```
ENTITY ffd IS
            PORT(
        clk: in bit;
        d: in bit;
        nclear: in bit;
        q: out bit;
        qn: out bit );
END ffd;

ARCHITECTURE comportamiento OF ffd IS

BEGIN
            PROCESS(clk)
            BEGIN
                        IF (nclear='0') THEN
                                    q <= '0';
                                    qn<='1';
                        ELSIF (rising_edge (clk)) THEN
                                    q <= d;
                                    qn <= not d;
                        END IF;
            END PROCESS;
END comportamiento;


ENTITY and3 IS
  PORT(e1, e2, e3 : IN bit; s : OUT bit);
END and3;
ARCHITECTURE comportamiento OF and3 IS

BEGIN
  s <= e1 AND e2 AND e3;
END comportamiento;


ENTITY and2 IS
  PORT(e1, e2 : IN bit; s : OUT bit);
END and2;
ARCHITECTURE comportamiento OF and2 IS

BEGIN
  s <= e1 AND e2;
END comportamiento;

ENTITY or2 IS
```

```
  PORT(e1, e2 : IN bit; s : OUT bit);
END or2;
ARCHITECTURE comportamiento OF or2 IS

BEGIN
  s <= e1 OR e2;
END comportamiento;


oooooooooo
ENTITY inv IS
  PORT(e : IN bit; s : OUT bit);
END inv;
ARCHITECTURE comportamiento OF inv IS

BEGIN
  s <= NOT e;
END comportamiento;
---------------------------------
---------------------------------
--- reconocedor del patron 010 ---
---------------------------------
---------------------------------
ENTITY reconocedor IS
  PORT(clk, nclear, x : IN bit; z : OUT bit);
END reconocedor;

ARCHITECTURE estructural OF reconocedor IS

--declaraciÃ³n de componentes, a completar por el alumn

COMPONENT ffd:
    PORT (clk, d, nclear : IN bit; q, qn OUT bit);
END ffd;

COMPONENT and3:
    PORT (e1, e2, e3 : IN bit; s : OUT bit);
END and3;

COMPONENT and2:
    PORT (e1, e2: IN bit; s : OUT bit);
END and2;

COMPONENT or2:
    PORT (e1, e2 : IN bit; s : OUT bit);
END or2;
```

```
COMPONENT inv:
   PORT (e : IN bit; s : OUT bit);
END inv;

--declaraciÃ³n de seÃ±ales . a completar por el alumno

SIGNAL sal_and2_1, sal_and3_1, sal_or2_1, sal_inv, sal_ffd1_1, sal_ffd1_2, sal_ffd2_1,
sal_fdd2_2: bit;


BEGIN
--descripciÃ³n estructural de la arquitectura
--utilizando las componentes, inv,or2,and2,and3 y ffd
i_inv : inv PORT MAP (x, sal_inv);
i_and1 : and2 PORT MAP (sal_ffd1_1, x, sal_and2_1);
i_and3 : and3 PORT MAP (sal_ffd1_1, sal_inv, sal_fdd2_2, sal_and3_1);
i_or1 : or2 PORT MAP (sal_and2_1, sal_and3_1, sal_or2_1);
i_ffd1 : ffd PORT MAP (clk, sal_or2_1, nclear, sal_ffd1_1, sal_ffd1_2);
i_fdd2 : ffd PORT MAP (clk, sal_inv, nclear, sal_ffd2_1, sal_fdd2_2);
i_and2 : and2 PORT MAP (sal_ffd1_1, sal_ffd2_1, z);
END estructural;
```