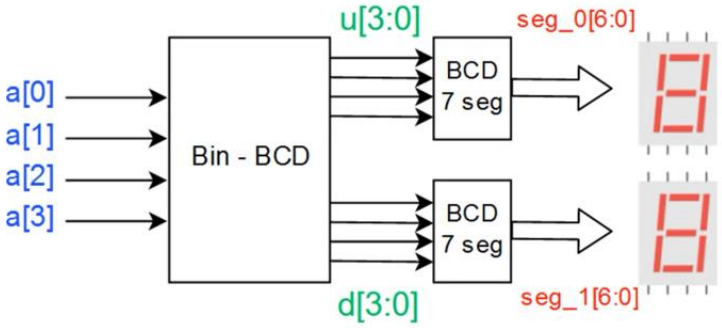
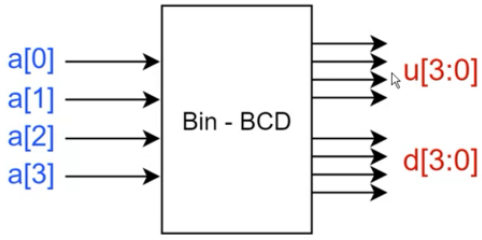
# CONVERTIDOR BINARIO A 7 SEGMENTOS

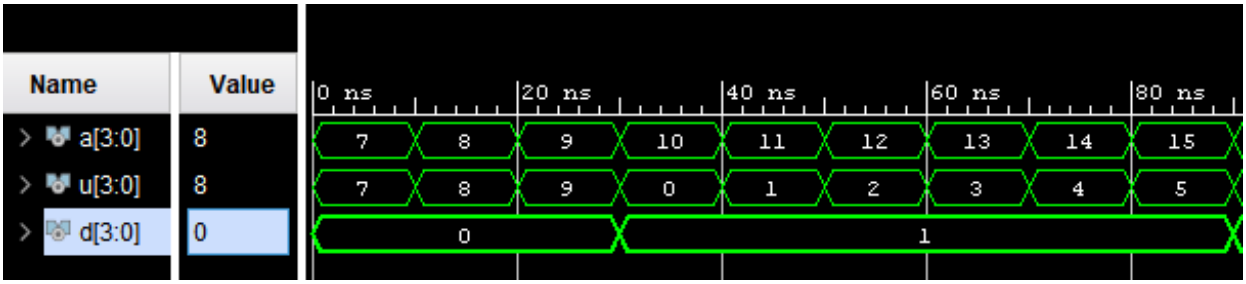
Implementar un convertidor de binario a 7 segmentos utilizando dos bloques interconectados, uno convertidor de binario a BCD y otro de BCD a 7 segmentos.




Primeramente, se hizo el convertidor binario a BCD donde se hizo uso de un ciclo for para realizar los desplazamientos, en este caso como solo recibimos 4 bits y en los primeros dos desplazamientos no tenemos un dato mayor o igual a 5, se puede proceder a realizar 3 desplazamientos al iniciar. Dicho lo anterior solamente se tiene que realizar un ciclo de comparación por lo cual se deja un for de 0 a 0. Se opta por dejarlo y no eliminarlo ya que posteriormente se puede parametrizar el mismo bloque.



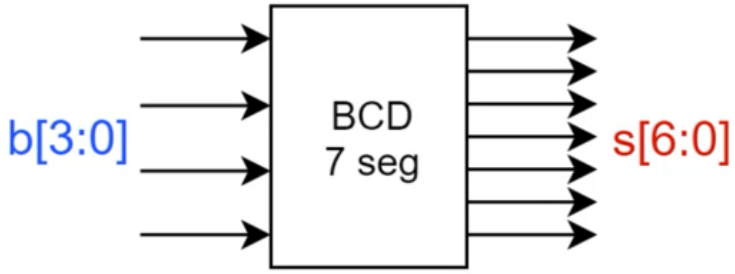

```
process(a)
    -- crear variables internas del proceso
    variable z: STD_LOGIC_VECTOR(11 downto 0);
    begin
        z := "000000000000"; -- := Asignar valor a variable
        z(6 downto 3) := a; -- primeros 3 corrimientos
        for i in 0 to 0 loop
            --unidades
            if z(7 downto 4) > "0100"  then
                z(7 downto 4) := z(7 downto 4) + "0011";
            end if;
            --decenas
            if z(11 downto 8) > "0100"  then
                z(11 downto 8) := z(11 downto 8) + "0011";
            end if;
            z(11 downto 1) := z(10 downto 0);
        end loop;
        u <= z(7 downto 4);
        d <= z(11 downto 8);
end process;
```




| 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|

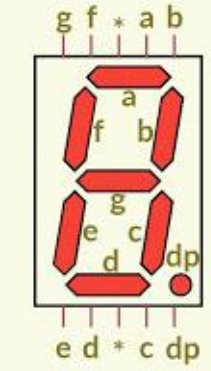
Posteriormente se realizó la implementación de convertidor de BCD a 7 segmentos, mediante la instrucción case para identificar el valor que se desea convertir. Una vez obtenidos los dos bloques se hizo un top level donde se encuentran los dos bloques.

.




```
case b is
    when "0000" => s <= "0111111";--0
    when "0001" => s <= "0000110";--1
    when "0010" => s <= "1011011";--2
    when "0011" => s <= "1001111";--3
    when "0100" => s <= "1100110";--4
    when "0101" => s <= "1101101";--5
    when "0110" => s <= "1111101";--6
    when "0111" => s <= "0000111";--7
    when "1000" => s <= "1111111";--8
    when "1001" => s <= "1101111";--9
    when others => s <= "0000000";--default
```

```
--Instance component
C_bin_bcd: bin_bcd
    PORT MAP(
        a => a,
        u => u,
        d => d
    );
C_tens: bcd_7seg
    PORT MAP(
        b => u,
        s => seg_0
    );
C_units: bcd_7seg
    PORT MAP(
        b => d,
        s => seg_1
    );
```




| Cátodo Común | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Común* | Número | g | f | e | d | c | b | a |
| GND | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| GND | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| GND | 2 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| GND | 3 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| GND | 4 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| GND | 5 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| GND | 6 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| GND | 7 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| GND | 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| GND | 9 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |