Take Home Final Examination

Assigned: May 12, 2021 Due: 11:59pm Tuesday May 18, 2021

The work on this examination is to be your own and you are expected to adhere to the UMass-Boston honor system. All questions can be answered by one or two short sentences. Do not try to make up for a lack of understanding by providing a rambling answer.

**Note: I give partial credit! Show all work!**

**1. (30 points) Short Questions:**

1. (3 points) The counter of the PIT is always changing. What hardware mechanism does the PIT provide to ensure that we are reading the right values?

Programmable interrupt controller

1. (6 points) Name 2 functions you can program the Programmable Interrupt Controller (PIC) to do.
2. \_\_\_\_ \_\_\_\_\_\_\_\_timer

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. \_\_\_\_\_count the number of interrupt requests\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. (6 points) Name two differences between L1 and L2 cache?
2. \_\_\_\_L1 is faster, L2 is slower \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. \_\_\_L1 is smaller, L2 is larger\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

d. (9 points) The output of sequential logic is based on:

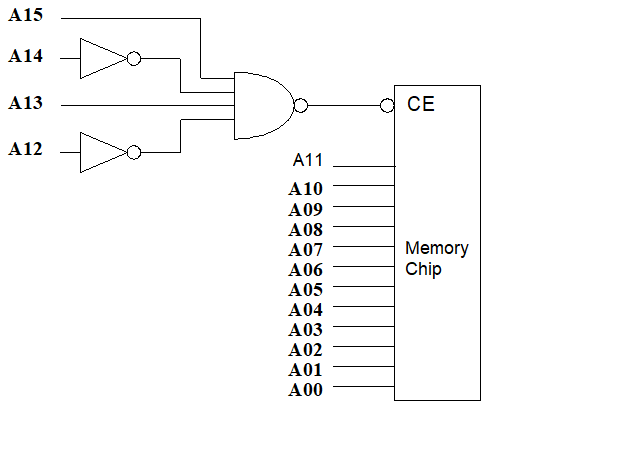
1. \_\_\_\_previous inputs,
2. \_\_\_order of inputs\_\_, and
3. \_\_\_\_\_current inputs\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

e. (6 points) Explain how the iret instruction differs from the ret instruction and explain where you would use each of them?

IRET is used to exit from an interrupt procedure and should only be used when dealing with interrupts

**2. (20 points) Combinational Logic:**

The design of a 16-bit address decoder is shown below:



The memory chip is enabled when CE =1. Use a truth table to determine what addressing bits turn on the chip:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A15 | A14 | A13 | A12 | CE |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

What range of memory addresses will be selected by this decoder:

1010000000000000 – 1010111111111111

Hex address range: 0xA000 to 0xAFFF\_\_\_\_\_\_\_\_\_\_

**3. (20 points) Timer Programming:**

Loading the PIT counter in mp4/mp5 with a value of 30000 will generate an interrupt every 25 msec.

(5 points) what is the counter value to generate an interrupt every 50 msec?

\_\_\_\_\_\_\_5536\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(5 points) what is the counter value to generate an interrupt every 75 msec?

\_\_\_\_\_\_71072\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(10 points) Write a set\_timer\_count() function below with the count value together with an ISR in C so that the PIT will call an output() function every 75 msec:

**4. (30 points) Assembly language program**

Write a C callable assembly language program

(google.s) to find out how many months google’s stock (goog) price is above a certain number in 2020.

The function prototype of the assembly language function in C is shown as:

extern int stock(int number);

Your assembly language function should get the dollar number from a C main function shown below and return the number of months that the stock price is equal or above the entered number. The program should exit when it reads the stop code 0.

------------------------------------------------------

/\* googlec.c: C driver for the stock analysis

function

\*/

#include <stdio.h>

extern int google(int number);

int main()

{

int n, number;

printf("Enter the stock price to compare: \n");

scanf("%d", &number);

n = google(number);

printf("Number of months that goog is equal or over the price of ", %d, "in 2020 is : %d\n", number, n);

return 0;

}

---------------------------------------------------------

----------------------------------------------------------

# GOOG Stock price analysis assembly program

# google.s

#

.globl google

.data

price: .long 1260 #11/1/19 price

.long 1289 #12/1/19 price

.long 1337 #1/1/20 price

.long 1434 #2/1/20 price

.long 1389 #3/1/20 price

.long 1105 #4/1/20 price

.long 1320 #5/1/20 price

.long 1431 #6/1/20 price

.long 1438 #7/1/20 price

.long 1482 #8/1/20 price

.long 1634 #9/1/20 price

.long 1490 #10/1/20 price

.long 0 #stop code

.text

google:

.end

---------------------------------------------------------

Instructions:

1. Copy the files you need from /courses/cs341/s21/cheungr/final\_exam/.
2. Edit the google.s to include your code. Use the provided make file to build your google.lnx program:

make A=google

c) Run and debug your program in the VMs. Create a typescript of the run of the lnx file in the VM. scp it to your cs341/final\_exam/ folder at uses.cs.umb.edu for grading.

**Please Note:**

1. Only use data in 2020 for comparison
2. Include the C stack frame in your assembly code.