Quiz 3 average was 6.5 / 11.

Here is the marking legend
FM - Flowchart/C Main (2 marks)
FS - Flowchart/C Subroutine (2 marks)
CM - Code Main (2 marks)
CS - Code Subroutine (2 marks)
SP - Followed Subroutine Specification (1 mark)
SX - Syntax (2 marks)

Common reasons for losing marks:

1. The code only checked for letters 'e' and 't'. Doing this made you get a MAXIMUM 1/2 for FS and CS.

2. In order to get the SP mark, you needed a subroutine as described in the handout. It had to accept a character, check to capitalize/replace it, and return the original/modified character. A subroutine that simply capitalized a character is not to specification. Also, the specification said SUBROUTINE, so you needed call/ret instruction - branching is not sufficient.

3. Since we are using characters, you must load and store only one byte at a time, so you must use the LDB/STB instructions. You get a MAXIMUM 1/2 for SX if you used any of LDW/LDH/ STW/STH.

Here are a few pointers on writing assembly code that many are struggling with:

1. MOV versus LDB/STB (load/store). Please make sure you completely understand the difference. MOV moves values between registers. If you want to access memory (data at TEXT and COPY), you must use LDB/STB.

2. Once you load a value into a register using LDB, there is no link formed between the register and memory. Changes to the register will not affect memory and vice versa. This is the same for STB.

3. You need to clearly define where your subroutines are, and never jump between them. You cannot jump from a subroutine back to main or vice versa. You must use CALL/RET!

If you feel you lost marks and don't know why:
- Check all of the common mistakes above
- Check whether your code performs the same as the example solution

NAME: _____ STUDENT #: _____

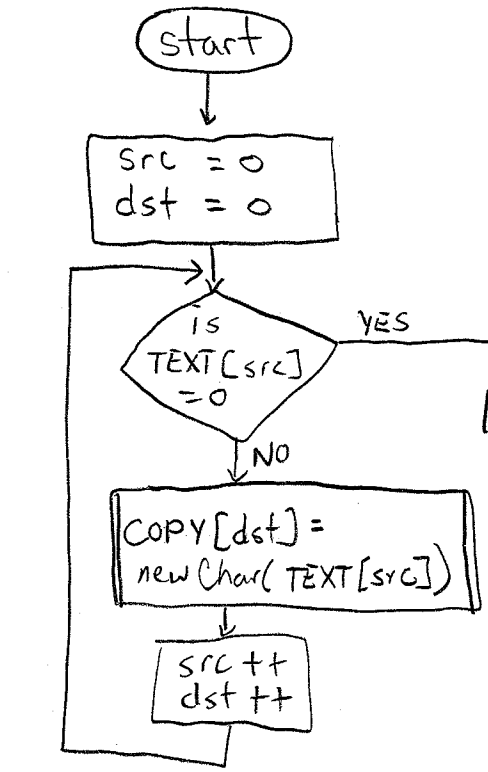
EECE 259: Introduction to Microcomputers **QUIZ 3 – 201** **Mar 11, 2011**

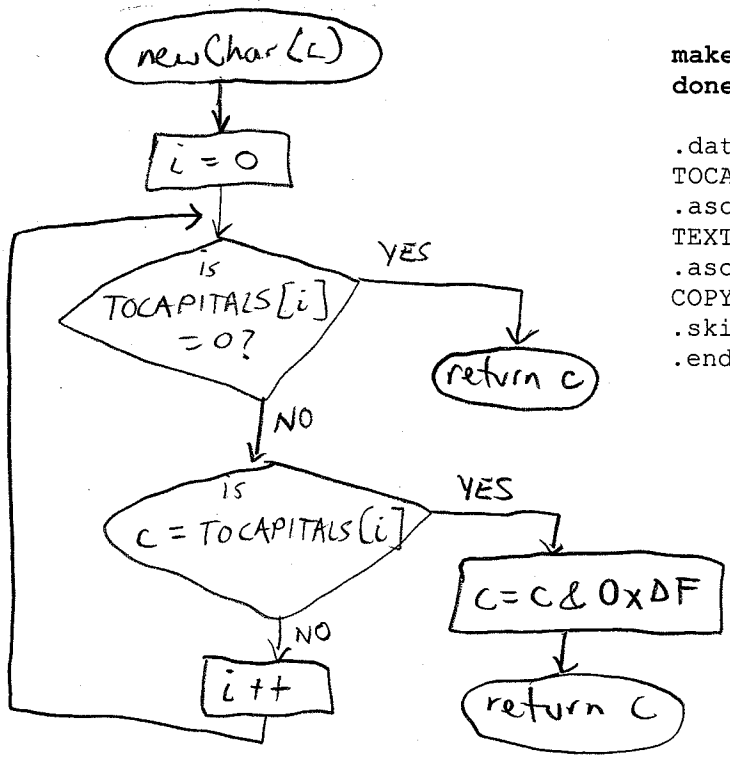
Write a **flowchart** and **Nios II assembly language** program to copy a string from label TEXT to label COPY. While copying, any characters also found in the string at label TOCAPITALS will be capitalized in the COPY. To capitalize a character, turn off bit 5 by masking the character value with (ANDing it) with 0xDF. All strings given are null-terminated (last character is the value 0); the copy should also end with 0.

- Your program must include a subroutine **newChar(c)** which returns a capitalized version of the character **c** if it is in the TOCAPITALS list, or returns the original character **c** if it is NOT the list.

Using the example strings below, the result at COPY should be the null-terminated string "somE TExT hErE".

You must use a flowchart or a C program. Using the stack is optional. Please obey register usage conventions.






```
.include "ubc-de1media-macros.s"
.global _start
.text
_start:         movia   sp, DRAM_END
                movia   r15, TEXT
                movia   r16, COPY

copyLoop:       ldb     r4, 0(r15)
                beq     r4, r0, doneCopy

                call    newChar
                stb     r2, 0(r16)

                addi    r15, r15, 1
                addi    r16, r16, 1
                br      copyLoop

doneCopy:       stb     r0, 0(r16)
stop:           br      stop

newChar:        movia   r8, TOCAPITALS
                mov     r2, r4
newLoop:        ldb     r3, 0(r8)
                beq     r3, r0, doneNewChar
                beq     r3, r4, makeCapital
                addi    r8, r8, 1
                br      newLoop
makeCapital:    andi    r2, r2, 0xDF
doneNewChar:    ret

.data
TOCAPITALS:
.asciz  "et"
TEXT:
.asciz  "some text here"
COPY:
.skip (COPY-TEXT)
.end
```

NAME: _____ STUDENT #: _____

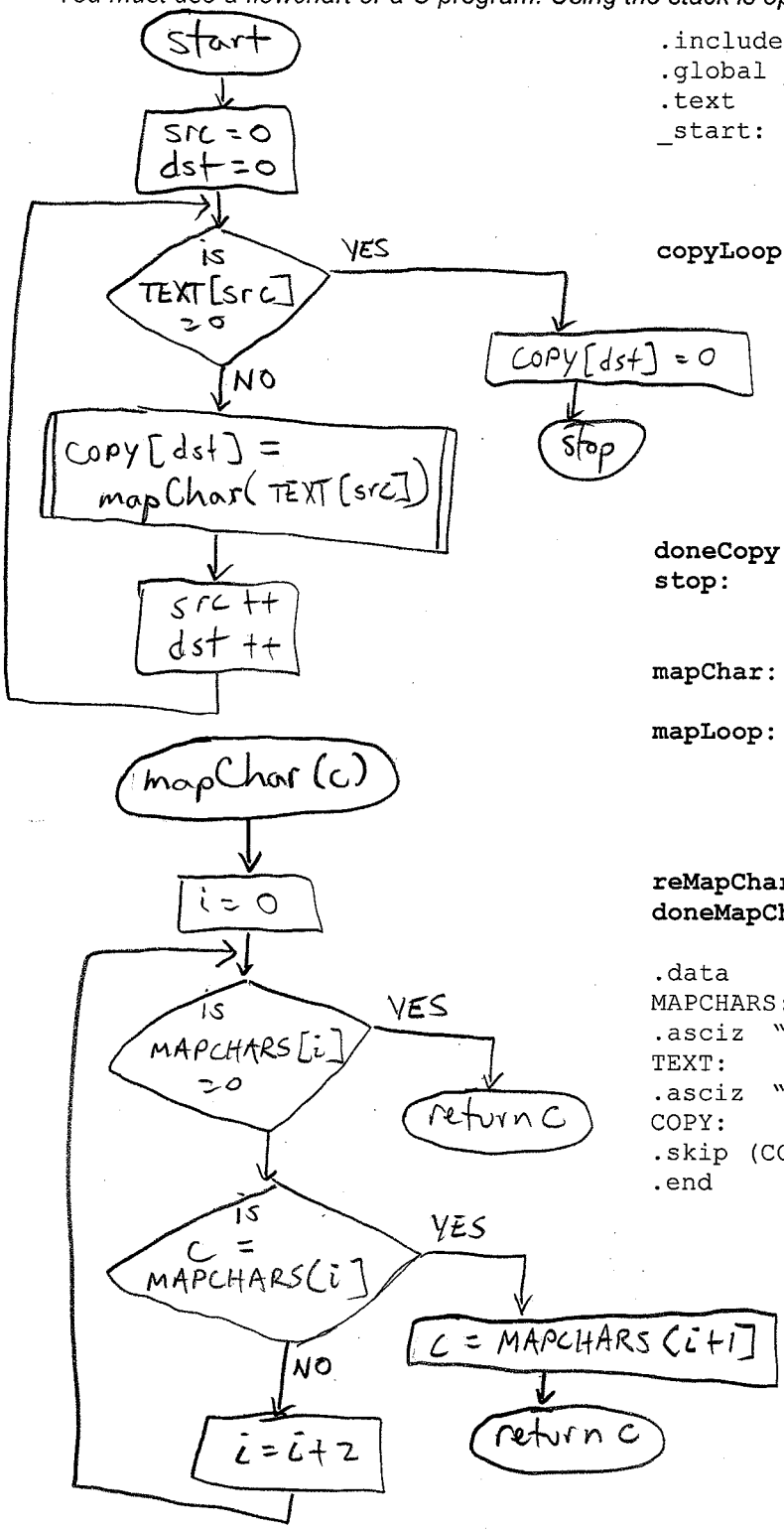
EECE 259: Introduction to Microcomputers **QUIZ 3 – 202** **Mar 11, 2011**

Write a **flowchart** and **Nios II assembly language** program to copy a string from label TEXT to label COPY. While copying, some characters will be replaced according to a string at label MAPCHARS. The characters in MAPCHARS always come in pairs: first the original character, followed by the new replacement character. There can be several pairs of characters in MAPCHARS. All strings given are null-terminated (last character is the value 0); the copy should also end with 0.

- Your program must include a subroutine **mapChar(c)** which returns the replacement for character **c** if it is in the MAPCHARS list, or returns the original character **c** if it is NOT in the list.

Using the example strings below, the result at COPY should be the null-terminated string "somE TExT hErE".

You must use a flowchart or a C program. Using the stack is optional. Please obey register usage conventions.




```
.include "ubc-de1media-macros.s"
.global _start
.text
_start:         movia   sp, DRAM_END
                movia   r15, TEXT
                movia   r16, COPY

copyLoop:       ldb     r4, 0(r15)
                beq     r4, r0, doneCopy

                call    mapChar
                stb     r2, 0(r16)

                addi    r15, r15, 1
                addi    r16, r16, 1
                br      copyLoop

doneCopy:       stb     r0, 0(r16)
stop:           br      stop


mapChar:        movia   r8, MAPCHARS
                mov     r2, r4
mapLoop:        ldb     r3, 0(r8)
                beq     r3, r0, doneMapChar
                beq     r3, r4, reMapChar
                addi    r8, r8, 2
                br      mapLoop
reMapChar:      ldb     r2, 1(r8)
doneMapChar:    ret

.data
MAPCHARS:
.asciz  "eEtT"
TEXT:
.asciz  "some text here"
COPY:
.skip (COPY-TEXT)
.end
```