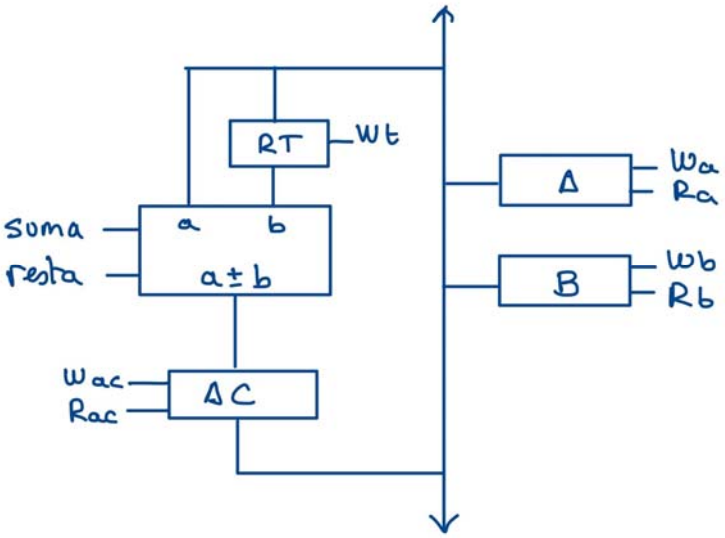
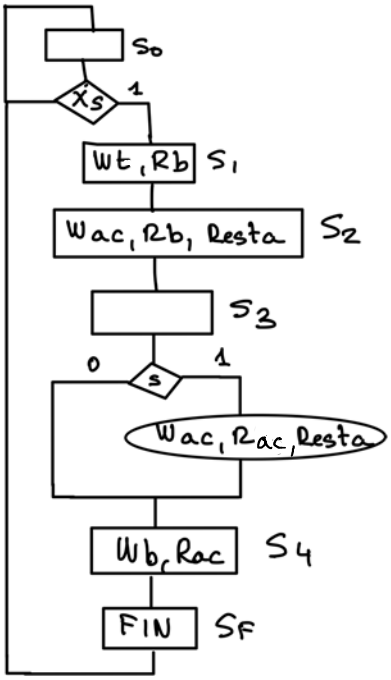
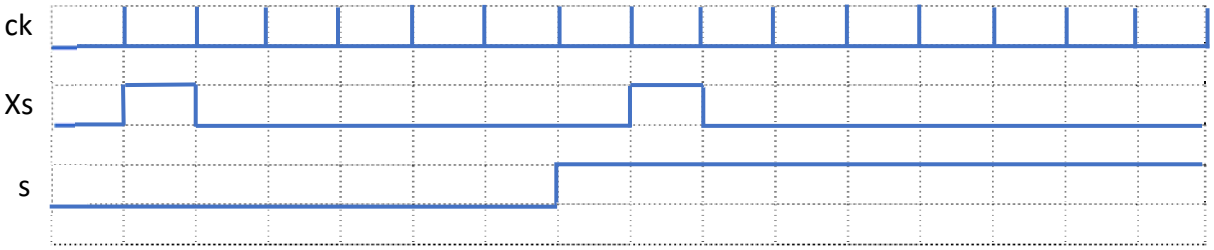
ALUMNO:_____

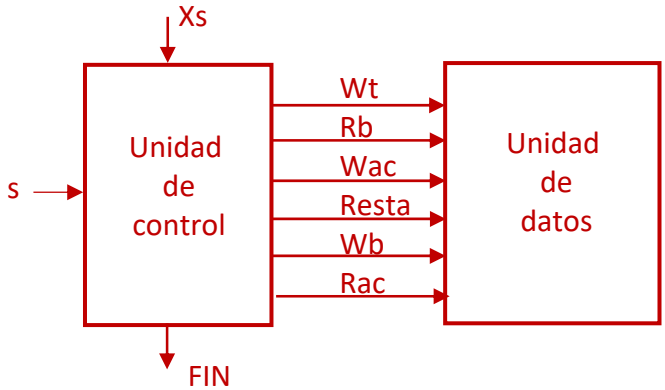
1. Se tiene un sistema digital cuya unidad de datos se muestra en la figura. También se muestra la carta ASM de la unidad de control y una secuencia de entrada que se utilizará en el apdo. d.







a. Haga un diagrama de bloques del sistema en el que aparezcan claramente especificadas las entradas y salidas de la unidad de control y las de la unidad de datos, indicando con flechas el sentido de las señales.



b. Describa el registro A a nivel RT y el registro AC en Verilog.

Wa	Ra	A ←	DAT =
0	0	A	HI
0	1	A	[A]
1	0	DAT	Entrada
1	1	Proh.	Proh.

c. Explique razonadamente cuál es la tarea que realiza el sistema para los dos valores de s.

Para s=0, las microoperaciones y los cambios en los registros son:

Ciclo 1.	RT ← B	aún no hay cambios	[B] en bus compartido
Ciclo 2.	AC ← B-RT	[RT] = B_0	[B] en bus compartido
Ciclo 3.	-	[AC] = 0	HI en bus compartido
Ciclo 4.	B ← AC	-	[AC] en bus compartido
Ciclo 5.	-	[B] = 0	HI en bus compartido

Por tanto para s=0, la macrooperación es B ← 0.

Para s=1, las microoperaciones y los cambios en los registros son:

Ciclo 1.	RT ← B	aún no hay cambios	[B] en bus compartido
Ciclo 2.	AC ← B-RT	[RT] = B_0	[B] en bus compartido
Ciclo 3.	AC ← B-RT	[AC] = 0	[AC] en bus compartido
Ciclo 4.	B ← AC	[AC] = $-B_0$	[AC] en bus compartido
Ciclo 5.	-	[B] = $-B_0$	HI en bus compartido

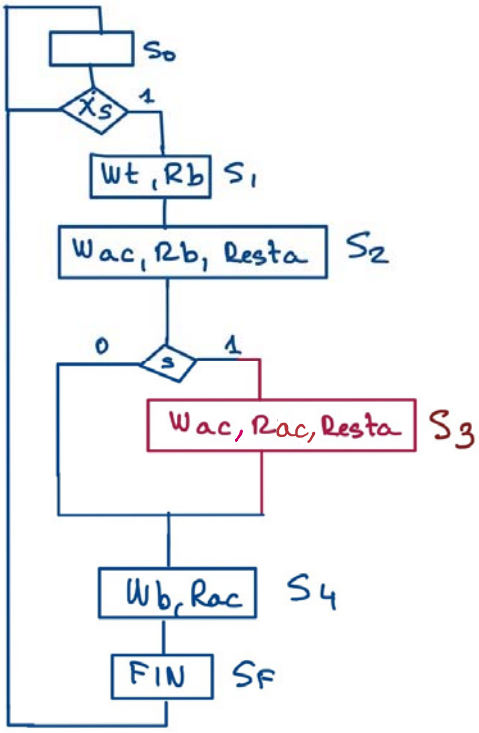
Por tanto para s=1, la macrooperación es B ← -B.

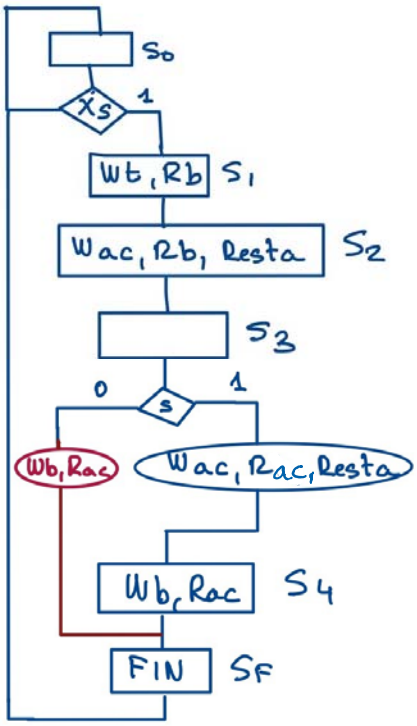
d. Complete en la plantilla del anexo el diagrama de ondas considerando que las entradas del sistema son las que se muestran.
(ver anexo)

e. Obtenga la descripción Verilog de la unidad de control sobre la plantilla del anexo.
(ver anexo)

f. Puede observar en la carta ASM que cuando s = 0 no se realiza ninguna acción en el estado S3. Proponga una optimización de la carta ASM que evite la pérdida de ese ciclo de reloj.

Hay dos posibilidades, señalo los cambios en rojo:



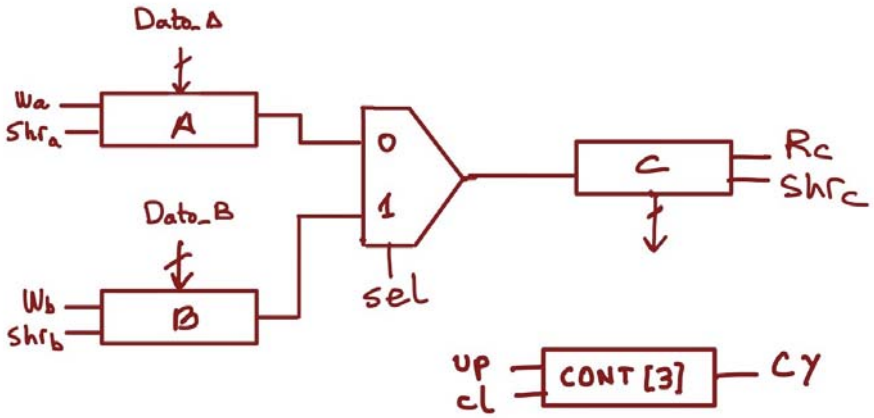


2. Se desea mover el contenido de dos registros A y B de 8 bits a un único registro C de 16 bits de modo que queden intercalados bit a bit, es decir, si [A] = $A_7A_6...A_1A_0$ y [B]= $B_7B_6...B_1B_0$, al final de la operación el contenido de C debe ser [C] = $A_7B_7A_6B_6...A_1B_1A_0B_0$.
Diseñe un sistema digital que realice la operación descrita, teniendo en cuenta que:

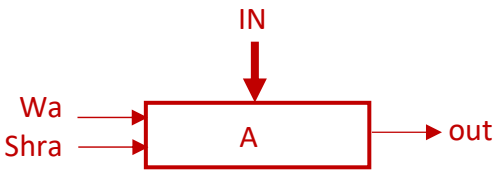
a. Al comienzo de la operación los datos (DatoA y DatoB) se cargan en los registros A y B respectivamente.
b. En la unidad de datos puede incluir los elementos que considere necesarios. La única restricción es que el registro C no posee carga en paralelo. Describa los componentes utilizados a nivel RT.
c. Debe hacer un diagrama de bloques del sistema especificando las entradas y salidas de la unidad de control y las de la unidad de datos, indicando con flechas el sentido de las señales.
d. Debe proporcionar la carta ASM de la unidad de control.

En este problema, no es necesario que realice ninguna descripción Verilog.

Unidad de datos:

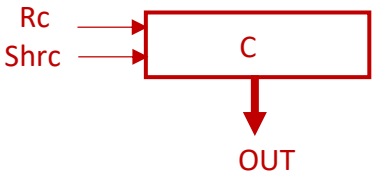


Descripción de los registros a nivel RT:



Wa Shra	A ←	out =
0 0	A	A_0
0 1	SHR(A,-)	A_0
1 0	IN	A_0
1 1	Proh.	Proh.

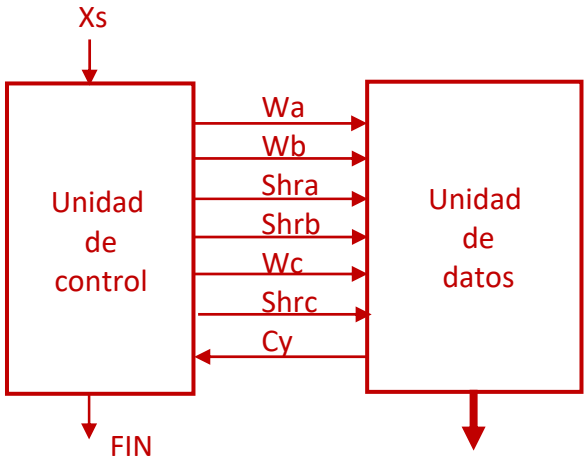
Válido para A y B



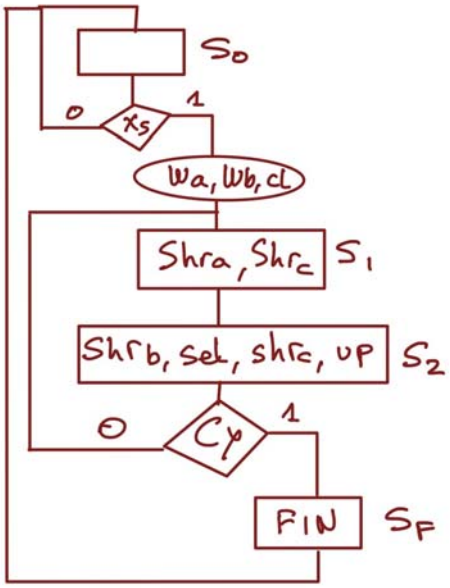
Rc Shrc	C ←	OUT =
0 0	C	[C]
0 1	SHR(C,-)	[C]
1 0	C	[C]
1 1	SHR(C,-)	[C]

up cl	CONT ←	Cy =
0 0	CONT	1 si
0 1	0	[CONT]=7
1 0	CONT+1	
1 1	Proh.	

Diagrama de bloques:

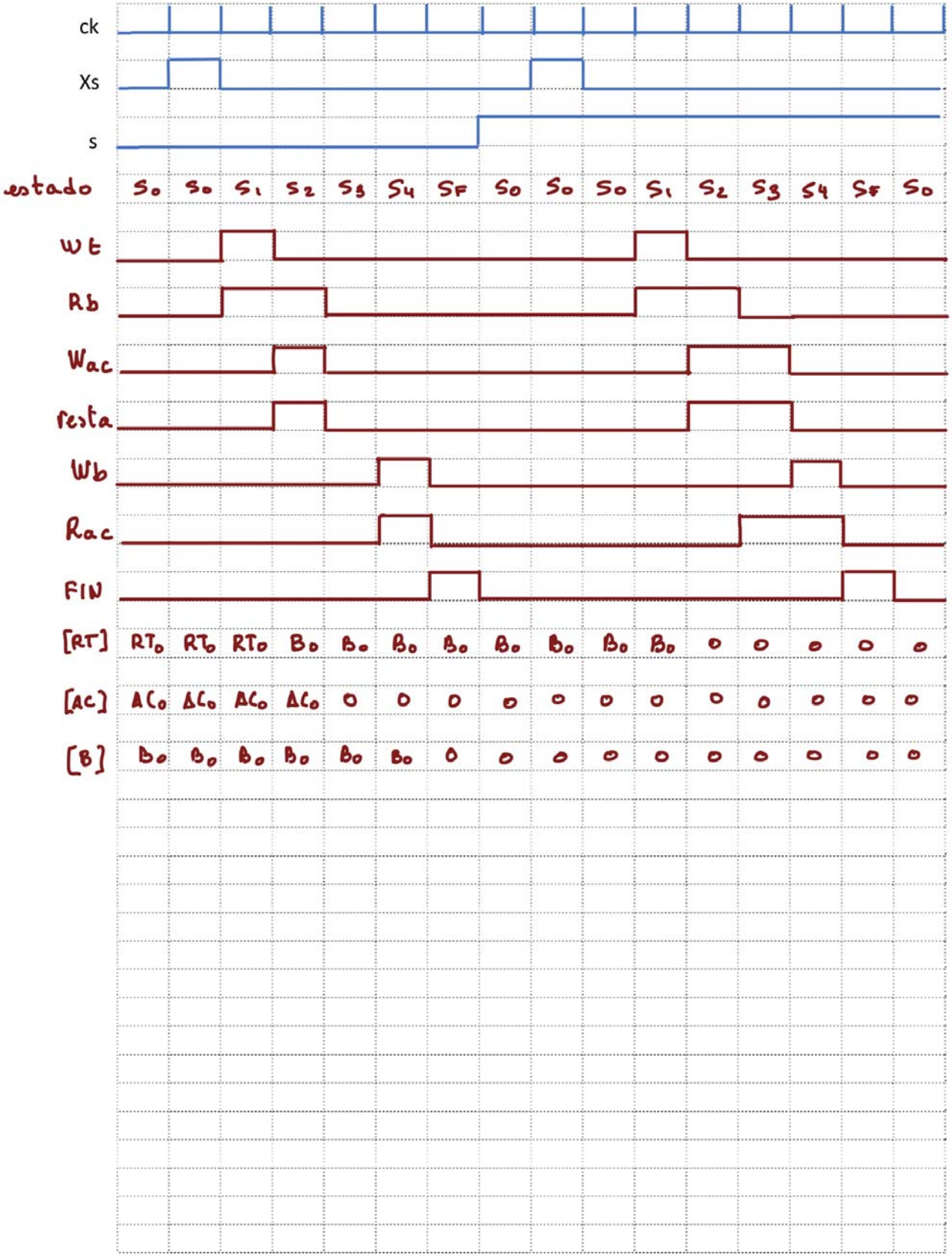


Carta ASM:



ANEXO

Diagrama de ondas



Descripción Verilog de la unidad de control

```
module Unidad_de_Control (
        input ck, reset, Xs,s                                   // Complete con el resto
        output reg Wt,Rb,Wac,resta,Wb,Rac
         // de entradas y salidas
        );

  parameter  S0 = 3'b000 ,                          // Complete la lista de estados
             S1 = 3'b001 ,
             S2 = 3'b010 ,
             S3 = 3'b011 ,
             S4 = 3'b100 ,
             SF = 3'b101 ;

  reg [2:0] estado, proximo_estado;          // Complete la declaración de vbles de estado

  always @ ( posedge clk, posedge reset )
        begin
             if ( reset)
                  estado <= S0;                        // Complete el “if”
             else
                  estado <= proximo_estado;            //Complete el “else”
        end

   always @ ( estado,   Xs, s )              //Complete con las entradas
        begin
        {Wt,Rb,Wac,resta,Wb,Rac}=0;
        proximo_estado = S0;
                                                       // Inicialice aquí las salidas
        case (estado) // Complete el “case”
             S0: begin
                  if(Xs)
                     proximo_estado=S1;
                end
             S1: begin
                  wt=1;
                  Rb=1;
                  proximo_estado=S2;
                end
             S2: begin
                  Wac=1;
                  Rb=1;
                  Resta=1;
                  proximo_estado=S3;
                end
```

```
                S3: begin
                        proximo_estado=S3;
                        if(s) begin
                                Wac=1;
                                Rb=1;
                                Resta=1;
                            end
                    end
                S4: begin
                        Wb=1;
                        Rac=1;
                        proximo_estado = SF;
                    end
                SF: begin
                        FIN=1;
                    end
        endcase
    end
endmodule
```