Lab 05

Merry Christmas!

The purpose of this assignment is to show how interrupt-driven Input/Output can interrupt a program that is running, execute the interrupt service routine, and return to the interrupted program, picking up exactly where it left off (just as if nothing had happened). In this assignment, we will use the Keyboard as the input device for interrupting the running program.

The assignment consists of three parts:

The user program.

Your user program will consist of continually producing the "HELLO WORLD checkerboard" by alternately outputting two different lines via an infinite loop. One line will consist of the pattern "HELLO" followed by **seven** spaces five times. The second line will consist of the pattern seven spaces followed by "WORLD" four times.

```
HELLO       HELLO       HELLO       HELLO       HELLO
       WORLD       WORLD       WORLD       WORLD
HELLO       HELLO       HELLO       HELLO       HELLO
       WORLD       WORLD       WORLD       WORLD
HELLO       HELLO       HELLO       HELLO       HELLO
       WORLD       WORLD       WORLD       WORLD
HELLO       HELLO       HELLO       HELLO       HELLO
```

To ensure the output on the screen is not too fast to be seen by the naked eye, the user program should include a piece of code that will count down from 2500 each time a line is output on the screen. A simple way to do this is with the following subroutine DELAY:

```
DELAY   ST  R1, SaveR1
LD  R1, COUNT
REP     ADD R1,R1,#-1
BRp REP
LD  R1, SaveR1
RET
COUNT   .FILL #2500
SaveR1  .BLKW 1
```

The keyboard interrupt service routine.

The keyboard interrupt service routine will examine the key typed to see if it is the capital letter 'M'. If the input key is 'M', the interrupt service routine will, **starting on a new line on the screen**, print

MERRY CHRISTMAS!
The service routine would then print a line feed (x0A) to the screen, and finally terminate with an RTI.

If the input key is not 'M', the interrupt service routine will, starting on a new line on the screen, print
HARD WORK
The service routine would then print a line feed (x0A) to the screen, and finally terminate with an RTI.

Your interrupt service routine should start at the location x1500.

VERY IMPORTANT instructions for constructing your interrupt service routine:

1. You may not use **any** TRAP instructions in your interrupt service routine. (However, you may use TRAP instructions as you wish in your user program.)
2. To display a character on the screen, you must poll the DSR before writing to the DDR.

If you use TRAP in the interrupt service routine, or if you do not properly poll the DSR before writing to the DDR, your program is not correct and will fail our tests, even though it may appear to work when you test it.

Hint: Don't forget to save and restore any registers that you use in the interrupt service routine.

The operating system enabling code.

Unfortunately, we have not YET installed Windows or Linux on the LC-3, so we are going to have to ask you to do the following three enabling actions in your user program first, before your user program starts outputing the checkerboard. Normally, these would be done by the operating system before your user program starts executing.

1. Normally, the operating system would have previously set up some stack space so that the PC and PSR can be pushed when an interrupt is encountered. (As you know, when the service routine executes RTI, both PC

and PSR will be popped, returning the machine to the interrupted program.) Since there is no operating system, please initialize R6 to x3000, indicating an empty stack.

2. Also, normally, the operating system establishes the interrupt vector table to contain the starting addresses of the corresponding interrupt service routines. You will have to do that for the keyboard interrupt. The starting address of the interrupt vector table is x0100 and the interrupt vector for the keyboard is x80. It is necessary for you to only provide the one entry in the interrupt vector table that is needed for this programming lab assignment.

3. Finally, normally, the operating system would set the IE bit of the KBSR. You will have to do that as well.

Your job.

Your job is to do three things:

a. Write the user program described above.
b. Write the keyboard interrupt service routine described above.
c. Write the code necessary to perform the three tasks described above (because we do not have an operating system installed) to enable interrupts to be handled, and insert that code at the front of your user program.

The user program must be named **user_program.asm** and will be of the form:

```
                .ORIG    x3000
                --       ---     ; initialize the stack pointer
                   ...
                --       ---
                --       ---     ; set up the keyboard interrupt
                                  ; vector table entry
                   ...
                --       ---
                --       ---     ; enable keyboard interrupts
                   ...
                --       ---
                --       ---     ; start of actual user program to
                                  ; print checkerboard
                   ...
                --       ---
              .END
```

The interrupt service routine must be named **interrupt_service_routine.asm** and will be of the form:

```
.ORIG     x1500
 --     ---      ; the code
    ...
 --     ---
RTI
 --     ---      ; buffer space as required
    ...
 --     ---
.END
```

We have provided a screenshot as an example of how the console should look when you run the program.

Notes:

- The **Linux LC-3 simulator does NOT** include support for interrupts. Therefore, you must use the **Windows LC-3 simulator** for this assignment.
- To test your program, first load the interrupt service routine, then the user program, and then execute.
- Since your user program contains an infinite loop, you will have to press the "Stop Execution" button in the simulator if you wish to stop the program.

Attention:

1. Your zip file should contain at least two files:
   .asm files and **report in pdf format**.
   As subject name , please refer to the notice on course web page.