Computer Science 605.611

Problem Set 4 Answers
(Please do not use outside resources such as simulators, disassemblers or web sites to complete this assignment.)

1. (4) Consider the following statement: "Every MIPS assembly language statement is translated by the assembler into a single 32-bit machine code instruction." Explain why the statement is true or is false.
The statement is false because an assembly language statement may be a pseudo-instruction that corresponds to multiple true-ops or may be an assembler directive that causes the generation of an amount of data other than a single 32-bit word.

2. a) (7) Prior to executing the MIPS machine instruction 0x10A1FFFE (which resides at address 0xB0000000), CPU registers $1 through $31 all contain the pattern 0xFFFFFFFA. What is the hex pattern in the PC (program counter) and any modified CPU general purpose registers ($0 through $31) immediately after this instruction is executed?
The 6-bit opcode=000100, so this is a beq instruction that branches if the contents of the rs register ($5) and the rt register ($1) are the same. Since the two registers contain the same value, the branch is taken and the PC is set to the branch target address computed as the incremented PC contents plus 4 times the sign-extended low 16 bits of the machine instruction.
The 16-bit offset 0xFFFE represents -2 (indicating a backwards branch of 2 instructions). 4 * (-2) = -8
So the branch target address = 0xB0000004 – 8
= 0xB0000004 + 0xFFFFFFF8 = 0xAFFFFFFC.

b) (7) Prior to executing the MIPS machine instruction 0x04A1FFFE (which resides at address 0xB0000000), registers $1 through $31 all contain the pattern 0xFFFFFFFA. What is the hex pattern in the PC (program counter) and any modified CPU general purpose registers ($0 through $31) immediately after this instruction is executed?
The 6-bit opcode=000001, so this is a conditional branch instruction that compares the rs register with the constant 0. The rt field (=00001) indicates that this is the instruction bgez which branches if the contents of the rs register is greater than or equal to 0. The rs field contains 00101 indicating register $5. Since register $5 contains 0xFFFFFFFA (= - 6), the branch is not taken and the PC will contain the address of the next instruction (0xB0000004).

3. (4) For our MIPS 9-instruction core subset (add, sub, and, or, slt, lw, sw, beq & jump), which bits within the machine instruction does the control unit examine to decide whether to set the RegDst control bit to 0 or to 1?
For the MIPS core instruction subset, the control unit examines bits 31 through 26 (the opcode). If these opcode bits = 000000 (R-type instruction) then RegDst is set to 1. Any other MIPS core instruction opcode will cause RegDst to be set to 0.

4. (4) List all assembly instructions within our 9-instruction MIPS core subset for which the control unit would set the MemtoReg control bit to 1 when it processes the corresponding machine instruction.
MemtoReg is set to 1 for instructions whose result comes from the data memory. The only instruction within our 9-instruction subset for which this is the case is the lw instruction.

5. (5) Suppose the instruction jr $t0 resides at memory address 0x04000000. What is the highest valid instruction address (expressed in hex) to which this instruction can transfer control?
This instruction uses a register to contain a full 32-bit destination address. Instruction addresses must be a multiple of 4. The highest 32-bit multiple of 4 is 0xFFFFFFFC.

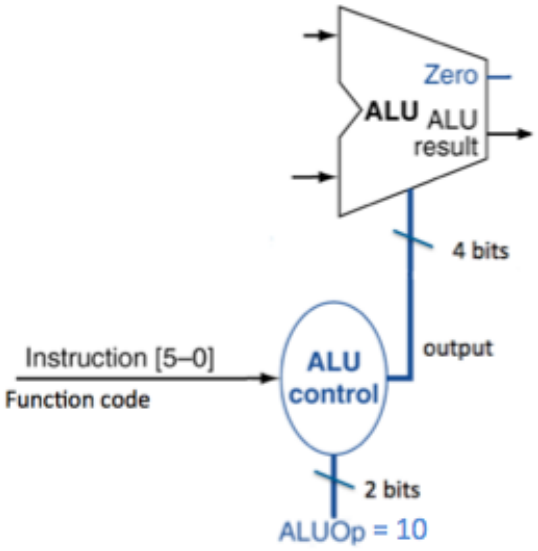
6. (5) List all assembly instructions within our MIPS 9-instruction core subset whose machine instruction format is the I-type format.
The beq, lw & sw instructions all use the I-format.

7. (4) The instructions: addi, ori, and lw all use the register identified by the rt field as the result register. Why can't these instructions use the register identified by the rd field as the result register?
These are all I-type instructions that use bits 15 – 11 within the machine instruction as the high bits of the 16-bit immediate field instead of specifying the rd field. These machine instructions contain only an opcode, rt, rs and immediate field.

8. (6) The diagram below shows how the ALU control unit directs the ALU during the execution of the instructions within our MIPS core subset. Assuming that the ALUOp bits = 10, complete the table below to show the 4-bit output from the ALU control unit for each of the listed 6-bit function code inputs described in module 4:

Function code	4-bit ALU control output
101010 (slt)	0111
100100 (AND)	0000
100000 (add)	0010
100101 (OR)	0001
100010 (subtract)	0110

These bit patterns were specified in the ALU control submodule within module 4.



9. a) (4) Why does the single-cycle datapath require separate instruction and data memories?
All instructions must be fetched from memory and instructions such as lw and sw also reference the data memory. A memory can only perform one operation (read or write) at a time. Therefore separate instruction and data memories are required to allow the instruction fetch and memory operand access to occur within the same clock cycle.

b) (4) Instead of using a single control bit Memctl to indicate whether to perform a memory read (Memctl =0) or a memory write (Memctl=1) with the single-cycle datapath, why are separate control bits (MemRead and MemWrite) used?
If only a single bit were used, then every instruction would be forced to either read from memory or write to memory. MemRead and MemWrite are both required so that when both bits are 0, neither a memory read nor a memory write occurs (as for the R-type instructions).

9. (5) Assume that $6 contains a two's complement integer that is greater than the two's complement integer in $5. After the instruction sub $4,$5,$6 is executed, what is the value of the MSB (bit31) in register $4 if an overflow occurs?
Subtracting a larger value from a smaller value should produce a negative result. A negative result will have a 1 in the MSB. However if an overflow occurs, the result will incorrectly appear to be non-negative. So the MSB = 0 if an overflow occurs.

10. (5) List all instructions within our MIPS core instruction subset (add, sub, and, or, slt, lw, sw, beq & jump) that do not require the output of the sign extension unit.
The instructions that use the output of the sign-extension unit are beq, sw and lw. The instructions that do not use the output of the sign-extension unit are add, sub, and, or, slt & j.

11. (10) Assume that the zero flag and registers $0 through $31 all contain 0 prior to executing the machine instruction 0x0C3A7E24 which resides at memory address 0x400B00C8. What value (expressed in hex) will be in any CPU registers that are modified and what value will be in the program counter (PC) register when the instruction completes execution?

The opcode corresponds to the leftmost 6 bits (000011 binary = 3), so this is a jal (jump and link) instruction (J-type). The jump address is produced by taking the upper 4 bits from the PC (0x4 in this case) and appending the rightmost 26 bits from the instruction shifted left 2 positions. Shifting the rightmost 26 bits left 2 positions yields the 28-bit pattern: 0x0E9F890 = (0x3A7E24 << 2). Prepending 0x4 (the leftmost 4 bits of the PC) to this pattern produces the 32-bit jump address 0x40E9F890, which is placed into the PC.

The address of the next instruction is placed into register $31 (the link register) as the return address. This return address is 0x400B00CC (or 0x400B00D0 taking into account the branch delay slot (which will be explained when pipelining is covered).

12. Show the 8-digit hex representation of the 32-bit machine instruction for each of the following true-op assembly language instructions:

(3) add $4,$5,$6 machine code = 000000 00101 00110 00100 00000 100000 = 0x00A62020

(3) or $4,$5,$6 machine code = 000000 00101 00110 00100 00000 100101 = 0x00A62025

13. Consider the MIPS assembly language instruction ori $6,$4,0xC8 .

a) (3) What is the operator mnemonic for this instruction?

The operator mnemonic is the name of the operation "ori'.

b) (3) What is the opcode for this instruction?

The leftmost 6 bits in the machine instruction contain the opcode = 001101 = 0x0D.

14. Suppose that registers $12 and $11 both contain the two's complement representation of the negative decimal integer -2147483648.

a) (3) What decimal value is represented by the result produced by the instruction: addiu $5,$12,-2 ? Decimal value = _____

This instruction adds the sign extended immediate operand plus $12 to produce the sum 0x80000000 + 0xFFFFFFFE = 0x7FFFFFFE in register $5. This pattern represents decimal +2147483646. Note that signed as well as unsigned arithmetic instructions sign extend the immediate operand.

b) (3) What decimal value is represented by the result produced by the instruction: andi $5,$11,-2 ? Decimal value = _____

This is a logical instruction. Logical instructions, unlike arithmetic instructions, zero-extend their immediate operand to 32 bits. So the pattern produced in register $5 is 0x80000000 AND 0x0000FFFE = 0x00000000. This pattern represents decimal 0.

c) (2) Does either or both of these instructions cause an overflow exception? If so, identify the instruction(s). The addi instruction causes an overflow since its result is interpreted as a signed integer. Neither the addiu instruction nor the andi instruction causes an overflow exception.

15. (6) Suppose that a hardware error, hack attack, or bit flip due to cosmic radiation inverts bit 20 within the machine instruction for bgez $4,exit. Bit 20 is the only bit affected. Would the modified 32-bit machine instruction still correspond to a valid instruction? If so, what is the assembly language instruction, when translated to machine code, corresponds to the modified 32-bit pattern?

Bit 20 within the machine instruction for bgez $4,exit is part of the rt field (00001). Inverting the bit changes the field to 10001. This makes the resulting 32-bit pattern appear to be the machine code for the assembly language instruction:
bgezal $4,exit.