PRÁCTICA 1

DISEÑO DE UN CONTADOR DE VOTOS

MEMORIA DE LA PRÁCTICA
ELECTRÓNICA DIGITAL. 2º GIERM

Víctor G. Mengual y Alba Correal
04/04/2022

VÍCTOR GABRIEL MENGUAL PIRPAMER
ALBA CORREAL OLMO

Contenido

VÍCTOR GABRIEL MENGUAL PIRPAMER
ALBA CORREAL OLMO

DISEÑO DE UN CONTADOR DE VOTOS

Desglosamos la implementación de cada función y su bloque.

1. Cuenta Votos

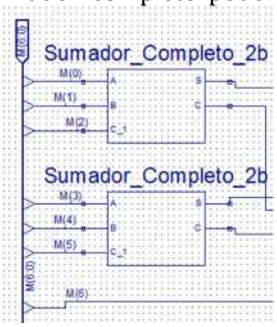
Debemos diseñar un bloque que sume bit a bit las entradas de los interruptores de la FPGA.

1.1 Sumar entradas bit a bit

C_{-1}	A	B	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

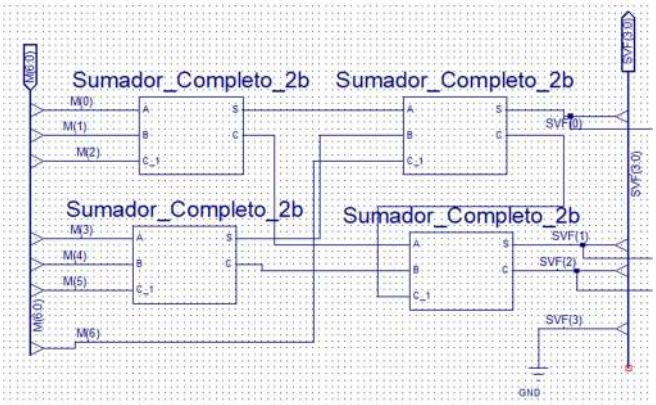
Si nos fijamos en la tabla de verdad de un sumador completo podemos apreciar que suma bit a bit cada una de las 3 entradas.

De este modo, podríamos construir un sumador de 7 bits a partir de bloques de suma completos de solo 3 bits, es decir, usamos dos sumadores para los 6 primeros bits.

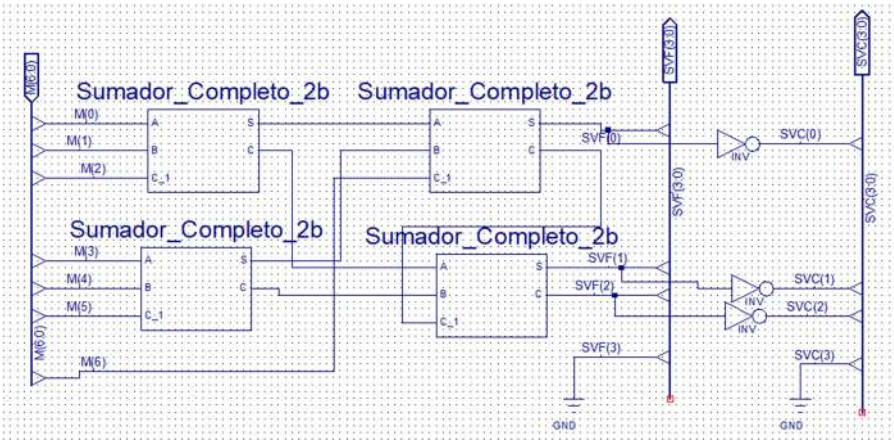


1.2 Obtener SVF(2:0) y SVC(2:0)

En la siguiente etapa, podemos configurar un sumador de números de 2 bits, para que se deduzca un único número binario de las dos sumas previas. Hemos implementado la arquitectura Ripple-Carry. Solo falta añadir el séptimo bit, para lo que bastará con conectarlo al C_{-1} de la primera puerta. Tras esto, habremos hallado SVF(2:0)



Por último, para obtener el número de votos en contra, habría que restar dicho número a 7. Sin embargo, como este número es el complemento a 7 del anterior, solo requiere que invirtamos las salidas de nuestro sumador, y las conectemos al otro bus, SVC(2:0).



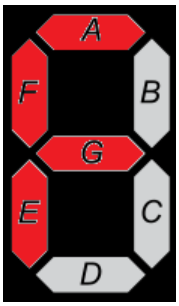
VÍCTOR GABRIEL MENGUAL PIRPAMER
ALBA CORREAL OLMO

2. Visualiza Cuenta

En este bloque, debemos mostrar en los displays el número de votos a favor y en contra que hemos obtenido en 1. Tenemos dos tipos de configuración de los 7-segmentos, una donde mostramos una letra fija y otra donde mostramos números que cambian en cada votación. Analicemos por separado cada una de estas situaciones.

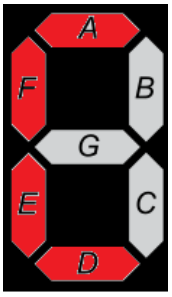
2.1 Letras F y C

Para las letras simplemente veremos qué valores necesitamos en el display.

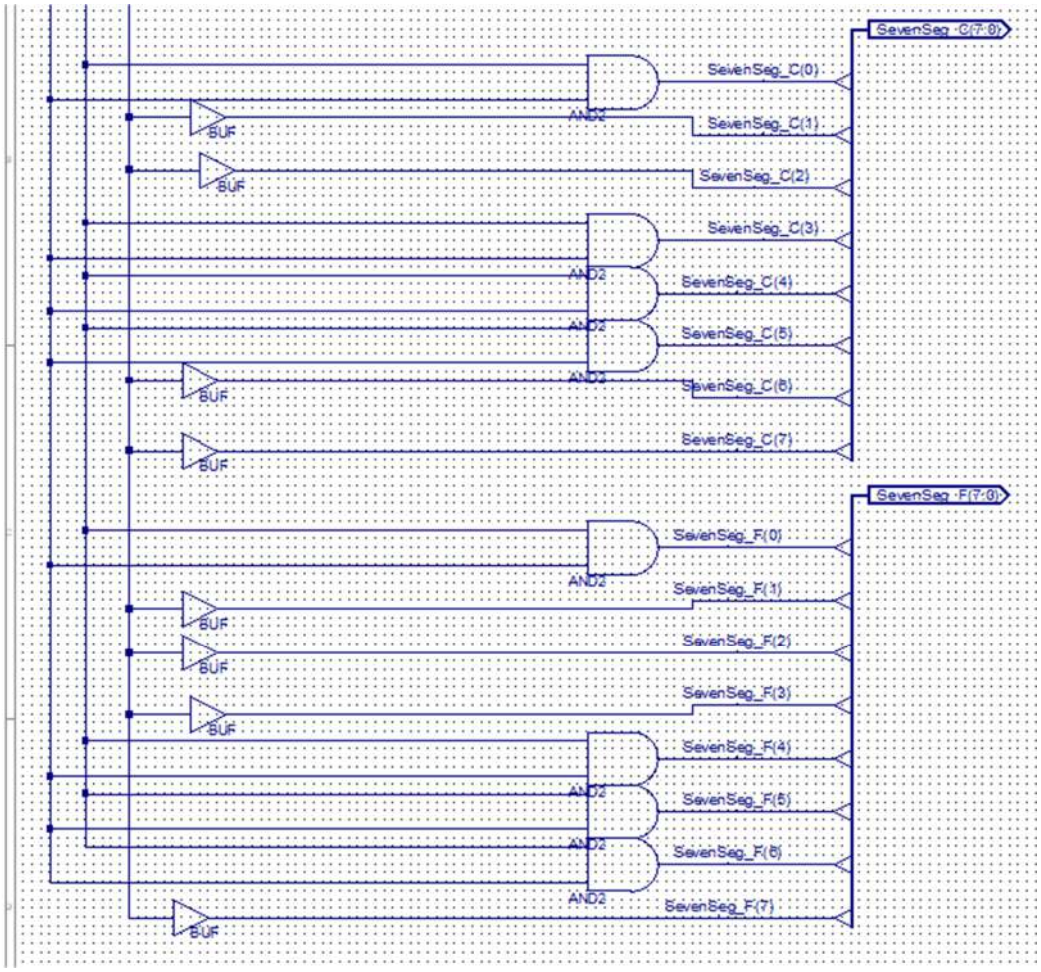


F	A	B	C	D	E	F	G
	1	0	0	0	1	1	1

C	A	B	C	D	E	F	G
	1	0	0	1	1	1	0



Simplemente bastará con ramificar un bus conectado a la alimentación entre los distintos segmentos del display.



Para evitar que se muestren las letras si el interruptor de "Enable" no está activado, pondremos una puerta AND en cada cable con un 1 donde la salida para VVal=0 sea 0 (GND) y la salida para VVal=1 sea SevenSeg_F(7:0) y Seven_Seg_C(7:0).

VÍCTOR GABRIEL MENGUAL PIRPAMER
ALBA CORREAL OLMO

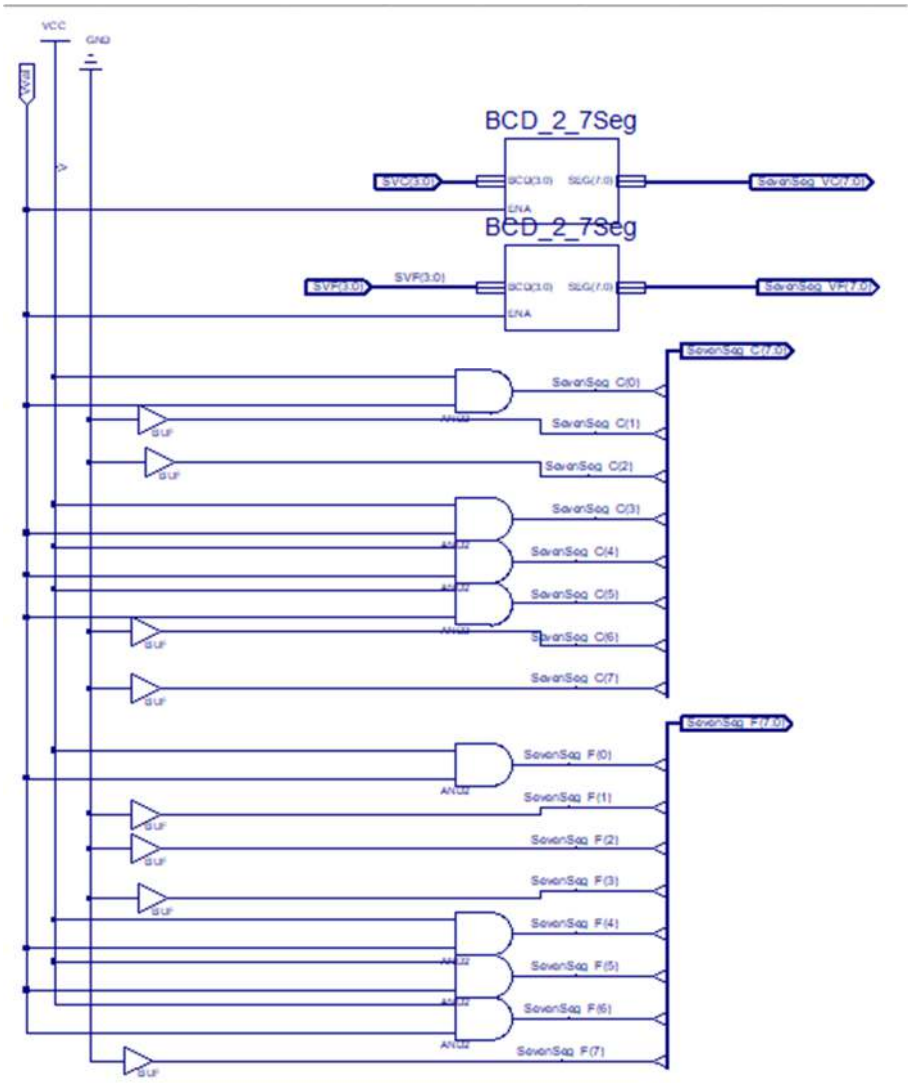
2.2 Números SVF(2:0) y SVC(2:0)

En este caso, reutilizaremos el conversor BCD-7 Segmentos de la práctica 0. Las tablas de verdad de cada uno de los displays son:

SVF(2:0)	A	B	C	D	E	F	G
000	1	1	1	1	1	1	0
001	0	1	1	0	0	0	0
010	1	1	0	1	1	0	1
011	1	1	1	1	0	0	1
100	0	1	1	0	0	1	1
101	1	0	1	1	0	1	1
110	1	0	1	1	1	1	1
111	1	1	1	0	0	0	0

SVC(2:0)	A	B	C	D	E	F	G
000	1	1	1	1	1	1	0
001	0	1	1	0	0	0	0
010	1	1	0	1	1	0	1
011	1	1	1	1	0	0	1
100	0	1	1	0	0	1	1
101	1	0	1	1	0	1	1
110	1	0	1	1	1	1	1
111	1	1	1	0	0	0	0

En total, el bloque quedaría:



VÍCTOR GABRIEL MENGUAL PIRPAMER
ALBA CORREAL OLMO

3 Resultado de Votación

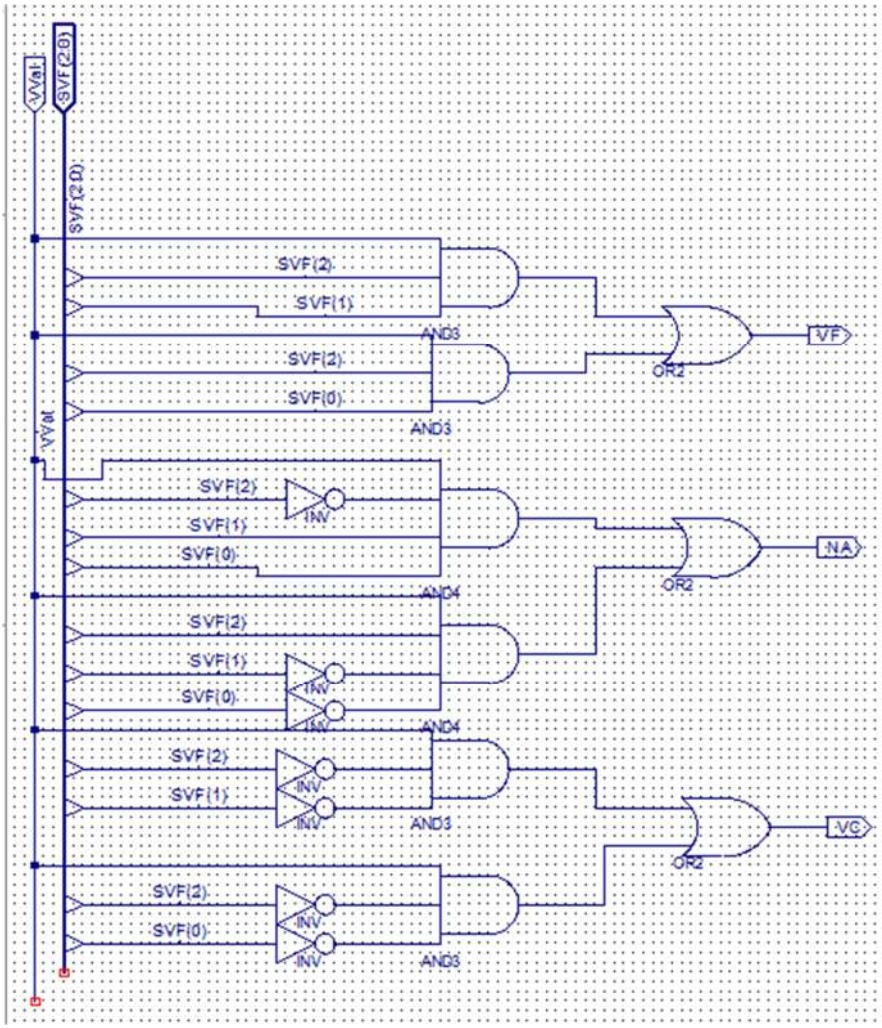
Este bloque consiste en definir si la votación es concluyente o no en función del número de votos obtenido en 1 (SVF(2:0)). En nuestro caso, la votación será a favor (VF) a partir de 5/7, en contra (VC) por debajo de 2/7 y no apta (NA) en el resto de casos. Comenzamos construyendo la tabla de verdad de este sistema

E	A	B	C	VF	NA	VC
0	X	X	X	0	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	0	1	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	0	0

Resolvemos el mapa de Karnaugh, para el que obtenemos una salida de:

- $EAB + EAC$ para VF
- $E\bar{A}BC + EA\overline{BC}$ para NA
- $E\overline{AB} + E\overline{AC}$ para VC

De este modo, el circuito asociado quedará de la forma:

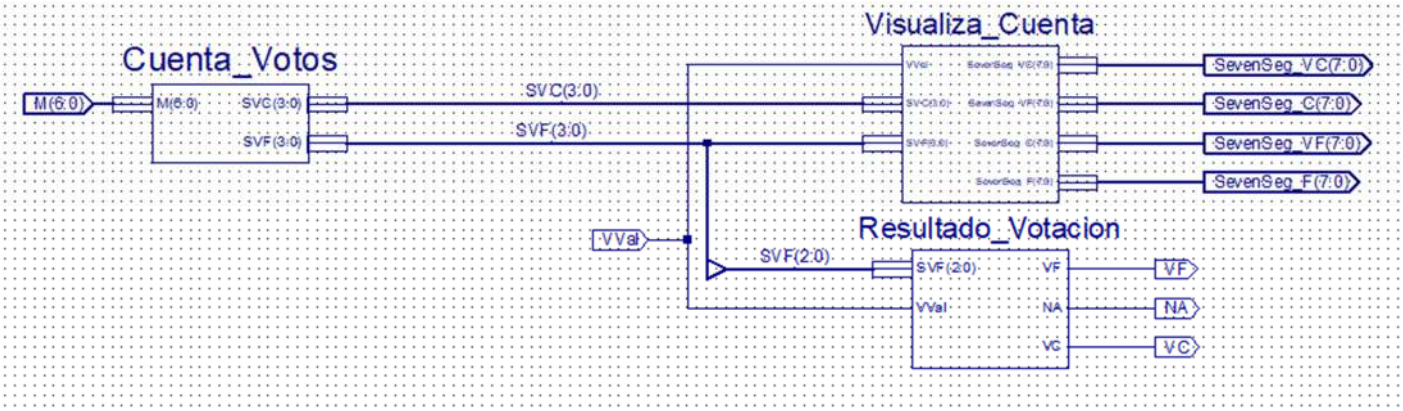


Cada uno de estos hilos se enviará a los leds de la FPGA. VF se conectará a los 3 últimos, NA a los 2 centrales y VC a los 3 iniciales.

VÍCTOR GABRIEL MENGUAL PIRPAMER
ALBA CORREAL OLMO

4 Vinculación de bloques.

Unimos todos los bloques en uno único que introduciremos en la FPGA. Visualmente:



5 Simulación.

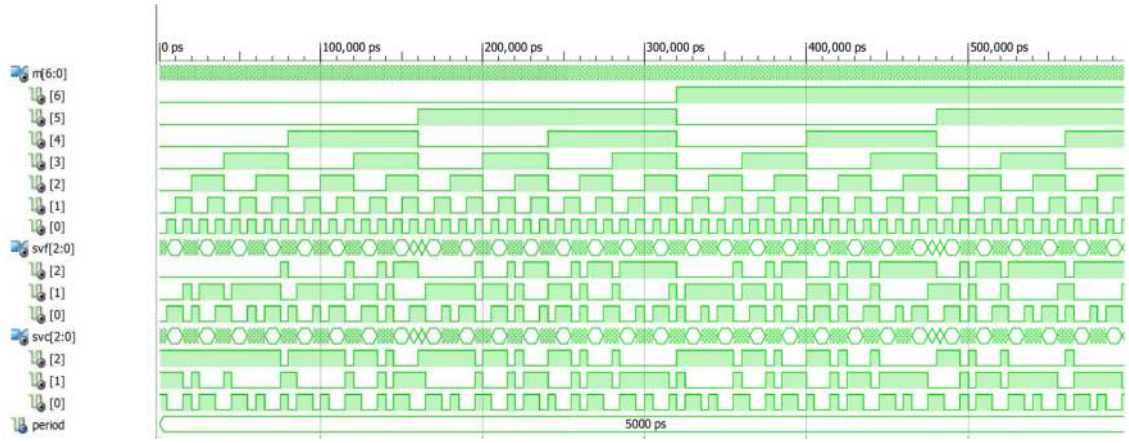
5.1 Simulación del bloque Cuenta_Votos

En el 'Test Bench' se simula la respuesta del bloque Cuenta_Votos ante todas las entradas posibles con 7 bits (el equivalente a 7 votantes).

El fichero 'Test Bench' utilizado para simular este bloque es el siguiente:

```
--
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE ieee.std_logic_arith.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;
ENTITY Cuenta_Votos_Cuenta_Votos_sch_tb IS
END Cuenta_Votos_Cuenta_Votos_sch_tb;
ARCHITECTURE behavioral OF Cuenta_Votos_Cuenta_Votos_sch_tb IS

   COMPONENT Cuenta_Votos
   PORT( M   :  IN STD_LOGIC_VECTOR (6 DOWNTO 0);
         SVF  :  OUT   STD_LOGIC_VECTOR (2 DOWNTO 0);
         SVC  :  OUT   STD_LOGIC_VECTOR (2 DOWNTO 0));
   END COMPONENT;

   SIGNAL M :  STD_LOGIC_VECTOR (6 DOWNTO 0);
   SIGNAL SVF  :  STD_LOGIC_VECTOR (2 DOWNTO 0);
   SIGNAL SVC  :  STD_LOGIC_VECTOR (2 DOWNTO 0);
   constant PERIOD: Time:= 5 ns;

BEGIN

   UUT: Cuenta_Votos PORT MAP(
      M => M,
      SVF => SVF,
      SVC => SVC
   );

-- *** Test Bench - User Defined Section ***
  tb : PROCESS
    variable i : integer;
    variable INPUT : std_logic_vector(6 downto 0);
    BEGIN
        for i in 0 to 128 loop
            INPUT := CONV_STD_LOGIC_VECTOR(i,7);
            M(6)<= INPUT(6);
            M(5)<= INPUT(5);
            M(4)<= INPUT(4);
            M(3)<= INPUT(3);
            M(2)<= INPUT(2);
            M(1)<= INPUT(1);
            M(0)<= INPUT(0);

        wait for PERIOD;
    end loop;
    INPUT := "0000000";
            M(6)<= INPUT(6);
            M(5)<= INPUT(5);
            M(4)<= INPUT(4);
            M(3)<= INPUT(3);
            M(2)<= INPUT(2);
            M(1)<= INPUT(1);
            M(0)<= INPUT(0);
    WAIT;
    END PROCESS;

--  End Test Bench - User Defined Section

END;
```

Y la respuesta obtenida es:



Se ha comprobado que el número binario obtenido en svf (siendo svf(2) el bit más significativo) es el equivalente al número de 1's introducidos en M(6:0), mientras que el número binario obtenido en svc (siendo svc(2) el bit más significativo) es el equivalente a 7 menos el número de votos a favor.

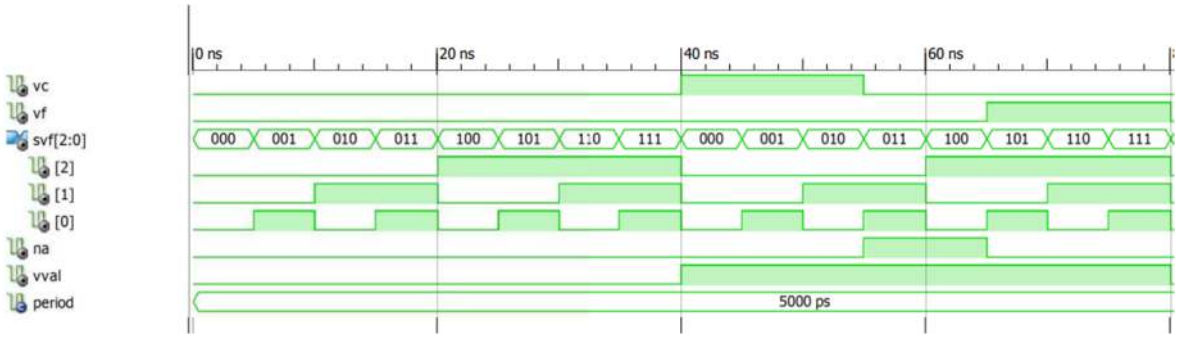
5.2 Simulación del bloque Resultado_Votación

En el 'Test Bench' se simula la respuesta del bloque Resultado_Votacion ante todas las entradas posibles provenientes del bloque Cuenta_Votos.

El fichero 'Test Bench' utilizado para simular este bloque es el siguiente:

```
--
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE ieee.std_logic_arith.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;
ENTITY Resultado_Votacion_Resultado_Votacion_sch_tb IS
END Resultado_Votacion_Resultado_Votacion_sch_tb;
ARCHITECTURE behavioral OF Resultado_Votacion_Resultado_Votacion_sch_tb IS

   COMPONENT Resultado_Votacion
   PORT( VC :  OUT   STD_LOGIC;
         VP   :  OUT   STD_LOGIC;
         SVP  :  IN STD_LOGIC_VECTOR (2 DOWNTO 0);
         NA   :  OUT   STD_LOGIC;
         VVal :  IN STD_LOGIC);
   END COMPONENT;

   SIGNAL VC   :  STD_LOGIC;
   SIGNAL VP   :  STD_LOGIC;
   SIGNAL SVP  :  STD_LOGIC_VECTOR (2 DOWNTO 0);
   SIGNAL NA   :  STD_LOGIC;
   SIGNAL VVal :  STD_LOGIC;
   constant PERIOD: Time:=5 ns;

BEGIN

   UUT: Resultado_Votacion PORT MAP(
      VC -> VC,
      VP -> VP,
      SVP -> SVP,
      NA -> NA,
      VVal -> VVal
   );

-- *** Test Bench - User Defined Section ***
    tb : PROCESS
    variable i : integer;
    variable INPUT : std_logic_vector(3 downto 0);
    BEGIN
        for i in 0 to 15 loop
            INPUT := CONV_STD_LOGIC_VECTOR(i,4);
            VVal<= INPUT(3);
            SVP(2)<= INPUT(2);
            SVP(1)<= INPUT(1);
            SVP(0)<= INPUT(0);
        wait for PERIOD;
    end loop;
    INPUT := "0000";
            VVal<= INPUT(3);
            SVP(2)<= INPUT(2);
            SVP(1)<= INPUT(1);
            SVP(0)<= INPUT(0);
    WAIT;
    END PROCESS;

--  End Test Bench - User Defined Section

END;
```

Y la respuesta obtenida es:



Se ha comprobado que cuando la señal 'ENABLE' está deshabilitada, no se obtiene conclusión sobre la votación, mientras que si está activa, cuando hay 5, 6 o 7 votos a favor la votación resulta favorable (VF), cuando hay 0, 1 o 2 votos a favor (es decir, 5, 6 o 7 en contra) la votación resulta en desfavorable; y cuando el número de votos a favor o en contra es 3 o 4 la votación resulta no aprobada.

Una vez comprobado el funcionamiento de los bloques, se incorporan al esquemático de MyDesign teniendo como resultado final: