1. Mark the **true** statements **with** an **X** and the **false** statements **with a –** in the last column. Each good answer is 0.5 points, each wrong answer -0.5 points, and an empty answer 0 point. Total of 0 to 3 points.

3p

Off-line I/O handling means, that the I/O operations of a program are executed on a separate processor than the other instructions of the same program.	
The spooling system also uses separate processors for I/O instructions than the normal program execution.	
A job is a set of different programs designed to be run in parallel.	
An operating system call usually requires a hardware mechanism to change between privilege levels in the processor.	
The operating systems usually implement logical IO device handling functions.	
A personal computer (PC) starts executing code from RAM immediately after power-up.	

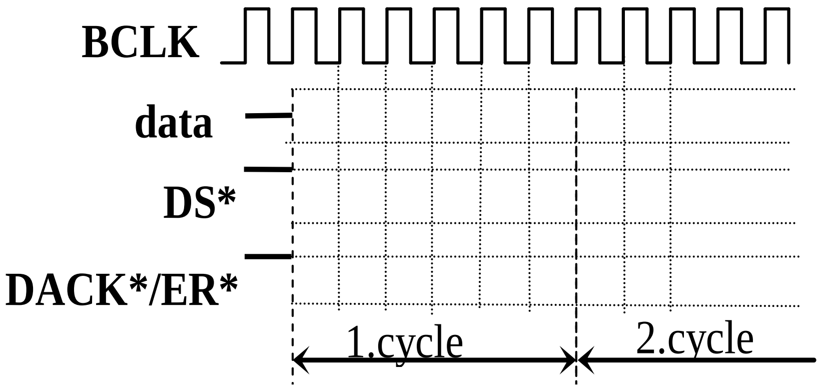
2. A fully interlocked semi synchronous protocol has a clock frequency of 40MHz.

3p

a) Give the **shortest possible time of a read cycle** with this protocol?

T_{WCmin}= ....

b) Draw the data and handshake signals in the following figure in case **o666f a read cycle**.



3. What can be done to avoid a deadlock? List at least 4 possible options: (0.5 points each)

2p

....

....

....

....

4. Draw the state diagram of a process in a multitasking operating system with a preemptive scheduler.

3p

5. There are three processes in a system (P1, P2, P3). Each consists of the operations given in the table. The processes run asynchronous to each other and do not “know” of each other at all. Only the resource request and release operations are presented in the table (RQ: request for a resource, RL: release a resource). There is only one instance of each resource (R1, R2, R3).
If a process in currently in the line that has a RQ() operation in it, then the process has requested for a resource, but has not yet taken control of it.

a./ Draw the resource allocation graph in the following state:
{P1 in line C; P2 in line F; P3 in line F}

b./ In which state can this system get into a deadlock?
(give one example, with the associated resource allocation graph)

3p

	P1	P2	P3
A	....	....	....
B	RQ(R1)	RQ(R2)	RQ(R3)
C	....	....	....
D	RL(R1)	RQ(R3)	RL(R3)
E	....	....	....
F	RQ(R2)	RQ(R1)	RQ(R1)
G	....	....	....
H	RQ(R3)	RL(R1)	RQ(R2)
I	....	....	...
J	RL(R3)	RL(R3)	RL(R2)
K	RL(R2)	RL(R2)	...
L	...	...	RL(R1)

6. There are the following resources in a system:

R1: **6** instances,
R2: **8** instances,
R3: **10** instances

There are 4 processes in this system: P1, P2, P3, P4.

	Maximal usage			**Actual usage**		
	R1	**R2**	**R3**	**R1**	**R2**	**R3**
P1	4	2	4	2	1	3
P2	2	2	1	0	1	1
P3	2	4	3	1	2	2
P4	3	7	4	1	1	2

4p

a. If the processes are in the actual state described in the upper table, i**s this state safe?**

b. From this state, **P1** requests **one of each R1,R2,R3** {P1: RQ(R1:1,R2:1,R3:1)}. If the resource manager is using **the banker's algorithm**, **would** it **serve this request** in the actual state **or not?**

7. In a virtual memory system, the physical memory access time is 40 ns, the page fault rate is $5*10^{-6}$ and the average service time of page faults is 70 ms.

2p

a. Give the effective memory access time.

b. Is this system in the state of thrashing? *(Hint: What would be the page fault rate at the limit of thrashing?)*

8. An operating system runs 3 processes: (P1, P2, P3).
The arrival times of processes: P1: **0**, P2: **3**, P3: **10**
Next CPU burst times: P1: **13**, P2: **3**, P3: **9**
Draw the Gantt chart for each of the following scheduler algorithms **and determine the average turnaround time**:

a. SJF
b. SRTF (shortest remaining time first)
c. RR(8)

6p

9. A multiprogrammed system runs processes P1 and P2 cyclically.
Use semaphore(s) to synchronize the processes so that the system never reads both sensors at the same time, and the result of Sensor1 is always printed first.

Modify the following code only by adding semaphore operations in some of the blank lines ($P(S_?)$ and $V(S_?)$).

Give the starting value of the semaphore(s).

4p

```
P1:                                 P2:
{                                   {
  int i;                                int i;

  i=GetSensorData(Sensor1);             i = GetSensorData(Sensor2);



  PrintToDisplay(i);                    PrintToDisplay(i);



}                                   }
```