Linh Nguyen

B04

ENCM 369 – Computer Organization

Lab 5

Ex. A)

$t8 = 0x0000\_0c54, $t9 = 0x0000\_0a36

lui $t0, 0x9163

$t0 is 0x9163\_0000 because lui copies the immediate operand to bits 31-16 of the destination register and makes bits 15-0 all 0.

srl $t1, $t0, 6

$t0 = 1001\_0001\_0110\_0011\_0000\_0000\_0000\_0000

$t1 will be $t0 shifted 6 bits to the right:

$t1 = 0000\_0010\_0100\_0101\_1000\_1100\_0000\_0000 = 0x0245\_8c00

or $t2, $t8, $t9

$t8 = 0000\_0000\_0000\_0000\_0000\_1100\_0101\_0100

$t9 = 0000\_0000\_0000\_0000\_0000\_1010\_0011\_0110

$t2 = 0000\_0000\_0000\_0000\_0000\_1110\_0111\_0110 = 0x0000\_0e76

andi $t3, $t8, 0x07fc

$t8 = 0000\_0000\_0000\_0000\_0000\_1100\_0101\_0100

0x07fc = 0000\_0000\_0000\_0000\_0000\_0111\_1111\_1100

$t3 = 0000\_0000\_0000\_0000\_0000\_0100\_0101\_0100 = 0x0000\_0454

nor $t4, $t8, $t8

OR-ing $t8 with $t8 gives the value of $t8. The NOR result is the inverse of all bits of $t8

$t8 = 0000\_0000\_0000\_0000\_0000\_1100\_0101\_0100

$t4 = 1111\_1111\_1111\_1111\_1111\_0011\_1010\_1011 = 0xffff\_f3ab

nor $t5, $t8, $zero

OR-ing $t8 with $zero results in the value of $t8 in $t5. The NOR result will be the inverse of the bits of $t8, therefore the result will be the same as $t4

$t5 = 1111\_1111\_1111\_1111\_1111\_0011\_1010\_1011 = 0xffff\_f3ab

xor $t6, $t8, $t9

$t8 = 0000\_0000\_0000\_0000\_0000\_1100\_0101\_0100

$t9 = 0000\_0000\_0000\_0000\_0000\_1010\_0011\_0110

$t6 = 0000\_0000\_0000\_0000\_0000\_0110\_0110\_0010 = 0x0000\_0662

xori $t7, $t8, 0x07fe

$t8 = 0000\_0000\_0000\_0000\_0000\_1100\_0101\_0100

0x07fe = 0000\_0000\_0000\_0000\_0000\_0111\_1111\_1111

$t7 = 0000\_0000\_0000\_0000\_0000\_1011\_1010\_1011 = 0x0000\_0bab

Ex B)

1. Bits 31-16 of the data address are given by 0x1003 and bits 15-0 are 0x49a0. Therefore, the assembly language for the given pseudoinstruction would be

lui $at, 0x1003

sb $t8, 0x49a0($at)

The opcode for lui is 001111, bits 25-21 are 00000, and bits 20-16 are 00001 for $at. The last 15-0 bits are given by the constant.

lui Machine Code in base two: 001111\_00000\_00001\_0001\_0000\_0000\_0011

To get hex, the machine code is grouped into 4 bits:

lui Machine Code in hex:

0011\_1100\_0000\_0001\_0001\_0000\_0000\_0011 = 0x3c01\_1003

$at is GPR 1 and $t8 is GPR 24. The opcode for sb is 101000. Bits 15-0 is the constant.

sb Machine Code in base two:

101000\_00001\_11000\_0100\_1001\_1010\_0000

sb Machine Code in hex:

1010\_0000\_0011\_1000\_0100\_1001\_1010\_0000 = 0xa038\_49a0

1. To make the constant 0x5dc70, two instructions are needed are needed and then the add operation is used

lui $at, 0x5dc7

ori $at, $at, 0x0000

add $s4, $s1, $at

lui Machine Code in base two: 001111\_00000\_00001\_0101\_1101\_1100\_0111

in hex:

0011\_1100\_0000\_0001\_0101\_1101\_1100\_0111 = 0x3c01\_5dc7

For ori, the opcode is 001101, bits 25-21 and 20-16 are 00001 for both the source and destination GPR $at, and bits 15-0 is the constant.

ori Machine Code in base two: 001101\_00001\_00001\_0000\_0000\_0000\_0000

in hex:

0011\_0100\_0010\_0001\_0000\_0000\_0000\_0000 = 0x3421\_0000

The two source GPRs for add are $s1 and $at (10001 and 00001 respectively), the destination GPR is $s4 (10100)

add Machine Code in base two: 000000\_10001\_00001\_10100\_00000\_100000

in hex:

0000\_0010\_0010\_0001\_1010\_0000\_0010\_0000 = 0x0221\_a020

Ex C)

Beq:

The address of the instruction that follows the beq instruction is given by

0x0040\_7320 + 4 + 4 = 0x0040\_7328

The offset, in bytes, between this instruction and the or, which is the branch target, can be found by doing hex subtraction:

0x0040\_73a4

* 0x0040\_7328

-----------

0x7c = 0111\_1100

beq use word offsets, therefore divide the result by 4 to get 0001\_1111. Extending that to 16 bits and adding the opcode and register numbers give

beq: 000100\_01000\_00000\_0000\_0000\_0001\_1111

in hex: 0001\_0001\_0000\_0000\_0000\_0000\_0001\_1111

= 0x1100\_001f

J:

The target address is

0x0040\_7320 = 0000\_0000\_0100\_0000\_0111\_0011\_0010\_0000

The machine code for the j instruction is simply the j opcode (000010) and then bits 27-2 of the above address:

J: 000010\_0000\_0100\_0000\_0111\_0011\_0010\_00

In hex: 0000\_1000\_0001\_0000\_0001\_1100\_1100\_1000

= 0x0810\_1cc8