ECE 375: Assignment 1

Jared Wasinger

October 11, 2016

1. Suppose a processor or CPU supports 64 different instructions and has a memory of 1K (K = 1024) words. Determine the size of each memory word for the following instruction formats:

   (a) i. 3-address instruction format:
   $Opcode = 2^6 = 6bits$
   $Address = 2^{10} = 10bits$
   Address bits * 3 + Opcode bits = 36 bits

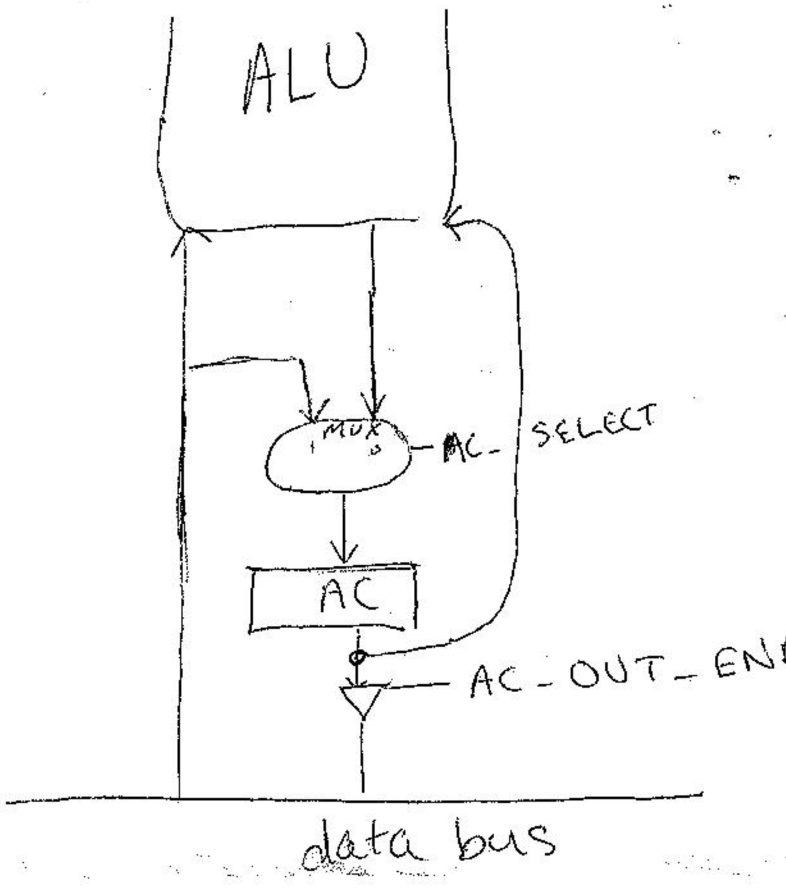
   ii. 2-address instruction format:
   $Opcode = 2^6 = 6bits$
   $Address = 2^{10} = 10bits$
   Address bits * 2 + Opcode bits = 26 bits

   iii. 1-address instruction format:
   $Opcode = 2^6 = 6bits$
   $Address = 2^{10} = 10bits$
   Address bits * 1 + Opcode bits = 16 bits

   (b) The advantage of a three-address instruction format vs a 2-address one is that it allows a wider variety of types of instructions. For example, if I wanted an instruction that could add two registers and store the results in a third, I could conceivably do this with one expression.
   However in a two address instruction format, this same operation would take more instructions. In order to perform addition and then move the result to a target register, the addition command would have to be invoked, then the result moved from the hardcoded output register of said command to the target register.

   The downside of a three-address instruction format is that it makes each instruction take up more memory than a two-address format. This would limit the total amount of instructions that a program could use and would potentially be severely limiting.

2



3. (a) This is not possible because AC and MAR would have to travel across the bus simultaneously.

   (b) $IR \leftarrow MDR, MAR \leftarrow MDR$
   This is possible as long as the number of bits being sent is specified. becuase IR_size<MDR_size and $MAR_size < MDR_size$.

   (c) $MAR \leftarrow MDR, MDR \leftarrow M(MAR)$ This is possible because MAR will have a value and thus, does not depend on $MAR \leftarrow MDR$ to get the address that needs to be retrieved from.

   (d) $MDR \leftarrow AC + 1$ Not possible: $AC + 1$ will take one step, and it will take another for MDR to receive the value.

   (e) $AC \leftarrow MDR, PC \leftarrow PC + 1$ Possible: PC can perform the addition using internal adder without sending anythin across the bus.

   (f) $PC \leftarrow PC + AC$ Not possible: This operation would take two cycles and more steps. First the value from PC must sent to AC over the bus. Then the addition must be performed and sent back to PC.

4. $STA^- : M(x) \leftarrow M(x) - 1, M(M(x)) \leftarrow AC$

   (a) $MAR \leftarrow X, TEMP \leftarrow AC$

   (b) $MAR \leftarrow MDR, AC \leftarrow MDR$
   Get $M(M(x))$ and store $M(X)$ in AC.

   (c) $MDR \leftarrow TEMP$ Store old AC in $M(M(x))$, and decrement current AC (holding $M(x)$)

   (d) $AC \leftarrow AC - 1$

   (e) ....

5. (a) $X \leftarrow X - 2, X = 0104$

   (b) $X_H \leftarrow 85, X = 8506$

   (c) $0x1B = 0b00011011- >$Rotate right through carry$- > 0b10001101; r2 = 0b10001101$

   (d) $r2 + r1 + C = 0x1B + 0x05 + 0x1 = 0x22; r2 = 0x22$

   (e) $M(0x0007) \leftarrow r28 = M(0x0007 \leftarrow X_L = M(0x0007)$; The registers and memory locations shown do not change.