Istanbul Technical University
Department of Computer Engineering

BLG 211E – Microprocessor Systems
Homework 2

Due Date: 10.01.2021, **Sunday,** 23.59.

QUESTION 1 (60 points):

You are expected to implement two assembly programs calculating factorials of sequential numbers. One program must be implemented with an iterative approach and the other must be implemented using a recursive approach. Both must be implemented in Arm Cortex M0+ assembly language. Pseudo codes and constraints for each approach are given below.

a) Pseudo Code for Recursive Approach (30 Points):

```
index <= 6
array[index+1]

factorial(n):
    if(n<2) then
        return 1
    return n*factorial(n-1)


int main():
    for i <= 0 to index:
      array[i] <= factorial(i)
    while(1)
```

Constraints:

- Index value must be assigned as a global value. (You must use EQU directive for declaration).
- The allocation of space for the array must use index value (We will only change index value to test your program). You must allocate enough (neither less nor more) space in the memory.
- At the beginning of your code, the start address of array must be stored in R5 register and the value of R5 register must remain unchanged for the rest of the program.
- The maximum factorial result is 32 bits. You can take the lowest 32 bits, if the result overflows the 32 bits.
- The result of the factorial(index) function must be returned in R2 register.
- You must use R0-R1 registers to pass parameters to factorial(index) function.
- You must use the stack to store return addresses and other register values if necessary.
- Your main function name or label must be "__main".
- Factorial of numbers should start at index 0. An example series is given below:
    - factorial(0)= 1, factorial(1) = 1, factorial (2)=2, factorial (3)=6, factorial (4)=24 ….
- Your code should include a comment for each line. Otherwise, points will be deducted.
- The file name for this program must be "recursive.s"
- Your assembly source file is expected to work in Keil µVision IDE v5.
- Default configuration should be sufficient to run your programs. If your program expects any different configuration parameter, please provide a ReadMe file.

b) Pseudo Code for Iterative Approach (30 Points):

```
index <= 6
array[index+1]

factorial(f_array, n):
    f_array[0]=1
    for i <= 1 to n:
        f_array[i]= i * f_array[i-1]


int main()
    factorial(array,index)
    while(1)
```

Constraints:

- Index value must be assigned as a global value. (You must use EQU directive for declaration).
- The allocation of space for the array must use index value (We will only change index value to test your program). You must allocate enough (neither less nor more) space in the memory.
- Your factorial function should not return any value. The factorial of numbers must be stored in the memory. (You must store factorial(1), factorial(2), …, factorial(index) in the allocated array).
- At the beginning of your code, the start address of array must be stored in R5 register and the value of R5 register must remain unchanged for the rest of the program.
- You must use R0-R1 registers to pass parameters to factorial(index) function.
- You must use the stack to store return addresses and other register values if necessary.
- Your main function name (label) must be "__main".
- Factorial of numbers should start at index 0. An example series is given below:
    - factorial(0)= 1, factorial(1) = 1, factorial (2)=2, factorial (3)=6, factorial (4)=24 ….
- Your code should include a comment for each line. Otherwise, points will be deducted.
- The file name for this program must be "iterative.s"
- Your assembly source file is expected to work in Keil µVision IDE v5.
- Default configuration should be sufficient to run your programs. If your program expects any different configuration parameter, please provide a ReadMe file.

QUESTION 2 (40 points):

You are expected to implement Sieve of Eratosthenes Algorithm for finding all prime numbers up to a given limit. The program must be implemented in Arm Cortex M0+ assembly language. Pseudo code and constraints are given below.

```
LIMIT <= 120
primeNumbers[LIMIT + 1];
isPrimeNumber[LIMIT + 1];

SieveOfEratosthenes(limit):
    for i <= 0 to limit:
        primeNumbers[i] = 0
        isPrimeNumber[i] = true

    for i <= 2, 3, 4, ... as long as i*i <= limit:
        if isPrimeNumber[i] then:
            for j <= i*i, i*i+i, i*i+2i, i*i+3i,... until j <= limit:
                isPrimeNumber[j] <=false

    index <= 0
    for i <= 2 to limit:
        if isPrimeNumber[i] then:
            primeNumbers[index]=i
            index++


int main():
    SieveOfEratosthenes(LIMIT)
```

Constraints:

- LIMIT value must be assigned as a global value. (You must use EQU directive for declaration).
- The allocation of space for the arrays must use LIMIT value (We will only change LIMIT value to test your program). You must allocate enough (neither less nor more) space in the memory.
- Your SieveOfEratosthenes function should not return any value. Prime numbers must be stored in the memory.
- At the beginning of your code, the start address of primeNumbers must be stored in R5 register, the start address of isPrimeNumber must be stored in R6 register, and the value of R5 and R6 registers must remain unchanged for the rest of the program.
- You must use R0-R1 registers to pass parameters to SieveOfEratosthenes(limit) function.
- You should use the stack to store return addresses and other register values if necessary.
- Your main function name (label) must be "__main".
- Your code should include a comment for each line. Otherwise, points will be deducted.
- The file name for this program must be "sieveOfEratosthenes.s"
- Your assembly source file is expected to work in Keil µVision IDE v5.
- Default configuration must be sufficient to run your programs. If your program expects any different configuration parameter, please provide a ReadMe file.

Submission: You should upload your homework codes in Ninova only. Sending any files via e-mail is not allowed. Please type your name and student ID at the top of each file as a comment. You are expected to submit your homework through the Ninova system before the due date. Late submissions will not be accepted.

Any solution must be your own work. If any plagiarism is detected, disciplinary regulations of the university will be followed.

Note: If you have any question regarding the homework, you may contact to teaching assistant of the course (kadir.ozlem@itu.edu.tr).