Thomas Brosh

ENEE350

HW 2

2. Direct Memory Access
3. Two fatal bugs in the code:
   * The student seems to have implemented a polling interface for IO transfer, however the student *does not* check if there was, in fact, a character has been input on the computer. This can be fixed by adding a conditional branch after checking the keybd\_status register.
   * The student is working with byte-sized memory entries (i.e. ‘lb,’ ‘sb’). This is fine for when the student loads keybd\_status into register $t0, because only the last two bits (MSB) are important. However, when the student calls ‘lb $v0, keybd\_status’ this will not load the entire keybd\_data memory word, and will result in reading something unexpected.

Fixed code:

#######################################################################

# Keyboard Read Device Driver: Called by OS File System

# $v0 contains character read from keyboard

#######################################################################

    .ktext # Make subsequent allocations in kernel text section

keybd\_read:

    lw $t0, keybd status

    andi $t1, $t0, 1

    beqz $t1, quit

    lw $v0, keybd data

    sw $t0, keybd status

quit:

    jr $ra # Return control to OS file system