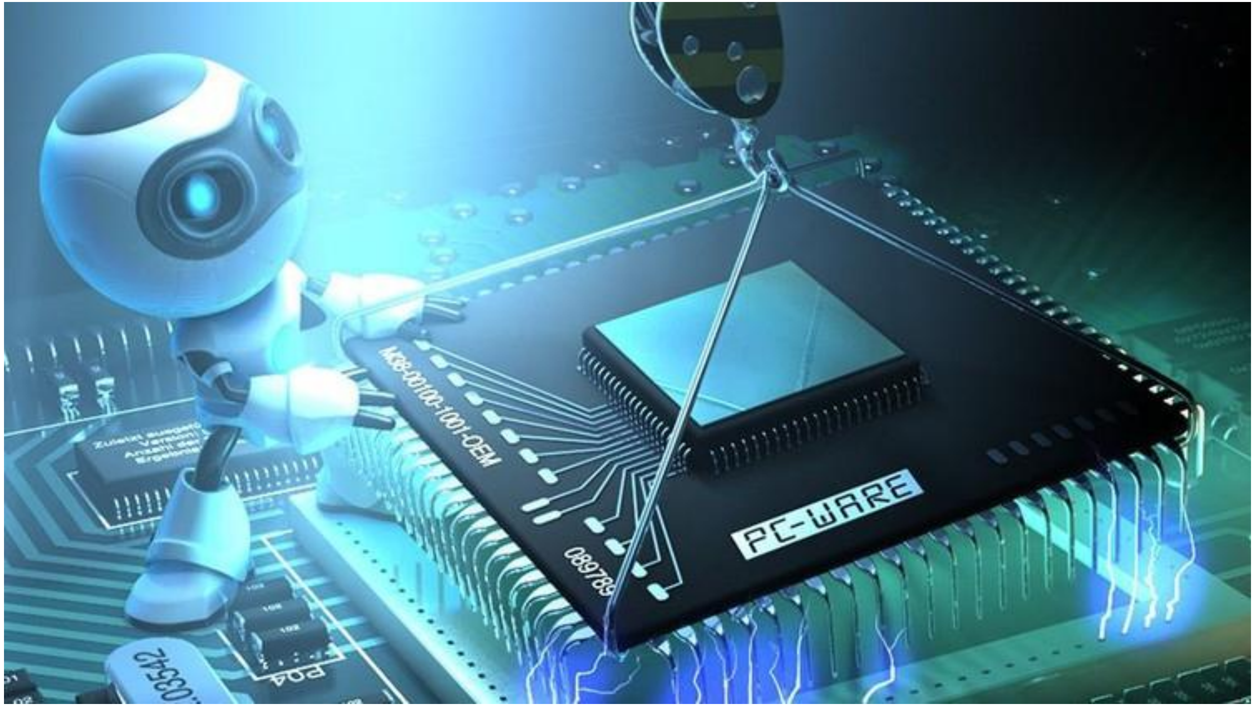
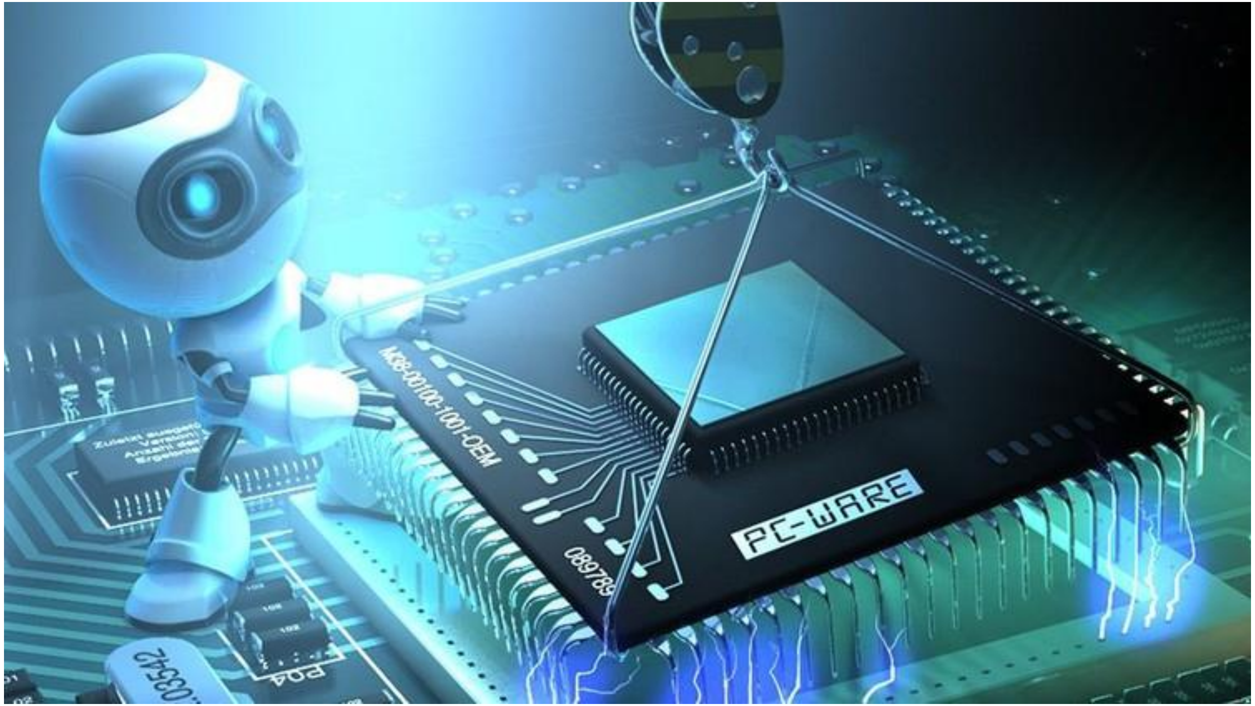
Exam 4



Exam 4 - Problem 1

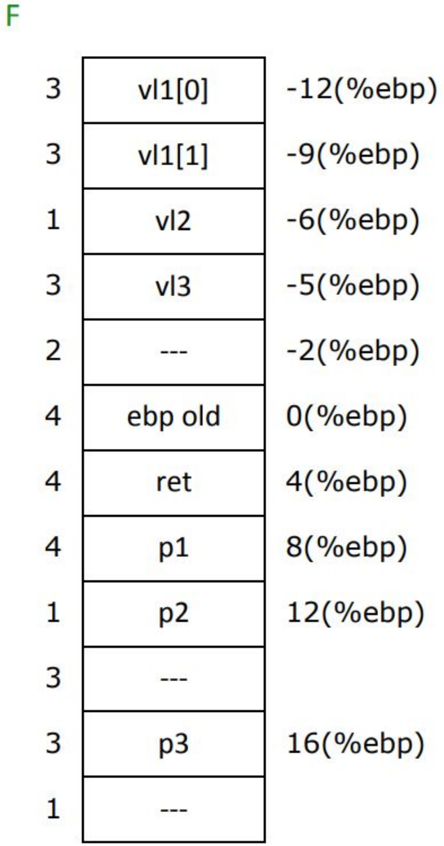


Exam 4 - Problem 1

Given the following code in C:

```
typedef struct {
   char a[3];
} s1;
char F(s1 p1[2], char p2, s1 p3){
   s1 vl1[2];
   char vl2;
   s1 vl3;
   ...
}
```

Answer the following questions.



F

3	vl1[0]	-12(%ebp)
3	vl1[1]	-9(%ebp)
1	vl2	-6(%ebp)
3	vl3	-5(%ebp)
2	---	-2(%ebp)
4	ebp old	0(%ebp)
4	ret	4(%ebp)
4	p1	8(%ebp)
1	p2	12(%ebp)
3	---	
3	p3	16(%ebp)
1	---	

Exam 4 - Problem 1

a) Draw how the structure `s1` and the activation block of the function `F` would be stored in memory, clearly indicating the displacements and the size of all the fields.

b) Translate the following statement to x86 assembler, assuming it's inside the `F` function:

```
return F(vl1,vl2,p3);
```

Note: chars are returned in %al
However, you can also use %eax to return this value

Exam 4 - Problem 1

a)

s1 (3 bytes)

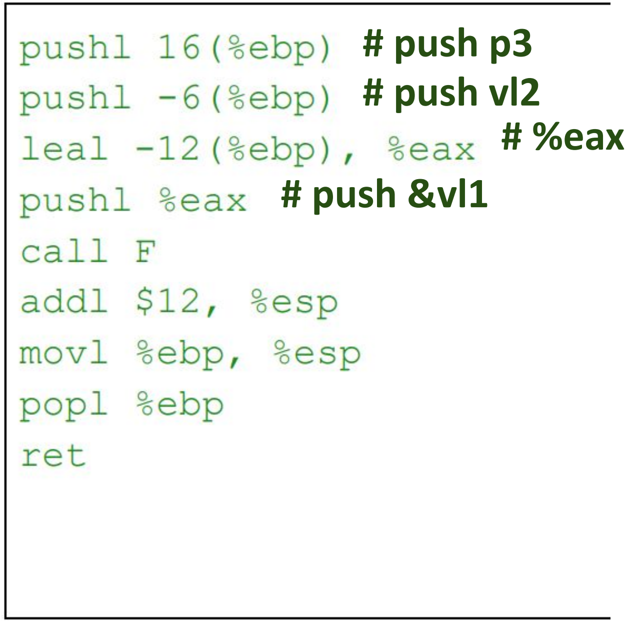
1	a[0]	+0
1	a[1]	+1
1	a[2]	+2

F

3	vl1[0]	-12(%ebp)
3	vl1[1]	-9(%ebp)
1	vl2	-6(%ebp)
3	vl3	-5(%ebp)
2	---	-2(%ebp)
4	ebp old	0(%ebp)
4	ret	4(%ebp)
4	p1	8(%ebp)
1	p2	12(%ebp)
3	---	
3	p3	16(%ebp)
1	---	

Exam 4 - Problem 1

b)



```
pushl 16(%ebp)      # push p3
pushl -6(%ebp)      # push vl2
leal -12(%ebp), %eax    # %eax = &vl1
pushl %eax      # push &vl1
call F
addl $12, %esp
movl %ebp, %esp
popl %ebp
ret
```

```
return F(vl1,vl2,p3);
```