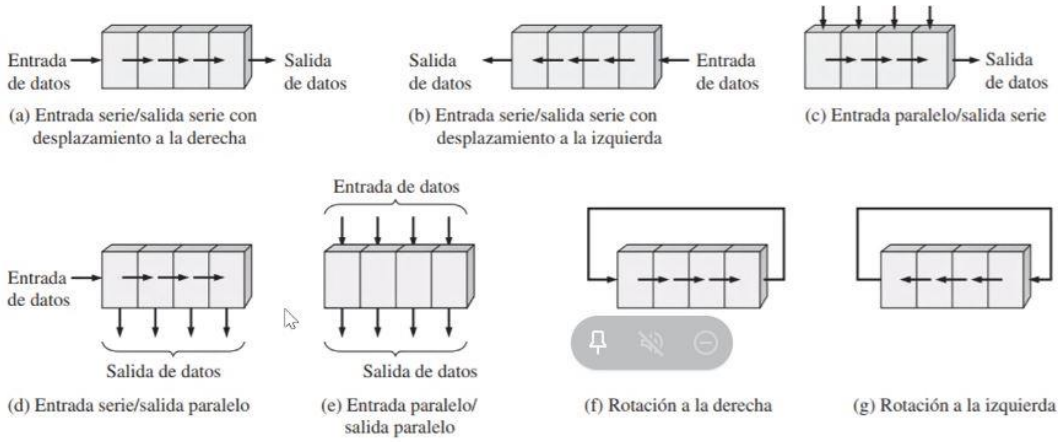
Registros

Un registro puede tener compuertas combinacionales que realizan ciertas tareas de procesamiento de datos.

Los flip-flop contienen la información binaria y las compuertas determinan cómo esa información se transfiere al registro.

La capacidad de almacenamiento de un registro le convierte en un tipo importante de dispositivo de memoria

Un contador es básicamente un registro que pasa por una sucesión predeterminada de estados. Las *compuertas* del contador están conectadas de tal manera que producen la sucesión prescrita de estados binarios.



(a) Entrada serie/salida serie con desplazamiento a la derecha

(b) Entrada serie/salida serie con desplazamiento a la izquierda

(c) Entrada paralelo/salida serie

(d) Entrada serie/salida paralelo

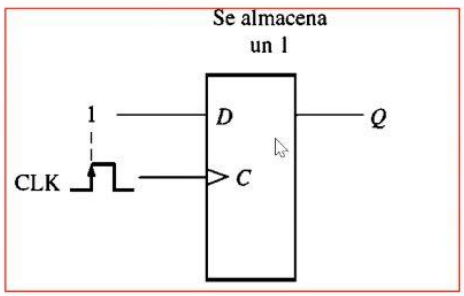
(e) Entrada paralelo/ salida paralelo

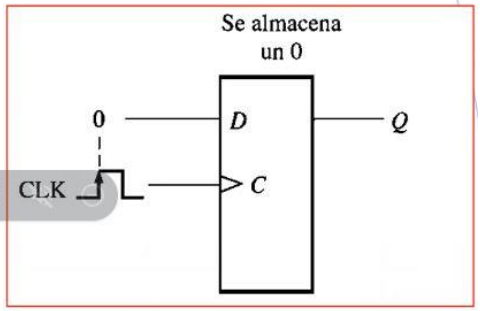
(f) Rotación a la derecha

(g) Rotación a la izquierda

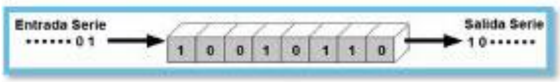
Tipos de registros de 4 bits.

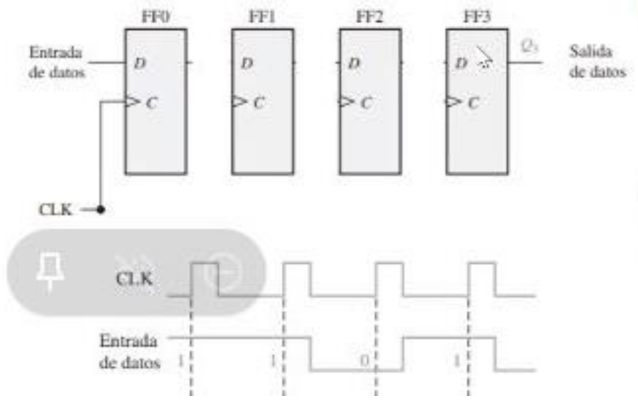
Flip – flop como elemento de almacenamiento

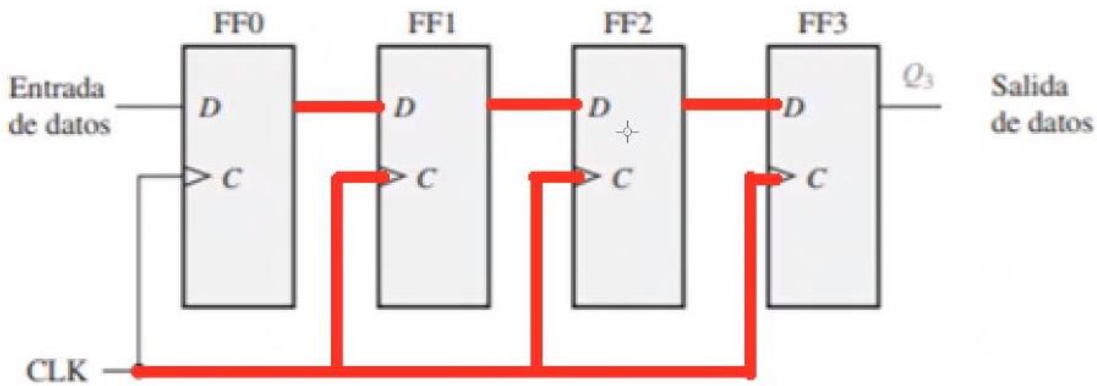


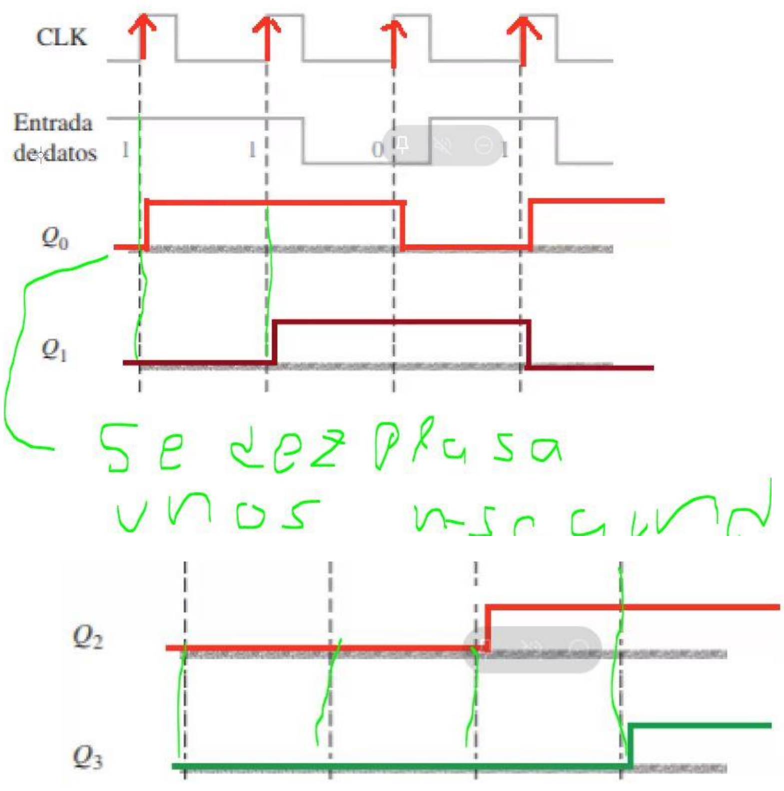


Registros de entrada serie - salida serie



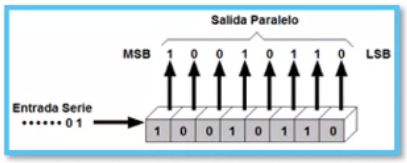


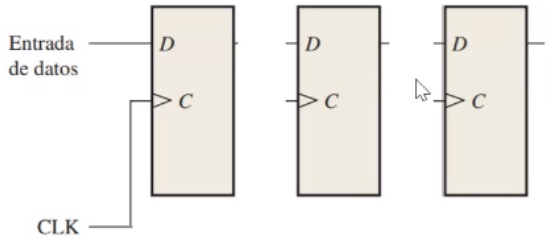




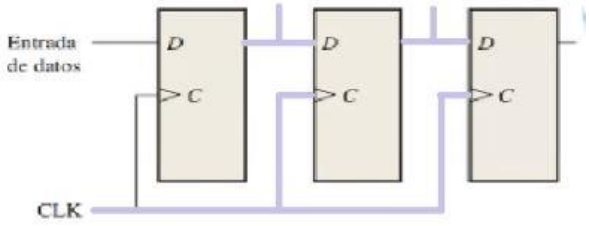
Pude se con cualquier tipo de FF

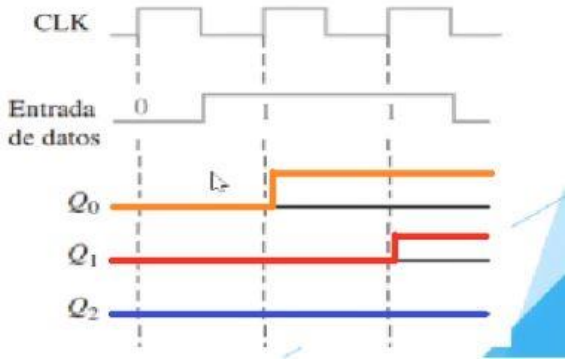
Registros de entrada serie - salida paralelo





Pude se con cualquier tipo de FF





Registros en VHDL.

Variables vs señales

```
-- Uso incorrecto de las señales
ARCHITECTURE ejemp1 OF entidad IS
SIGNAL a,b,c,x,y: integer;
BEGIN
 p1: PROCESS(a,b,c)
 BEGIN
    c<=a;  -- Se ignora
    x<=c+2;
    c<=b;  -- Se mantiene
    y<=c+2;
 END PROCESS p1;
END ejemp1;
```

```
-- Uso correcto de las variables
ARCHITECTURE ejemp1 OF entidad IS
SIGNAL a,b,x,y: integer;
BEGIN
 p1: PROCESS(a,b)
 VARIABLE c: integer;
 BEGIN
    c:=a;  -- Inmediato
    x<=c+2;
    c:=b;  -- Inmediato
    y<=c+2;
 END PROCESS p1;
END ejemp1;
```

Latch con una entrada ***d***, salida ***q*** y señal de habilitación ***ena***

▶NOTA: En VHDL no es necesario poner dos biestables en serie para definir un registro; al utilizar una descripción comportamental, lo que se hace es definir su función, es decir, que la salida capture la entrada cuando se produzca uno de los flancos de reloj y que no haga nada en caso contrario..

```
PROCESS(ena,d)
BEGIN
  IF ena='1' THEN q<=d;
  END IF;
END PROCESS;
```

Declaración de un flip-flop en VHDL

```
PROCESS(clk,reset)
BEGIN
  IF reset<='1' THEN q<='0';
  ELSIF clk='1' AND clk'event THEN q<=d;
  END IF;
END PROCESS;
```

Descripción de Lógica Secuencial

- Una de las propiedades más importantes de un ***process*** es la capacidad de su estructura para almacenar los valores de las señales que se asignan en su interior. Debido a esta característica se utilizarán los process para generar HW secuencial

Hardware Secuencial

```
if (CLK'event and CLK='1') then …
```

Creación de un biestable D con reset

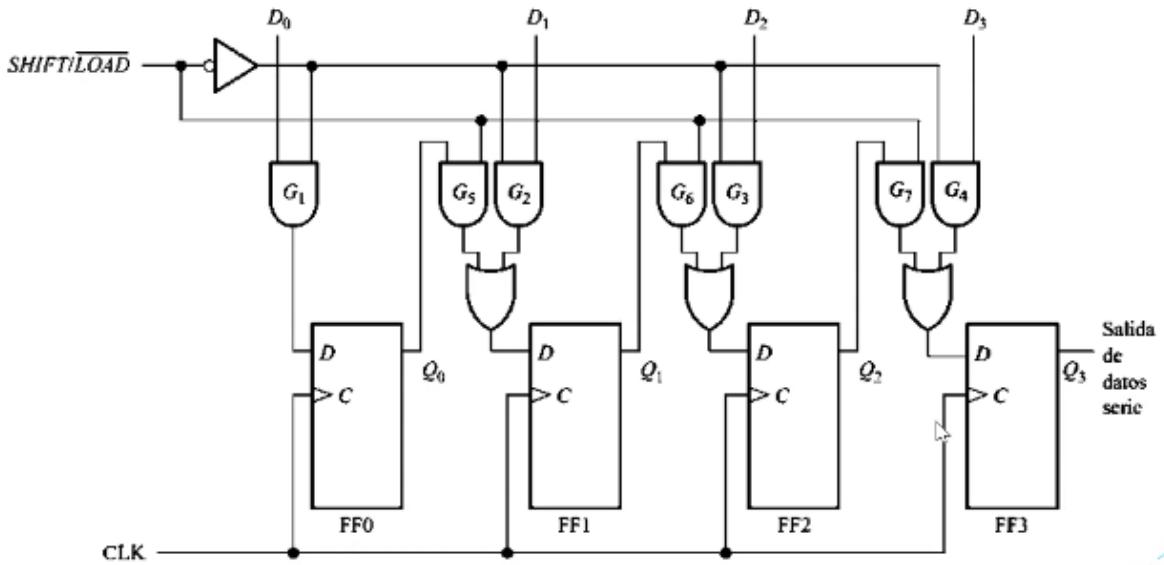
```
entity Biestable_rD is
     port(d, clk, reset: in std_logic; q: out std_logic);
end Biestable_rD;

architecture ARCH_ASYN of Biestable_rD is
begin
     process (clk, reset, d)
     begin
          if (reset = '1') then q <= '0';
          elsif clk = '1' and clk'event then q <= d;
          end if;
     end process;
end ARCH_ASYN;
```

```
architecture ARCH_SYN of Biestable_rD is
begin
     process (clk, reset, d)
     begin
          if clk = '1' and clk'event then q <= d;
               if (reset = '1') then q <= '0';
               end if;
          end if;
     end process;
end ARCH_SYN;
```

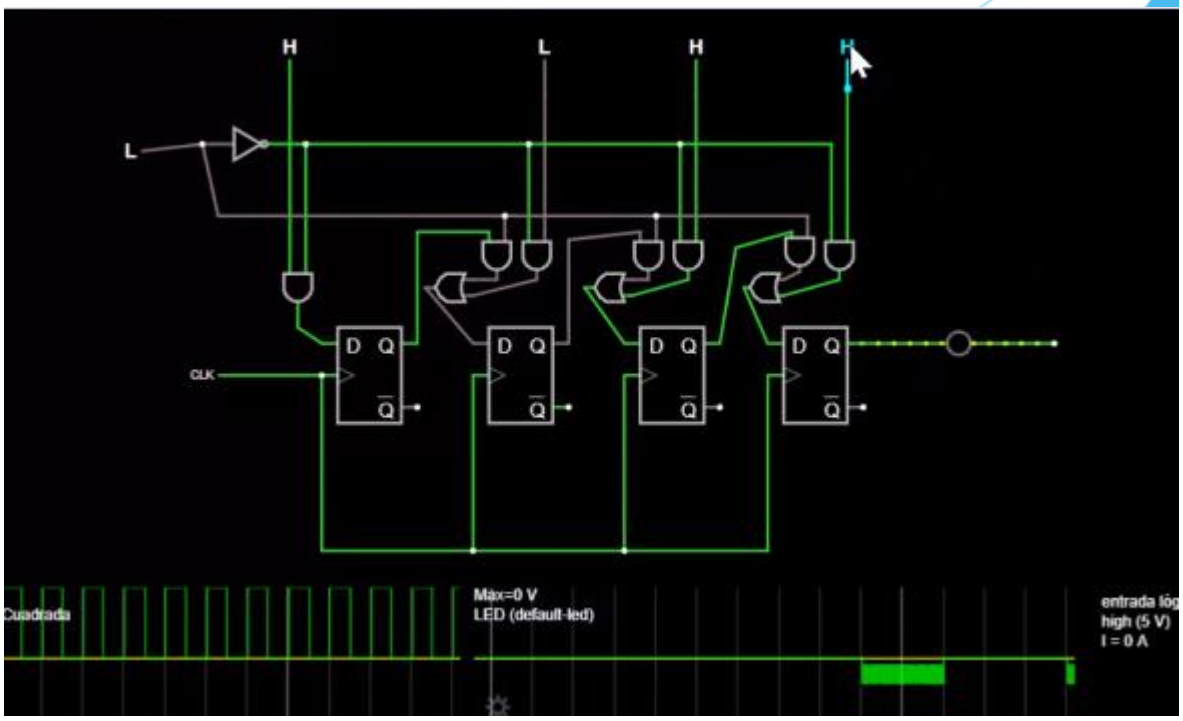
Para conseguir crear el HW secuencial esperado hay que cumplir una serie de reglas: - Una sentencia *if* que tenga por condición una especificación de flanco no puede tener rama ***else***, en caso contrario la rama **else** debería realizarse en todo momento menos en el preciso instante en el que el reloj cambia.

- En sentencias ***if-then-elsif*** la especificación de flanco sólo podrá ser la condición del último ***elsif*** (que no podrá tener rama else).
- Una sentencia *if* que tenga por condición una especificación de flanco puede tener internamente sentencias ***if-else*** encadenadas.
- En un ***process*** solo puede existir una única especificación de flanco, en caso contrario se estaría especificando HW secuencial sensible a varios relojes.

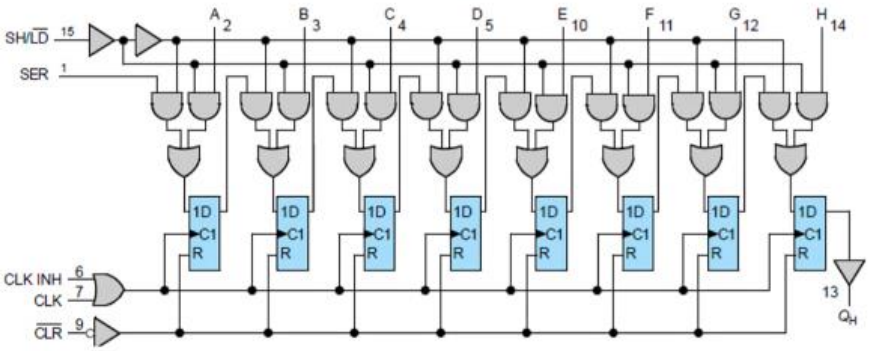
Registros de entrada paralela- salida serie



http://tinyurl.com/yxmnched

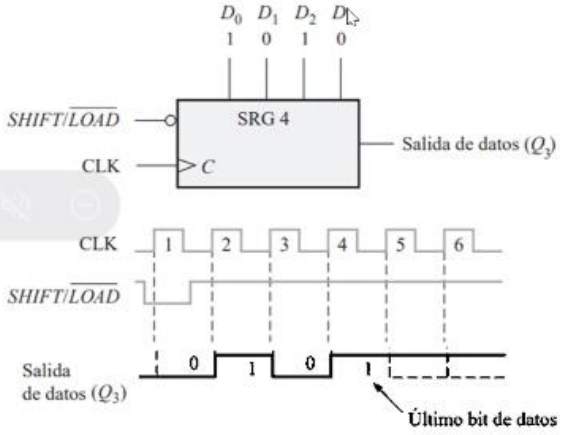


Serie TTL 74166 registro de desplazamiento de 8 bits



▶En el impulso de reloj 1, los datos paralelo (***D0D1D2D3*** = 1010) se cargan en el registro, poniendo la salida Q3 a 0. En el impulso de reloj 2, el 1 de Q2 se desplaza a Q3; en el impulso de reloj 3, el 0 se desplaza a Q3; en el impulso de reloj 4, el último bit de datos (1) se desplaza a Q3 y en el impulso de reloj 5 todos los bits se han desplazado y salido del registro, y sólo quedan 1s en el mismo (suponiendo que la entrada D0 permanece a 1)

La carga de datos en paralelo del Registro puede ser del tipo Síncrona o asíncrona



Registro de desplazamiento de 4 bits con entrada paralelo salida serie

Registro de desplazamiento

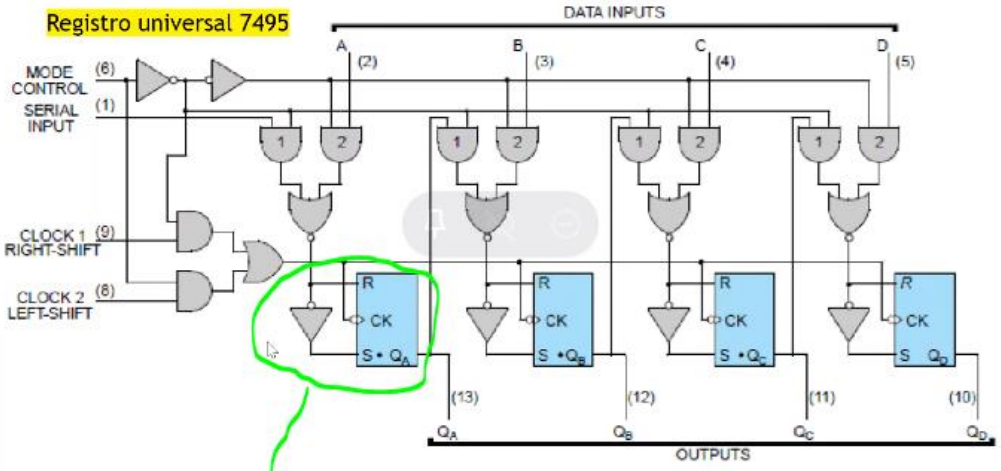
- Consiste en una cadena de flip-flops en cascada, con la salida de un flip-flop conectada a la entrada del siguiente flip-flop. Todos los flip-flops reciben pulsos de reloj comunes, que activan el desplazamiento de una etapa a la siguiente.

Registro bidireccional

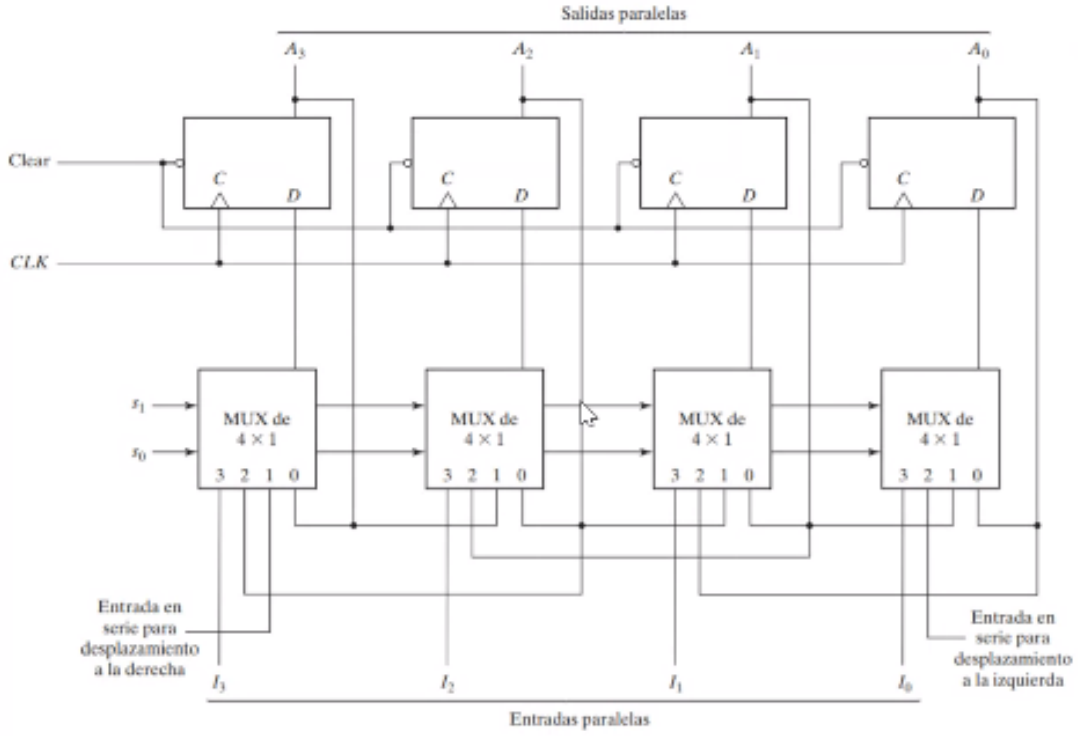
- Este tipo de registro tiene la opción de elegir la dirección en que se transmiten los datos. Estos registros tienen una señal de control que permite seleccionar el sentido de desplazamiento de los datos.

Un registro que sólo puede desplazar en una dirección es un registro de **desplazamiento unidireccional**. Uno que puede hacerlo en ambas direcciones es un registro de desplazamiento **bidireccional**. Si el registro tiene ambos desplazamientos y capacidad de carga paralela, se denomina **registro de desplazamiento universal.**

Registro universal 7495



puede ser tipo D



Registro de desplazamiento universal de 4 bits

Control de modo		
s_1	s_0	**Operación del *registro***
0	0	Sin cambio
0	1	Desplazamiento a la derecha
1	0	Desplazamiento a la izquierda
1	1	Carga en paralelo

Tabla de operación de un registro universal de 4 bits

RESUMEN DE REGISTROS.

- *Serie:* los bits se transfieren uno a continuación del otro por una misma línea.

- *Paralelo:* se intercambian todos los bits al mismo tiempo, utilizando un número de líneas de transferencia igual al número de bits.

- El uso de los registros de desplazamiento dentro de la electrónica secuencial es de gran utilidad, ya que pueden utilizarse para diferentes aplicaciones como:

Multiplicadores: realizan la multiplicación mediante sumas y desplazamientos.

Circuitos de Retardo: pueden utilizar (para retardar un bit) un número entero de ciclos de reloj. Consiste simplemente en un conjunto de biestables en cascada (tantos como ciclos de reloj deseemos retardar los bits).