CSE 334 Microprocessors Spring 2016

Assignment 2

Due date April 4, Tuesday 17:00 (No late submissions)

Write an assembly program that performs the following operations when the following commands are given:

1. **Put**: "put <data> <address>"
   Puts the given number to the given address. For instance
   `put 23 $1200`
   Then loads number 23 to $1200.

2. **Copy**: "cp <address> <address>"
   Copies an address vcontent to another address. For instance
   `cp $1500 $1200`
   Then copies the content of $1500 to the location $1200.

3. **Clear**: "clr <address> <numberOfLocations>"
   Clears a number of locations starting with an address. For instance:
   `clr $1500 12`
   Then writes zero to 12 locations between $1500 and $150B.

4. **Add**: "add <address> <address> <resultAddress>"
   Adds two numbers taken from the given addresses and puts the result to the resultAddress. For instance:
   `add $1200 $1201 $1202`
   Then adds the contents of $1200 and $1201 and puts the result to $1202.

5. **Gcd**: "gcd <address> <address> <resultAddress>"
   Computes greatest common divisor of two numbers taken from the given addresses and puts the result to the resultAddress. For instance:
   `gcd $1200 $1201 $1202`
   Then computes greatest common divisor (OBEB) of the contents of $1200 and $1201 and puts the result to $1202. (You will get 3x more grade when compared with other commands if you implement this instruction correctly)

Your assembly program will understand if a command different than the above 4 command is given. If such a case occurs your program will output $FF on PORTB. If a right command is given your program should count the commands and output the number of commands from PORTB. The commands will be taken starting from the address $1800. The end of commands will be understood when the program realizes the word "OVER".

Example: For instance lets assume that there is the following string starting at address $1800.

$1800	'p'	'u'	't'	' '	'1'	'8'	' '	'$'	'1'	'2'	'0'	'5'	'\n'	'c'	'p'	' '
$1810	'$'	'1'	'2'	'0'	'5'	' '	'$'	'1'	'2'	'0'	'6'	'\n'	'a'	'd'	'd'	' '
$1820	'$'	'1'	'2'	'0'	'5'	' '	'$'	'1'	'2'	'0'	'6'	' '	'$'	'1'	'2'	'0'
$1830	'7'	'\n'	'O'	'V'	'E'	'R'										

For such memory contents your program will:

- Put 18 to address $1205.
- Copy 18 to address $1206.
- Add contents of $1205 and $1206 and put the result to address $1207.
- Then finish the program realizing "OVER" at the end.
- At the end PORTB will be 3 as three commands are executed.
- If there were a wrong usage or unknown command PORTB would be $FF.

Comment almost all lines of your assembly. Use a subroutine for each of the commands.

BONUS

1. Insert a new command of your own. (+10 points)
2. Support hexadecimal numbers with '$' sign in put command (+10 points)

Assembly code that does not build or execute will get at most 20 points whether the error is small or big.
Cheating will result in -100 for both sides. Any help or cooperation is forbidden. Do on your own!