Electrical and Computer Engineering Department
University of Maryland
College Park, MD 20742-3285
Glenn L. Martin Institute of Technology ♦ A. James Clark School of Engineering
Dr. Charles B. Silio, Jr.
Telephone 301-405-3668
Fax 301-314-9281
silio@umd.edu

ENEE 350 Homework Set No. 4

(Due: Class 9, Mon., June 16, 2014)

1. Read textbook Chapt. 3 and work the following problems:

   a. Problem 3-23.

   b. Problem 3-24.

   c. Problem 3-25.

   d. Problem 3-28.

   e. Problem 3-30.

   f. Problem 3-31.

   g. Problem 3-32.

   h. Problem 3-42.

2. Print out and read the notes on an example microprogrammed microarchitecture. These notes are entitled "Microarch.ps" or "Microarch.pdf" in the Notes subdirectory of the course webpage, and they describe a microarchitecture called the Mic-1 that by way of microprogramming implements an instruction set architecture (or macroarchitecture) called the Mac-1. Then do the following.

   (1) Log into a Glue workstation.

   (2) Type "tap ee350" (without the quotes, of course)

   (3) Copy the two files halt and halt.pascal to your working directory:

   ```
   ‘‘cp $EE350/halt .’’
   ‘‘cp $EE350/halt.pascal .’’
   ```

   (4) Then print out these two files:

   ```
   ‘‘qpr -q <your favorite printer> halt’’
   ‘‘qpr -q <your favorite printer> halt.pascal’’
   ```

   (5) Keep the listings themselves for your reference.

In problems 3 through 8 below assume that new OP Codes have been assigned for some new MAC-1 machine instructions and that these instructions have been fetched into the IR during an instruction fetch cycle, that they have been completely decoded, and that a microprogram jump has occurred to the control store starting address for their execution sequences. Write the MAL routines that accomplish execution of each of these new MAC-1 instructions. Execution is complete when an unconditional transfer of control back to the start of the instruction fetch sequence occurs.

3. Implement execution of the machine instruction:

   CALNZ x

   which calls subroutine (procedure) x if the content of the accumulator (ac) is NOT equal to zero. Assume that this microinstruction sequence starts at control memory address 100 and that locations above 100 are available for your use.

-over-

4. Implement execution of the machine instruction:

Mnemonic	Meaning
JSR x	m[x]:= pc; pc:= x+1

where JSR stands for "Jump to subroutine" (with entry point at main memory address x specified in the address field of the instruction). It differs from the subroutine call instruction in that it stores the return address (the content of the program counter) in location x (the first location in the subroutine) and jumps to location x+1 rather than storing the return address on the stack. Assume that this microinstruction sequence starts at control memory address 120 and that locations above 120 are available for your use.

5. Implement execution of the machine instruction: **RAR**
Assume that the sequence of microinstructions to execute a new Mac-1 conventional machine language instruction "RAR" starts in address 150 of control store and that all control store locations > 150 are free and available for your use. RAR stands for "rotate accumulator right" which does a one bit position right circular shift of the bits in the 16-bit ac register specified by the following, where the bit positions in the ac register are numbered right to left in their natural powers of two order from 0 to 15: For i = 0, 1, ..., 14, bit $ac_i := ac_{i+1}$; and $ac_{15} := ac_0$.

6. Implement execution of the machine instruction: **ACTOE**
which transfers (i.e., copies) the contents of the AC register into the E register. Assume control store locations 160 and above 160 are available for this.

7. Implement execution of the looping control instruction: **JGDE x**
which first checks the contents of register E, and if the content of E is strictly greater than zero, then jumps to memory address "x"; if the content of E is not strictly greater than zero, then no jump is taken (i.e., the execution cycle terminates by going directly to the instruction fetch cycle); in either case the content of register E is decremented by one (i.e., 1 is subtracted from the contents of register E). Assume control store locations 170 and above 170 are available for this.

8. Assuming a 1's complement representation of negative numbers, implement execution of the machine instruction:

ASHR1

Assume that the sequence of microinstructions to execute a new Mac-1 conventional machine language instruction "ASHR1" starts in address 180 of control store and that all control store locations > 180 are free and available for your use. ASHR stands for "arithemetic shift right the accumulator" which does a one bit position arithmetic (i.e., algebraic) right shift of the bits in the 16-bit ac register specified by the following, where again the bit positions in the ac register are numbered right to left in their natural powers of two order from 0 to 15: For i = 0, 1, ..., 14, bit $ac_i := ac_{i+1}$; and $ac_{15} := ac_{15}$. In other words, it preserves the sign bit (ac_{15}) while right shifting the contents of ac.

9. Now assuming a 2's complement representation of negative numbers (and the Mac-1 instruction set does so) implement execution of the machine instruction:

ASHR2

as in problem 8 for this case. The difference is that if the number in ac is negative (i.e. $ac_{15} = 1$) **and** the rightmost bit in ac is a 1 (i.e., $ac_0 = 1$) prior to the right shift, then we must add 1 (i.e. $ac_0 = 1$ and for i = 1, ..., 15, bit $ac_i = 0$) to the right shifted result in ac. In other words, if a 1 falls out the right end of the ac during the ASHR of a negative number, then we must add it back into the result in the ac. Assume control store locations 190 and above 190 are available for this.