Rubén Comerón Galán

Sesión 3 - Síntesis e implementación del sumador completo

El sumador creado mediante puertas OR de 4 entradas y puertas AND de 3 entradas es:

```
ENTITY sumador IS
	PORT (x, y, c_in: IN BIT; s, c_out:OUT BIT);
END sumador;

ARCHITECTURE estructural OF sumador IS
	COMPONENT and3
		PORT (e1, e2, e3: IN BIT; sal: OUT BIT);
	END COMPONENT;
	COMPONENT or4
		PORT (e1, e2, e3, e4: IN BIT; sal: OUT BIT);
	END COMPONENT;
	COMPONENT not2
		PORT (e: IN BIT; sal: OUT BIT);
	END COMPONENT;
	SIGNAL not_x, not_y, not_c_in, s_and1, s_and2, s_and3, s_and4, c_out_and1, c_out_and2,
c_out_and3: BIT;
BEGIN
	notx: not2 PORT MAP (x, not_x);
	noty: not2 PORT MAP (y, not_y);
	notc_in: not2 PORT MAP (c_in, not_c_in);
	and1_sal: and3 PORT MAP (x, not_y, not_c_in, s_and1);
	and2_sal: and3 PORT MAP (not_x, y, not_c_in, s_and2);
	and3_sal: and3 PORT MAP (not_x, not_y, c_in, s_and3);
	and4_sal: and3 PORT MAP (x, y, c_in, s_and4);
	sal: or4 PORT MAP (s_and1, s_and2, s_and3, s_and4, s);
	and1_c_out: and3 PORT MAP (x, y, '1', c_out_and1);
	and2_c_out: and3 PORT MAP (x, c_in, '1', c_out_and2);
	and3_c_out: and3 PORT MAP (y, c_in, '1', c_out_and3);
	cout: or4 PORT MAP (c_out_and1, c_out_and2, c_out_and3, '0', c_out);
END estructural;
```

El sumador creado a partir de puertas OR y puertas AND de 2 entradas es:

```
ENTITY sumador IS
	PORT (x, y, c_in: IN BIT; s, c_out:OUT BIT);
END sumador;

ARCHITECTURE estructural OF sumador IS
	COMPONENT and2
```

```
        PORT (e1, e2: IN BIT; sal: OUT BIT);
    END COMPONENT;
    COMPONENT or2
        PORT (e1, e2: IN BIT; sal: OUT BIT);
    END COMPONENT;
    COMPONENT not2
        PORT (e: IN BIT; sal: OUT BIT);
    END COMPONENT;
    SIGNAL not_x, not_y, not_c_in, s_and1_1, s_and1_2, s_and2_1, s_and2_2, s_and3_1,
s_and3_2, s_and4_1, s_and4_2, c_out_and1, c_out_and2, c_out_and3, cout_aux, s1, s2: BIT;
BEGIN
    notx: not2 PORT MAP (x, not_x);
    noty: not2 PORT MAP (y, not_y);
    notc_in: not2 PORT MAP (c_in, not_c_in);
    and1_sal_1: and2 PORT MAP (x, not_y, s_and1_1);
    and1_sal_2: and2 PORT MAP (s_and1_1, not_c_in, s_and1_2);
    and2_sal_1: and2 PORT MAP (not_x, y, s_and2_1);
    and2_sal_2: and2 PORT MAP (s_and2_1, not_c_in, s_and2_2);
    and3_sal_1: and2 PORT MAP (not_x, not_y, s_and3_1);
    and3_sal_2: and2 PORT MAP (s_and3_1, c_in, s_and3_2);
    and4_sal_1: and2 PORT MAP (x, y, s_and4_1);
    and4_sal_2: and2 PORT MAP (s_and4_1, c_in, s_and4_2);
    sal1: or2 PORT MAP (s_and1_2, s_and2_2, s1);
    sal2: or2 PORT MAP (s_and3_2, s_and4_2, s2);
    sal: or2 PORT MAP (s1, s2, s);
    and1_c_out: and2 PORT MAP (x, y, c_out_and1);
    and2_c_out: and2 PORT MAP (x, c_in, c_out_and2);
    and3_c_out: and2 PORT MAP (y, c_in, c_out_and3);
    aux_cout: or2 PORT MAP (c_out_and1, c_out_and2, cout_aux);
    cout: or2 PORT MAP (cout_aux, c_out_and3, c_out);
END estructural;
```

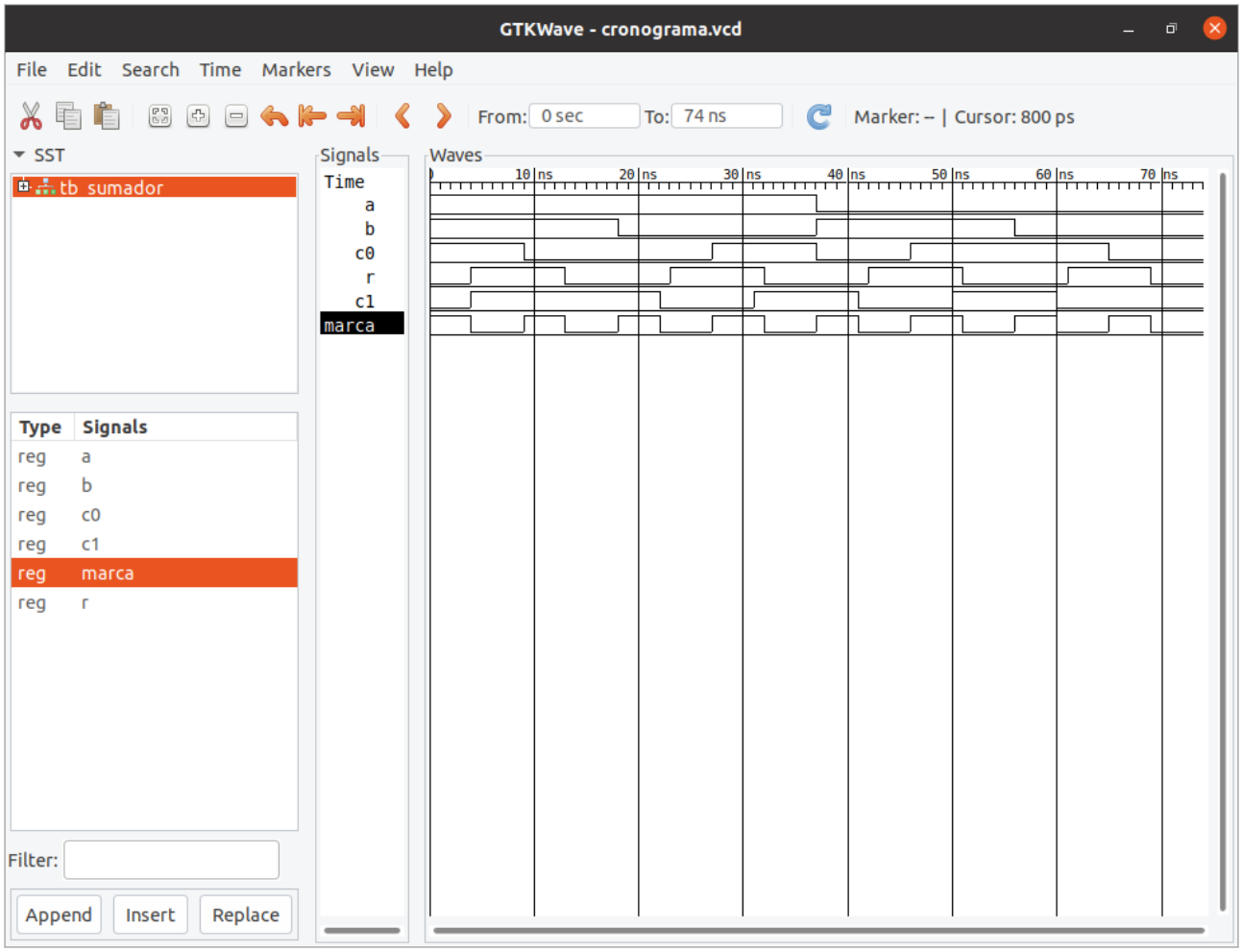
El test-bench utilizado para ambos sumadores es:

```
ENTITY tb_sumador IS
END tb_sumador;

ARCHITECTURE estructural OF tb_sumador IS

    COMPONENT sumador
        PORT (x, y, c_in: IN BIT; s, c_out: OUT BIT);
    END COMPONENT;


    FOR componente: sumador USE ENTITY WORK.sumador(estructural);
```

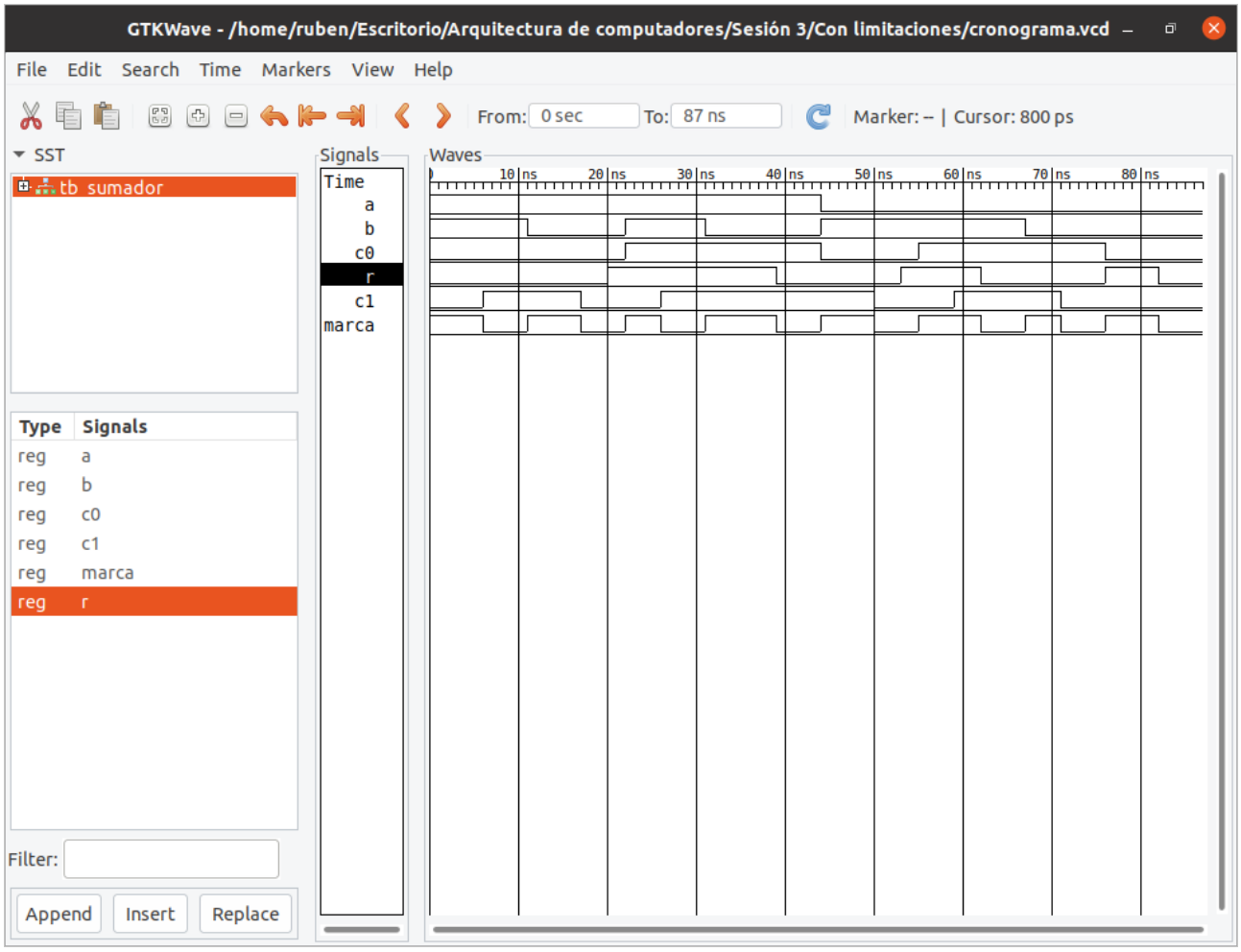
```
        SIGNAL a, b, c0, marca: BIT:='0';
        SIGNAL r, c1: BIT:='0';


BEGIN

        componente: sumador PORT MAP (a, b, c0, r, c1);


        PROCESS
        BEGIN
                a<= '1';
                b<= '1';
                c0<= '1';
                marca<= '1';
                WAIT UNTIL r<= '1' and c1<= '1';
                marca<= '0';
                WAIT FOR 5 ns;

                a<= '1';
                b<= '1';
                c0<= '0';
                marca<= '1';
                WAIT UNTIL r<= '0' and c1<= '1';
                marca<= '0';
                WAIT FOR 5 ns;

                a<= '1';
                b<= '0';
                c0<= '0';
                marca<= '1';
                WAIT UNTIL r<= '1' and c1<= '0';
                marca<= '0';
                WAIT FOR 5 ns;

                a<= '1';
                b<= '0';
                c0<= '1';
                marca<= '1';
                WAIT UNTIL r<= '0' and c1<= '1';
                marca<= '0';
                WAIT FOR 5 ns;

                a<= '0';
                b<= '1';
                c0<= '0';
                marca<= '1';
                WAIT UNTIL r<= '1' and c1<= '0';
                marca<= '0';
                WAIT FOR 5 ns;
```

```
		a<= '0';
		b<= '1';
		c0<= '1';
		marca<= '1';
		WAIT UNTIL r<= '0' and c1<= '1';
		marca<= '0';
		WAIT FOR 5 ns;

		a<= '0';
		b<= '0';
		c0<= '1';
		marca<= '1';
		WAIT UNTIL r<= '1' and c1<= '0';
		marca<= '0';
		WAIT FOR 5 ns;

		a<= '0';
		b<= '0';
		c0<= '0';
		marca<= '1';
		WAIT UNTIL r<= '0' and c1<= '0';
		marca<= '0';
		WAIT FOR 5 ns;

		WAIT;
	END PROCESS;
END estructural;
```

El cronograma del primer sumador usando este test-bench es:



El cronograma del segundo sumador usando este test-bench es:



En el segundo cronograma podemos observar una longitud del camino crítico mayor debido al uso de más puertas lógicas, lo que conlleva mayor retardo. También existe, en la señal r, un retardo mayor respecto a la señal c1 debido, también, al uso de más puertas lógicas. Por último señalizar el cambio inexplicable en la señal marca aún cuando r no cumple la condición para ello.