Instituto Politécnico Nacional

Escuela Superior de Computo

Práctica 7.

Archivo de Registros.

Nombre: Flores Castro Luis Antonio.

Arquitectura de Computadoras.

Profesora: Vega García Nayeli.

Código VHDL de Archivo de Registros.

```
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 FileReg is
7.         Port( writeReg, readReg1, readReg2,
   shamt: in STD_LOGIC_VECTOR(3 downto 0);
8.                 writeData: in STD_LOGIC_VECTOR(15 downto 0);
9.                 clk,clr,wr,she,dir: in STD_LOGIC;
10.                     readData1,
   readData2: out STD_LOGIC_VECTOR(15 downto 0));
11.     end FileReg;
12.
13.     architecture Behavioral of FileReg is
14.     type palabra is array (0 to 15) of std_logic_vector(15 d
   ownto 0);
15.     signal banco : palabra;
16.     begin
17.     process(clk,clr)
18.     variable aux:std_logic_vector(15 downto 0);
19.     begin
20.             if(clr='1')then
21.                     banco<= (others=>(others=>'0'));
22.             elsif (rising_edge(clk))then
23.             --escritura
24.             if(wr='1' and she='0')then--carga
25.                     banco(conv_integer(writeReg))<=writeData
   ;
26.             elsif(wr='1' and she='1' and dir ='0')then --
   corrimiento a la derecha
27.             --aqui meter barrel shifter
28.                       aux:=banco(conv_integer(readReg1));
29.                       for i in 0 to 3 loop
30.                           for j in 0 to 15-2**i loop
31.                               if shamt(i)='0' then
32.                                   aux(j):=aux(j);
33.                               else
34.                                   aux(j):=aux(j+2**i);
35.                               end if;
36.                           end loop;
37.                           for j in 15-2**i+1 to 15 loop
38.                               if shamt(i)='0' then
39.                                   aux(j):=aux(j);
40.                               else
41.                                   aux(j):='0';
42.                               end if;
43.                            end loop;
44.                       end loop;
```

```
45.                         banco(conv_integer(writeReg))<=aux;
46.             elsif(wr='1' and she='1' and dir ='1')then
47.             --izquierda
48.                     aux:=banco(conv_integer(readReg1));
49.                     for i in 0 to 3 loop
50.                       for j in 15 downto 2**i loop
51.                           if shamt(i)='0' then
52.                               aux(j):=aux(j);
53.                           else
54.                               aux(j):=aux(j-2**i);
55.                           end if;
56.                       end loop;
57.                       for j in 2**i-1 downto 0 loop
58.                           if shamt(i)='0' then
59.                               aux(j):=aux(j);
60.                           else
61.                               aux(j):='0';
62.                           end if;
63.                       end loop;
64.                     end loop;
65.                     banco(conv_integer(writeReg))<=aux;
66.             end if;
67.     end if;
68.     end process;
69.
70.     readData1<=banco(conv_integer(readReg1));
71.     readData2<=banco(conv_integer(readReg2));
72.
73.     end behavioral;
```

Código VHDL Test-Bench de Archivo de Registros.

```
1. library IEEE;
2. library STD;
3. use STD.TEXTIO.ALL;
4. use IEEE.STD_LOGIC_TEXTIO.ALL;
5. use IEEE.STD_LOGIC_1164.ALL;
6. use ieee.std_logic_arith.all;
7. use ieee.std_logic_unsigned.all;
8.
9. entity Tbreg is
10.     end Tbreg;
11.
12.     architecture Behavioral of Tbreg is
13.
14.     component FileReg is
15.             Port( writeReg, readReg1, readReg2,
   shamt: in STD_LOGIC_VECTOR(3 downto 0);
16.                   writeData: in STD_LOGIC_VECTOR(15 downto
    0);
17.                   clk,clr,wr,she,dir: in STD_LOGIC;
18.                   readData1,
   readData2: out STD_LOGIC_VECTOR(15 downto 0));
19.     end component;
20.
21.
22.     --NUEVAS--
23.     signal wrD :STD_LOGIC_VECTOR(15 downto 0);--writeData
24.     signal wrReg:STD_LOGIC_VECTOR(3 downto 0);--writeReg
25.     signal readR1: STD_LOGIC_VECTOR(3 downto 0);--read1
26.     signal readR2: STD_LOGIC_VECTOR(3 downto 0);--read2
27.     signal sh: STD_LOGIC_VECTOR(3 downto 0);--shamt
28.     signal clk,clr,she,rw,dir: STD_LOGIC;
29.
30.     signal readD2  : STD_LOGIC_VECTOR (15 downto 0);--
   readDatas
31.     signal readD1  : STD_LOGIC_VECTOR (15 downto 0);--
   salidas
32.
33.     begin
34.
35.     clock : process
36.             begin
37.                 clk <= '0';
38.                 wait for 5 ns;
39.                 clk <= '1';
40.                 wait for 5 ns;
41.             end process;
42.
43.     TestFR: FileReg
44.         Port map(
```

```
45.             writeReg => wrReg,
46.             readReg1 => readR1,
47.             readReg2 => readR2,
48.             shamt => sh,
49.             writeData => wrD,
50.             clk=>clk,
51.             clr=>clr,
52.             wr=>rw,
53.             she => she,
54.             dir => dir,
55.             readData1 => readD1,
56.             readData2 => readD2
57.             );
58.
59.     stimulators: process
60.         file RES : TEXT;
61.             variable L_RE : line;
62.             variable vrd1: STD_LOGIC_VECTOR(15 downto 0);--
   salidas
63.             variable vrd2: STD_LOGIC_VECTOR(15 downto 0);--
   salidas
64.
65.         file STIMU : TEXT;
66.             variable L_E : line;
67.             variable vclr: STD_LOGIC;
68.             variable vrR1:STD_LOGIC_VECTOR(3 downto 0);--
   readReg
69.             variable vrR2: STD_LOGIC_VECTOR(3 downto 0);
70.             variable vshamt: STD_LOGIC_VECTOR(3 downto 0);
71.             variable vwrRe: STD_LOGIC_VECTOR(3 downto 0);
72.             variable vwd: STD_LOGIC_VECTOR(15 downto 0);
73.             variable vwr: STD_LOGIC;
74.             variable vshe: STD_LOGIC;
75.             variable vdir: STD_LOGIC;
76.
77.             --lectura de cadenas
78.             variable CADENA: STRING(1 to 5);
79.
80.             begin
81.             file_open(STIMU, "C:\Users\Luis
   FC\Documents\Semestre 21-2\Arquitectura de
   Computadoras\AR2\ESTIMULOS.TXT", READ_MODE);
82.             file_open(RES,"C:\Users\Luis
   FC\Documents\Semestre 21-2\Arquitectura de
   Computadoras\AR2\RESULTADO.TXT", WRITE_MODE);
83.
84.             CADENA:= "RR1  ";
85.             write(L_RE,CADENA,left,1);
86.             CADENA:= "RR2  ";
87.             write(L_RE,CADENA,left,1);
88.             CADENA:= "SHAMT";
89.             write(L_RE,CADENA,left,6);
```

```
90.             CADENA:= "WREG ";
91.             write(L_RE,CADENA,left,1);
92.             CADENA:= "WD   ";
93.             write(L_RE,CADENA,left,1);
94.             CADENA:= "CLR  ";
95.             write(L_RE,CADENA,left,1);
96.             CADENA:= "WR   ";
97.             write(L_RE,CADENA,left,1);
98.             CADENA:= "SHE  ";
99.             write(L_RE,CADENA,left,1);
100.            CADENA:= "DIR  ";
101.            write(L_RE,CADENA,left,1);
102.            CADENA:= "RD1  ";
103.            write(L_RE,CADENA,left,2);
104.            CADENA:= "RD2  ";
105.            write(L_RE,CADENA,left,2);
106.            writeline(RES,L_RE);
107.
108.            wait for 80ns;
109.
110.            FOR i in 0 to 11 loop
111.                readline(STIMU,L_E);
112.                hread(L_E,vrR1);
113.                readR1<=vrR1;
114.
115.                hread(L_E,vrR2);
116.                readR2<=vrR2;
117.
118.                hread(L_E,vshamt);
119.                sh<=vshamt;
120.
121.                hread(L_E,vwrRe);
122.                wrReg<=vwrRe;
123.
124.                hread(L_E,vwd);
125.                wrD<=vwd;
126.
127.                read(L_E, vclr);
128.                clr<=vclr;
129.
130.                read(L_E, vwr);
131.                rw<=vwr;
132.
133.                read(L_E, vshe);
134.                she<=vshe;
135.
136.                read(L_E,vdir);
137.                dir<=vdir;
138.
139.                wait until rising_edge(clk);
140.
141.                vrd1:=readD1;
```

```
142.                vrd2:=readD2;
143.
144.                hwrite(L_RE, vrR1, left, 5);
145.                hwrite(L_RE, vrR2, left, 5);
146.                hwrite(L_RE, vshamt, left, 6);
147.                hwrite(L_RE, vwrRe, left, 5);
148.                hwrite(L_RE, vwd, left, 5);
149.                write(L_RE, vclr, left, 5);
150.                write(L_RE, vwr, left, 5);
151.                write(L_RE, vshe, left, 5);
152.                write(L_RE, vdir, left, 5);
153.                hwrite(L_RE, vrd1, left, 5);
154.                hwrite(L_RE, vrd2, left, 5);
155.
156.                writeline(RES,L_RE);
157.                end loop;
158.                file_close(STIMU);
159.                file_close(RES);
160.
161.                wait;
162.                end process;
163.    end Behavioral;
```

Estimulos ingresados al Archivo de Registros.

1. Reset
2. Banco[1] = 89
3. Banco[2] = 72
4. Banco[3] = 123
5. Banco[4] = 53
6. Leer Banco[1] y Banco[2]
7. Leer Banco[3] y Banco[4]
8. Banco[2] = Banco[1] << 3
9. Banco[4] = Banco[3] >> 5
10. Leer Banco[1] y Banco[2]
11. Leer Banco[3] y Banco[4]
12. Reset

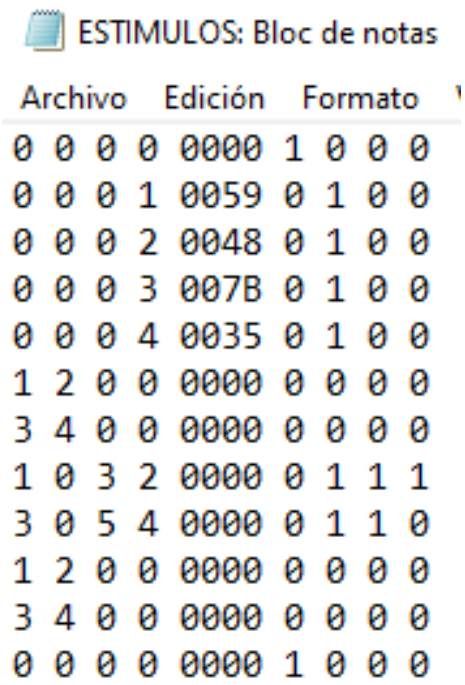
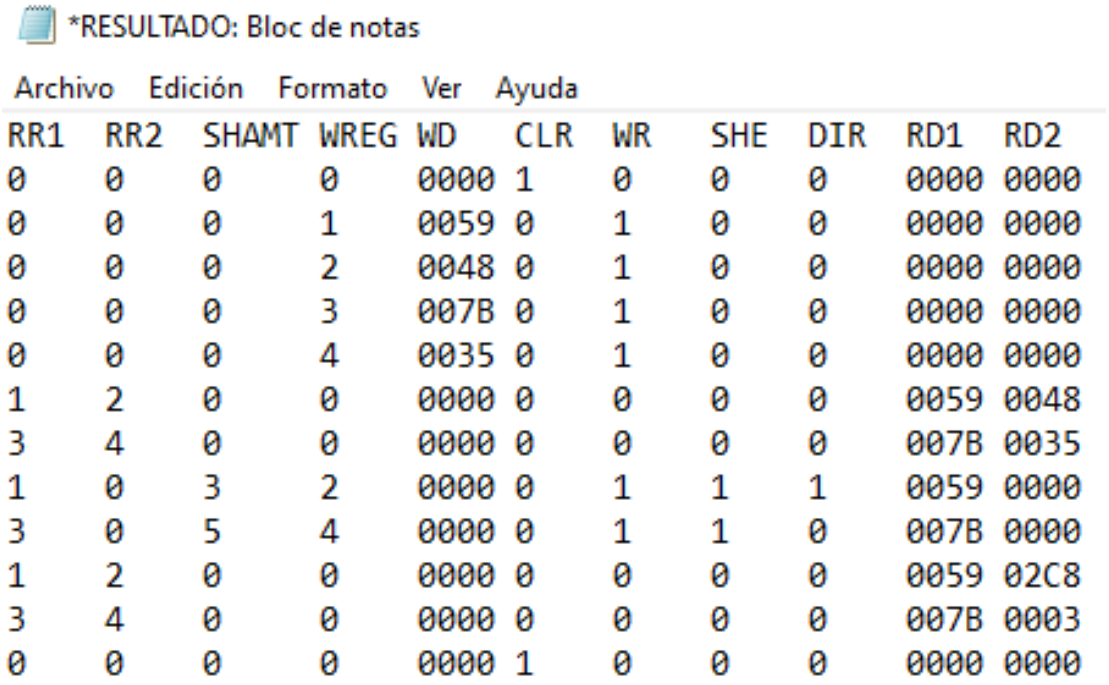


Figura 1. Estímulos ingresados por medio de archivo de textos a Test Bench.

Resultados obtenidos mediante la simulación del Test Bench del Archivo de Registros.



*RESULTADO: Bloc de notas

Archivo Edición Formato Ver Ayuda

RR1	RR2	SHAMT	WREG	WD	CLR	WR	SHE	DIR	RD1	RD2
0	0	0	0	0000	1	0	0	0	0000	0000
0	0	0	1	0059	0	1	0	0	0000	0000
0	0	0	2	0048	0	1	0	0	0000	0000
0	0	0	3	007B	0	1	0	0	0000	0000
0	0	0	4	0035	0	1	0	0	0000	0000
1	2	0	0	0000	0	0	0	0	0059	0048
3	4	0	0	0000	0	0	0	0	007B	0035
1	0	3	2	0000	0	1	1	1	0059	0000
3	0	5	4	0000	0	1	1	0	007B	0000
1	2	0	0	0000	0	0	0	0	0059	02C8
3	4	0	0	0000	0	0	0	0	007B	0003
0	0	0	0	0000	1	0	0	0	0000	0000

Figura 1.1. Estímulos ingresados por medio de archivo de textos a Test Bench.

Diagrama RTL.

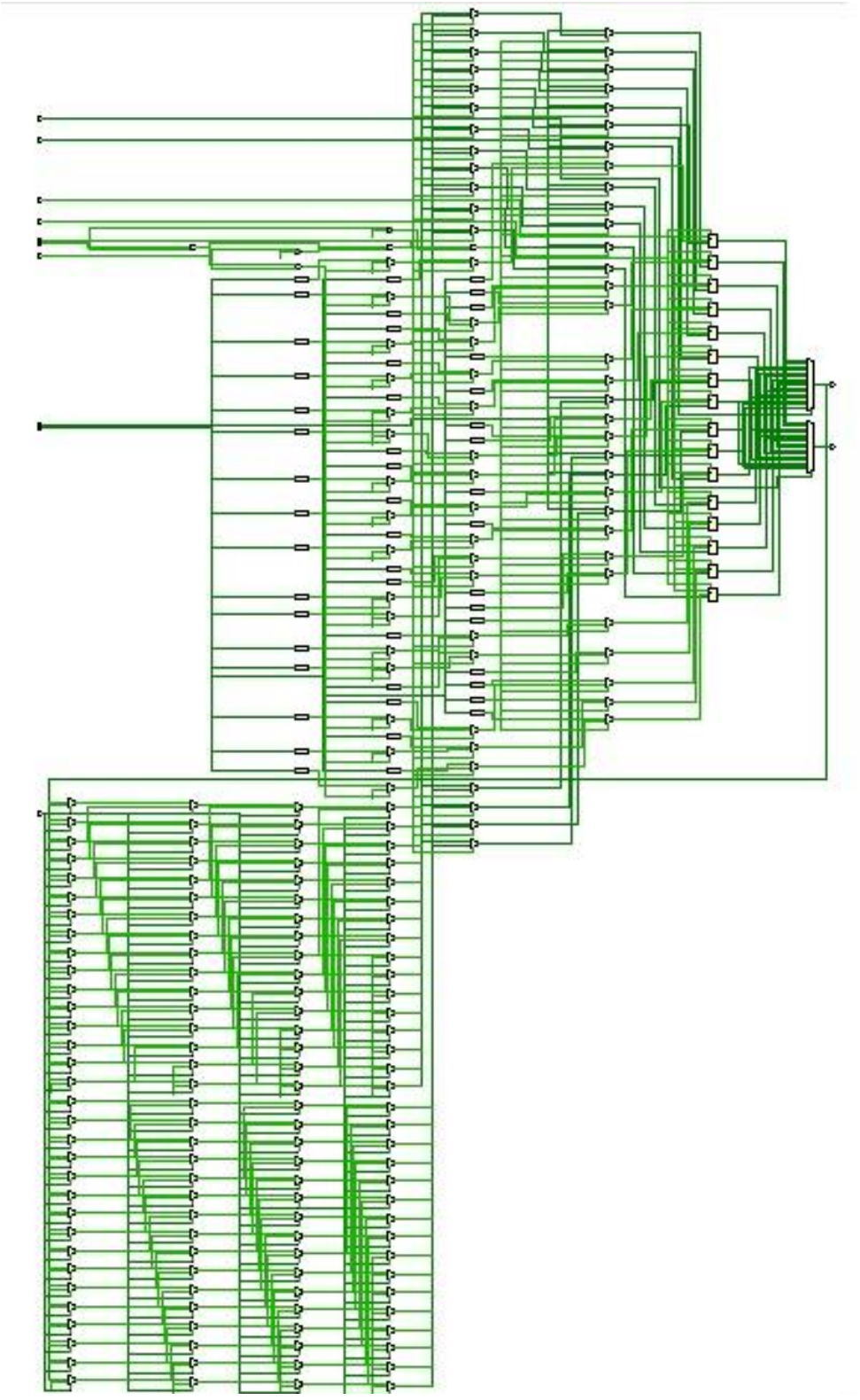


Figura 2. Circuito esquemático del Archivo de Registros.

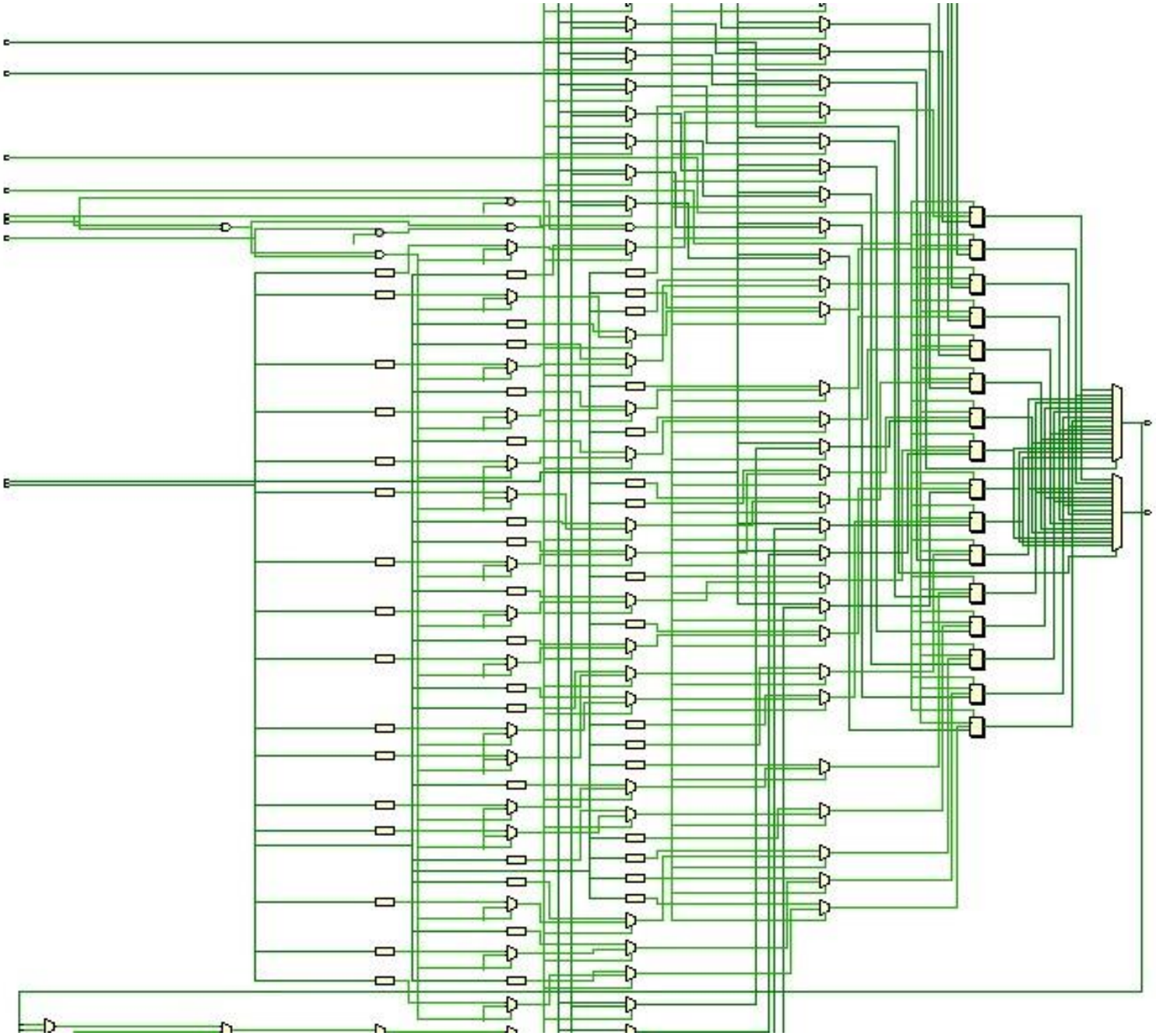


Figura 2.1 Circuito esquemático del Archivo de Registros (expandido).

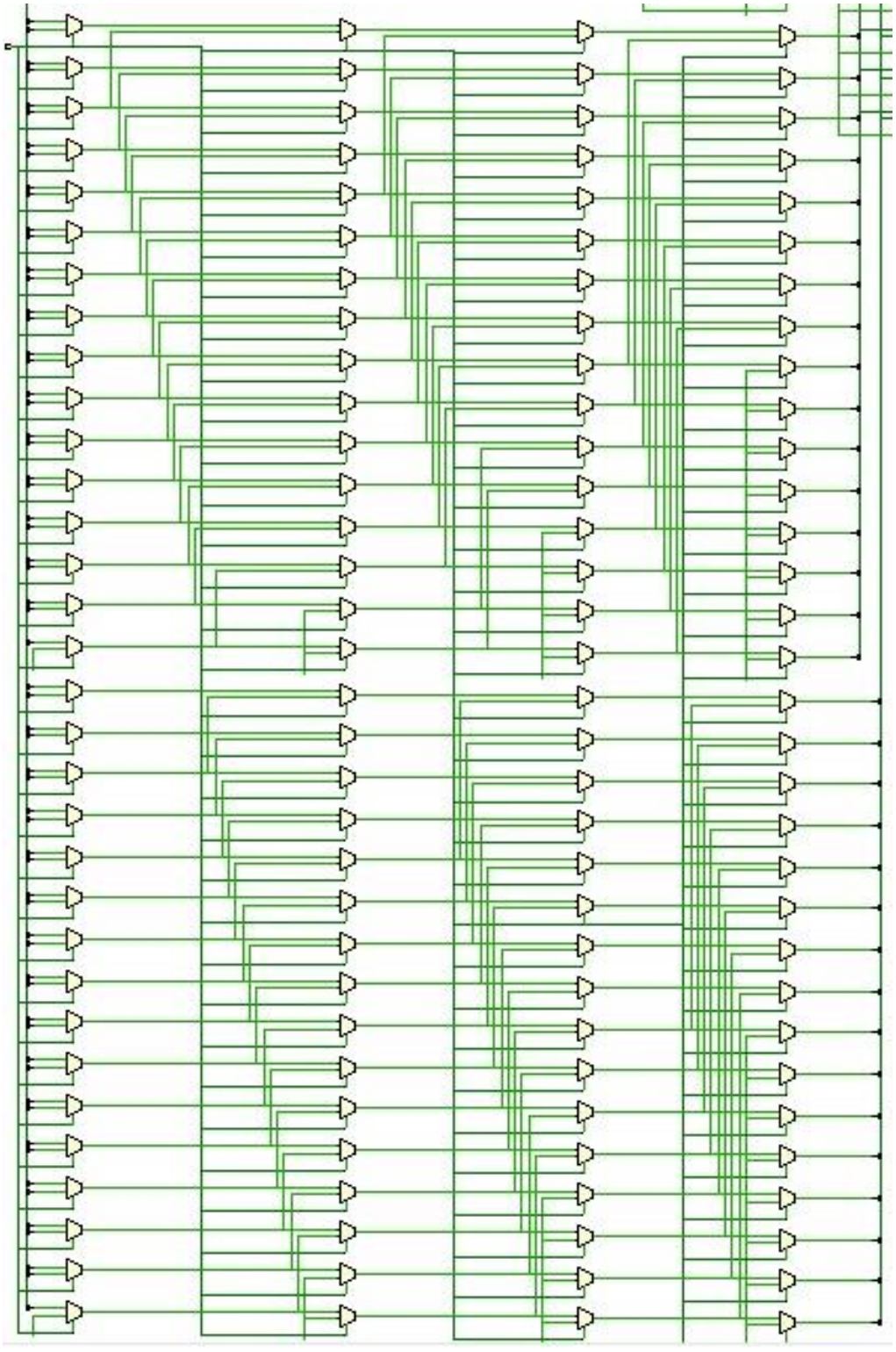


Figura 2.2 Circuito esquemático del Archivo de Registros (expandido parte Barrel Shifter).

Diagrama RTL.

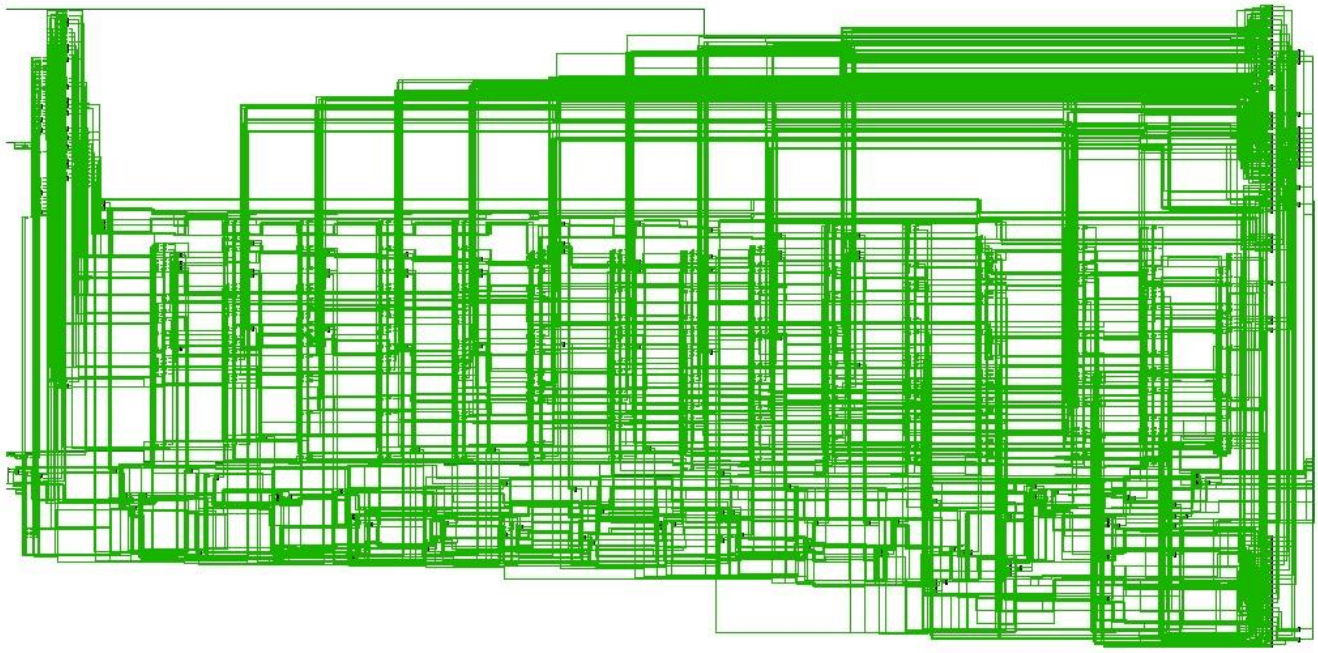


Figura 2.3. Circuito lógico del Archivo de Registros.

Diagrama de Onda del Archivo de Registros.

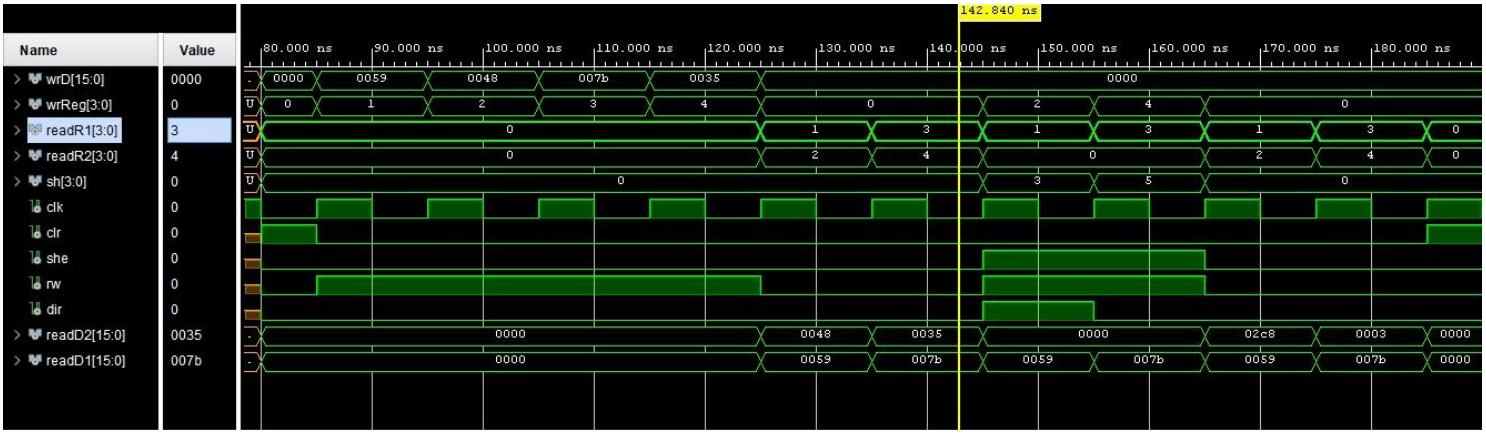


Figura 3. Simulación del Test-Bench del Archivo de Registros.