Introduction to Computer Systems

COMP-273

Assignment #3

Due: April 11, 2017 on myCourses at 23:30

Question 1: How many days (2 points)

Write a MIPS program using the MARS interpreter. The program calculates the number of days someone has been alive given their age, assuming that each year is exactly 365 days. This program must compute the number of days using the ADD instruction and a loop, it cannot use the multiply instruction.

Assume in the .DATA section is a label called YEARS that is .WORD and a label called DAYS which is also a .WORD type. Assign a number to YEARS in your source file. Your program uses the number stored in YEARS, computes the number of days and saves the result in DAYS. The program then terminates.

This program does not output anything on the screen. The user will have to look into memory to see if the computation was carried out correctly.

Question 2: Factorial (6 points)

Write a MIPS program using the MARS interpreter. The program has a main program and a function (subroutine) that computes the factorial recursively. The subroutine's name is FACT. The main program prompts the user for an integer number. It then passes that number to the subroutine FACT using the ANSI C convention. FACT recursively computes the factorial of the number entered by the user. FACT returns the result to the main program. The main program then displays the result to the screen. The program then terminates.

Your program will display the following output:

```
Calculate Factorial

Enter a number: 3

The factorial is 6

END
```

Question 3: Peripheral Programming (7 points)

SPIM only simulates two hardware devices: the keyboard and the text-screen. These are both referred to as the console I/O devices.

In MIPS programming we have the following conventions: a MIPS subroutine can be implemented in two ways, as a driver or as a function. If a MIPS subroutine is implemented as a driver then it does not use the run-time stack but uses only the register convention for subroutine access. In other words a0-a4 for parameter passing and v0-v1 for returning values. It only uses the run-time stack for saving the s registers. If a MIPS subroutine is implemented as a function then it follows the ANSI C function calling convention then all parameters, local variables, and saving registers are placed on the run-time stack, and returned values are placed in vo-v1. You must use the above conventions when writing the answer to this question.

Write a MIPS program that asks the user to enter their first and then last name. The program then displays: "You entered: <last name>, <first name>". Where <first name> and <last name> are the words they inputted. The program then stops. Your program must adhere to the following instructions:

- Create a driver called GETCHAR: char getchar(void) in MIPS, that interfaces with the keyboard's status and data registers. This driver when called returns the character in the keyboard buffer to register $V0. Do not use the OS syscall command. Access the peripheral directly. A slide in class was given to help you. Implement this program as a driver using the driver polling flowchart we discussed in class.

- Create a driver called PUTCHAR: void putchar(char c) in MIPS, that interfaces with the screen's (console) status and data registers. This driver when called assumes that register $A0 contains the character that needs to be output. Only one character is output. Do not use the OS libraries. Access the peripheral directly. A slide in class was given to help you. Implement this program as a driver using the driver polling flowchart we discussed in class.

- To test this, you will build the following functions in MIPS:
  4. int gets(char *buffer, int limit)
     int puts(char *buffer)

     The function GETS reads (and PUTS writes) ASCII from the keyboard into a memory space pointed to by BUFFER (from BUFFER to the screen for PUTS). It stops reading when the user either presses the enter key or when LIMIT characters is reached (stops printing when NULL is found for PUTS). The string is terminated with a '\0' (if there is space). The function also returns the number of characters it actually read (outputted for PUTS) into $v0. The function GETS must use GETCHAR and the function PUTS must use PUTCHAR.
  5. The function gets() and puts() are used to read and write the user's name, and any other string based input or output, from the main program.

QUESTION 4: Performance Issues [5 points]

Assume we have a hard drive that can operate in both polling or interrupt mode. Assume further that the disk drive can access data in either block or byte mode. Block mode uses the disk drive's internal buffer to store 10K bytes of data. In block mode the hard drive can run on its own after receiving the address on disk and the number of bytes to read from the CPU. All of these bytes are loaded into the buffer. If the number of bytes to read is greater than the size of the buffer or if all the requested bytes have been read, an interrupt is sent to the CPU once the buffer is full or once the operation is completed. The CPU then needs to download the buffer to RAM, clear the buffer, and instruct the drive to continue reading (if needed). Byte mode simply downloads a single byte of data from the hard disk given an address on disk to the first byte in the disk drive buffer. No interrupt is sent. It is up to the CPU to know when to extract that byte from the buffer using polling. The disk drive has a status register that is set to integer 1 when the drive is busy, 0 when it is not busy and no data is in the buffer, 2 when it is not busy but a single byte is in the buffer, and 3 if it is not busy with a full buffer.

Assume that polling takes 200 ticks, while interrupts takes 500 ticks. Assume we want to copy a one meg file from disk to RAM. Assume further that all other assembler instructions take only 1 tick to execute. Assume you have a 500MHz processor.

Answer the following questions:

- How long will it take for polling to load the file into RAM using byte mode?
- How long will it take for polling to load the file into RAM using block mode?
- How long will it take for interrupts to load the file into RAM in byte mode?
- How long will it take for interrupts to load the file into RAM in block mode?
- Compare the impact in percentage of processing time for the above 4 calculation
  - Where does polling loose all it time compared to where an interrupt looses all its time?
- Assuming that DMA has an overhead of 1000 ticks and can transfer the entire file:
  - Calculate the number of ticks it will take to load the file into RAM.
  - Calculate the impact on percentage of processing time.
  - Where does the DMA loose all its time?
  - How does DMA compare with polling and interrupts?
- Why do we still have these three methods of accessing peripherals?
- State a situation where polling, interrupts and DMA can be applied appropriately.

ASSIGNMENT SUBMISSION INSTRUCTIONS

- Submit your solution to myCourses before the due date
- 5% is removed for each day late up to 2 days at which time the submission box closes
- Please submit your assignment in electronic form:
    - Submit the MIPS source files executable by the TA using MARS
    - Written questions in PDF, RTF or TXT file formats.
- You must work on your own. A grade of zero will be applied to copied or shared solutions.

GRADING INSTRUCTIONS

- This assignment is worth a total of 20 points
- The breakdown is stated on each question
- Each question is graded proportionally. In other words, if your question is 50% correct you get 50% of the marks.