University of Puerto Rico – Mayagüez Campus
School of Engineering

INEL 4206 – Microprocessors

Problem Set 3
Due: April 22, 2003

This problem set is designed to give you practice and test your knowledge and skills in MIPS assembly language programming. You are given three problems of increasing difficulty, each has an associated template (assembly file) and a reference file (C File), to aide you in your solution to the problem set. Read carefully all instructions before submitting your solution.

Problem 1: Procedure - X^Y

Provide a MIPS assembly language program that includes a function called *power*. The function takes two positive integer parameters x and y, and returns the result of the mathematical operation X^Y (X to the Y^{th} power).

Template: p1.asm
Reference p1.c

Problem 2: Procedure – Polynomial Solution

A polynomial is stored in memory in an array of integer numbers (at location labeled *polynomial*). Each number represents the integer coefficient a single monomial. The offset index of that number indicates the power of the variable. In other words the number stored at A[P] is the coefficient of the monomial with the variable x^p. The following example illustrates the polynomial associated with a C language definition of an array called polynomial.

int polynomial[5]={3,4,6,8,1} → $x^4 + 8x^3 + 6x^2 + 4x^1 + 3x^0$

Provide a MIPS assembly program that includes a function called *solve* that receives three parameters: an integer parameter *x*, the address *poly* of the first cell of the coefficient array, and the number *n* of terms in the polynomial. The function iteratively evaluates the polynomial and returns the value of poly(x).

Template: p2.asm
Reference p2.c

Problem 3: Recursion - Horner's Method for Polynomial Evaluation

Horner's Method is a way to optimize the task of evaluating a polynomial. The method splits the polynomial into its individual terms and solves them incrementally. This method separates the lowest degree term in the polynomial from the rest, by grouping any terms with a higher degree into one unit, with degree 1, thus making any polynomial K*x+C, where K is the group of all terms with a degree higher than one, and C is the term with degree 0. When the terms are grouped into K, their degrees decrease by one, as follows.

$$x^4 + 8x^3 + 6x^2 + 4x^1 + 3x^0 \rightarrow K*x+ 3 \quad \text{where } K = (x^3 + 8x^2 + 6x^1 + 4)$$

This method can then be applied recursively inside the K to simplify the polynomial that results after grouping, by repeating these steps until there are no terms with a degree higher than 1 and will yield an equation where there is no need to calculate x^n for any term. After applying this method to the sample polynomial we end up with:

$$x^4 + 8x^3 + 6x^2 + 4x^1 + 3x^0 \rightarrow ((((1)*x + 8)*x + 6)*x + 4)*x + 3$$

For problem 3 you must provide a MIPS assembly language function called *horner* that receives four parameters: the value of the variable *x*, the index *i* of the highest degree of the polynomial or subpolynomial to be evaluated by that instance of the function call, the maximum index *max* of the polynomial, and the address *poly* of the first element of the coefficient array. Thus a function call to *horner* would look like *horner(x, i , max, poly)*. The first time *horner* is called *i* should be 0, since you are starting from the first term. When you reach the maximum index n-1, the function simply returns the value the coefficient of the corresponding term.

Template: p3.asm
Reference: p3.c

FILES

File you receive: ***ps3pkg.tar.gz***

This is a compressed file, which you have to decompress using the command, "`tar -xzf ps3pkg.tar.gz`" on the console of the Linux workstation. This will create a folder named ***ps3files*** and inside you will find:

ps3.pdf → This file (or future versions)

p1.asm → Template file for problem 1
p1.c → C Reference file for problem 1

p2.asm → Template file for problem 2
p2.c → C Reference file for problem 2

p3.asm → Template file for problem 3
p3.c → C Reference file for problem 3

submit → Answer Submission program
answer files MUST have the filenames as follow
p1.asm for problem 1
p2.asm for problem 2
p3.asm for problem 3

You must not edit anything above and including the line containing "#Start your code here", with the exception of the place provided for your name, student number and account number. Any editing above that line might have an effect on your grade.

This files will be submitted using the submit script, by just running the command "./submit" inside the ps3files folder. You must use the account provided for this class to submit the files since they will be identified by the username (account name).

To run the MIPS emulator (Xspim) in the Linux console type 'xspim'. If you need additional information go to my website (http://micro.geekie.org) where an explanation on setting up your account so you can use spim and xspim. There is also information helpful if you get an error regarding the trap.handler.

Correction criteria:

Criteria	Weight(%)
Correctness	60%
Design	20%
Efficiency	10%
Style	10%

Remember the policy on late submission stated on the "Course Information Sheet".