Solution: TP3

Part I:

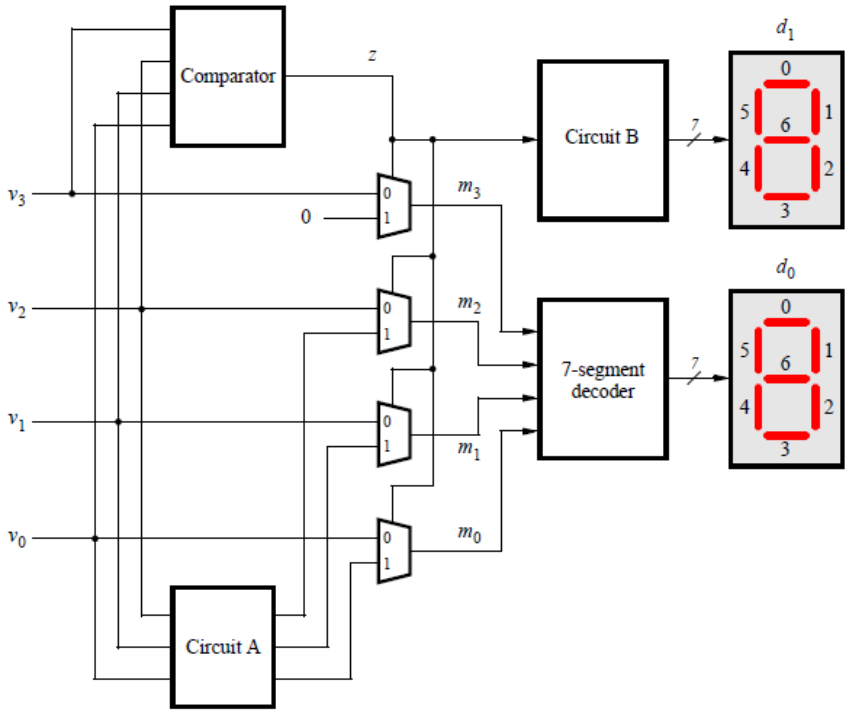
```
library ieee;
use ieee.std_logic_1164.all;
entity part1 is
port ( sw0 : in std_logic_vector(3 downto 0);
                    sw1 : in std_logic_vector(7 downto 4);
                    sw2 : in std_logic_vector(11 downto 8);
                    sw3 : in std_logic_vector(15 downto 12);
                    hex0 : out std_logic_vector(6 downto 0);
                    hex1 : out std_logic_vector(6 downto 0);
                    hex2 : out std_logic_vector(6 downto 0);
                    hex3 : out std_logic_vector(6 downto 0));
end part1;
architecture behaviore of part1 is
begin
hex0 <= "0000101" when sw0 <= "0000" else
                     "1011001" when sw0 <= "0001" else
      "0000111" when sw0 <= "0010" else
                     "0011010" when sw0 <= "0011"else
                     "0010101" when sw0 <= "0100" else
                     "1101001" when sw0 <= "0101" else
                     "1001101" when sw0 <= "0110" else
                     "0010101" when sw0 <= "0111" else
                     "1100101" when sw0 <= "1000" else
                     "1010110" when sw0 <= "1001" else
                     "1111111" ;

hex1 <= "0000101" when sw1 <= "0000" else
                     "1011001" when sw1 <= "0001" else
      "0000111" when sw1 <= "0010" else
                     "0011010" when sw1 <= "0011"else
                     "0010101" when sw1 <= "0100" else
                     "1101001" when sw1 <= "0101" else
                     "1001101" when sw1 <= "0110" else
                     "0010101" when sw1 <= "0111" else
                     "1100101" when sw1 <= "1000" else
                     "1010110" when sw1 <= "1001" else
                     "1111111" ;

hex2 <= "0000101" when sw2 <= "0000" else
                     "1011001" when sw2 <= "0001" else
      "0000111" when sw2 <= "0010" else
                     "0011010" when sw2 <= "0011"else
                     "0010101" when sw2 <= "0100" else
                     "1101001" when sw2 <= "0101" else
```

```
                    "1001101" when sw2 <= "0110" else
                    "0010101" when sw2 <= "0111" else
                    "1100101" when sw2 <= "1000" else
                    "1010110" when sw2 <= "1001" else
                    "1111111" ;

hex3 <= "0000101" when sw3 <= "0000" else
                    "1011001" when sw3 <= "0001" else
      "0000111" when sw3 <= "0010" else
                    "0011010" when sw3 <= "0011"else
                    "0010101" when sw3 <= "0100" else
                    "1101001" when sw3 <= "0101" else
                    "1001101" when sw3 <= "0110" else
                    "0010101" when sw3 <= "0111" else
                    "1100101" when sw3 <= "1000" else
                    "1010111" when sw3 <= "1001" else
                    "1111111" ;

end behaviore;
```

Part II:

Binary value	Decimal digits	
0000	0	0
0001	0	1
0010	0	2
...	...	...
1001	0	9
1010	1	0
1011	1	1
1100	1	2
1101	1	3
1110	1	4
1111	1	5

Table 1. Binary-to-decimal conversion values.



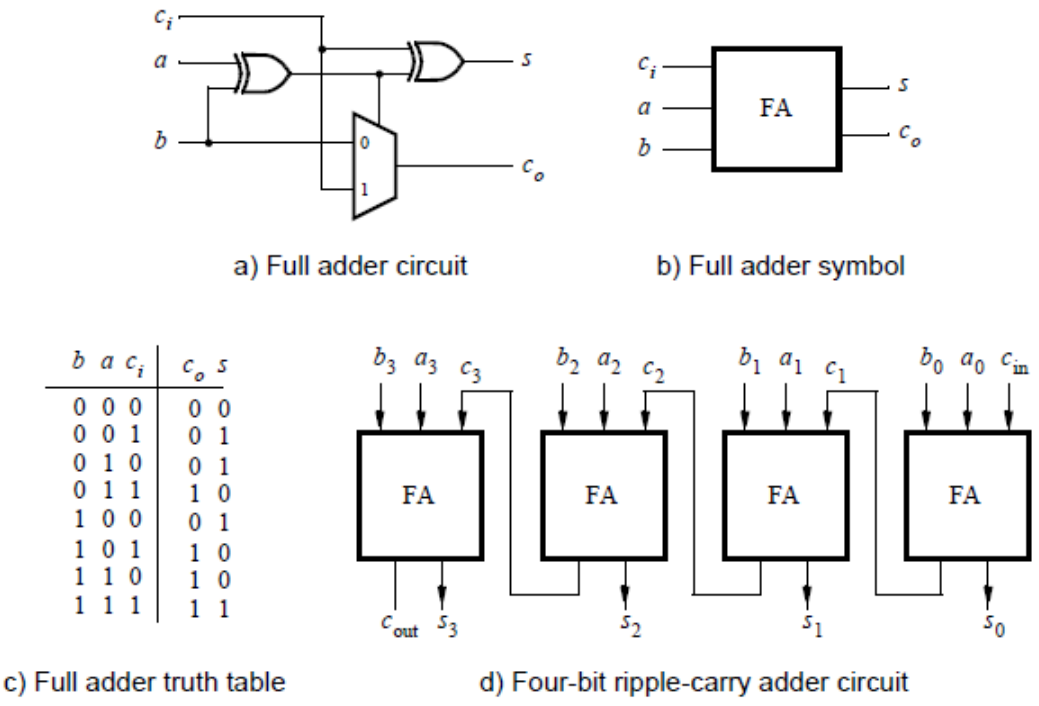

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
----------comparator----------------------
entity part2 is
PORT( c: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                t: OUT std_logic  );
END part2;
ARCHITECTURE BEHAVIOR OF part2 IS
BEGIN
t <=  '0' when c <= "0000" else
                '0' when c <= "0001" else
                '0' when c <= "0010" else
                '1' when c <= "0011" else
                '1' when c <= "0100" else
                '1' when c <= "0101" else
                '1' when c <= "0110" else
                '0' when c <= "0111" else
                '0' when c <= "1000" else
                '1' when c <= "1001" else
                '0' when c <= "1010" else
                '0' when c <= "1011" else
                '1' when c <= "1100" else
                '0' when c <= "1101" else
                '1' when c <= "1110" else
                '1';
END BEHAVIOR;
```

```
----------mux-------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity multiplexeur is
port ( n0,n1 : in std_LOGIC;
                        z : in std_logic;
                        l: out std_LOGIC );
end multiplexeur;
architecture behAVIOR of multiplexeur is
begin
l <= (not(z) and n0) or (z and n1);
end behAVIOR;
----------circuit a----------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity circuita is
port( k : in std_LOGIC_VECTOR(2 downto 0);
                      b : out std_LOGIC_VECTOR(2 downto 0));
end circuita;
architecture behAVIOR of circuita is
begIN
b <= "000" when k <= "001" else
    "001" when k <= "010" else
            "010" when k <= "011" else
            "011" when k <= "100" else
            "100" when k <= "101" else
            "101" when k <= "110" else
            "110" when k <= "111" else
            "111";
end behAVIOR;
----------circuit b----------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity circuitb is
port ( h : in std_logic;
                      s : out std_LOGIC_VECTOr(6 downto 0)    );
end circuitb;
arcHITECTURE behAVIOR of circuitb is
begin
s <= "1010101" when h <='0' else
    "1100101" when h <='1' else
    "1111111";
end behAVIOR;
```

```
------------------decoder-------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity decoder is
port ( g : in std_LOGIC_VECTOR( 3 downto 0);
      j : out std_LOGIC_VECTOR(6 downto 0));
end decoder;
arcHITECTURE behAVIOR of decoder is
begin
j <= "1010101" when g <="0000" else
    "1100110" when g <="0001" else
    "1001101" when g <="0010" else
    "1001000"      when g <="0011" else
              "0110011" when g <="0100" else
              "0000000" when g <="0101" else
              "0000101" when g <="0110" else
              "0010010" when g <="0111" else
              "1110011" when g <="1000" else
              "0001110" when g <="1001" else
              "1111111";
end behAVIOR;
----------------main----------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity lab2_2 is
port ( v : in std_logic_vector(3 downto 0);
                        d0 : out std_logic_vector(6 downto 0);
                        d1 : out std_LOGIC_VECTOR(6 downto 0));
end lab2_2;
architecture behAVIOR of lab2_2 is
component comparator is
PORT( c: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                       t: OUT std_logic
                       );
end component;
component multiplexeur is
port ( n0,n1 : in std_LOGIC;
                        z : in std_logic;
                        l: out std_LOGIC );
end component;
component circuita is
port( k : in std_LOGIC_VECTOR(2 downto 0);
                       b : out std_LOGIC_VECTOR(2 downto 0));
end component;
component circuitb is
port ( h : in std_logic;
                        s : out std_LOGIC_VECTOr(6 downto 0)    );
```

```
end component;
component decoder is
port ( g : in std_LOGIC_VECTOR( 3 downto 0);
      j : out std_LOGIC_VECTOR(6 downto 0));
end component;
signal u : std_logic;
signal m: std_LOGIC_VECTOr(3 downto 0);
signal a: std_LOGIC_VECTOR(2 downto 0);
begin
g1 : comparator port map(v,u);
g2: circuita port map(v(2 downto 0),a);
g3 : multiplexeur port map(v(0),a(0),m(0));
g4 : multiplexeur port map(v(1),a(1),m(1));
g5 : multiplexeur port map(v(2),a(2),m(2));
g6 : multiplexeur port map(v(3),'0',m(3));
g7 : circuitb port map(u,d1);
g8 : decoder port map(m,d0);
end behAVIOR;
```

Part III:



b	a	c_i	c_o	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 2. A ripple-carry adder circuit.

```
library ieee;
use ieee.std_logic_1164.all;
entity part3 is
port ( di,e0,e1: in std_logic;
                                f,l : out std_logic);
end part3;
architecture behavior of part3 is
begin
f <= (e0 xor e1) xor di;
l <= e0 or (di and (e0 xor e1));
end behavior;

----------main----------
library ieee;
use ieee.std_logic_1164.all;

entity lab2_3 is
port ( ci : in std_logic;
      a : in std_logic_vector(3 downto 0);
      b : in std_logic_vector(3 downto 0);
                  s : out std_logic_vector(3 downto 0);
                  c0: out std_logic );
end lab2_3;
architecture behavior of lab2_3 is
signal h : std_logic_vector(2 downto 0);
component add is
port (  di,e0,e1: in std_logic;
                                f,l : out std_logic);
end component;
begin
g0 : add port map(ci,a(0),b(0),s(0),h(0));
g1 : add port map(h(0),a(1),b(1),s(1),h(1));
g2 : add port map(h(1),a(2),b(2),s(2),h(2));
g3 : add port map(h(2),a(3),b(3),s(3),c0);
end behavior;
```