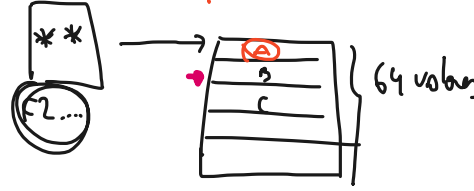


Final 2/12/20

Periferico = memoria

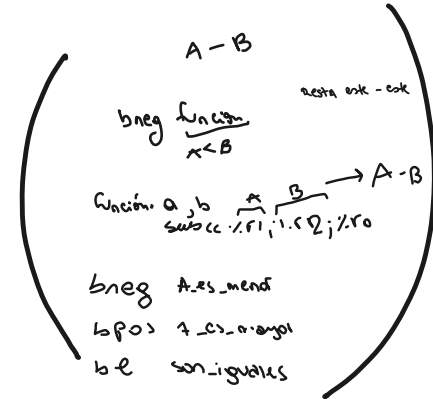
perif = A (1 pos)



Ejercicio 2
Un periférico mapeado en la dir 0x10000000 entrega datos de 32 bits en formato de punto flotante simple precisión. Escribe un programa que lee 64 valores de ese periférico y devuelve por sí mismo los cambios de valores positivos entregados por el mismo. la lectura del periférico...

c=1 -> numero negativo
c=0 -> numero positivo

```
1100 0010 0001 0000 0000 0000 1010 0001
1000 0000 0000 0000 0000 0000 0000 0000
...
0100 0010 0001 0000 0000 0000 1010 0001
C=1
Z=0
```



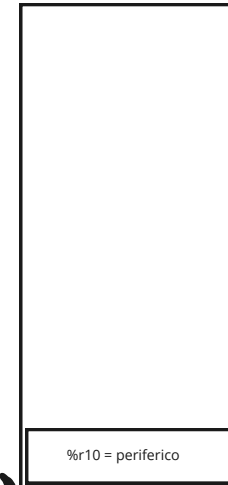
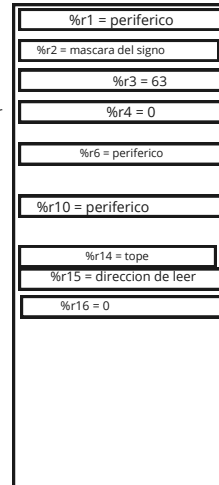
bit-signo . equ 00000001 0000 0000 0000 0000 0000 0000
sethi bit-signo, %r2, 1 #2 contiene la mascara 10 0000 0000 0000 0000 0000 0000 0000
addi %r3, %r2, %r3 #1 mi para leer (r3)
addi %r0, %r0, %r4 #4 es mi acumulador

bcc branch on carry clear
bcs branch on carry set

addi %r3, %r2, %r3 #1 mi para leer (r3)
addi %r0, %r0, %r4 #4 es mi acumulador
for: addcc %r3, -1, %r3
bneg fin
addi %r0, %r0, %r4 #4 es mi acumulador
call leer

pop %r6 #1 leído
andcc %r6, %r2, %r0
bcc positivo mira el carry y salta si no esta prendido el carry
ba for ba: salta always

iterador
acumulador



%r14
tope de la pila

se guarda el valor en memoria de %r1
leer: ld %r1, %r10
push %r10
jmp %r14, %r0
macro push reg
addi %r4, %r4, 4
st reg, %r4
endmacro
macro pop reg
ld %r14, %r4
addi %r4, %r4, 4
endmacro
positivo: addi %r4, %r4, 1
fin: push %r3
jmp %r14, %r0
end