Homework 2

ECE2500 CRN:82943

Jacob Abel

September 14, 2018

Problem 1: Extend each of the 16-bit numbers given below into 32-bit numbers using both the ZeroExtImm and SignExtImm schemes. Write your answers in hex.

1. 0xC943

   ZeroExtImm: 0x0000C943

   SignExtImm: 0xFFFFC943

2. 0x3A04

   ZeroExtImm: 0x00003A04

   SignExtImm: 0x00003A04

Problem 2: The C language permits a variable to be left- or right-shifted by a non-constant amount (e.g. i » j where i and j are variables), but the MIPS instruction set only supports shifts by a constant value (e.g. i » 2). In words rather than code, describe how a variable-length right shift could be performed using MIPS instructions.

To implement a variable length right shift using MIPS instructions, you can create a loop over the shift length j of a right shift operation on i.

Problem 3: Provide the instruction format type (R/I/J) and 32-bit hexadecimal encoding for each of the following instructions:

1. `add $s1, $t0, $t1`

   Format: R

   32-bit hex encoding: (0x00, 0x08, 0x09, 0x11, 0x00, 0x20)
   -> 000000 01000 01001 10001 00000 100000
   -> 0000 0001 0000 1001 1000 1000 0010 0000
   -> 0x02284820

2. `lw $t2, 32($s0)`

   Format: I

   32-bit hex encoding: (0x23, 0x10, 0x0A, 0x10)
   -> 100011 10000 01010 0000000000010000
   -> 1000 1110 0000 1010 0000 0000 0001 0000
   -> 0x8B0A0010

Problem 4: Provide the type (R/I/J) and MIPS assembly language instruction for the instructions described by the following MIPS fields:

1. op=0x0, rs=0x10, rt=0x11, rd=0x08, shamt=0x0, funct=0x27

   Format: R

   MIPS Instruction: nor $t0, $s0, $s1

2. op=0xB, rs=0x18, rt=0x2, immediate=0xF21D

   Format: I

   MIPS Instruction: sltiu $v0, $t8, 61981

Problem 5: Provide the type (R/I/J) and MIPS assembly language instruction for the following machine code: 00000001000010010101000000100011

-> 00000001000010010101000000100011

-> 000000 01000 01001 01010 00000 100011

op=0x0, rs=0x08, rt=0x09, rd=0x0A, shamt=0x0, funct=0x23

Format: R

MIPS Instruction: subu $t2, $t0, $t1

Problem 6: Translate the following C code to MIPS assembly code. Assume that the values of a, b and c are in registers $\$s1$, $\$s2$ and $\$s3$, respectively.

```
if (a <= b)
        b = 4*a + c;
else
        b = 0;

        ble   $s0, $s1, Else
        sll   $t0, $s0, 2
        add   $s1, $t0, $s3
        beq   $zero, $zero Endif
Else:   sw    $zero, 0($s1)
Endif:  ..
```

Problem 7: Translate the following C code to MIPS assembly code. Assume that the values of i and n are in registers $\$s0$ and $\$s1$, respectively. Also, assume that register $\$s2$ holds the base address of the array A.

```
for(i=2*n; i > n; i--) {
        A[i] = A[i] + A[i-1] - i;
}
```

```
        sll $s0, $s1, 1
For:    ble $s0, $s1, Efor
        sll $t1, $s0, 2
        add $t0, $t1, $s2
        lw  $t1, 0($t0)
        sub $t2, $t1, 4
        lw  $t2, 0($t2)
        add $t1, $t1, $t2
        sub $t1, $t1, $s0
        sw  $t1, 0($t0)
        addi $s0, -1
        beq $zero, $zero For
Efor:   ..
```