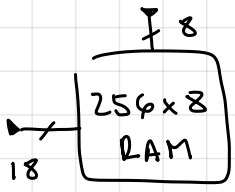
1) Un microprocesador puede direccionar 1 Mbyte de RAM, ancho de palabra, 8 bits. Es decir: bus de address, 20 bits y bus de direcciones 8 bits.
Si se cuenta con una cantidad ilimitada de chips de RAM de 256 kilobytes y de decodificadores, diseñar el mayor mapa de memoria posible para ese microprocesador. Indicar la primera y la última dirección de la RAM que mapea las direcciones más altas.

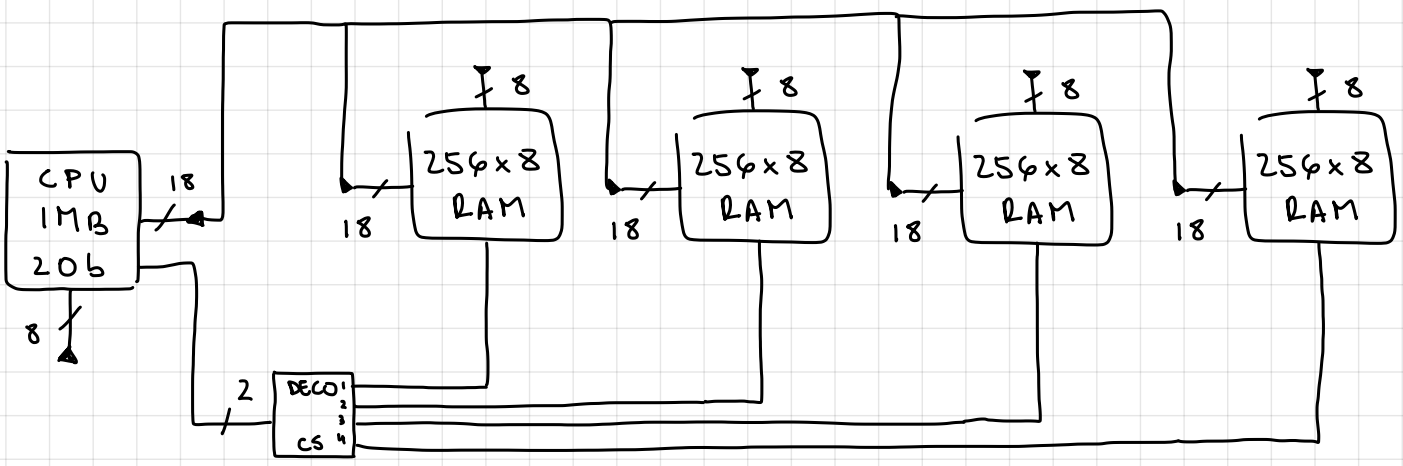


tengo que expandir a lo alto.

MAPA de Memoria. $\log_2(18) = 4,17 \Rightarrow$ necesito 4x5 bits

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MÓDULO 0
3	F	F	F	F	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	MÓDULO 1
7	F	F	F	F	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MÓDULO 2
B	F	F	F	F	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
C	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	MÓDULO 3
F	F	F	F	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

8 bits



las direcciones más altas son C 0 0 0 0 y F F F F F.
primera última

2)

En un procesador ARC el registro Program Counter apunta a la siguiente instrucción guardada en RAM:

ld %r10, %r1, %r7

Detallar los pasos de microprograma que la decodifican y para cada uno de ellos indicar los valores presentes en las entradas y en las salidas de cada uno de los siguientes bloques funcionales:

* multiplexor de direcciones de la memoria de control
* incrementador de direcciones de la memoria de control
* decodificadores de los buses A B y C
* multiplexores que intervienen en la decodificación de los buses A B y C
* multiplexor de datos del bus C
* bus de datos A B y C
* entrada de direcciones del módulo de memoria RAM

Considere que antes de la ejecución de esta instrucción:
%r10=3000 , %r1=8, %r7=1

(Nota: es idéntico al ejercicio 8 del TP7)

ld $\underbrace{\%r10}_{rs1}, \underbrace{\%r1}_{rs2}, \underbrace{\%r7}_{rd}$

Construyo el registro IR:

op = 1 1
rd = 0 0 1 1 1
op3 = 0 0 0 0 0 0
rs1 = 0 1 0 1 0
i = 0
rs2 = 0 0 0 0 1

decode = $1\ 1\ \underbrace{1}_{op}\ \underbrace{0\ 0\ 0\ 0\ 0\ 0\ 0\ 0}_{op3} = 1792_{10}$

Luego, se procesa la instrucción 1792 según lo interpretado por el decoder:

1792 || · R[temp0] ← ADD(R[rs1], R[rs2]); IF $\overbrace{R[IR[13]]}^{=0}$ THEN GOTO 1794;

A = x
AMUX = 1
B = x
BMUX = 1
C = 1 0 0 0 0 1
CMUX = 0
RD = 0
WR = 0
ALU = 1 0 0 0
COND = 0 0 0
JUMP ADDR = 1 1 1 0 0 0 0 0 0 0 1

1793 || · R[ra] ← AND (R[temp0], R[temp0]) ; READ ; GOTO 2047 ;

A = 1 0 0 0 0 1

AMUX = 0

B = 1 0 0 0 0 1

BMUX = 0

C = x

CMUX = 1

RD = 1

WR = 0

ALU = 0 1 0 1

COND = 1 1 0

JUMP ADDR = 0 1 1 1 1 1 1 1 1 1 1 1

2047 || · R[pc] ← INCPC (R[pc]) ; GOTO 0;

A = 1 0 0 0 0 0

AMUX = 0

B = x

BMUX = x

C = 1 0 0 0 0 0

CMUX = 0

RD = 0

WR = 0

ALU = 1 1 1 0

COND = 1 1 0

JUMP ADDR = 0 0 0 0 0 0 0 0 0 0 0 0

Luego, se lee la dirección de memoria 3008.

3) Escribir código que recibe a través de %r10 un número en punto flotante y devuelve su valor absoluto (también en punto flotante).

(Nota: es idéntico al ejercicio 15 del TP5)

```
.begin
    .org 2048
    ld [comparador], %r1
    and %r10, %r1, %r10        ! pone en 0 el bit más significativo
comparador: 7fffffff           ! 7fffffff = 0 111 1111 1111 1111 1111 1111 1111 1111
.end
```