

Computer architecture assignment E11

Jukka Honkanen

S N

36400T

09.07.1999

1. Original assignment

```
#####
#
#   Computer architecture assignment E11
#
#####
#
#   Write a subroutine which converts ascii characters in a null ending string
#   from lowercase to uppercase.
#
#   The arguments of the subroutine are passed as follows:
#
#       $a0 contains the address of string 'str'
#
#   The C-prototype of the subroutine is as follows:
#
#       unsigned int strupr(unsigned char *str);
#
#   The subroutine should return in register $v0 the number of converted
#   characters.
#####
#
# Main program to test strupr()
#
        .text
        .globl main
main:
    # Create a stack frame for main program
    subu    $sp,$sp,8
    sw     $ra,4($sp) # return address
    sw     $fp,0($sp) # old frame pointer
    addu   $fp,$sp,8 # update frame pointer

    # Test data

    .data

str1:    .asciiz "test string1\n"
teststr: .word str1
results: .word 10

        .text

    # Test routine
    # Set test data

    la     $s0,teststr
    la     $s1,results
    li     $s2,1 # test counter

testloop:

    # Print string before strupr call

    lw     $a0,($s0) # set pointer to string
    li     $v0,4
    syscall

    # Call strupr

    lw     $a0,($s0) # set pointer to string
    jal    strupr    # call

    # Check that result in $v0 is OK
    lw     $t1,($s1)
    bne    $v0,$t1,failure

    # Print string after strupr call

    lw     $a0,($s0) # set pointer to string
    li     $v0,4
    syscall

    # Update test counters
    add    $s2,$s2,-1
    addu   $s0,$s0,4
    addu   $s1,$s1,4
```

```
        bgtz    $s2, testloop

        # All tests done
        # Tell user that subroutine works..
        .data
ok:     .asciiz "The strupr() passed all tests\n"

        .text

        li     $v0, 4
        la    $a0, ok
        syscall

        # Exit from test program
exit:   li     $v0, 10
        syscall

        # Test failed
failure: .data
fail:   .asciiz " The strupr() failed\n"

        .text
        li     $v0, 4
        la    $a0, fail1
        syscall
        j      exit

#####
#
# The subroutine strupr
#

        .text
        .globl strupr

strupr:

        #
        # write your own subroutine here
        #

        jr     $ra

        .end
```

2. Function of the subroutine

The function of the subroutine is to convert lowercase characters in a string to uppercase. Additionally, the subroutine returns the number of converted characters. The subroutine operates according to the following syntax

```
unsigned intstrupr(unsigned char const *s1);
```

Where *s1* is a pointer to the string. Input parameter *s1* is in register \$a0 and the number of the converted characters is set by the subroutine to register \$v0. Input string is converted to the same place where the original string exists before calling the subroutine.

3. Description of the subroutine

Subroutine *strupr* converts lowercase characters to uppercase by adding conversion offset to all lowercase samples. For example uppercase character 'B' = 'b' + offset, where offset is 'A' - 'a'. The operation of the subroutine can be described by the following C-code.

```
unsigned intstrupr(unsigned char const *s1)
{
    unsigned counter = 0;          /* output counter */

    while (*s1 != NULL)
    {
        if ((*s1 > 'a') && (*s1 < 'z'))
        {
            /* lowercase character found */
            *s1 = s1+'A'-'a'; /* to uppercase*/
            counter++;        /* increment counter */
        }
        s1++;                  /* next character */
    }
    return (counter);
}
```

Subroutine does not use callee-saved registers (\$sX), therefore, stack frame is not required. The following registers are used:

\$a0 pointer to next character
 \$t0 last character that has been read from string
 \$t1 conversion offset 'A'-'a'
 \$v0 number of converted characters

4. Testing

The purpose of testing is to verify that the subroutine operates correctly with all possible input parameters. Since, this subroutine is not complicated the following test cases are estimated to be sufficient:

Test case 1

Input string "test string1\n" with 10 lowercase samples.

Expected output: string contains "TEST STRING1\n" and \$v0 = 10.

Test case 2

Input string "123 string2 ABC\n" with 6 lowercase samples.

Expected output: string contains "123 STRING2 ABC \n" and \$v0 = 6.

Test case 3

Empty input string "", no lowercase samples.

Expected output: string contains "" and \$v0 = 0.

Test case 4

Empty input string "4\n", no lowercase samples.

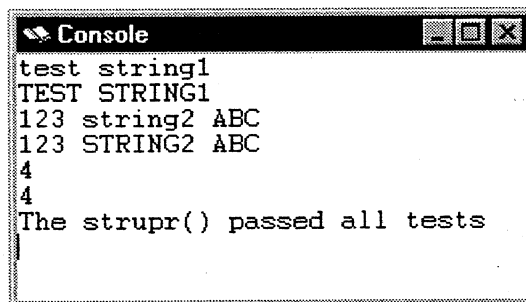
Expected output: string contains "4\n" and \$v0 = 0.

Test cases are performed consecutively in one test loop. For test purposes the main program prints the string before and after the function call. Also, the output of function (\$v0) is compared with the expected results in the test loop.

The result of a test run is listed in the next chapter.

5. Run-time log file

The following result was obtained while running the test program using spimwin. The program seems to be OK.



```
test string1
TEST STRING1
123 string2 ABC
123 STRING2 ABC
4
4
The strupr() passed all tests
```

6. Program code

```
#####
#
#   Computer architecture assignment E11
#
#####
#
#   Write a subroutine which converts ascii characters in a null ending string
#   from lowercase to uppercase.
#
#   The arguments of the subroutine are passed as follows:
#
#       $a0 contains the address of string 'str'
#
#   The C-prototype of the subroutine is as follows:
#
#       unsigned int strupr(unsigned char *str);
#
#   The subroutine should return in register $v0 the number of converted
#   characters.
#####
#
#   Main program to test strupr()
#
#       .text
#       .globl main
main:
#   Create a stack frame for main program
subu    $sp,$sp,8
sw      $ra,4($sp) # return address
sw      $fp,0($sp) # old frame pointer
addu    $fp,$sp,8 # update frame pointer

#   Test data

#       .data

str1:   .asciiz "test string1\n"
str2:   .asciiz "123 string2 ABC\n"
str3:   .asciiz ""
str4:   .asciiz "4\n"

teststr:
.word  str1,str2,str3,str4
results:
.word  10,6,0,0

#       .text

#   Test routine
#   Set test data

la      $s0,teststr
la      $s1,results
li      $s2,4 # test counter

testloop:

#   Print string before

lw      $a0,($s0) # set pointer to string
li      $v0,4
syscall

#   Call strupr

lw      $a0,($s0) # set pointer to string
jal     strupr    # call

#   Check that result in $v0 is OK
lw      $t1,($s1)
bne     $v0,$t1,failure

#   Print string after

lw      $a0,($s0) # set pointer to string
li      $v0,4
syscall

#   Update test counters
```

```

add    $s2,$s2,-1
addu   $s0,$s0,4
addu   $s1,$s1,4
bgtz   $s2,testloop

# All tests done
# Tell user that subroutine works..
.data
ok:    .asciiiz "Thestrupr() passed all tests\n"

.text

li     $v0,4
la     $a0,ok
syscall

# Exit from test program
exit:  li     $v0,10
       syscall

# Test failed
failure:
fail:  .data
       .asciiiz " Thestrupr() failed\n"

       .text
       li     $v0,4
       la     $a0,fail1
       syscall
       j      exit

#####
#
# The subroutinestrupr
#
# This subroutine converts characters in ascii string to upper case
# Input:      $a0 = pointer to string
# Output:     $v0 = number of converted characters
#
       .text
       .globlstrupr

strupr:
li     $v0,0           # init result
li     $t1,'A'
sub    $t1,$t1,'a'    # set offset A-a to $t1

loop:  lbu     $t0,($a0) # load next character
       beqz   $t0,ret   # return if null encountered

       blt    $t0,'a',endloop
       bgt    $t0,'z',endloop

       # character is between a..z

       addu   $t0,$t0,$t1 # convert it to upper case
       sb     $t0,($a0)   # store converted to string
       addu   $v0,$v0,1   # one more character converted

endloop:
addu   $a0,1           # set pointer to next char
b      loop

ret:   jr      $ra

       .end

```