## Q1 Paging 4 Points

#### Q1.1 2 Points

Let us say that we have a 32 bit architecture with 4KB pages. How many page table entries are required to cover the full address space of a process? Assume that we have a flat page table i.e, not a hierarchical page table. (Take log base 2 of your answer and respond).

$\Gamma$				 -	 _	-	-	-	-	_	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	7
į.																																							ı
i.	•	20	`																																				i
	-	۷,	J																																				i
i .																																							1
1																																							ı

#### Q1.2 1 Point

Dirty bit signifies that the page was written.

False

True

#### Q1.3 1 Point

x86 hardware immediately raises a page fault when a page cannot be found in TLB.

False

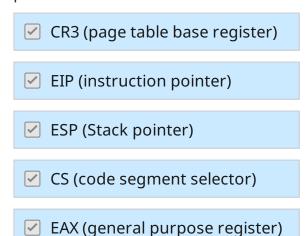
True

#### **Q2 Context switch**

#### 2.5 Points

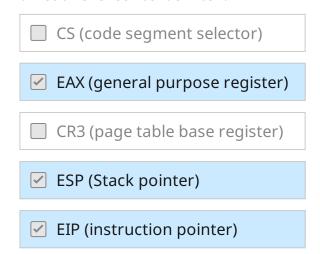
#### Q2.1 1.25 Points

For each x86 register, mark whether the register is saved and restored upon a process-level context switch:



### Q2.2 1.25 Points

For each x86 register, mark whether the register is saved and restored upon a thread-level context switch:



#### Q3 CPU control

3.5 Points

Q3.1									
1	<b>Point</b>								

On a single CPU machine, when a user process is running, the OS is not running.

- False
- True

# Q3.2

1.5 Points

Briefly describe how an OS can gain back control from a user process that is just running an infinite loop while(1); i.e, the user process is not doing any IO or making system calls. Assume that the machine has a single CPU.

The timer interrupt (which is a hardware interrupt) let's the OS take the control.

# Q3.3

1 Point

Code running in ring 3 can modify the interrupt descriptor table.

- False
- True