**Assignment 1**

**PART 1**

Q.1 Explain the difference between relative delay and absolute delay using your own understanding.

Relative Delay:

* It is used delaying the task for certain amount of time.
* It can be accessed by providing command “Delay”.
* Expiration time for relative delay is rounded up to the nearest clock tick.
* Form of the relative delay:- Delay Duration\_expression.
* This Duration\_expression should be provided by the user in Minutes.

Absolute Delay:-

* It is used for delaying the task for a certain of amount of time by giving a reference time.
* It can be accessed by providing command “Delay until”.
* Form of the relative delay:- Delay Time\_expression.
* This Time\_expression should be provided by the user in by accessing internal clock.

Q2 Suppose we know the exact execution times of F1, F2 and F3. Is it possible to use relative delay for avoiding drift? Why?

For example, Suppose Relative delay is having delay of 0.5 sec, it means that delay value would be minimum of 0.5 sec. so it can be also greater than 0.5. Therefore by using the relative delay we can avoid drift by some extent, but it’s not possible to completely eliminate the drift

**PART 2**

Q.1 How did you synchronize the watchdog task with the start and the end of a F3execution?

Provide entry points to watchdog task for both times of start and at the end of process F3.We can note the time of these events in the watchdog for further usage.

1. Q.2 Does it makes sense to use absolute delay in watchdog timeouts? Why/why not?

It does make sense to use absolute delay in watchdog timeouts, because

Watchdog timeouts used so that we can return to the particular instruction line defined by the user when there is no input received or when the condition for the watchdog timer gets satisfied.

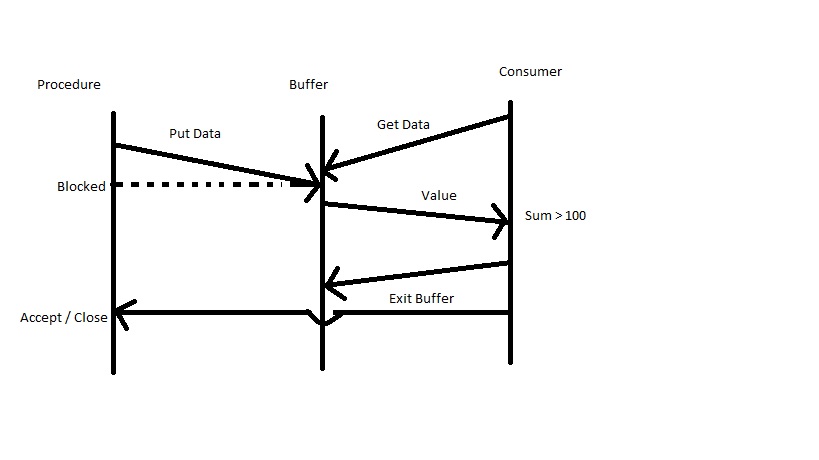
1. Q.3 Explain the way you re-synchronize the cyclic executive when F3 misses its deadline.

A cyclic executive is a control structure or program for explicitly interleaving the execution of several periodic processes on a single CPU; the interleaving is done in a deterministic fashion so that execution timing is predictable.

So when F3 misses its deadline it is an error, called frame overrun. So after

**PART 3**

Q.1 Show by a concrete scenario that the producer-consumer-buffer program using blocking rendezvous communication can have deadlocks and explain the mistake that can cause it.



1. Procedure starts creating arbitrary number and put it in a buffer.
2. Consumer send a “Get Data” command and receives arbitrary number from buffer.
3. This process continues till the summation at consumer end is 100.
4. After this point Consumer Exit the buffer, because of that Procedure gets block automatically.
5. In the next state Consumer Sends Exit signal to procedure but, procedure is already blocked by the buffer.
6. This situation is known as deadlock.

**PART 4**

1. Q.1 Does the producer-consumer-buffer program using protected object suffer from any potential deadlock? If not, please explain the main reason behind it.

The producer-consumer-buffer program using protected object does not suffer from any potential deadlock. Because as soon as Producer put data in buffer it creates next number and get block. Now the consumer will get data from buffer. When the summation gets completed Consumer put exit signal to buffer. As Buffer is empty it takes the value of procedure. Because of this reason t he producer-consumer-buffer program using protected object does not suffer from any potential deadlock