**Homework Wan Huzaifah bin Wan Azhar**

**Answer:**



Dx Thread 0

0 -

-1 1000 sub 1, dx

-1 1001 test 0 dx

-1 1002 jgte .top

-1 1003 halt



Dx

3

2

2

2

1

1

1

0

0

0

-1

-1

-1

-1

3

2

2

2

1

1

1

0

0

0

-1

-1

-1

-1



Dx

3

2

2

2

3

2

2

2

2

1

1

2

1

1

1

0

1

1

1

0

0

0

-1

1

0

-1

-1

-1

0

0

0

-1

-1

0

-1

-1

-1

-1

-1

* Interrupt frequency changes thread every three instruction.
* As such, it takes a bit of longer to terminate the program
* Changing interrupt interval does not change much on data as each thread uses its local register dx



2000

0  
0  
0  
1  
1  
1  
1  
1



2000

0  
0  
0  
1  
1  
1  
1  
1  
1  
1  
2  
2  
2  
2  
2  
2  
3  
3  
3  
3  
3  
3

---halt  
3  
3  
4  
4  
4  
4  
4  
4  
5  
5  
5  
5  
5  
5  
6  
6  
6  
6  
6

* Final value of value is 6
* Each thread loops three times because loop counter, bx is set to 3 on each thread stack.



* It is possible to tell what the final value will be but it is difficult.
* Timing of interrupt matters because you don’t want OS to switch to another thread when the existing thread runs code in critical section.
* In essence, you don’t want timing interval to be too short so that thread will be interrupt in critical section.
* Interrupt is safe if it is not in critical section.
* Critical section is from instruction 1000 to 1002. Another thread must not run in this section when another thread is already running.



* Final value of address 2000 in seed 1 is 1
* Timing interval of 3 gives correct answer of 2.



* Timing interval of multiple of 3 gives the correct answer (e,g, 3, 6, 9) of 2000.
* This is because as critical section have three instruction, it needs to be run without interruption.
* Anything that is not multiple of 3 will gives different result.



2000 ax

0 1  
0 1

0 1

1 1

1 1

-- halt

1 0

1 0

1 0

1 0

1 0

1 0

* Final value of 2000 will always be 1 regardless of which threads run first.
* This is because the second thread will always wait for the first thread to finish before it can run its code.
* First thread is called signaller and second thread is waiter. Waiter is waiting for signaller before starting.



* The first thread is a waiter thread. It waits until there is a value of 1 in memory address 2000.
* So it will do a continuous loop until cx is not equal to 1
* When the OS switch to second thread, it saves 1 into 2000 so that the first thread can do its job.
* Changing interrupt frequency will not change the final value, as there is only one critical section, which is address 1006.
* Even if interrupt occurs before 1006 or after will not change the final value of 2000.
* No, the thread is not efficiently using the CPU as the first thread is busy waiting. The better approach is to sleep the thread and wake it.