

Sistemas Digitales Laboratorio

Semestre 2020-2
Sesión Laboratorio Semana 6
Profesor: Kalun José Lau Gan

1

Preguntas previas: (5minutos)

- Sobre constantes en VHDL (usar constant):
- Sobre alias, type, array, record en VHDL
- Explicación del item F de la PC1:

f) Describir la siguiente señal interna en términos de VHDL: (1p)

PC1_REG					
1	0	0	1	0	1
bit0	bit1	bit2	bit3	bit4	bit5

signal PC1_REG : std_logic_vector(0 upto 5);

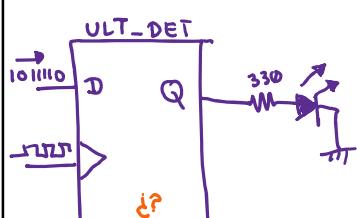
2

Agenda:

- Ejemplo de Máquinas de estado en VHDL (Mealy)
- Memorias

3

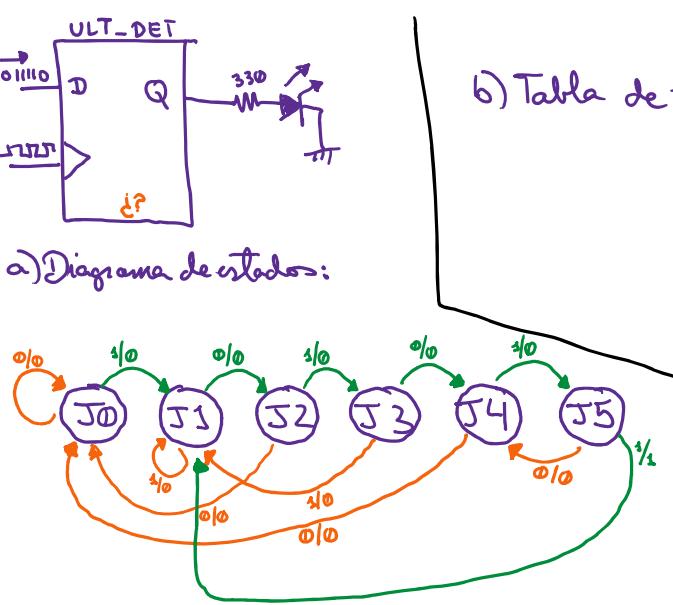
Ejemplo: Detector de secuencia $\rightarrow 101011$ en MEF Mealy



b) Tabla de transiciones:

D	E.P.	E.S.	Q
0	J0	S0	0
1	J0	S2	0
0	J1	S1	0
1	J1	S0	0
0	J2	S3	0
1	J2	S3	0
0	J3	S4	0
1	J3	S1	0
0	J4	S0	0
1	J4	S5	0
0	J5	S4	1
1	J5	S1	1

a) Diagrama de estados:



4

Ejemplo: Detector de secuencia ->101011 en MEF Mealy

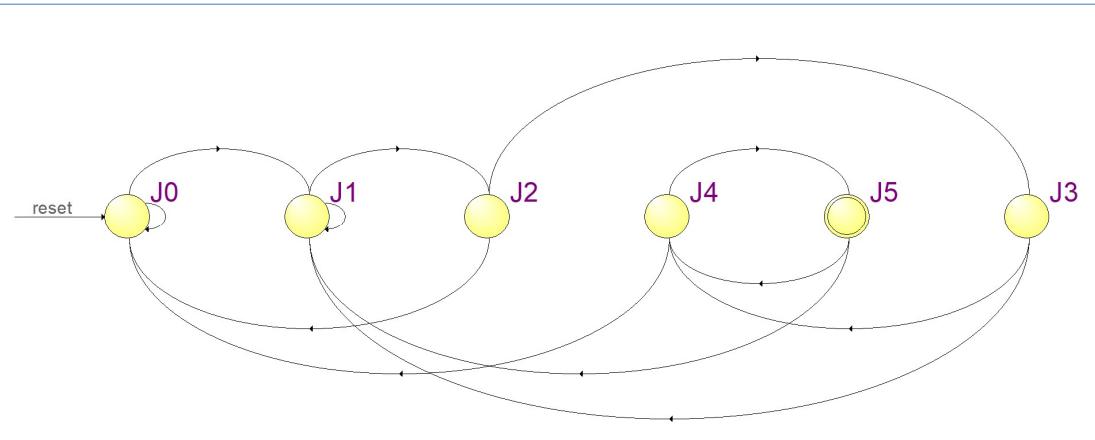
```

1 library IEEE;
2 use IEEE.std_logic_1164.all;
3 use IEEE.std_logic_arith.all;
4 use IEEE.std_logic_unsigned.all;
5
6 entity el_ultimo is
7 port ( clk, D: in std_logic;
8         Q: out std_logic);
9 end el_ultimo;
10
11 architecture flujo of el_ultimo is
12
13 TYPE estados is (J0, J1, J2, J3, J4, J5);
14 SIGNAL es_p: estados;
15
16 begin
17 process (clk)
18 begin
19 if rising_edge(clk) then
20 case es_p is
21 when J0 =>
22 if D = '0' then
23 es_p <= J0;
24 else
25 es_p <= J1;
26 end if;
27 when J1 =>
28 if D = '0' then
29 es_p <= J2;
30 else
31 es_p <= J1;
32 end if;
33 when J2 =>
34 if D = '0' then
35 es_p <= J0;
36 else
37 es_p <= J3;
38 end if;
39 when J3 =>
40 if D = '0' then
41 es_p <= J4;
42 else
43 es_p <= J1;
44 end if;
45 when J4 =>
46 if D = '0' then
47 es_p <= J0;
48 else
49 es_p <= J5;
50 end if;
51 when J5 =>
52 if D = '0' then
53 es_p <= J4;
54 else
55 es_p <= J1;
56 end if;
57 when others =>
58 es_p <= J0;
59 end case;
60 end if;
61 end process;
62
63 process(es_p, D)
64 begin
65 if es_p = J5 then
66 Q <= D;
67 else
68 Q <= '0';
69 end if;
70 end process;
71 end flujo;

```

5

Ejemplo: Detector de secuencia ->101011 en MEF Mealy



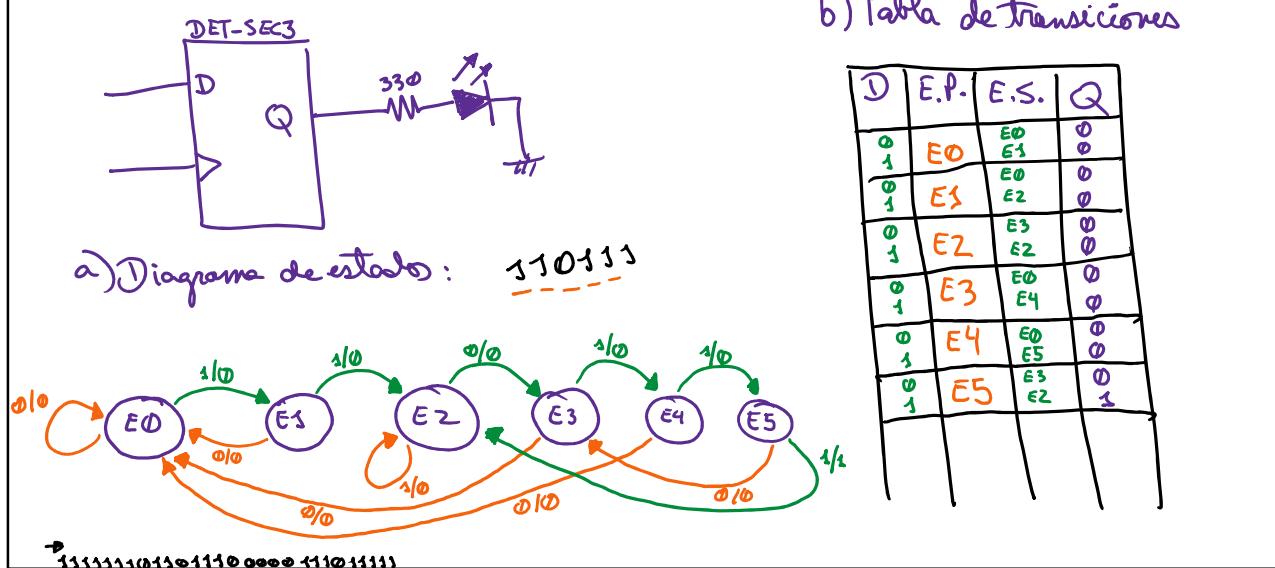
6

Ejemplo: Detector de secuencia ->101011 en MEF Mealy



7

Ejemplo: Detector de secuencia ->110111 con MEF Mealy



8

Ejemplo: Detector de secuencia ->110111 con MEF Mealy

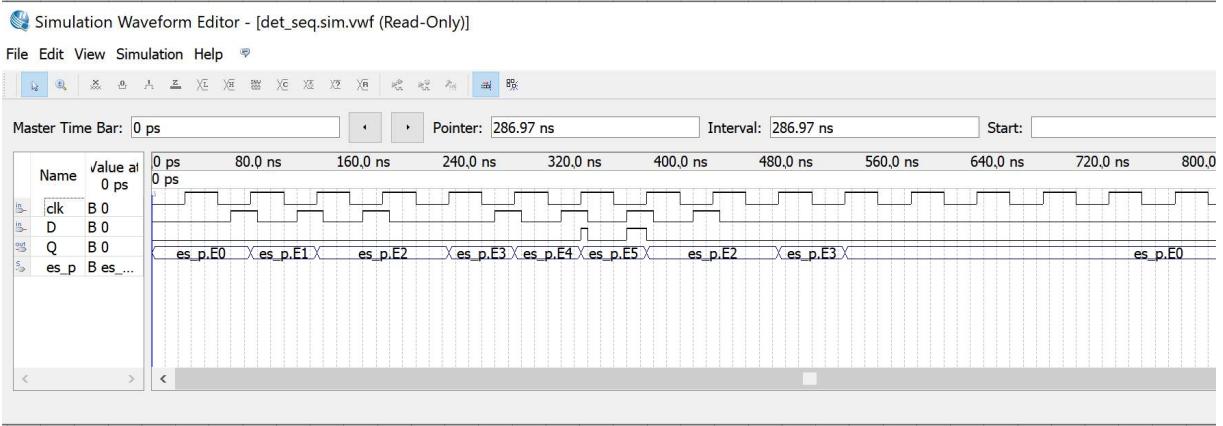
```

1 library IEEE;
2 use IEEE.std_logic_1164.all;
3 use IEEE.std_logic_arith.all;
4 use IEEE.std_logic_unsigned.all;
5
6 entity det_seq is
7   port ( clk, D: in std_logic;
8         Q: out std_logic);
9 end det_seq;
10
11 architecture flujo of det_seq is
12
13 TYPE estados is (E0, E1, E2, E3, E4, E5);
14 SIGNAL es_p: estados;
15
16 begin
17   process (clk)
18   begin
19     if rising_edge(clk) then
20       case es_p is
21         when E0 =>
22           if D = '0' then
23             es_p <= E0;
24           else
25             es_p <= E1;
26           end if;
27         when E1 =>
28           if D = '0' then
29             es_p <= E0;
30           else
31             es_p <= E2;
32           end if;
33         when E2 =>
34           if D = '0' then
35             es_p <= E3;
36           else
37             es_p <= E2;
38           end if;
39         when E3 =>
40           if D = '0' then
41             es_p <= E0;
42           else
43             es_p <= E4;
44           end if;
45         when E4 =>
46           if D = '0' then
47             es_p <= E0;
48           else
49             es_p <= E5;
50           end if;
51         when E5 =>
52           if D = '0' then
53             es_p <= E3;
54           else
55             es_p <= E2;
56           end if;
57         when others =>
58           es_p <= E0;
59       end case;
60     end if;
61   end process;
62
63   process(D)
64   begin
65     if es_p = E5 then
66       Q <= D;
67     else
68       Q <= '0';
69     end if;
70   end process;
71 end flujo;
72
73
74

```

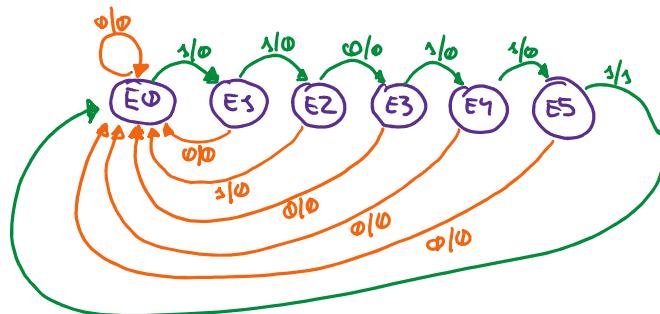
9

Resultado de la simulación VWF



10

Ejemplo: Detector de clave ->110111 con MEF Mealy (nótese no hay traslape)



11

Detector de secuencia $\rightarrow 011010$ empleando MEF Mealy

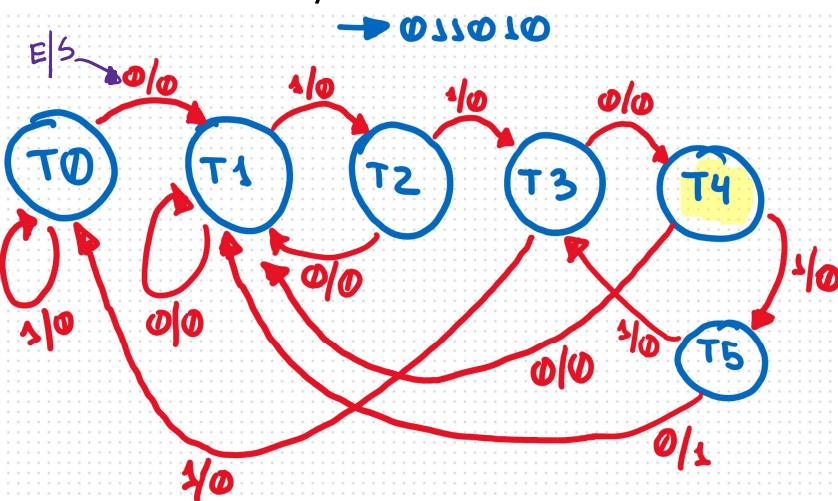
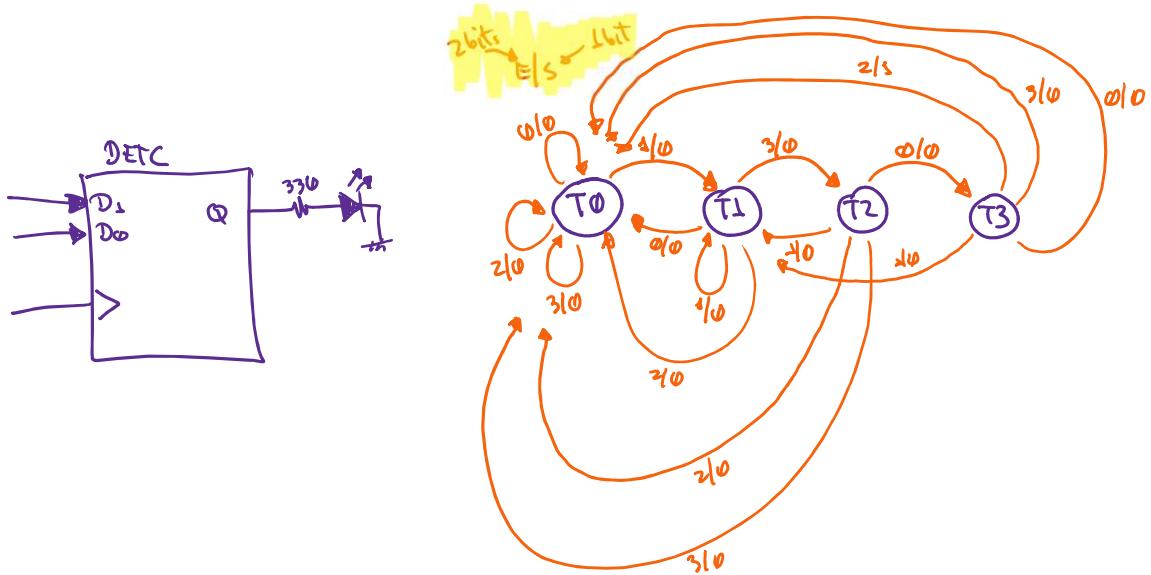


Tabla de transiciones

E	E.P.	E.S	S
0	T0	T0	0
1	T0	T0	0
0	T1	T1	0
1	T1	T2	0
0	T2	T2	0
1	T2	T3	0
0	T3	T3	0
1	T3	T4	0
0	T4	T4	0
1	T4	T5	0
0	T5	T5	1

15

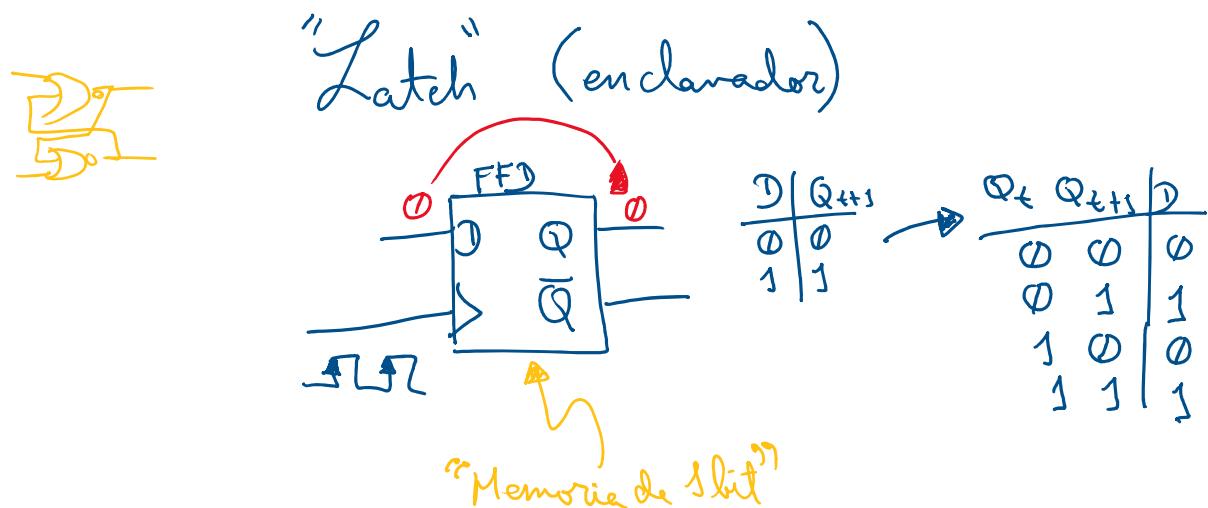
Detector de secuencia de dos bits ->1,3,0,2



16

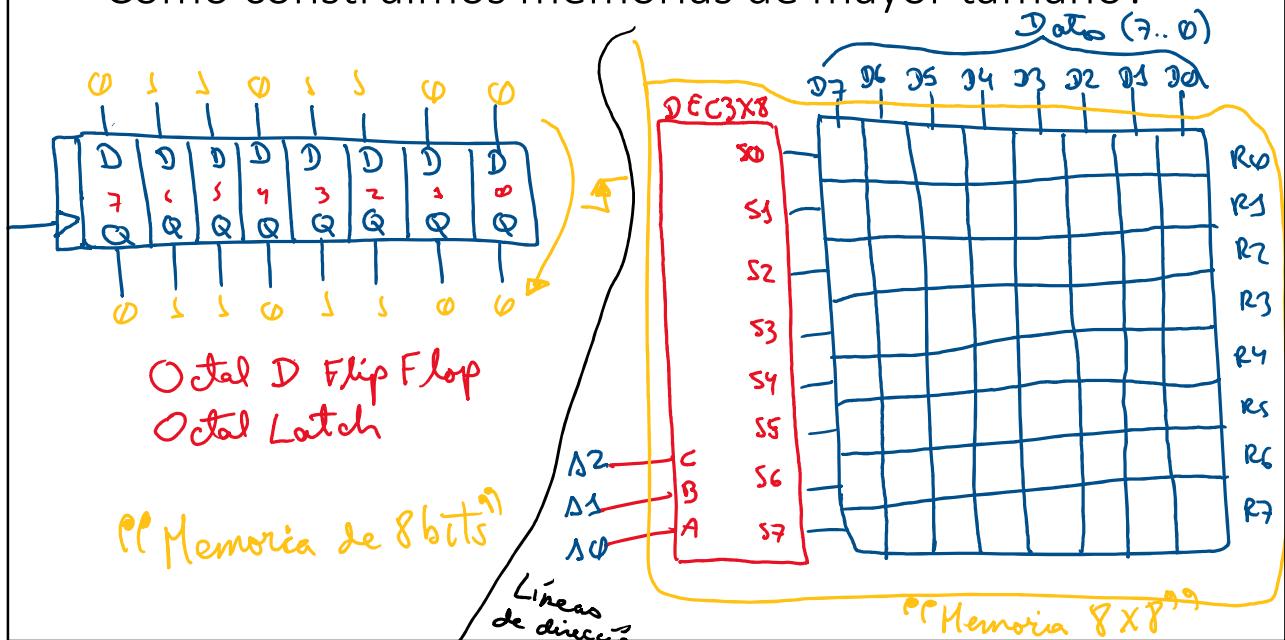
Memorias

- Las memorias son dispositivos que almacenan información



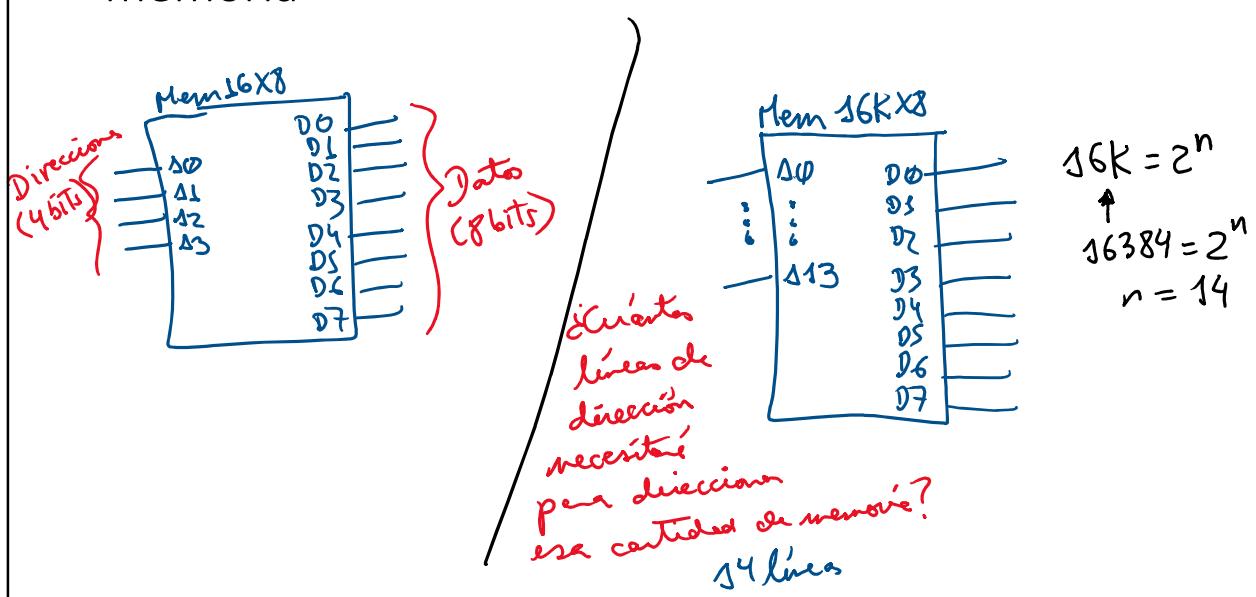
17

Cómo construimos memorias de mayor tamaño?



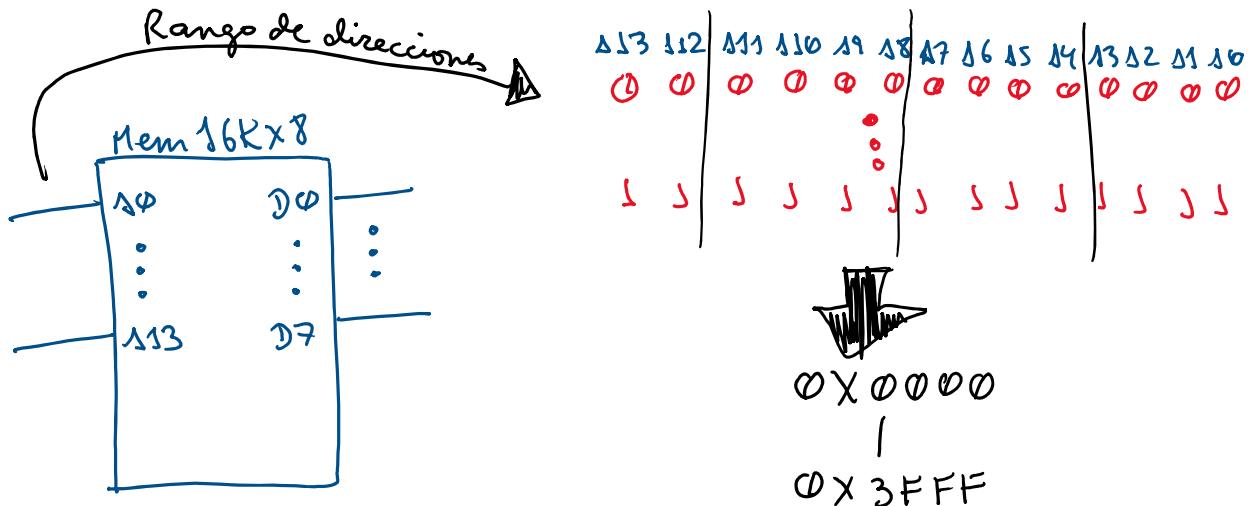
18

Relación entre líneas de dirección y tamaño de la memoria



19

En la memoria anterior de 16Kx8:



20

Analizando lo siguiente:

https://www.pjrc.com/tech/8051/board3/board_image.html

New Rev 4 Board

8051 Development Board: What It Does

The 8051 development board provides an easy-to-use and low-cost way to develop your 8051 based microcontroller projects, with equipment, such as IC programmers or emulators. See below for a [detailed list of the boards features](#).

Close-Up View of the Development Board

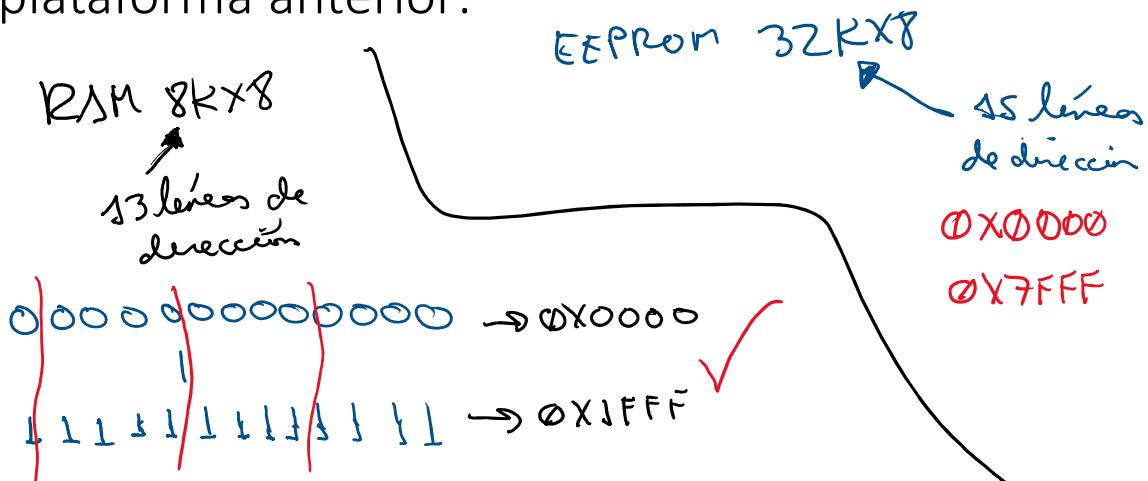
- Tarjeta embebida basada en el 8051.
- Tiene Memoria RAM M5M5165P (8Kx8)
- Tiene Memoria EEPROM (32Kx8)
- Tiene interface periférica programable (PPI) 82C55
- Precursor de los actuales microcontroladores “all-in-one”

Memory Map

0000 - 1FFF	PAULMON2 Monitor
2000 - 3FFF	SRAM
4000 - 5FFF	82C55 I/O Chip
	4000: Port A (read or write)
	4001: Port B (read or write)
	4002: Port C (read or write)
	4003: Configure (write only)
6000 - 7FFF	User Expansion (Y6 signal assert low)
8000 - FFFF	Flash ROM

21

Verificación de mapeo de memoria de la plataforma anterior:



22

Preguntas sobre memorias:

- Si la memoria EEPROM es no volátil. ¿Por qué se sigue usando memoria RAM si éste al quitarle la alimentación se borra lo que grabó?
 - La EEPROM es lenta (en los eventos de escritura) a comparación de la RAM que tiene tiempos de acceso más cortos.

Hay que tener en cuenta la jerarquía de memorias.

23

Tecnologías de Memorias

- ROM (Read Only Memory – Memoria de Sólo Lectura)
 - Éstas memorias vienen programadas de fábrica, no se altera el contenido en cortes de alimentación.
 - No es posible borrar ni modificar el contenido.

24

Tecnologías de Memorias

- OTP (One Time Programmable – Memoria programable una única vez)
 - También conocidas como PROM
 - Éstas memorias sólo pueden ser grabadas una sola vez.
 - No es posible borrar ni modificar el contenido después de haberla grabado.

25

Tecnologías de Memorias

- EPROM (Erasable Programmable Read Only Memory)
 - Para programarlas hay que borrarlas, generalmente son las memorias con una ventana en la cara superior en donde se aplica radiación UV para el proceso de borrado.
 - Luego de programadas, su contenido no se borra si se le corta la alimentación.



26

Tecnologías de Memorias

- EEPROM (Electrically Erasable Programmable Read Only Memory)
 - Éstas se borran de manera eléctrica, es decir, aplicando un voltaje más elevado que el voltaje del nivel lógico y en un pin especial.
 - Luego de programadas, su contenido no se borra si se le corta la alimentación.

27

Tecnologías de Memorias

- Flash EEPROM (Electrically Erasable Programmable Read Only Memory)
 - Los eventos de borrado/escritura con mayores.
 - Son mucho mas rápidas que las EEPROM convencionales.
 - De mucha mayor capacidad.

28

Tecnologías de Memorias

- RAM (Random Access Memory – Memoria de Acceso Aleatorio)
 - Son volátiles (al desconectarle la alimentación se pierde la información que pueda haber contenido).

29

Tecnologías de Memorias

- SRAM (Static Random Access Memory)
 - Son volátiles (al desconectarle la alimentación se pierde la información que pueda haber contenido).
 - Poseen un costo muy elevado
 - Empleado para memoria CACHÉ y registros temporales.

30

Tecnologías de Memorias

- DRAM (Dynamic Random Access Memory)
 - Son volátiles (al desconectarle la alimentación se pierde la información que pueda haber contenido).
 - Menor costo que las SRAM.
 - Requieren ser “refrescadas” para que puedan almacenar y retener información
 - De grandes capacidades.

31

Ubicación de la Memoria en una PC

- CPU
 - La que se encuentra dentro del procesador
 - La de acceso más rápido
- Memoria Interna
 - Almacenamiento de mayor capacidad que la del CPU (memoria principal)
- Memoria Externa
 - Almacenamiento de mucha mayor capacidad (secundaria)

32

Método de Acceso de las Memorias

- Acceso Secuencial
 - Para acceder a una posición debe de ir trasladándose desde la posición actual a la deseada, pasando y obviando cada registro intermedio.
 - Método empleado en los “Tape Backup”.
- Acceso Directo
 - Se organiza en bloques para buscar más rápido un registro, una vez hallado el bloque en donde se encuentra la información solicitada, se emplea el acceso secuencial para hallarla.
 - Método empleado en las unidades de disco.

33

Método de Acceso de las Memorias

- Acceso Aleatorio (random)
 - Cada posición tiene un mecanismo de acceso cableado físicamente, el tiempo de acceso a una posición es constante.
 - Método empleado en la memoria principal
- Asociativa
 - Similar como la de acceso aleatorio, posee un mecanismo de comparación según valores dados para recuperar la información basándose en una porción de su contenido en lugar de su dirección.
 - Método empleado en las memorias caché.

34

Desempeño de las Memorias

Tiempo de acceso:

- Es el tiempo entre la asignación de memoria y la obtención de un dato válido.

Ciclo de la Memoria:

- Es el tiempo que la memoria requiere para "recuperarse" antes de un nuevo evento de acceso.

Tasa de Transferencia:

- Cuanta información se pueda mover

35

Jerarquía de las Memorias en una PC



- Registros
- Caché Nivel 1
- Caché Nivel 2
- Memoria Principal
- Caché de Disco Duro
- Disco Duro
- Discos Ópticos
- Cintas Tape Backup

36

Jerarquía de las Memorias en una PC

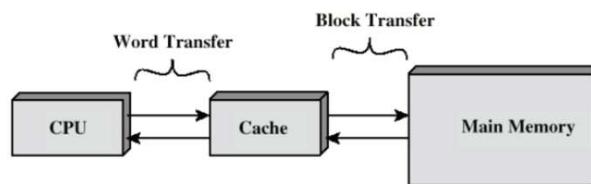
● Relaciones:

- A menor tiempo de acceso, mayor coste por bit.
- A mayor capacidad, menor coste por bit
- A mayor capacidad, mayor tiempo de acceso

37

Memoria CACHÉ

- Basado en memoria del tipo SRAM
- En pequeña cantidad
- Se encuentra entre la memoria principal y el μP
- Puede estar embebida en el mismo μP o de manera externa



38

Funcionamiento de la Memoria CACHÉ

- El μP pide el contenido de una posición de memoria
- Revisa la caché para ver si contiene esa información
- Si está presente, lo obtiene de la caché
- Si no está presente, lee el bloque requerido desde la memoria principal hacia la caché.
- Luego la información es entregada desde la caché hacia el μP.
- La caché incluye etiquetas para identificar cuál bloque de la memoria principal está en una casilla de la caché.

39

¿Porqué Memoria CACHE?

- Un sistema de memoria el cual almacena temporalmente información accedida frecuentemente y obtenida de dispositivos de almacenamiento con mayor tiempo de acceso.
- Con esto, se busca aminorar el tiempo de acceso, mientras la memoria CACHE es mayor, nuestra PC será mas rápida

40

¿Necesitamos velocidad?

- Es posible implementar una PC con sólamente memoria del tipo SRAM
- Sería sumamente rápido
- No se necesitaría memoria CACHE
- El costo sería extremadamente exorbitante!

41

Cuestionario

- Revisar modelos de ALU en VHDL

10 decimal
0x10 hexadecimal
10h DEH

42

Fin de la sesión

43