CompSci 210 Computer Systems 1 Mini-assignment 3

Due Date: 12 Noon, Friday, May 11, 2007

1. Write an Alpha assembly language programme that reads in an integer *n* and prints out factorial(*n*).

Write an Alpha assembly language programme that performs the following tasks:

- Prints out your name, UPI, student ID, then the text "CompSci 210 2007 Semester 2 Assignment 3". For example,

  ```
  James Goodman, jgoo052, ID xxxxxxx CompSci 210 2007 Semester 2 Assignment 3
  ```

  (Change the name, UPI, and ID to your own).

- Prints a message requesting input of an integer and specifying the maximum permissible size.

- Reads in the line of text, interprets an initial string of digits terminated by white space (spaces, tabs, and newlines) as an integer, discards the rest of the string.

- Computes the value of the factorial for that integer.

- Prints out a message including the original number and its factorial and exits.

Sample Output

```
Please enter an integer less than (specify):
10
Factorial(10) = 3628800
```

Requirements

- Use the same simulation environment for running the Alpha simulator as in the previous assignment.

- Write an assembly language programme for the Alpha to behave as described above. The programme must be documented so that it is easy to understand.

- Your programme must solve this problem recursively. That is, you should write a function that returns the correct value directly for n=1, but calls itself with an argument of n - 1 for n > 1. When the called function returns a value, the calling function multiplies the returned result by n and returns the product as its result.

  ```
  int fact(int n) {
        if (n==1) return (1);
        return(n * fact(n-1));
        }
  ```

- You may use the library functions in block IO (newline(), print(), readLine(), and printf(). You should initially get your programme running using these functions. For full credit, you should use only the library functions in block Sys. You can receive a maximum of 96% if your assignment uses functions in block IO. (*Note: get your programme working correctly first! Using only block IO is a lot more work for a tiny bit of additional credit.*)

- To receive 100% credit your programme also must handle all input correctly and exit properly,

indicating when an input is not understood or otherwise illegal. Your code should indicate the largest integer for which the factorial can be computed correctly (the larger the better).

- Your main programme should do nothing but assign arguments and call functions. All other code, e.g., code to read and interpret input and generate output, must be in separate, properly named functions. Functions must follow the conventions described in section 11.3 of Bruce Hutton's notes.

Marking

Must satisfy assignment requirements. Generally well written, and easy to understand. Documented by comments explaining high-level actions. Uses meaningful identifiers for variables and labels. Appropriate use of registers and calling conventions.

2. Assemble an Alpha programme into Alpha machine code.

Assemble the following code and data segments. The data begins at location 0x0150 0000 and the code segment begins at location 0x0085 0000.

```
import "../IMPORT/callsys.h";
data {
     align quad;
     string:
     asciiz "Message:\n";
     align quad;
     readbuffer:    byte[100];
     } data


code {
     public enter:
        ldiq $s2, readbuffer;
     loop:
        ldbu $a0, ($s1);
        beq $a0, endstring;
        ldiq $a0, CALLSYS_PUTCHAR;
        call_pal CALL_PAL_CALLSYS;
        addq $s1, 1;
        br loop;
     endstring:
        cmpeq $v0, '\n', $t1;
        blbs $t1, loop;
        stb $v0, ($s1);
        addq $s1, 1, $s1;
     } code
```

Requirements

- Your assembly should should show for each address affected (1) the address (in hex), (2) the instruction or data assembled into that location. You should also show the status of the Symbol Table on completion.

Submit

1. An Alpha programme, A3.user.s, compatible with the project provided from the web.
2. A .txt, .rtf, .doc, or .pdf file showing the layout of memory for the assembled instructions of part 2.

Submit electronically using the assignment drop box.

Due Date: 12 Noon, Friday, May 11, 2007