E E 552

Final Report

Text Messaging Centre

Instructor : Dr. Duncan Elliott
Group Member : Name : Ben Lee

Name : Alan Mak

Name : Wing Yee Chan

Name : Christina Kwok

Due Date : November 27, 2001

Declaration of original content signed by all group members

"The design elements of this project and report are entirely the original work of the authors and have not been submitted for credit in any other course except as follows:"

1. schematic in figure 9 was taken from reference [c]
2. lcd driver "lcddriver.vhd", “lcd.vhd” , and “lcd_pkg.vhd” were modified from reference [e] and taken from reference [g]
3. keyboard driver “key.vhd” and “keypush.vhd” were modified from [f]
4. supporting reference on FPGA board was taken from reference [i]
5. tmc_ram.vhd was modified from from reference [c]

Signature : -

Ben Lee, Alan Mak, Wing Yee Chan, Christina Kwok

Abstract

The design project was the text message center. It acted like a text network communication program (i.e. ICQ) in the sense that, a text message was sent from a remote device with the keyboard to the center through network transmission (which was not included as a part of the project), and stored at the center side after data compression. Having the center side to send the message desired from the memory after decompression, and display it on the LCD can retrieve the stored text. The system was tested and verified by both software and hardware simulations. This project overall is successful.

Table of Content

I. Achievements

This design is capable to record up to 8 compressed messages, which each message having a maximum of fifty characters each. Any message can be retrieved by a press of button, and display it on to the Liquid Crystal Display. All lower case letters and numbers, space, comma and period are valid keyboard input. Shift and enter key are used as function keys.

The system cannot delete messages as planned due to time constrain.

In summary:
-Store up to 8 messages
-Compress data from 8 bits to 5 bits by pattern matching technique
-Onboard EBA ram is used to store the compressed data
-Keyboard as input, LCD as output
-All lower case letters and numbers, space, comma, and period are accepted as keyboard inputs
-Not able to delete messages

II. Description of Operation

a. The Interface:

The interface consisted of an ordinary keyboard for text input, and a LCD for text display. The driver for the keyboard could read the serial input, including a starting bit and a stopping bit, and converted it into standard ASCII codes. The LCD driver simply drove the LCD to output characters on the LCD, from 8-bit ASCII coded input and a data ready signal. It also responsible for other LCD related operation, such as clear the screen and returned the cursor to the beginning when the cursor hit the end.

b. The de/compression:

For ordinary ASCII representation, each word would be represented by 8-bit. For the design of the simple algorithm, 16 characters that would use most frequently had been chosen to be represented by 5-bit. The first bit would be used to indicate it for the compressed bit. In our design, “1” had been used to indicate it was a compressed group, and “0” would be for normal group. Also, the normal group, each character would be represented by two 5-bit strings, and each will start with “0”. Handshaking signals with the memory were implemented, so the de/compression unit and the memory could be in phase. Also the compression will be operating as function decoder for the read and delete of the system.

c. The memory:

Using the EABs onboard and lpm_counter was used to count for the addresses that control the memory board. The size of the memory was taken up 5-bit per character, 1024 memory blocks for 8 messages storage. Each messages contained a maximum of 50 characters. After compression, the worse case would be 10-bit per character, with 50 characters. That is equivalent to taking up 100 spaces per message.

d. Hand-shaking:

It is critical to use handshaking signal to operation, since the system will be using two sub-clock period, ms for keyboard, and ns for the regular operation. On the input operation to store into ram, the master part would be ram since the compression is an intermediate state, and the keyboard is running in ms clock cycle. The process will be for ram component is signal a input ready signal to the compression engine, and the engine will begin feeding data to the ram one at time depend on the output ready generate by the ram component. Also, since the keyboard is operating in ms,

which mean it will generate a long trigger pulse for the compression, as a result, the compression engine is design to begin to operate after the input valid is finish in order to avoid multiple input to the ram.

For the decompression part, the process will be operation in similar base, but when the decompression is output the LCD display, the trigger pulse will design to last long enough to suit for display which operating in ms. The length of the trigger pulse will last until for the display's input ready is low, which mean the display is retrieve the data already, and the decompression will be ready for the next data.

III. **Design Details**

The design of the compression and decompression could basically separate into two parts: compression & decompression, and the handshaking state.

a. The compression / decompression:

For ordinary ASCII representation, each word would be represented by 8-bit. For the design of the simple algorithm, 16 characters that would use most frequently were represented by 5-bit. The first bit would be used to indicate it for the compressed bit. In our design, "1" had been used to indicate it was a compressed group, and "0" would be for normal group. Also, the normal group, each character would be represented by two 5-bit strings, and each would start with "0". Also the compression would be operating as a function decoder for the read and delete for the whole system. For the whole compression and decompression, it would be based on the state machine, inside the state machine, it would use case statement to sort out the 16 chosen case and output, also, for the not chose type input, compression would separate the input bit into two 4 bit data with extend bit of '0' of the data and output them into three state, the intermediate state were to be test when the ram component was ready for the next input.

For the decompression part, the process would be operation in similar base, but when the decompression was output the LCD display, the trigger pulse would design to last long enough to suit for display which operating in ms. The length of the trigger pulse would last until for the display's input ready was low, which meant the display was to retrieve the data already, and the decompression would be ready for the next data. Also, one thing was different of the compression logic was that it had to reset the LCD display when it began to read the message coming out of the ram base.

b. Hand-shaking:

It was critical to use handshaking signal to operation, since the system would be using two sub-clock range, millisecond for keyboard, and nanosecond for the internal operation. On the input operation from keyboard to store into ram, the master part would be ram since the compression was

an intermediate state, and the keyboard was running in millisecond clock cycle. The process would be for ram component to give a input ready signal to the compression engine, and the engine would begin feeding data to the ram, one at time depended on the output ready generated by the ram component. Also, since the keyboard was operating in millisecond range, which meant it would generate a long trigger pulse for the compression. As a result, the compression engine was designed to start the operation after the input valid signal had been finished, in which to avoid multiple inputs to the ram.

IV. **Datasheet for FPGA**

Tmc datasheet

Text Message Center

Description

This is a text network communication program. A text message was sent from a remote device with the keyboard to the center through network, and stored at the center side after data compression. Having the center side to send the message desired from the memory after decompression, and display it on the LCD can retrieve the stored text.

Features

-Keyboard Input
-LCD Output
-Pattern matching de/compression algorithm
-Maximum Clock Speed of 16.4MHz
-851 of total 1152 Logic Cells
-5V power supply

Pin Descriptions

keyclock	Input	66	Flex Expansion A - Hole 30
Keyboard	Input	67	Flex Expansion A - Hole 31
dataout0	Output	51	Flex Expasion A - Hole 20
dataout1	Output	53	Flex Expasion A - Hole 21
dataout2	Output	54	Flex Expasion A - Hole 22
dataout3	Output	55	Flex Expasion A - Hole 23

			Flex Expansion A - Hole
dataout4	Output	56	24
			Flex Expansion A - Hole
dataout5	Output	61	25
			Flex Expansion A - Hole
dataout6	Output	62	26
			Flex Expansion A - Hole
dataout7	Output	63	27
			Flex Expansion A - Hole
dataoutvalid	Output	64	28
Select	Input	28	Push Button 1

Figure 1 : Detailed FPGA user IO signal

V. Estimate / measure of number of FPGA logic blocks

All the value were based on measurement from MaxPlus II

Vhdl program	clock	Frequency	LCs used	LC utilized
Tmc	61.0ns	16.4MHz	851	73.9%

Figure 2: FPGA logic blocks, clock and frequency

EAB used: 3 /6 (50%)
Embedded cells used 20/48 (41%)
Memory bits 5120
Memory Utilized: 41%

V. Results of experiments and characterization

Compression:

On the design of the compression engine, the pattern matching coding was used. There are two method including in the pattern coding, auto set pattern, and pre-set pattern coding. By inspection, auto generate coding can done a better compression (approx. 73%), but in a trade off for the programming size. On the other hand, pre-set coding will be a simpler program but in a less compression rate. In this case, C++ is used to analysis the use of the pre-set coding. Result of the C++ base on a 16 pattern matching over a set of sentence extract from a commercial is around 70-90%, the average of an alphabet base text is around 75%. Since the project is a alphabet character base, pre-set coding is well do the job.

On the design of VHDL code, a state machine is used for implement a de/compression engine. First an IF statement is used while in the state machine for the pre-set pattern, the program can compile without error and warning, but in the simulation, the program can produce a correct result in first couple if statement and producing same result on the later if statement, cause of this result is un-found. Lately, a experiment case on a Case statement is used to implement for the pre-set pattern, the program can compile and simulate perfectly, while the speed and the used of logic cell remain unchanged.

VI. References

a) Title : http://www.altera.com/literature/ds/dsf10k.pdf

-- the website address was the title as well. The EAB datasheets have been used as a reference of the EAB

b) Title : Implementing Built in RAM-Function

http://www.ee.ualberta.ca/~elliott/ee552/studentAppNotes/2000f/vhdl/lpm_ram/Appnotes.htm

-- the reference on building and constructing the dq-ram.vhd

c) Title : Pipelining and Testbenches

http://www.ee.ualberta.ca/~elliott/ee552/labs/lab5/lab5.html

-- the modification of the ram.vhd to the TMC_RAM.vhd

d) Text book from EE 350 – Microelectronic Circuit – Sedra and Smith

-- the op-amp circuit had been used as a reference in building the RF circuit.

c) Text book from EE315 – Field and Wave Electromagnetics –Cheng

-- for the reference of the antennas

d) Text book from EE390 – Modern Digital and Analog Communication System – Lathi

-- the filter from this text book had been used as a reference in building the RF circuit

e) http://www.ee.ualberta.ca/~elliott/ee552/studentAppNotes/2000_w/interfacing/lcd_driver/lcd.htm

-- this website had been used as a reference in building the LCD driver

f) http://www.beyondogic.org

-- this website had been used as a reference in building the keyboard.vhd

g) http://seiko_usa_ecd.com/lcd.products/char_mods/1167200j000.html

-- this website provides the datasheets for the LCD.

h) MC68000 Programming Reference Card

-- look up the ASCII table

i) Title :Seiko LCD : Character Modules

http://www.seiko-usa-ecd.com/lcd/products/char_mods/

-- this website had been used as a reference of building the LCD driver

VIII. Datasheets for chip used

Please see the attachment.

IX. Index to test cases / verification

a) Compression Simulation

- Case use to test to output data to see if it function properly.

b) Decompression Simulation

- Case use to test to output data to see if it function properly.

c) MsgCounter Simulation

- Case use to test the address counter function properly.

- Case use to test the message counter function properly.

- Retrieve the proper message as required from the user.

d) tmc_ram simulation

- Case use the lpm-ram-dq to have proper input to the input register

X. **Design verification**

a) Compression simulation

For the compression simulation, there are several section of the compression has to be function to work properly, first, the continuality of the data in different form (set or non-set), second is the data_out_valid correspond to the right data, and third is the functionality of the compression. Which mean, the compression engine is outputting the correct compressed data. Based on the above criteria, two-test cases are performing to test on the above three sections.

Case 1)

The input for this case is a character "a" for hex "61", since this the set pattern for the compression engine, the output will binary "10000" with the data_out_valid set high for two clock edge.

Case 2)

The input for this case would be a non-set data hex "23", for output in this case, it will separate in two 5 bit binary number, in this case, two 4 bit binary number begin with a add on bit "0". Which mean b"0"&h"2", and b"0"&h"3".

Case 3)

The input for this case would test for extend input pulse generate by the keyboard.

By the analysis of the simulation, the above two is perform as predicted, in the first section, the continuouality are solid ok since the two input data are performed. The second and third sections are good as indicate by the output waveform of the simulation.

b) Decompression simulation

For the decompression simulation, there are several section of the compression has to be function to work properly. First, the continuity of the data in different form (set or non-set), second is the data_out_valid correspond to the right data, and third is the functionality of the compression, which mean the is the compression engine the correct compressed data. Based on the above criteria, two test cases are performing to test on the above three sections.

Case 1)

The input for this case would a set pattern in binary "10101", since this is a set pattern, the output would be h"72" for the character "r".

Case 2)

The input for this case would be a non-set pattern, which two 5-bit data begin with zero bit. 1111 and 0001, the two data will be hold for a while to test the waiting state over the decompression engine toward the non-set input. For the above input, the output would be h"F1".

Case 3)
The input for this case would test for extend output pulse generate by decompression to the LCD display.

By analysis of the simulation, the above test case is performing correctly, the continuity, functionality, waiting state, and data_out_valid and done signal are perform correctly as predicted.

c) MsgCounter Simulation

Using the timing analysis from MaxPlusII, the MsgCounter.vhd does not lose any indata value as seen from the simulation graph. Also, the address counter and the message counter can increment the counter after reset.
The message can be stored and retrieved when the message number had been required from the user, the message could then pass to the decompression part.

d) The C++ program and the simulation result had been attached for reference. From the result of the simulation, the compression ratio would be around 70%~90%. The worst case from the result was 89.62%, and the best case from the result was 72.92%. The compression was tested upon passing a string into the C++ program, and checks the size before and after the compression. Thus, getting the value for the compression ratio.

e) Keypush

Keypush is used to determine the output from the keyboard. F0 is sent whenever a key is release, the output which signal by the F0 character will be the main purpose of this simulation.

Case 1)
Input: H”1C”
Keyoutput (scancode): H”1C”
Key_out (ASCII code): H”61”

Case 2)
Input: H”2E”
Keyoutput (scancode): H”2E”
Key_out (ASCII code): H”35”

By analyze the waveform; the key controller is function properly.

f) LCDDriver

Lcddriver is simulate the time for initialize the lcd, and output the ready signal for the actual LCD display.

The first waveform shows the time required to initialize the LCD display.
Once the LCD display is initializing, it needs to clear the screen, and then it starts the normal operation.
For initialization, the lcd_data that sends to the lcd is 0x38 (Hex).
The second waveforms shows that once the initialization is complete, it enters the entrymode, giving lcd_data=0x06, then it enters the display on mode, giving lcd_data=0x0E, and the last step before normal operation is the clear mode, and it gives lcd_data=0x01.
The third waveform shows that when the screen is clear, it enters the address set mode, and then the lcd is ready for displaying the character.

Case1)
Message: H”36”
Data: H”36”

Case2)
Message: H”22”
Data: H”22”

By analyze the waveform, the lcddriver shows that the time required for initialization is 400 clock edge (approx 24ms), two clock edge for entrymode and display_on mode, and 30 clock edge for clearing screen, and then two clock edge for address set, and then it displays the corresponding output.

g) LCD
This code masks all the component together that used for final pin assignment, since all the parts component has been simulated, as a result, only two test cases were perform to verify the functionality of the project interface.
Case 1)
Message: b”01100001”
Data: b”01100001”
Case 2)
Message: b”00011100”
Data: b”00011100”

h) TMC (final)
Since the final combine system is come with different clock range, it is omit the LCD display and keyboard component, as a result, the test case of the final system would base on the compression,

decompression, and ram component.

Case 1)

Input data will be one of each chosen and non-chosen type, and follow by end message. Next thing is the read message and the message number.

Result) By analysis of the output waveform, the system are able to compress, store, and decompress correctly.

XI. Diagram of design hierarchy

The design hierarchy had been attached on the separate page, and all the vhdl had been tested and simulated with Timing analysis with no known bugs.

XII. Index to VHDL code

All the vhd file had been compile and simulated with no known bugs

- tmc.vhd -- entity tmc----------Pg
- key.vhd -- entity key----------Pg
  - keypush.vhd -- entity keypush ----------Pg
  - keyAsCII.vhd – entity keyAsCII----------Pg
  - compression.vhd -- entity compression ---------- Pg
  - compress engine which compress 8 bit data to either 5 bit or 10 bit data depend if it is pre-set data or not.
  - decompression.vhd -- entity decompression ---------- Pg
  - decompress design which decompress either 5 bit or 10 bit data to 8 bit output depend if it is pre-set data or not.
  - MsgCounter.vhd -- entity MsgCounter ----------Pg
    - use to count the message and the address and to store and
    - retrieve the message
  - tmc_ram.vhd ---- entity tmc_ram ----------Pg
- transform the EAB board into the configuration that needed for our project memory.
  - tmc_ram_pkg.vhd ---- entity tmc_ram_pkg ----------Pg

  - counter_lcd.vhd -- entity counter_lcd ----------Pg
  - lcd.vhd – entity lcd----------Pg
  - lcd_driver.vhd – entity lcddriver---------- -Pg
  - lcd_pkg.vhd -- entity lcd_pkg----------Pg

XIII. **VHDL design**

All the vhdl program listed above under session XII and the corresponding simulated timing analysis waveform had been attached.

XIV. **Test bench index (with descriptions) and test benches**

- testbench.vhd ----- entity testbench----------Pg 42-43
- compression unit using mentor graphic program to do the software simulation.

Testbench simulation

For the testbench simulation, since this part is perform as required indicate, and most of the simulation is done over maxplus2, there is only one test case perform by indicate the ability to the use of the testbench. The part, which uses the mentor graphic, is the compression unit.

Case 1)

The test case will be compression a set pattern binary input b"01100001", and the corresponding output would be a set output b"10000".

The testbench are simulated and the results are match with predicted.

XV. **Schematics**

XVI. Self evaluation of group

In our group, everybody tried their part, as originally there are four parts for this project. But one of the part is totally omitted due to time constrain, and that member had helped out other group members. Therefore everyone in the group should receive the mark equally.