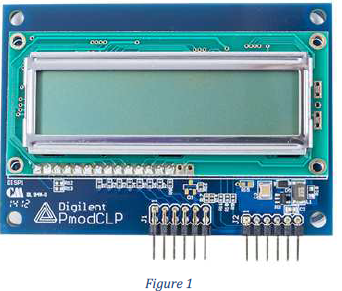
|  |  |  |
| --- | --- | --- |
| NAME | ID NUM | SEC |
| 1. أحمد عبد الله جبر القرم | 11819195 | 3/8:00-11:00 |
| - | - | - |

**Experiment 5**: LCD Driver

**Introduction :**

In this lab , I will design and implement a driver that will handle the parallel communication between LCD and the ZedBoard.

1. **PmodCLP :**

The Character LCD with Parallel Interface module from Digilent is a 16×2 character LCD to let the system boards display up to 32 different characters.

The LCD consists of two lines. Each line can display up to 16 characters. To display a

character, first you need to send the location address where you want the character to be

displayed. Then the character code of the character to be displayed.

The LCD device has three internal regions of memory. The Data Display RAM (DD

RAM), which references the data to be displayed on the screen, the Character Generator

RAM (CG RAM), which stores user-defined patterns and the Character Generator ROM

(CG ROM), which includes a number of predefined patterns that correspond to ASCII

symbols. We will only use the DD-RAM and the CG-ROM. To reference a value in the

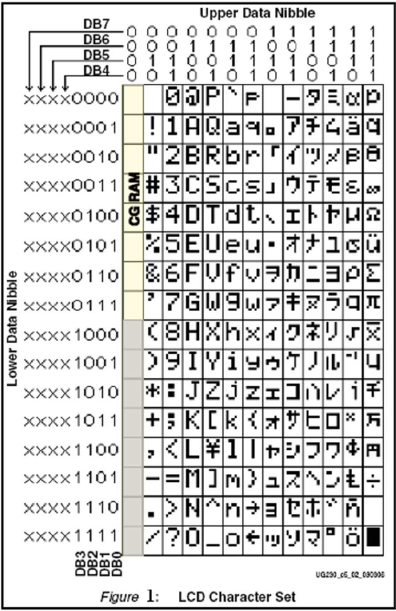
CG-ROM, the value in the figure 1 needs to be written into the DD-RAM. For example,

the character ‘S’ from the CGROM would have the value “01010011”.

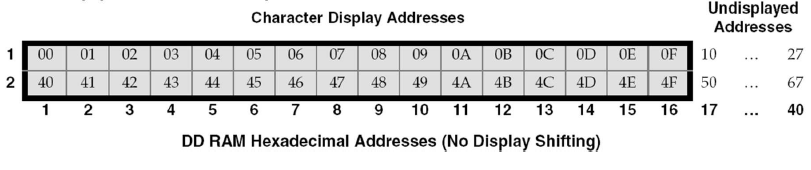
These values are written to the location address of the LCD. The address of DD-RAM

location is first sent to LCD then the code of the symbol to be displayed. The codes for

characters are as follows:



The LCD physical locations correspond to addresses in the DD-RAM as follows:

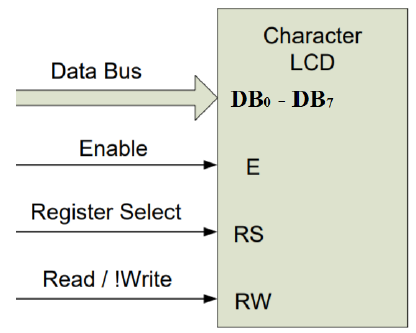


1. **Interfacing with Pmod :**

The PmodCLP utilizes a Samsung KS0066 LCD controller to display information to a 16x2

LCD panel which has both kinds of interface types: 4-bit bus and 8-bit bus. The PmodCLP

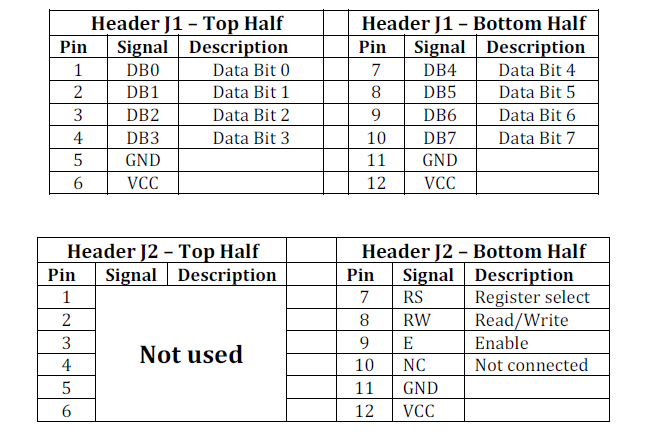
module has 8 bits interface as shown in the next figure.



Pmod CLP module should be connected with two Digilent Pmod™ compatible headers

(2x6). Therefore, it can be connected to JA and JB connectors from the ZedBoard. The

pinout for this module is described in the following table.

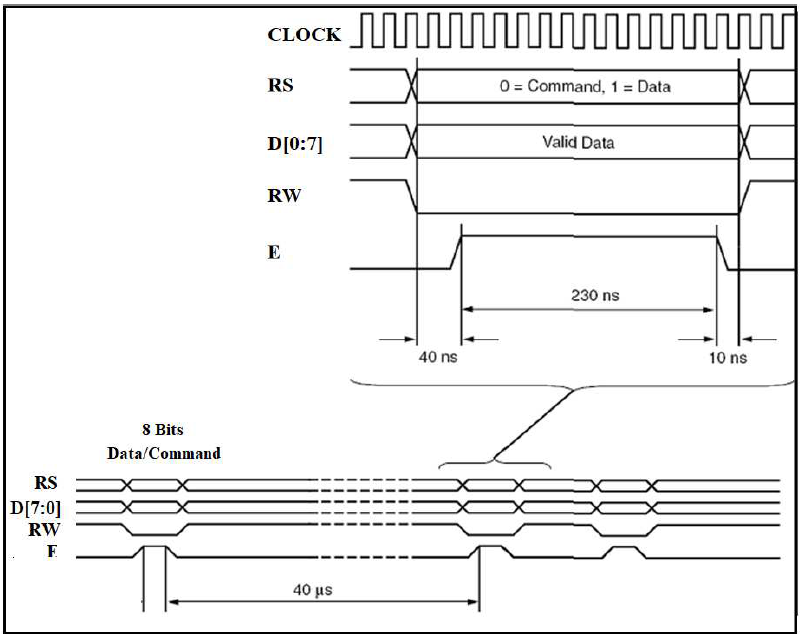


1. **Timing Requirements:**

The Pmod CLP module communicates with the host board via the GPIO protocol. This

particular module requires specific timings in order to program the LCD correctly.

Because the Pmod CLP module has 8 bits interface, any 8-bit data/command can be sent

in a single transition. The general timing diagram is shown below.

In case of two successive commands, they have to be separated with 40 μ s delay

(4000 clock cycles).

From the above diagram, it is important to notice the following:

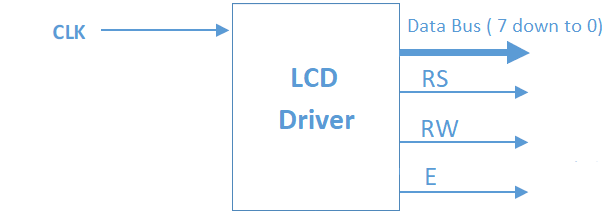
* Setup time (time for the outputs to stabilize) is 40ns (4 clock cycles).
* The hold time (time to assert the enable (E) pin) is 230ns (23 clock cycles).
* The fall time (time to allow the outputs to stabilize) is 10ns (1 clock cycle).

You can simplify the process as follows:

1. Select the correct value for RS, RW, and Data bus
2. Set E to 1
3. At least 230ns must elapse. (Use 1 μ s : 100 Clock cycles)
4. Set E to 0

**Objectives:**

* Lern how to display my name in the LCD
* Lern how to display counter in the LCD



**Tools used in Lap :**

1. Computer lap.
2. Vivado software .
3. Zboard from Xilinx.
4. VHDL
5. PmodCLP
6. two Digilent Pmod™ compatible headers

(2x6)

**Procedure :**

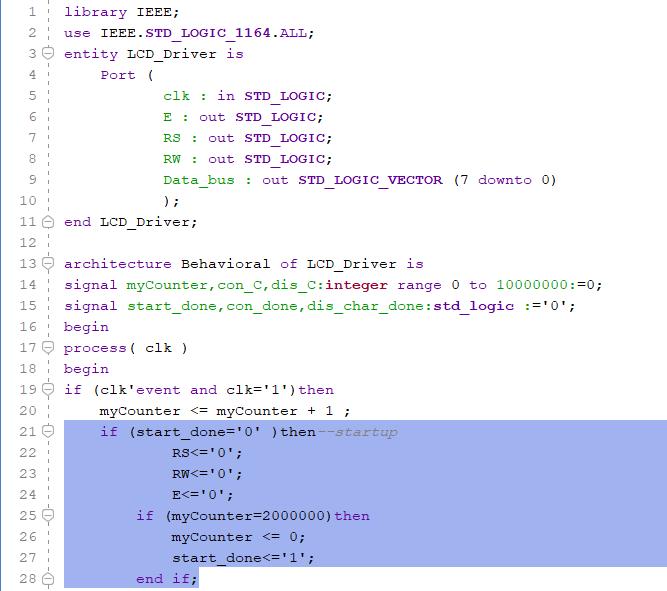
**Part 1** : Display my full Name (with shifted display )

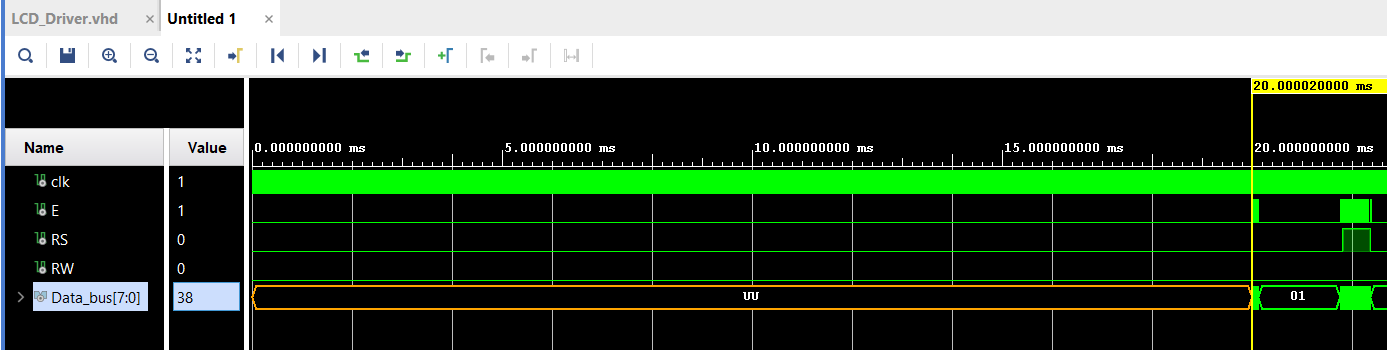
* **Startup:**

After the power on I wait for 20 ms (2 000 000 clock cycles at 100 MHz) before start configuration Code and

through this duration RS , RW and E was in logic '0' as we see :

VHDL code :



Simulation :

* **Configuration :**

1. Issue a Function Set command, 0x38, to configure the display for operation on 8-

bit data.

2. Wait for 40 μs (4000 clock cycles).

3. Issue an Entry Mode Set command, 0x06, to set the display to automatically

increment the address pointer.

4. Wait for 40 μs (4000 clock cycles).

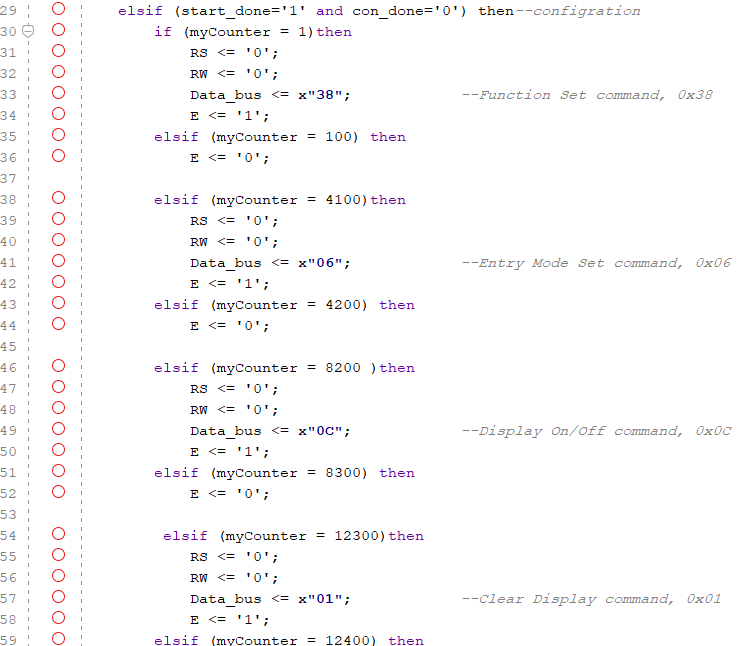
5. Issue a Display On/Off command, 0x0C, to turn the display on and disables the

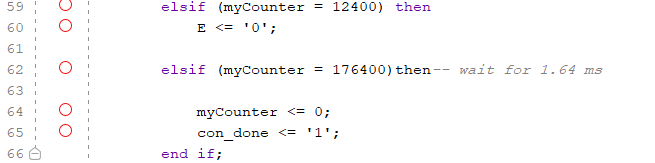
cursor and blinking.

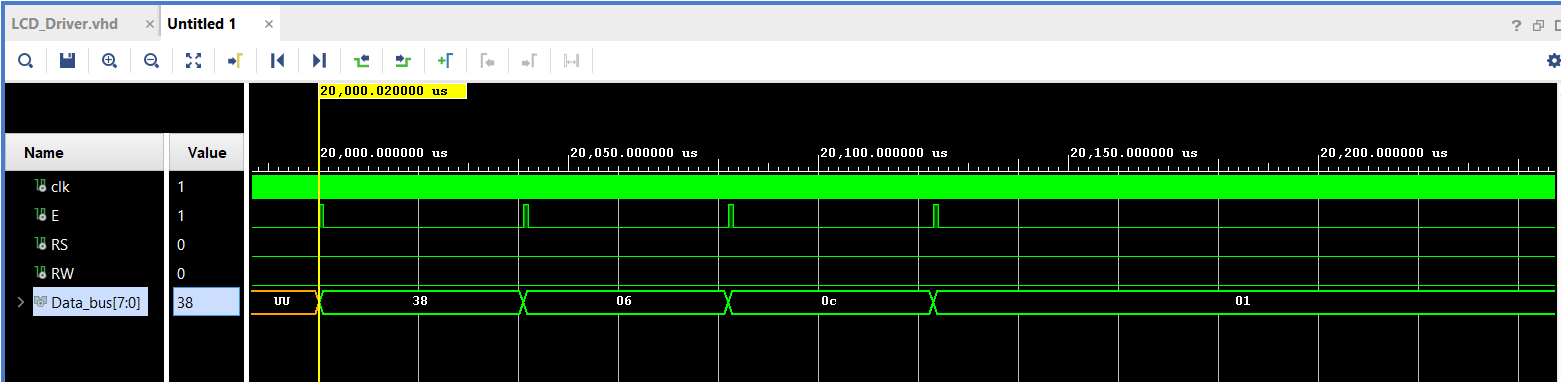
6. Wait for 40 μs (4000 clock cycles)

7. Finally, issue a Clear Display command. Allow at least 1.64 ms (164,000) clock

cycles).

VHDL code :



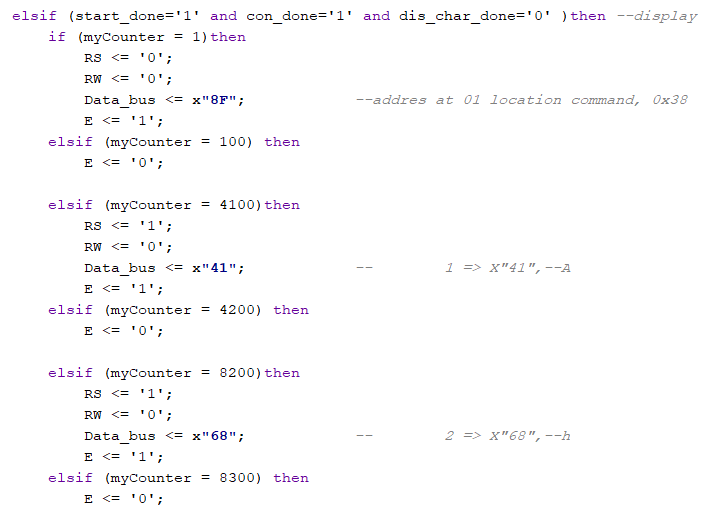
Simulation:

* Display :

In this stage I will send the commands that will print characters on LCD ,To show a character on the LCD, I have to:

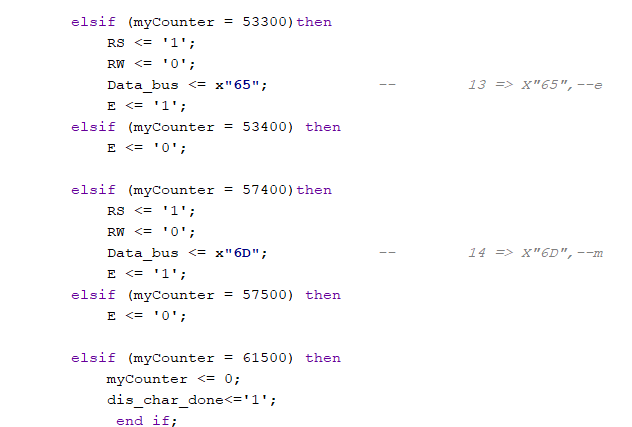
1. Specify the start address with a Set DD-RAM Address command.

2. Display a character with a Write Data command.

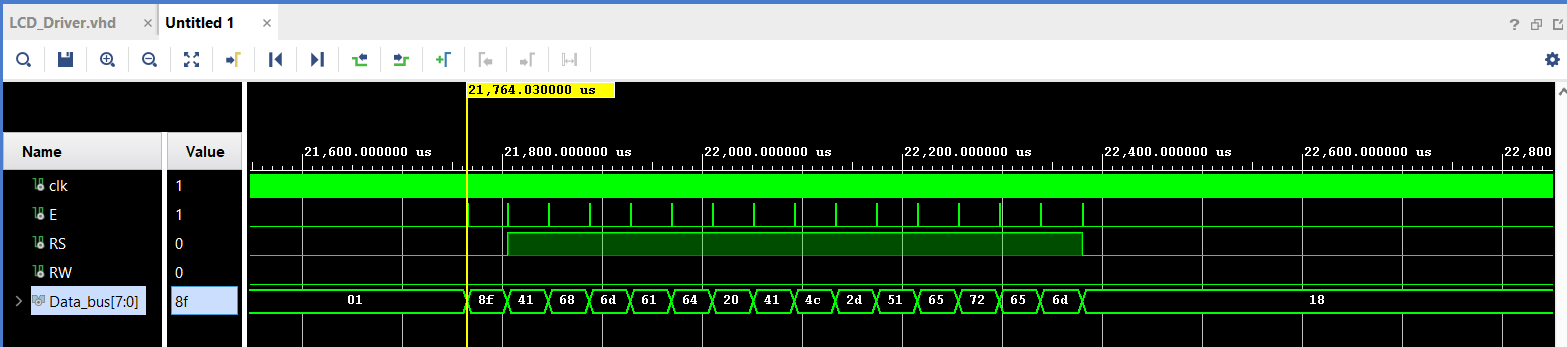
VHDL code:

**.**

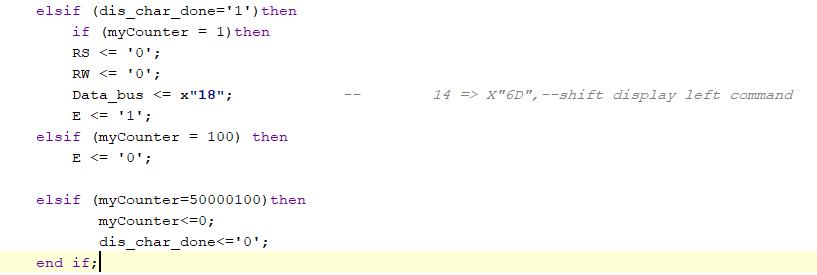
**.**

**.**

Simulation:

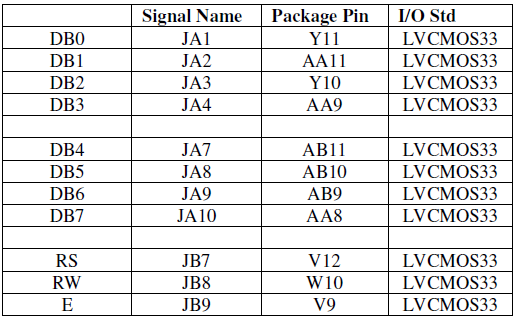
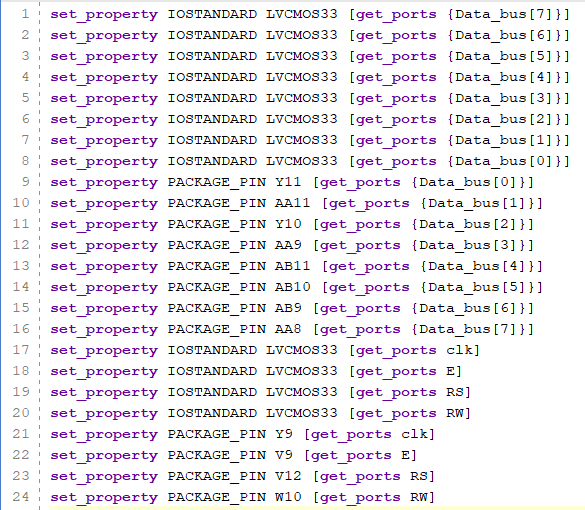


* Shift Display (bonus part ):

After write data in the memory in the LCD . I will shift display left every 1 s and show charters one by one using shift display left command (0x18)

**Synthesis, Implementation, and Bitstream Generation :**

I had connected Pins in the board as shown in the Table



And check the result :



**Part 2: Decimal counter from ( 00 to 99 ):**

Top level Entity LCD DRIVER :

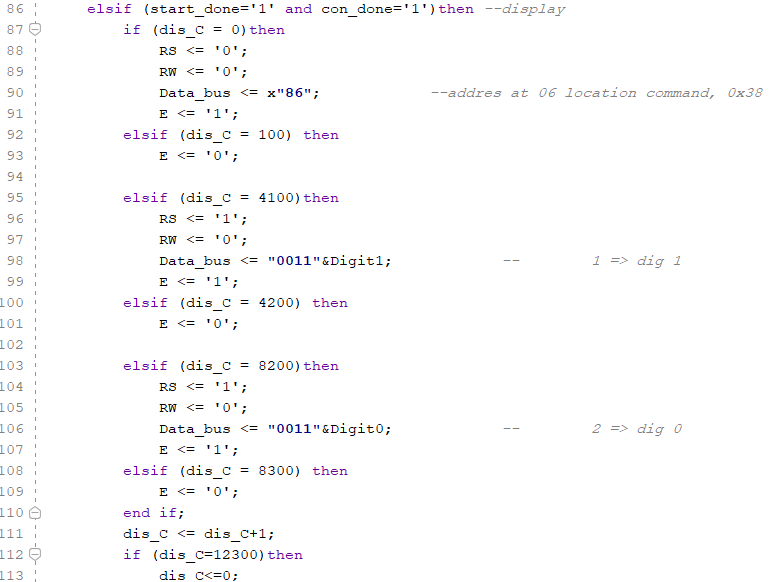
contain two component

1. Clk Divider.
2. Two digit BCD Counter.

* Startup : the same as the previous part .
* Configuration : also the same as the previous part.
* Display :

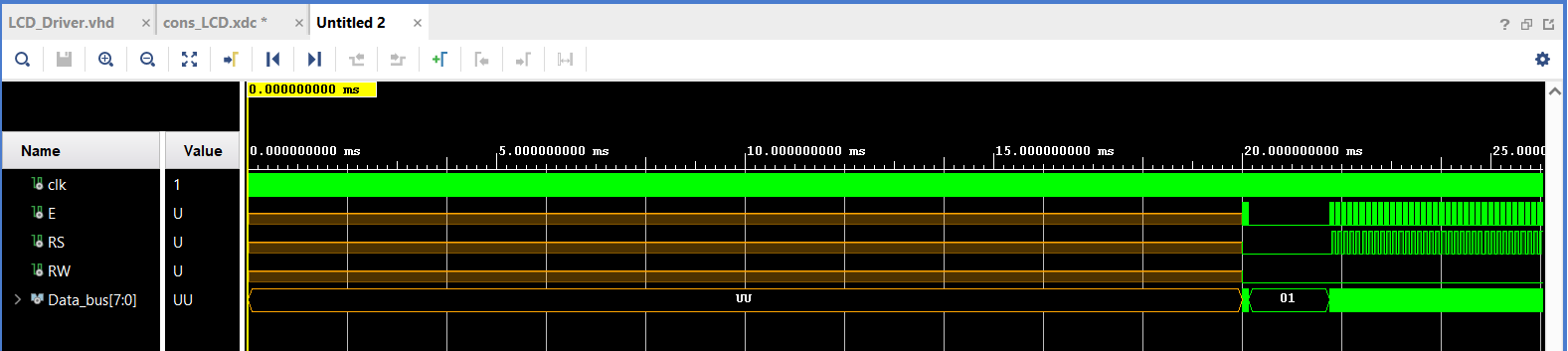
In this stage I selected the address to display the two digits and then I can easily create the number to display command by concatenated the output from the BCD counter with value " 3 " to become as : "3"+Digit 0 and "3"+Digit 1.

VHDL code :

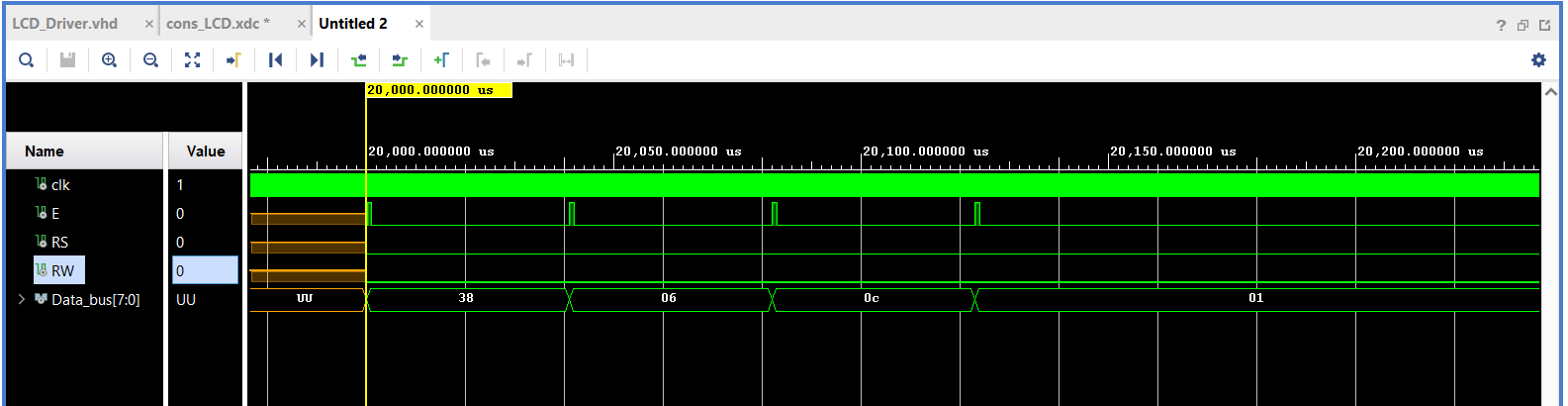


Simulation :

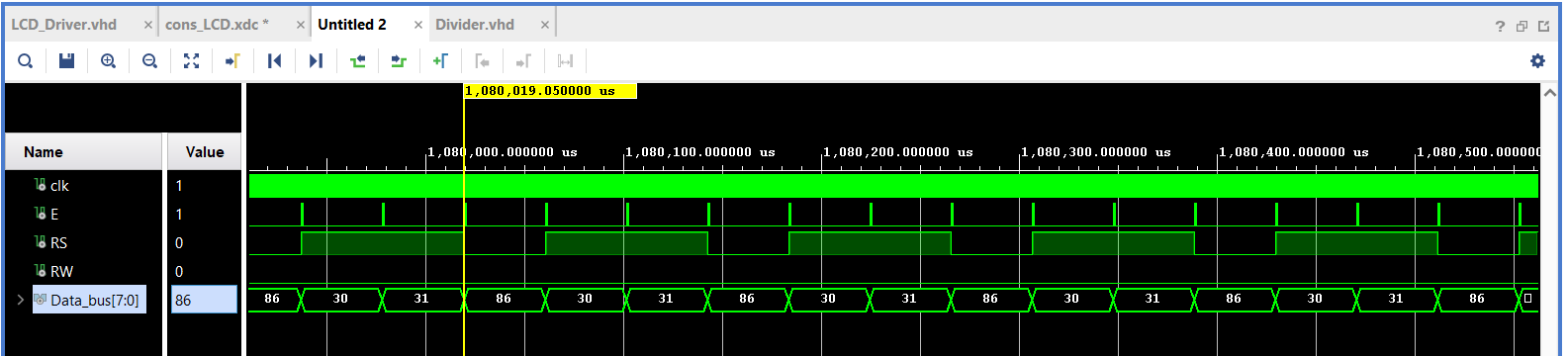
1. Wait for 20 ms:



1. Configuration commands :



1. Set address and display two digits :



**Synthesis, Implementation, and Bitstream Generation :**

I had connected Pins in the board as shown previous in the Table and tested the circuit .

**Conclusion:**

In this experiment, I learned how to deal with the LCD module attached with the

Zedboard. I also learned how an LCD Driver works by giving extremely

specifically timed commands that get the module working in the preferred mode.

I also learned how the RAMs and ROM present in the module operate to show data

on the screen.

In conclusion, the LCD module is an important part of a system for showing textual

information to the user, and one that can be easily integrated and programmed with

special commands and instructions.

**Source and References**:

* ZedBoard User’s Guide:

<http://zedboard.org/sites/default/files/documentations/ZedBoard_HW_UG_v2_2.pdf>

<https://reference.digilentinc.com/_media/reference/pmod/pmodclp/ks0066.pdf>

* Pmod CLP Reference Manual

<https://reference.digilentinc.com/reference/pmod/pmodclp/reference-manual>

* Programmable Logic Master User Constraints

<https://reference.digilentinc.com/_media/reference/pmod/pmodclp/ks0066.pdf>

<http://zedboard.org/sites/default/files/documentations/zedboard_master_XDC_RevC_D_v3.zip>

* stack overflow

<https://stackoverflow.com/>

* Youtube

<https://www.youtube.com/>