C-355 Credit HW # 2(Due Friday 07/21/2017 at 6 PM)

Group 1: Chris Kasper, Alfonzo Desantis, Matt Long, Dallas Foglia

Q1.- Chris Kasper

Convert the following MIPS-32 assembler instructions to machine code . Fill up the

different fields in decimal or hexadecimal notation as appropriate

Example: Here first and last fields are hexadecimal, others are decimal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 hex | 16 | 17 | 9 | 0 | 23 hex |

(i) add $s7,$t3,$s5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 hex | 11 | 21 | 23 | 0 | 20 hex |

(ii) sll $v0,$a0,2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 hex | 0 | 4 | 2 | 2 | 00 hex |

(iii) lw $t4, 300($t0)

|  |  |  |  |
| --- | --- | --- | --- |
| 23 hex | 8 | 12 | 300 |

(iv) bne $s6, $s7, L1 where L1 is the 10th instruction after this one (4 points)

|  |  |  |  |
| --- | --- | --- | --- |
| 5 hex | 22 | 23 | 9 |

Q2. Matthew Long

Write MIPS-32 Assembler code to store the 32 bit integer in $s0 in locations beginning 0xABCD5678 in the main memory (1 point extra credit for a 2;line code)

|  |  |  |  |
| --- | --- | --- | --- |
| 0xpp | 0xqq | 0xrr | 0xss |

<--------------------------------$s0----------------------------------->

Addi $s1, 0xABCD5678, $zero

sw $s0, $s1

|  |  |
| --- | --- |
| Address | Content |
|  |  |
|  |  |
|  |  |
|  |  |
| ABCD5678 | 0xpp |
| ABCD5679 | 0xqq |
| ABCD567A | 0xrr |
| ABCD567B | 0xss |

Q3. - Alfonzo Desantis

Convert the following C code to equivalent MIPS-32 Assembler code.

Assume x,y,z are in $s4,$s5,$s6 respectively. You can use additional temporary registers as necessary.

int x,y,z;

if(x >= y)

z= z - 3;

else

z= 6;

Slt $t0, $s4, $s5

Beq $t0, $zero, FALSE

Addi $s6, $s6, 6

J Exit:

FALSE: addi $s6, $s6, -3

Exit:

Q4.- Chris Kasper

Develop the MIPS-32 Assembly language code corresponding to the C code given below:

(Assume the base of x is in $s0 and the loop variable i is in $t0)

int x[10];

int i;

for (i=0;i<10;i++)

{

x[i]=8\*x[i] - 1;

}

Add $t0, $zero, $zero # initialize i to 0

Loop: slti $t1, $t0, 10 # check if i< 10

Beq $t1, $zero, Exit # if i is not less than 10, jump out of loop

Sll $t3, $t0, 2 # i times 4 ← This line was included after HW was submitted.

Add $t2, $t3, $s0 # get address of x[i]

Lw $t3, 0($t2) # load x[i]

Sll $t3, $t3, 3 # multiply x[i] by 8

Addi $t3, $t3, -1 # subtract 1 from (x[i]\*8)

Sw $t3, 0($t2) # store new x[i]

Addi $t0, $t0, 1 # add 1 to i

J Loop # go to loop

Exit: