CS270 Recitation 9
“LC-3 Input and Output”

Goals

1. To extend your knowledge of LC-3 programming with more programming.
2. To solidify your knowledge of how to use the LC-3 assembler and simulator to debug assembly code.

The Assignment

Make a subdirectory called R9 for the recitation; all files should reside in this subdirectory. Copy the file from the link to the R9 directory, a listing of the code is shown below.

http://www.cs.colostate.edu/~cs270/.Fall14/recitations/R9/R9.asm

```
; Recitation 9
; Author: <name>
; Date:   <date>
; Email:  <email>
; Class:  CS270
; Description: Converts hexadecimal number in ASCII to binary number
;--------------------------------------------------------------------------
; Begin reserved section: do not change ANYTHING in reserved section!

                .ORIG x3000
                JSR read            ; read number from keyboard
                JSR convert         ; convert to binary number
                HALT

; Decinal value
Result          .BLKW 1             ; space to store result

; String value
String          .BLKW 4             ; space to store input (4 characters)

; End reserved section: do not change ANYTHING in reserved section!
;--------------------------------------------------------------------------

; Constants
Four            .FILL 4             ; the number 4
AsciiA          .FILL xFFBF         ; minus ASCII 'A'
Ascii0          .FILL xFFD0         ; minus ASCII '0'
Prompt          .STRINGZ "Hexadecimal: "
;--------------------------------------------------------------------------
; read - reads four ASCII digits from console and stores them in string

read_ra         .BLKW 1             ; return address
read            ST R7,read_ra       ; save return address

                LD R1,Four          ; R1 = loop counter = 4
```

```
                                    ; R2 = string pointer
                                    ; Display prompt using PUTS
                                    ;
loop0                               ; Input character using GETC
                                    ; Output character using PUTC
                STR R0,R2,0         ; Store next character
                                    ; Increment pointer
                ADD R1,R1,-1        ; Decrement counter
                BRp loop0           ; Loop if positive

                LD R7,read_ra       ; restore return address
                RET
;---------------------------------------------------------------------------
; convert - converts four ASCII digits to binary number

convert_ra      .BLKW 1             ; return address
convert                             ; save return address

                LD R1,Four          ; R1 = loop counter = 4
                                    ; R2 = string pointer
                                    ; R3 = result = 0
loop1                               ; R0 = load next character
                LD R4,AsciiA        ; Minus 'A'
                ADD R4,R0,R4        ; Compare 'A'
                                    ; Branch to letter if >= 'A'

                                    ; What is true to get here?
digit           LD R4,Ascii0        ; Minus '0'
                ADD R4,R0,R4        ; Compare '0'
                BR continue         ; Processed digit

                                    ; What is true to get here?
letter          ADD R4,R4,10        ; Letter to digit

continue                            ; result *= 16
                                    ;
                                    ;
                                    ;
                                    ; result += digit
                                    ; Increment pointer
                                    ; Decrement counter
                BRp loop1           ; Loop if positive
                ST R3,Result        ; Store result

                                    ; restore return address
                RET
;---------------------------------------------------------------------------
               .END
```

1) Use the LC-3 assembler to transform your assembly code into object code that can run on the LC-3 simulator:

```
$ ~cs270/lc3tools/lc3as R9.asm
```

2) Load the LC-3 simulator and the TA will help you step through an invocation of one of the LC-3 subroutines:

```
$ ~cs270/lc3tools/lc3sim-tk &
```

3) Add code wherever there is a comment line but no assembly code. Empty semicolons are a hint about how many instructions are needed.

4) Test the program with the input values “1357”, “ABCD”, and “A4D6”. The decimal equivalent of these numbers should be in Result when the program hits the HALT instruction.

5) Be prepared to answer the two questions in the program, no code is required for these lines.

6) Submit to the drop box in RamCT for Recitation 9 and show your code to the TA.