MIPS: How to sort

I need your help with this problem of sorting in MIPS assembly :

how to write a MIPS program to read a text file containing only decimal integers and sort them in descending order.

The program should do the following:

■ Open a text file and read its content into an array of characters. The array should be limited to 1000 characters. MARS provides the system calls for opening and reading from a text file.

■ Traverse the array character by character. Convert each decimal string into binary. A decimal string consists of one or multiple decimal characters. It should terminate by white space or a newline character. Ignore and skip all other characters. Store all the decimal integers into an array of words. The size of the integer array should be limited to 100 words.

■ Sort the integer array in descending order.

■ Display the sorted array

Actualy I have no problem with sorting the array since I have it, but the problem with dealing with the text file, reading from it, converting to decimal plugging in the array.

Do you have any ideas ? comments ? suggestions ?

Thx in advance

Update: some has asked what is the question? the question is how to read from a txt file, convert the numbers to decimal ? this is the question.

assembly mips mars-simulator

edited Mar 4 '13 at 22:05
quetzalcoatl
10.5k ● 2 ● 19 ● 44

asked Mar 27 '11 at 20:17
Fahd
135 ● 2 ● 13

1 What have you tried, and where are you stuck? – Michael Petrotta Mar 27 '11 at 20:18

1 Please ask a specific question. – Gabe Mar 27 '11 at 20:27

Q: How to read from a txt file, convert the numbers to decimal ? this is the question. – Fahd Mar 28 '11 at 17:53

2 Answers

Ok ,,, I will post some parts of the solution to a problem that is similar to yours (much complicated though) ,,,

Data Initialization

```
# Data Buffers
  m1Buff:       .space  20000           # File 1 Data Buffer
  numBuff:       .space  200              # String Number Buffer
# Arrays
  m1:            .double    1:100
```

Code to read from a file and save into a buffer

```
############################################################
#### Open the file for reading
```

```
li  $v0, 13             # system call for open file
la  $a0,m1File          # input file name
li  $a1, 0              # flags
li  $a2, 0              # Open for reading (mode is 0: read, 1: write)
syscall                # open a file (file descriptor returned in $a0)
move $t0, $a0            # save the fd (syscall below will overwrite $a0)
############################################################
#### read from file just opened
li  $v0, 14             # system call to read from file
la  $a1,m1Buff          # address of buffer to store read data.
li  $a2, 20000      # max num of cahrs to read.
syscall                # read from file

############################################################
  #### Getting number of characters read from file.
  move $s0,$a0         # $s0 = number of read characters
############################################################
#### Close the file
li  $v0, 16             # system call for close file
move $a0, $t0        # Restore fd
syscall                # close file
############################################################
```

As for converting to decimal numbers ,,, I've written this procedure 2 years ago to convert a string to a floating point number (it also reads exponents i.e. 2.23E5) ,,, In your case you need to convert a string to a decimal which is much easier ,,, you should be able to figure out how to do it on your own giving the following procedure ,,,

```
###########################################################################################



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

## String To FP Procedure
##
## INPUTS : A String That Is Terminated With Null Char.
##   $a0 = address of String
##
## Author : Manaf Abu.Rous
###########################################################################################

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

str2float:

    move    $t1,$a0             # load Address Of Sttring In $t1

    li      $t0,10              # t0 = 10
    mtc1    $t0,$f2             # move $t0 to $f2
    cvt.d.w $f2,$f2             # f2 = 10 ( Used for Multiplication & Division )

    li      $t0,0               # t0 = 0
    mtc1    $t0,$f4             # move $t0 to $f4
    cvt.d.w $f4,$f4             # f4 = 0 = Integer Part. (Used To Generage The Integer
Part Of The Number)

    li      $t0,0               # t0 = 0
    mtc1    $t0,$f6             # move $t0 to $f6
    cvt.d.w $f6,$f6             # f6 = 0 = Fraction Part. (Used To Generage The Fraction
Part Of The Number)

    li      $t6,0               # $t6 = 0 = Exponent Part. (Used To Generage The Exponent
Part Of The Number)

    li      $t3,0               # $t3 = 0 ( Used To Determine The Sign Of The Number, +ve
= 0, -ve = 1 )
    li      $t4,0               # $t4 = 0 ( Used To Count The Number of Digits In a
Fraction )
    li      $t5,0               # $t5 = 0 ( Used To Determine If The Next Char Is a
Fraction , Yes = 1, No = 0)
    li      $t7,0               # $t5 = 0 ( Used To Determine If The Next Char Is an
Exponent , Yes = 1, No = 0)

    #--------------------------------------------------------

    #---------------------------------------------------------------------------
    #  Loop for visiting the buffer Char by Char and Constructing an Integer String
    #---------------------------------------------------------------------------

readLoop:

    lb      $t2,0($t1)          # load byte (char) in $t2

    #-----------------------------------------------------------------
    # -------- Check For "-" Sign
    bne     $t2,0x2d,skipNeg    # check if char = "-" ? if yes > sign reg = 1.
    li      $t3,1               # $t3 = 1 ( Used To Determine The Sign Of The Number )
    j       NextChar
    skipNeg:
    #-----------------------------------------------------------------

    #-----------------------------------------------------------------
    # -------- Check For "." Dot
    bne $t2,0x2e,skipDot        # check if char = "." ? if yes > Go Read Fraction
    li      $t5,1               # $t3 = 1 > Next Chars Are Fractinos
    j       NextChar
    skipDot:
    #---------------------------------------------------------------------

    #---------------------------------------------------------------------
    # -------- Check For "E" Exponent
    bne $t2,0x45,skipE          # check if char = "E" ? if yes > Go Read Exponent
    li      $t7,1               # $t3 = 1 > Next Chars Are Exponent
    j       NextChar
    skipE:
    #---------------------------------------------------------------------

    #---------------------------------------------------------------------
    # -------- Check For Null Char ( End Of String)
    beq     $t2,0x00,FinishNumber   # check if char = Null ? if yes > we are done with the
number.
    #---------------------------------------------------------------------

    #--------------------------------
    # Check What's The Current Char
    #--------------------------------
    beq     $t7,1,readExponent      # if $t7 = 1 (Next Char is Exponenet) Jump To
ReadExponent
    beq     $t5,1,readFraction      # if $t7 = 1 (Next Char is Exponenet) Jump To
ReadFraction

    #--------------------------------------------------------------------------
    # --------- Read Integer Part Of Char And Convert Them To Float
readInteger:
    subi    $t2,$t2,0x30        # subtract 0x30 from char to convert it to a number.
    mtc1    $t2,$f10            # move $t2 to $f6
    cvt.d.w $f10,$f10           # f6 = converte integer to a FP number.
    mul.d   $f4,$f4,$f2         # =((old)+2 X 10)
    add.d   $f4,$f4,$f10        # =((old)+ 2)
    j       NextChar
    #--------------------------------------------------------------------------
    #--------------------------------------------------------------------------
    # --------- Read Fraction Part Of Char And Convert Them To Float
readFraction:
    subi    $t2,$t2,0x30        # subtract 0x30 from char to convert it to a number.
    mtc1    $t2,$f10                # move $t2 to $f6
    cvt.d.w $f10,$f10           # f6 = converte integer to a FP number.
    mul.d   $f6,$f6,$f2         # =((old)+2 X 10)
    add.d   $f6,$f6,$f10            # =((old)+ 2)
    addi    $t4,$t4,1           # Increment Fractions Digits Counter
    j       NextChar
    #--------------------------------------------------------------------------
    #--------------------------------------------------------------------------
    # --------- Read Exponenet Part Of Char And Convert Them To Integer
readExponent:
    li      $t0,10
    subi    $t2,$t2,0x30        # subtract 0x30 from char to convert it to a number.
    mult    $t6,$t0             # =((old)+2 X 10)
    mflo    $t6
    add     $t6,$t6,$t2         # =((old)+ 2)
    j       NextChar
    #--------------------------------------------------------------------------



NextChar:
    addiu   $t1,$t1,1           # increment the address for the next buffer byte.
    j       readLoop

###################################### Finish Number Code
######################################

FinishNumber:
    # Finalizing Fraction Part And Adding It To Integer Part ------------------
    beq     $t5,0,skipFrac      # If There's No Fraction Part Then Skip. ($t5 != 1) > Skip

    li      $t0,1               # $t0  = 1
    mtc1    $t0,$f20            # move $t0 (Counter) to $f20
    cvt.d.w $f20,$f20           # f20 = 1

FracLoop:       #-------- Loop To Multiply 10 By It Self (Fraction Ditigs Counter) Times
                mul.d   $f20,$f20,$f2       # $f20 = $f20 X $f2 ($f20 = $f20 X 10)
                addi    $t4,$t4,-1          # Decrement Fraction Digits Counter
                bgtz $t4,FracLoop

    div.d   $f6,$f6,$f20        # $f6 = $f6 / $f20 ($f6 = Fraction Part / FractionsDigits
X 10 )
    add.d   $f4,$f4,$f6         # $f4 = $f4 + $f6 ( $f4 = IntegerPart + Fraction Part )
    skipFrac:
    # ------------------------------------------------------------------------



    # Exponent Part ----------------------------------------------------------
    beq     $t7,0,skipExp       # If There's No Exponenet Part Then Skip. ($t7 != 1) >
Skip
    beq     $t6,1,skipExp       # If The Exponent Is = 1 Then Skip
    addi    $t6,$t6,-1          # Correct Exponent.
    mov.d   $f18,$f4
```

```
ExpLoop:    #-------- Loop To Compute The Exponenet (Multiply Number By It Self (Expoenet)
Times)
            mul.d   $f4,$f4,$f18        # $f4 = $f4 X $f4 ( Multiply Number By It Self )
            addi    $t6,$t6,-1          # Decrement Fraction Digits Counter
            bgtz $t6,ExpLoop

    skipExp:
    # -------------------------------------------------------------------------


    # Checking Sign Register --------------------------------------------------
    bne     $t3,1,skipSign      # if $t3 != 1  Then Skip
    neg.d   $f4,$f4             # $f4 = - $f4
    skipSign:
    # -------------------------------------------------------------------------

    # Save The Number In $f0 To Be Returned
    mov.d   $f0,$f4             # $f0 = Converted String ( To Be Returned )

################################### End Of Finish Number Code
##################################
    jr      $ra         # Return

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

#                                         End Of Procedure
############################################################################################
```

Now what's left is reading from the file buffer ,,, looping through the chars ,,, converting them to decimal numbers ,,,, and saving them into the array.

edited Mar 29 '11 at 3:38

answered Mar 28 '11 at 23:04
Manaf Abu.Rous
1,624 ● 12 ● 21

Use syscall with $v0=13,$a1=0,$a2=0 to open a file.

Then use syscall with $v0=14 to read it into a buffer.

answered Mar 28 '11 at 19:31
Blux
46 ● 5