INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE COMPUTO

Sumador de 8 bits

Practica 6

Materia:

Arquitectura de computadoras

Profesor:

Castillo Cabrera Gelacio

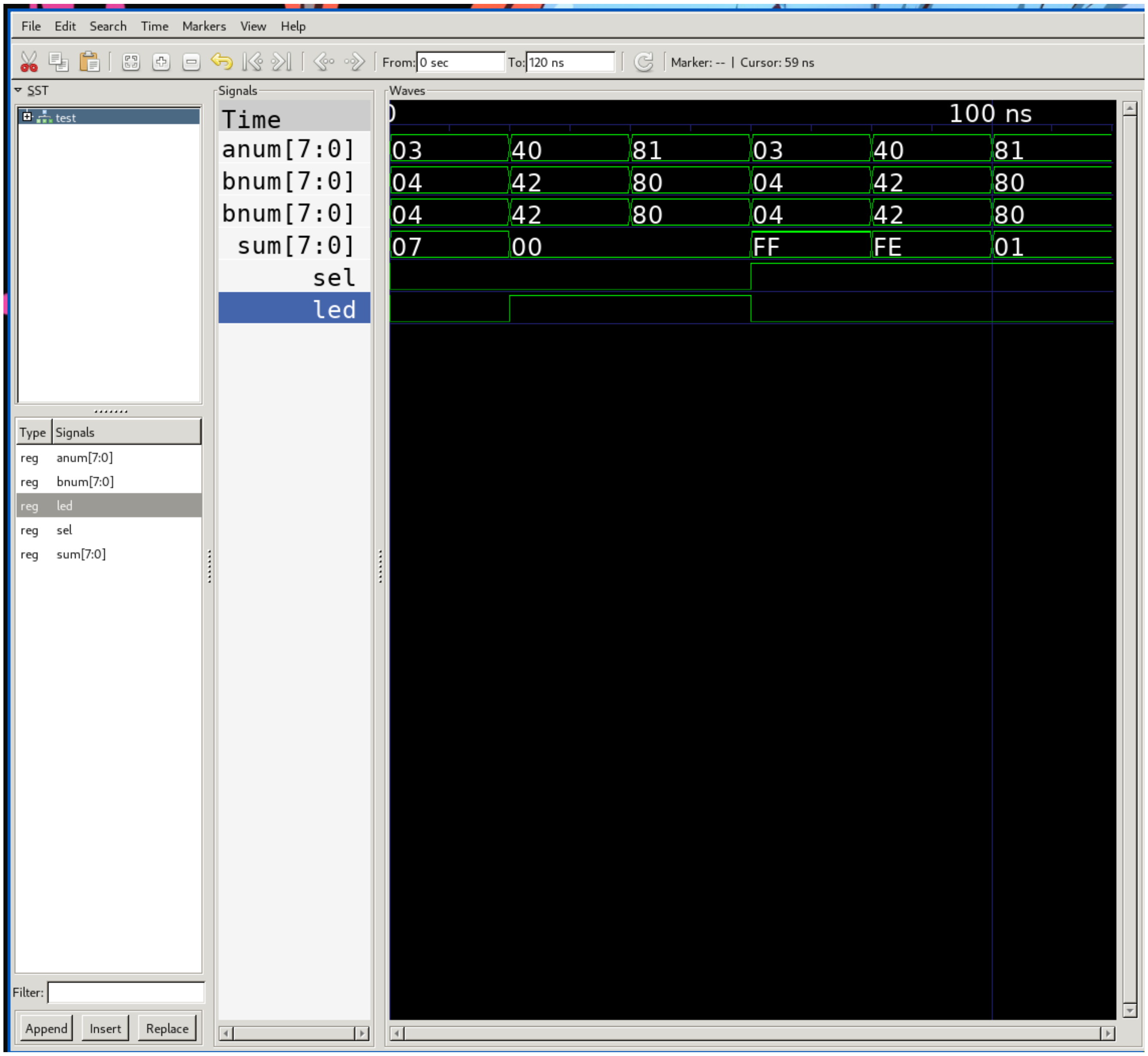
Alumno:

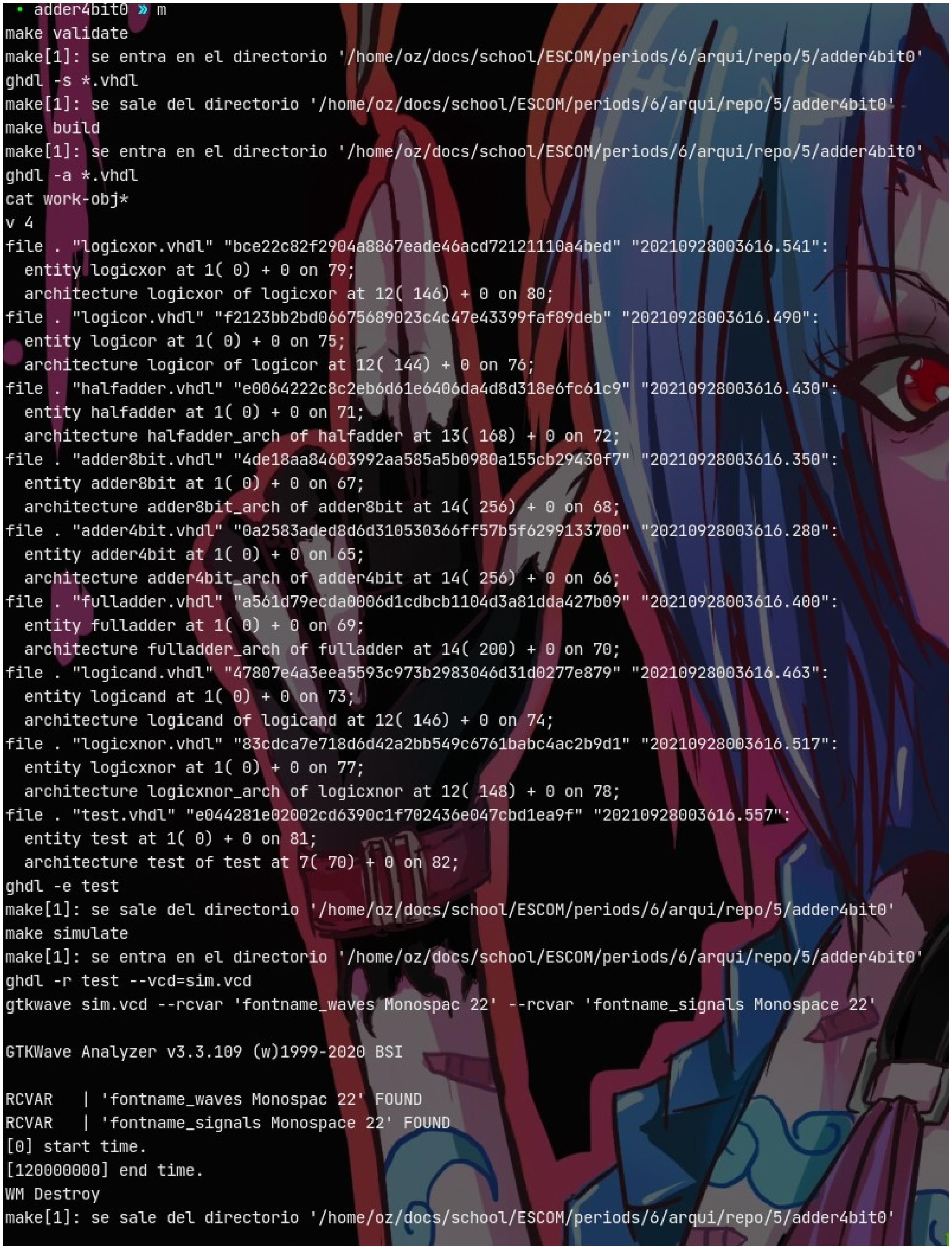
Cortés Piña Oziel

Grupo:

3CM12

Simulación en GHDL y GTLWAVE





```
• adder4bit0 » m
make validate
make[1]: se entra en el directorio '/home/oz/docs/school/ESCOM/periods/6/arqui/repo/5/adder4bit0'
ghdl -s *.vhdl
make[1]: se sale del directorio '/home/oz/docs/school/ESCOM/periods/6/arqui/repo/5/adder4bit0'
make build
make[1]: se entra en el directorio '/home/oz/docs/school/ESCOM/periods/6/arqui/repo/5/adder4bit0'
ghdl -a *.vhdl
cat work-obj*
v 4
file . "logicxor.vhdl" "bce22c82f2904a8867eade46acd72121110a4bed" "20210928003616.541":
  entity logicxor at 1( 0) + 0 on 79;
  architecture logicxor of logicxor at 12( 146) + 0 on 80;
file . "logicor.vhdl" "f2123bb2bd06675689023c4c47e43399faf89deb" "20210928003616.490":
  entity logicor at 1( 0) + 0 on 75;
  architecture logicor of logicor at 12( 144) + 0 on 76;
file . "halfadder.vhdl" "e0064222c8c2eb6d61e6406da4d8d318e6fc61c9" "20210928003616.430":
  entity halfadder at 1( 0) + 0 on 71;
  architecture halfadder_arch of halfadder at 13( 168) + 0 on 72;
file . "adder8bit.vhdl" "4de18aa84603992aa585a5b0980a155cb29430f7" "20210928003616.350":
  entity adder8bit at 1( 0) + 0 on 67;
  architecture adder8bit_arch of adder8bit at 14( 256) + 0 on 68;
file . "adder4bit.vhdl" "0a2583aded8d6d310530366ff57b5f6299133700" "20210928003616.280":
  entity adder4bit at 1( 0) + 0 on 65;
  architecture adder4bit_arch of adder4bit at 14( 256) + 0 on 66;
file . "fulladder.vhdl" "a561d79ecda0006d1cdbcb1104d3a81dda427b09" "20210928003616.400":
  entity fulladder at 1( 0) + 0 on 69;
  architecture fulladder_arch of fulladder at 14( 200) + 0 on 70;
file . "logicand.vhdl" "47807e4a3eea5593c973b2983046d31d0277e879" "20210928003616.463":
  entity logicand at 1( 0) + 0 on 73;
  architecture logicand of logicand at 12( 146) + 0 on 74;
file . "logicxnor.vhdl" "83cdca7e718d6d42a2bb549c6761babc4ac2b9d1" "20210928003616.517":
  entity logicxnor at 1( 0) + 0 on 77;
  architecture logicxnor_arch of logicxnor at 12( 148) + 0 on 78;
file . "test.vhdl" "e044281e02002cd6390c1f702436e047cbd1ea9f" "20210928003616.557":
  entity test at 1( 0) + 0 on 81;
  architecture test of test at 7( 70) + 0 on 82;
ghdl -e test
make[1]: se sale del directorio '/home/oz/docs/school/ESCOM/periods/6/arqui/repo/5/adder4bit0'
make simulate
make[1]: se entra en el directorio '/home/oz/docs/school/ESCOM/periods/6/arqui/repo/5/adder4bit0'
ghdl -r test --vcd=sim.vcd
gtkwave sim.vcd --rcvar 'fontname_waves Monospac 22' --rcvar 'fontname_signals Monospace 22'

GTKWave Analyzer v3.3.109 (w)1999-2020 BSI

RCVAR   | 'fontname_waves Monospac 22' FOUND
RCVAR   | 'fontname_signals Monospace 22' FOUND
[0] start time.
[120000000] end time.
WM Destroy
make[1]: se sale del directorio '/home/oz/docs/school/ESCOM/periods/6/arqui/repo/5/adder4bit0'
```

Código VHDL

Logicand

```
library ieee;
use ieee.std_logic_1164.all;

entity logicand is
      port(
            a:    in    std_logic;
            b:    in    std_logic;
            o:    out   std_logic
      );
end logicand;

architecture logicand of logicand is
begin
      o <= a and b;
end logicand;
```

Logicor

```
library ieee;
use ieee.std_logic_1164.all;

entity logicor is
      port(
            a:    in    std_logic;
            b:    in    std_logic;
            o:    out   std_logic
      );
end logicor;

architecture logicor of logicor is
begin
      o <= a or b;
end logicor;
```

Logicxor

```
library ieee;
use ieee.std_logic_1164.all;

entity logicxor is
      port(
            a:    in    std_logic;
            b:    in    std_logic;
            o:    out   std_logic
      );
end logicxor;

architecture logicxor of logicxor is
begin
      o <= a xor b;
end logicxor;
```

Logicxnor

```
library ieee;
use ieee.std_logic_1164.all;

entity logicxnor is
      port(
            a:    in    std_logic;
            b:    in    std_logic;
            o:    out   std_logic
      );
end logicxnor;

architecture logicxnor_arch of logicxnor is
begin
      o <= a xnor b;
end logicxnor_arch;
```

Halfadder

```
library ieee;
use ieee.std_logic_1164.all;

entity halfadder is
      port(
            a:    in    std_logic;
            b:    in    std_logic;
            o:    out   std_logic;
            c:    out   std_logic
      );
end halfadder;

architecture halfadder_arch of halfadder is
      component logicand is
            port(
                  a:    in    std_logic;
                  b:    in    std_logic;
                  o:    out   std_logic
            );
      end component;
      component logicxor is
            port(
                  a:    in    std_logic;
                  b:    in    std_logic;
                  o:    out   std_logic
            );
      end component;
begin
      sum:  logicxor port map( a => a, b => b, o => o );
      carry:      logicand port map( a => a, b => b, o => c );
end halfadder_arch;
```

Fullader

```
library ieee;
use ieee.std_logic_1164.all;

entity fulladder is
      port(
            anum: in    std_logic;
            bnum: in    std_logic;
            cin:  in    std_logic;
            sum:  out   std_logic;
            cout: out   std_logic
      );
end fulladder;

architecture fulladder_arch of fulladder is
      component halfadder is
            port(
                  a:    in    std_logic;
                  b:    in    std_logic;
                  o:    out   std_logic;
                  c:    out   std_logic
            );
      end component;
      component logicor is
            port(
                  a:    in    std_logic;
                  b:    in    std_logic;
                  o:    out   std_logic
            );
      end component;
      signal carryA, carryB, sumPass : std_logic;
begin
      firtsum:    halfadder   port map( a => anum,    b => bnum,        o => 
sumPass,    c => carryB );
      secondsum:  halfadder   port map( a => cin,           b => sumPass,     o => 
sum,        c => carryA );
      handlec:    logicor           port map( a => carryA,  b => carryB,      o => 
cout );
end fulladder_arch;
```

Adder8bit

```
library ieee;
use ieee.std_logic_1164.all;

entity adder8bit is
      port(
            sel:   in std_logic;
            anum:  in    std_logic_vector(7 downto 0);
            bnum:  in    std_logic_vector(7 downto 0);
            led:   out std_logic;
            sum:   out   std_logic_vector(7 downto 0)
      );
end adder8bit;

architecture adder8bit_arch of adder8bit is
      component fulladder  is
```

```
		port(
			anum:	in	std_logic;
			bnum:	in	std_logic;
			cin:	in	std_logic;
			sum:	out	std_logic;
			cout:	out	std_logic
		);
	end component;
	component logicxor  is
		port(
			a:	in	std_logic;
			b:	in	std_logic;
			o:	out	std_logic
		);
	end component;
	component logicxnor is
		port(
			a:	in	std_logic;
			b:	in	std_logic;
			o:	out	std_logic
		);
	end component;
	component logicand  is
		port(
			a:	in	std_logic;
			b:	in	std_logic;
			o:	out	std_logic
		);
	end component;
	signal carry:std_logic_vector(7 downto 0);
	signal bneg: std_logic_vector(7 downto 0);
	signal sneg: std_logic_vector(7 downto 0);
	signal cend: std_logic;
begin
	rest0:	logicxor	port map( a => sel, b => bnum(0), o => bneg(0) );
	rest1:	logicxor	port map( a => sel, b => bnum(1), o => bneg(1) );
	rest2:	logicxor	port map( a => sel, b => bnum(2), o => bneg(2) );
	rest3:	logicxor	port map( a => sel, b => bnum(3), o => bneg(3) );
	rest4:	logicxor	port map( a => sel, b => bnum(4), o => bneg(4) );
	rest5:	logicxor	port map( a => sel, b => bnum(5), o => bneg(5) );
	rest6:	logicxor	port map( a => sel, b => bnum(6), o => bneg(6) );
	rest7:	logicxor	port map( a => sel, b => bnum(7), o => bneg(7) );
	sneg0:	fulladder	port map(anum => anum(0), bnum => bneg(0),	cin => sel,
	cout => carry(0),	sum => sneg(0) );
	sneg1:	fulladder	port map(anum => anum(1), bnum => bneg(1),	cin => carry(0),
	cout => carry(1),	sum => sneg(1) );
	sneg2:	fulladder	port map(anum => anum(2), bnum => bneg(2),	cin => carry(1),
	cout => carry(2),	sum => sneg(2) );
	sneg3:	fulladder	port map(anum => anum(3), bnum => bneg(3),	cin => carry(2),
	cout => carry(3),	sum => sneg(3) );
	sneg4:	fulladder	port map(anum => anum(4), bnum => bneg(4),	cin => carry(3),
	cout => carry(4),	sum => sneg(4) );
	sneg5:	fulladder	port map(anum => anum(5), bnum => bneg(5),	cin => carry(4),
	cout => carry(5),	sum => sneg(5) );
	sneg6:	fulladder	port map(anum => anum(6), bnum => bneg(6),	cin => carry(5),
	cout => carry(6),	sum => sneg(6) );
	sneg7:	fulladder	port map(anum => anum(7), bnum => bneg(7),	cin => carry(6),
	cout => carry(7),	sum => sneg(7) );
	ledh:	logicxor	port map( a => carry(7), b => carry(6), o => led  );
	idkw:	logicxnor	port map( a => carry(7), b => carry(6), o => cend );
	sum0:	logicand	port map( a => cend,	b => sneg(0),	o => sum(0) );
	sum1:	logicand	port map( a => cend,	b => sneg(1),	o => sum(1) );
	sum2:	logicand	port map( a => cend,	b => sneg(2),	o => sum(2) );
	sum3:	logicand	port map( a => cend,	b => sneg(3),	o => sum(3) );
```

```
      sum4:  logicand     port map( a => cend,       b => sneg(4),      o => sum(4) );
      sum5:  logicand     port map( a => cend,       b => sneg(5),      o => sum(5) );
      sum6:  logicand     port map( a => cend,       b => sneg(6),      o => sum(6) );
      sum7:  logicand     port map( a => cend,       b => sneg(7),      o => sum(7) );
end adder8bit_arch;
```

Test

```
library ieee;
use ieee.std_logic_1164.all;

entity test is
end test;

architecture test of test is
      component adder8bit is
            port(
                  sel:  in    std_logic;
                  anum: in    std_logic_vector(7 downto 0);
                  bnum: in    std_logic_vector(7 downto 0);
                  led:  out   std_logic;
                  sum:  out   std_logic_vector(7 downto 0)
            );
      end component;
      signal anum, bnum, sum:       std_logic_vector(7 downto 0);
      signal sel, led:  std_logic;
begin
      conecctions : adder8bit port map( anum => anum, bnum => bnum, sel => sel,
sum => sum, led => led );
      process begin
            sel <= '0';
            anum <= "00000011";
            bnum <= "00000100";
            wait for 20 ns;
            anum <= "01000000";
            bnum <= "01000010";
            wait for 20 ns;
            anum <= "10000001";
            bnum <= "10000000";
            wait for 20 ns;
            sel <= '1';
            anum <= "00000011";
            bnum <= "00000100";
            wait for 20 ns;
            anum <= "01000000";
            bnum <= "01000010";
            wait for 20 ns;
            anum <= "10000001";
            bnum <= "10000000";
            wait for 20 ns;
            wait;
      end process;
end test;
```

Análisis de vectores

Vectores de entrada

A		B	
Binario	Hexadecimal	Binario	Hexadecimal
00000011	3	00000100	4
01000000	40	01000010	42
10000001	81	10000000	80

Valores de salida

A		B		Selector		
Binario	Hexa	Binario	Hexa	Binario	Binario	Hexa
00000011	3	00000100	4	0	00000111	7
01000000	40	01000010	42	0	00000000	0
10000001	81	10000000	80	0	00000000	0
00000011	3	00000100	4	1	11111111	FF
01000000	40	01000010	42	1	11111110	FE
10000001	81	10000000	80	1	00000001	1

Conclusión

Los circuitos para sumar en los procesadores buscan ser eficientes en su tarea, a pesar de que hacen uso repetitivo de módulos buscando la simplificación del problema a resolver hacen uso de las propiedades de los número en su base binario para añadir muy pocos componentes para convertir el sumador en un restador que de haber hecho uso de un circuito distinto pudo haber sido perjudicial para el uso de recursos que supondría.