**EASTERN MEDITERRANEAN UNIVERSITY**

**DEPARTMENT OF COMPUTER ENGINEERING**

**CMSE 443**

**Real-Time Systems Design**

**Final Exam (50 points)**

**2019/20 Spring Semester**

**June 30, 2020, Tuesday, 12.30**

Assoc. Prof. Dr. Alexander CHEFRANOV

1. Upload your pdf file with answers **to your personal chat** in Microsoft Team CMSE443:
2. In the **case of uploading problems**, e-mail to Alexander.chefranov@emu.edu.tr

Time: 110 minutes

**Please, consider the instructions below. The questions are in pages 2-5.**

1. Your answers shall be hand-written
2. On each page, at the top, write a header: “CMSE443 FE 30.06.2020”, followed by your Name, Surname, Student ID, page number
3. There are **9** questions in total. Try answering each question. There are 5 pages in total.
4. You may rewrite the text of the question in your paper, or not. It is up to you. But at the beginning of an answer, write “Question <i> answer:” substituting <i> by a particular question number
5. Open book, open notes, work yourself
6. Copies are not allowed and will be zero graded
7. In **15 minutes after the exam finishing**, you shall do **all** the following. 1. Make photo of each your page so that its full content, including the header, is in the image, and clearly readable. 2. Then, the images shall be assembled in the page number order into a pdf file named “CMSE443 FE 30062020 Name Surname.pdf”. 3. **Finally,** Upload the pdf file with answers **to your personal chat** in Microsoft Team CMSE443; In the **case of uploading problems**, e-mail to Alexander.chefranov@emu.edu.tr
8. ***papers uploaded (sent) outside of the 15 minutes Stated in item 7 will not be considered for checking!!!***
9. ***Only one pdf file uploaded will be considered for checking!!!***

Good Luck!

**Task 1. (4 points)** Why analog-to-digital and digital-to-analog converters are used in RTS? What is sampling? How the sampling frequency depends on the controlled object?

ADC/DAC are necessary because computers deal with discrete information, and controlled objects are characterized by continuous signals. Sampling means getting with some frequency values of a continuous signal. Sampling frequency shall be at least twice the highest frequency of the signal.

**Task 2. (9 points)** Consider 2-address machine, with 4 floating-point registers, R1, R2, R3, R4, available, and instruction set below

Load R,mem// load register R from memory cell, mem

Sto R, mem// store content of register, R, to memory cell, mem

Add R1,R2// add content of registers R1 and R2, result

 //is saved in R1

Sub R1,R2// subtract from register, R1, content of register R2; result is

 //saved in R1

Mul R1, R2// multiply register, R1, and register R2, ~~mem~~, content;

 //result is saved in R1

Div R1, R2 //divide content of register R1 by content of register R2; result is

//in R1; floating-point division result has at most 7 decimal digits

//in the mantissa

Assume that memory cell allocated to variable, x, is x.

For the expression $y=\frac{z}{y+z-w}∙\left(w-y\right)+z$

1. Write the code to implement it (4 points)

Load r1, w//r=w

Load r2, y//r2=y

Sub r1, r2//r1=r1-t2=w-y

Load r3,z/r3=z

Mul r1,r3//r1=r1/r3=(w-y)\*z

Add r2,r3//r2=r2+r3=y+z

Load r4, w//r4=w

Sub r2,r4//r2=r2-r4=y+z-w

Div r1,r2//r1=r1/r2=(w-y)\*z/(y+z-w)

Add r1,r3//r1=r1+r3=(w-y)\*z/(y+z-w)+z

Sto r1,y//y=r1=(w-y)\*z/(y+z-w)+z

1. Trace the code assuming the floating-point w=3E2, y=4E1, z=5E-1: show state of the memory cells allocated for the variables and state of the registers initially and after each instruction completion. In the operations results, keep 7 significant digits in the fractional part. (5 points)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Instruction | R1 | R2 | R3 | R4 | w | y | z |
|  | Load r1, w //r=w | 3e2 |  |  |  | 3e2 | 4e1 | 5e-1 |
|  | Load r2, y //r2=y |  | 4e1 |  |  |  |  |  |
|  | Sub r1, r2 //r1=r1-t2=w-y | 2.6e2 |  |  |  |  |  |  |
|  | Load r3,z //r3=z |  |  | 5e-1 |  |  |  |  |
|  | Mul r1,r3 //r1=r1\*r3=(w-y)\*z | 1.3e2 |  |  |  |  |  |  |
|  | Add r2,r3//r2=r2+r3=y+z |  | 4.05e1 |  |  |  |  |  |
|  | Load r4, w//r4=w |  |  |  | 3e2 |  |  |  |
|  | Sub r2,r4//r2=r2-r4=y+z-w |  | -2.595e2 |  |  |  |  |  |
|  | Div r1,r2//r1=r1/r2=(w-y)\*z/(y+z-w) | -0.5009634e0 |  |  |  |  |  |  |
|  | Add r1,r3//r1=r1+r3=(w-y)\*z/(y+z-w)+z | -0.9633911e-3 |  |  |  |  |  |  |
|  | Sto r1,y//y=r1=(w-y)\*z/(y+z-w)+z |  |  |  |  |  | -0.9633911e-3 |  |

**Task 3. (9 points)** Use PLA to create a ROM with 4 memory cells containing the following numbers in the given order: 15, 12, 13, 14. For each number, specify its address in your ROM. Each memory cell shall have one and the same number of bits, and this number of bits shall be minimal possible to accommodate the numbers. What is the number of bits of each memory cell? Why? What is the number of address bits? Why?

Hint: PLA example from the Lecture notes

Two address bits, since 4 numbers; 4 content bits, since the numbers are <16.





**Task 4. (4 points)**. How preemptive multitasking is organized in a system with a single CPU? What data structures are used to support multi-tasking? What is the most important information kept in the data structures? How do they help organizing the multitasking?

Preemptive multitasking is organized by allocating CPU to a process for a time quantum, and switching to the next process when the time quantum expires. Task control blocks (TCB) are used to support the multi-tasking. The most important information kept in a TCB are the Instruction pointer and Processor status word allowing resuming the process from the point of interruption.

**Task 5. (13 points).** Consider the code and diagram below

|  |  |
| --- | --- |
| Driver{ while(1){ if(data\_for\_I/O){ prepare(command);  V(busy); P(done);} }}Controller{while(1){ P(busy);  exec(command); V(done); }} |  |

Assume, there are $N$ Device controllers interacting with one driver in a mutually exclusive mode. Write the code for the Driver and $i$-th Controller providing their correct interaction using binary semaphores, where $i$ is from 1$..N$. Specify initial values of the semaphores you use.

Sema busy=0 /\*closed\*/, done=0/\*closed\*/;

Driver{

 While(1)

 If(data\_for\_IO){

 Prepare(command); V(busy); P(done);

 }

}

Controller-i{while(1){

 P(busy); exec(command); V(done);

 }

}

**Task 6 (10 points).** Calculate processor utilization and hyper-period for the following task set:

|  |  |  |
| --- | --- | --- |
| **Task#** | **E** | **P** |
| **1** | **2** | **6** |
| **2** | **1** | **7** |
| **3** | **2** | **8** |
| **4** | **1** | **9** |

$$U=\frac{2}{6}+\frac{1}{7}+\frac{2}{8}+\frac{1}{9}=\frac{4}{9}+\frac{1}{7}+\frac{1}{4}=\frac{37}{63}+\frac{1}{4}=\frac{148+63}{252}=\frac{211}{252}=0.837$$

Hyperperiod=6\*7\*4\*3=42\*12=420+84=504.

**Task 7. (17 points).** Build a Rate-Monotonic schedule for tasks:

|  |  |  |
| --- | --- | --- |
| Task# | E | P |
| 1 | 2 | 12 |
| 2 | 1 | 6 |
| 3 | 3 | 9 |

What are the utilization and hyperperiod for this set of tasks? Are the deadlines met by the RM schedule? Explain why they are met (not met)?

$$U=\frac{2}{12}+\frac{1}{6}+\frac{3}{9}=\frac{6}{36}+\frac{6}{36}+\frac{12}{36}=\frac{24}{36}$$

Hyperperiod=12\*3=36

2-1

3-1

3-1

 3-1

1-1

 1-1

 2-2

3-2

 3-2

 3-2

2-3

 1-2

 1-2

 2-4

 3-3

 3-3

3-3

2-5

 1-3

1-3

 3-4

 3-4

 3-4

 1-6

i-j denotes j-th release of i-th task

**Task 8 (17 points)**. Assume, three parallel processes, A1, A2, A3, infinitely run. Processes A2, A3 wait for a signal from A1, execute, and signal A1 each after completion. A1, on getting signals from A2, A3, executes, and signals them again once execution finished, as shown in the figure below:

Write a pseudo-code using binary semaphores to provide necessary synchronization of the processes A1, A2, A3. Define necessary data structures. Show initial settings of the semaphores you use.

|  |
| --- |
| Sema s1=0/\*closed\*/, s2=0/\*closed\*/,s3=0/\*closed\*/,mutex=1/\*open\*/;Int count=0; |
| A1{ while(1){ V(s2); V(S3); If(count<2) P(S1); Count=0; }} | A2{ while(1){ P(s2); P(mutex); Count++; If(count==2)  V(s1) V(mutex) }} | A3{ while(1){ P(s3); P(mutex); Count++; If(count==2)  V(s1) V(mutex) }} |

**Task 9 (17 points)**. Assume that a system has 4 processes and resources of two types: special processors (totally, 10 SP), and memory blocks (totally, 12 MB) and . Processes’ resources required and maximal requirements are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Process | SP required | SP maximal requirement | MB required | MB maximal requirement |
| 1 | 2 | 5 | 2 | 6 |
| 2 | 1 | 6 | 2  | 5 |
| 3 | 2 | 4 | 3  | 6 |
| 4 | 2 | 7 | 1 | 4 |

Currently, each process has no any SP or MB allocated to it. Use the Banker algorithm to decide on safety of granting required resources. Show all the steps of the algorithm you make to come to your decision.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Process | SP required | SP maximal requirement | SP max need left | MB required | MB maximal requirement | MB max need left |
| 1 | 2 | 5 | 3 | 2 | 6 | 4 |
| 2 | 1 | 6 | 5 | 2  | 5 | 3 |
| 3 | 2 | 4 | 2 | 3  | 6 | 3 |
| 4 | 2 | 7 | 5 | 1 | 4 | 3 |
|  |  | Total currently available SP: | 3 |  | Total currently available MB: | 4 |

If allocating all required resources, then

SP left=10-(2+1+2+2)=10-7=3;

MB left=12-(2+2+3+1)=12-8=4

P1 max need (Sp=3,Mb=4) can be satisfied, and no resources left

S=(P1)

After P1 termination it releases (Sp=5, Mb=6), and resources left are (Sp=5, Mb=6)

Then P2 max needs can be satisfied, and resources left are (Sp=0, Mb=3)

Hence, S=(p1, p2)

After P2 termination, (Sp=6, Mb=8)

Then P3 max needs can be satisfied, and resources left are (Sp=4, Mb=5)

Hence, S=(p1, p2,p3)

After P3 termination, (Sp=8, Mb=11)

Then P4 max needs can be satisfied, and resources left are (Sp=3, Mb=8)

Hence, S=(p1, p2,p3,p4)

After P4 termination, (Sp=10, Mb=12).

Hence, such resources allocation is safe, and safe sequence is S=(p1, p2,p3,p4).