Rubén Comerón Galán

Sesión 6 – Sumador CSK

El código del bloque propagador es:

```
ENTITY Bloque_P IS
	GENERIC (c: INTEGER:=4);
	PORT(x_in, y_in: IN BIT_VECTOR(c-1 downto 0); P: OUT BIT);
END ENTITY;

ARCHITECTURE estructural OF Bloque_P IS

COMPONENT and2 IS
	GENERIC (retardo: TIME:= 1 ns);
	PORT (e1, e2: IN BIT; sal: OUT BIT);
END COMPONENT;

COMPONENT xor2 IS
	GENERIC (retardo: TIME:= 3 ns);
	PORT (e1, e2: IN BIT; sal: OUT BIT);
END COMPONENT;

SIGNAL xor_0, xor_1, xor_2, xor_3, and_1, and_2: BIT;

BEGIN

	bit_0: xor2 PORT MAP(x_in(0), y_in(0), xor_0);
	bit_1: xor2 PORT MAP(x_in(1), y_in(1), xor_1);
	bit_2: xor2 PORT MAP(x_in(2), y_in(2), xor_2);
	bit_3: xor2 PORT MAP(x_in(3), y_in(3), xor_3);
	primer_and: and2 PORT MAP(xor_0, xor_1, and_1);
	segundo_and: and2 PORT MAP(xor_2, xor_3, and_2);
	tercer_and: and2 PORT MAP(and_1, and_2, P);

END ARCHITECTURE;
```

El código del bloque CSK es:

```
ENTITY Bloque_CSK IS
	GENERIC (c: INTEGER:=4);
	PORT(x_in, y_in: IN BIT_VECTOR(c-1 downto 0); c_in: IN BIT; sal: OUT BIT_VECTOR(c-1 downto 0); c_out: OUT BIT);
END ENTITY;
```

```
ARCHITECTURE estructural OF Bloque_CSK IS

COMPONENT RCA IS
	GENERIC (c: INTEGER:=4);
	PORT(x_in, y_in: IN BIT_VECTOR(c-1 downto 0); c_in: IN BIT; sal: OUT BIT_VECTOR(c-1 downto 0); c_out: OUT BIT);
END COMPONENT;
COMPONENT Bloque_P IS
	GENERIC (c: INTEGER:=4);
	PORT(x_in, y_in: IN BIT_VECTOR(c-1 downto 0); P: OUT BIT);
END COMPONENT;
COMPONENT and2 IS
	GENERIC (retardo: TIME:= 1 ns);
	PORT (e1, e2: IN BIT; sal: OUT BIT);
END COMPONENT;
COMPONENT or2 IS
	GENERIC (retardo: TIME:= 1 ns);
	PORT (e1, e2: IN BIT; sal: OUT BIT);
END COMPONENT;
SIGNAL P, c_out_aux, and_aux: BIT;
BEGIN

	sumador: RCA PORT MAP(x_in, y_in, c_in, sal, c_out_aux);
	propagacion: Bloque_P PORT MAP(x_in, y_in, P);
	and_cout: and2 PORT MAP(P, c_in, and_aux);
	or_cout: or2 PORT MAP(and_aux, c_out_aux, c_out);

END estructural;
```

El código del sumador CSK es:

```
ENTITY CSK IS
	GENERIC (n: INTEGER:=16);
	PORT(x_in, y_in: IN BIT_VECTOR(n-1 downto 0); c_in: IN BIT; sal: OUT BIT_VECTOR(n-1 downto 0); c_out: OUT BIT);
END ENTITY;

ARCHITECTURE estructural OF CSK IS

COMPONENT Bloque_CSK
	GENERIC (c: INTEGER:=4);
	PORT(x_in, y_in: IN BIT_VECTOR(c-1 downto 0); c_in: IN BIT; sal: OUT BIT_VECTOR(c-1 downto 0); c_out: OUT BIT);
END COMPONENT;
COMPONENT RCA
	GENERIC (c: INTEGER:=4);
```

```
	PORT(x_in, y_in: IN BIT_VECTOR(c-1 downto 0); c_in: IN BIT; sal: OUT BIT_VECTOR(c-1 downto 0); c_out: OUT BIT);
END COMPONENT;

CONSTANT c: integer:=4;
SIGNAL c_aux: BIT_VECTOR ((n/4)-1 DOWNTO 0);

BEGIN

	primeros_bits: RCA PORT MAP(x_in(3 DOWNTO 0), y_in(3 DOWNTO 0), c_in, sal(3 DOWNTO 0), c_aux(0));
	G: FOR i IN 1 TO (n/4)-1 GENERATE
		siguientes: IF i < (n/4)-1 GENERATE
			intermedios: Bloque_CSK PORT MAP(x_in((i+1)*c-1 DOWNTO i*c), y_in((i+1)*c-1 DOWNTO i*c), c_aux(i-1), sal((i+1)*c-1 DOWNTO i*c), c_aux(i));
		END GENERATE siguientes;
		msb: IF i = c-1 GENERATE
			ultimos: Bloque_CSK PORT MAP(x_in((i+1)*c-1 DOWNTO i*c), y_in((i+1)*c-1 DOWNTO i*c), c_aux(i-1), sal((i+1)*c-1 DOWNTO i*c), c_out);
			END GENERATE msb;
	END GENERATE G;
END estructural;
```

El test-bench para el sumador CSK es:

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

	CONSTANT TAMANO_PALABRA: integer := 4;
	CONSTANT TAMANO_PALABRA_RCA: INTEGER := 2;

	SIGNAL a, b, r: BIT_VECTOR(TAMANO_PALABRA-1 downto 0);
	SIGNAL cin, cout, marca: BIT;

BEGIN

	componente: CSK GENERIC MAP (TAMANO_PALABRA) PORT MAP (A, B, Cin, R, Cout);

	PROCESS
	VARIABLE valor, limite, cuenta_carry, sol: INTEGER;
	BEGIN

		limite:= 2**TAMANO_PALABRA-1;

		FOR i IN 0 TO limite LOOP
			valor:=i;

			FOR k IN B'REVERSE_RANGE LOOP
				A(k)<=BIT'VAL(valor REM 2);--cambiamos A
```

```
                    valor:=valor/2;
                END LOOP;

                FOR j IN 0 TO limite LOOP
                    valor:=j;

                    FOR k IN B'REVERSE_RANGE LOOP
                        B(k)<=BIT'VAL(valor REM 2);--cambiamos B
                        valor:=valor/2;
                    END LOOP;
                    FOR carry IN 0 TO 1 LOOP
                        cin<=BIT'VAL(carry REM 2);
                        WAIT FOR 30 ns;
                    END LOOP;
                END LOOP;
            END LOOP;
        WAIT;
        END PROCESS;
END estructural;
```