Devoir d'architecture DEUST TSIC
Janvier à Juillet 2006

1– Arithmétique

Effectuez la multiplication des deux nombres suivants exprimés en base 16 :

```
      42BB49
*      F5AB
---------------
```

Développer tous les calculs intermédiaires.

Soit XX le résultat en base 16. La vérification du calcul se fera en transformant les deux nombres en base 10, puis à effectuer la multiplication en base 10 de ces deux nombres et en comparant le résultat avec XX exprimé en base 10.

2– Analyse d'un code assembleur

2.1 Expliquez le programme suivant. En particulier, dites ce que représentent les trois entiers qui sont affichés à la fin du programme. Dégagez les structures d'organisation et donnez en langage algorithmique l'algorithme correspondant.

2.2 Si les entiers saisis sont tous strictements positifs ou négatifs, est-ce que le résultat affiché vous semble correct. Si non, proposez un code assembleur pour remédier au problème.

```
################################################################
################################################################
##
##  DESCRIPTION:
##
##
##  AUTHOR: C. Cérin <christophe.cerin@lipn.univ-paris13.fr>
##  DATE:   February 3, 2006
##  VERSION:      1.0
##
################################################################
################################################################

################################################################
## Data Segment
################################################################
      .data
STR1: .asciiz "Please enter a number (0 to stop): "
STR2: .asciiz "Result is "
NL:   .asciiz ".\n"

################################################################
## Text Segment
################################################################
      .text
      .globl main
#---------------------------------------------------------------
# Main Start
#---------------------------------------------------------------
```

```
main: addi  $s0, $zero, 0     # $s0 = 0
      addi  $s1, $zero, 0     # $s1 = 0
      addi  $s2, $zero, 0     # $s0 = 0
loop: li    $v0, 4            # system call for print_str
      la    $a0, STR1   # address of string to print
      syscall               # print the string

      li    $v0, 5            # system call for read_int
      syscall               # read the integer

      ## Check if $v0 == 0, if true goto exit
      beq   $v0, $zero, exit
      ## Result from read_int is stored in $v0
      add   $s0, $s0, $v0     #
      blez    $v0, loop1
      add     $s2, $s2, $v0   #
      j     loop        # goto loop
loop1:      add   $s1, $s1, $v0     #
      j     loop

exit: li    $v0, 4            # system call code for print_str
      la    $a0, STR2         # address of string to print
      syscall               # print the string

      li    $v0, 1            # system call code for print_int
      add   $a0, $s0, $zero   # integer to print,
      syscall               # print the integer

      li    $v0, 4            # system call code for print_str
      la    $a0, NL           # address of string to print
      syscall               # print the string

      li    $v0, 4            # system call code for print_str
      la    $a0, STR2         # address of string to print
      syscall               # print the string

      li    $v0, 1            # system call code for print_int
      add   $a0, $s1, $zero   # integer to print,
      syscall               # print the integer

      li    $v0, 4            # system call code for print_str
      la    $a0, NL           # address of string to print
      syscall               # print the string

      li    $v0, 4            # system call code for print_str
      la    $a0, STR2         # address of string to print
      syscall               # print the string

      li    $v0, 1            # system call code for print_int
      add   $a0, $s2, $zero   # integer to print,
      syscall               # print the integer

      li    $v0, 4            # system call code for print_str
      la    $a0, NL           # address of string to print
      syscall               # print the string

      jr    $ra               # exit program
# Main End
#-----------------------------------------------------------
```

3– Technologie

Sur une page au grand maximum, faites une introduction aux normes PCI-Express et PCI-X en les comparant à la norme PCI (http://www.pcisig.com).