1.

paddb: Sumar un paquete de bytes (conjunto) de un registro mmx con otro registro mmx.

movdqa: Mover de un double quadword del operando de origen (primero) al segundo. Cuando se accede a memoria, se debe estar alineado a 16-bytes o se generara una excepcion de tipo (GP) "general protection".

movdqu: Mover de un double quadword del operando de origen (primero) al segundo. En esta no hay necesidad de alineamento.

emms: Setea a vacio un (todo 1s en coma flotante) todos los registros x87. No tiene operandos. Se usa para restaurar estos registros ya que se comparten partes con los mmx.

2.

Se usa para alinear una variable o structs en C usando el atributo aligned.
ex: float vect[4] __attribute__((aligned(16)));/* aqui nos aseguramos que cada elemento vect esta alinado a 16*/

3.

```
procesar:
        movl 8(%ebp), %esi      # esi <- mata
        movl 12(%ebp), %edi     # edi <- matb
        movl 16(%ebp), %eax
        sall  $2,%eax           # eax <- n*n
        xor %ecx, %ecx          # ecx = 0 usado para index
bucle:
        cmpl %ecx, %eax
        jge fi                  #index < n*n
        movb (%esi), %bl
        salb   $4, %bl          #bl = mata[i*n+j]*16
        movb %bl, (%edi)        #matb[i*n+j]=(mata[i*n+j]*16)
        inc %esi
        inc %edi                #esi i edi + stride (1)
        inc %ecx
        jmp bucle
fi:
```

4.

Guardando ese valor en la region .data i luego cargandolo con la instruccion movdqu con el tag de dicho valor.

5.

```
procesar:
        movl 8(%ebp), %esi      # esi <- mata
        movl 12(%ebp), %edi     # edi <- matb
        movl 16(%ebp), %eax
        sall  $2,%eax           # eax <- n*n
        xor %ecx, %ecx          # ecx = 0 usado para index
bucle:
        cmpl %ecx, %eax
        jge fi                  #index < n*n
        movdqu (%esi), %mm1
```

```
        psllW %mm1, $4
        movdqu %mm1, (%edi)
        addl $16, %esi
        addl $16, %edi
        addl $16, %ecx
        jmp bucle
fi:
```

6.

```
and $FFF0H, %reg
```