CH2_LED_1.vhd

```
--LED 霹靂燈 1:查表法
--EP3C16Q240C8 50MHz LEs:15,408 PINs:161 ,gckP31 ,rstP99

Library IEEE;                                --掛載零件庫
Use IEEE.std_logic_1164.all;                 --使用套件
Use IEEE.std_logic_unsigned.all;             --使用套件

entity CH2_LED_1 is
port(   gckp31,rstP99:in std_logic;     --系統時脈、系統重置
        LED16:buffer std_logic_vector(15 downto 0) --LED
        87,93,95,94,100,101,102,103
        106,107,108,110,111,112,113,114
      );
end entity CH2_LED_1; --或 end CH2_LED_1;  或 or end;

architecture Albert of CH2_LED_1 is
    signal FD:std_logic_vector(24 downto 0);   --除頻器
    type LED_T is array(0 to 127) of std_logic_vector(15 downto 0);
    --LED 樣板格式
    constant LED_Tdata:LED_T:= (    --LED 樣板資料
X"0000",X"8001",X"C003",X"E007",X"F00F",X"F81F",X"FC3F",X"FE7F",
X"FFFF",X"7FFE",X"3FFC",X"1FF8",X"0FF0",X"07E0",X"03C0",X"0180",
X"0000",X"0180",X"03C0",X"07E0",X"0FF0",X"1FF8",X"3FFC",X"7FFE",
X"FFFF",X"FE7F",X"FC3F",X"F81F",X"F00F",X"E007",X"C003",X"8001",
X"0000",X"0001",X"0005",X"0015",X"0055",X"0155",X"0555",X"1555",
X"5555",X"5557",X"555F",X"557F",X"55FF",X"57FF",X"5FFF",X"7FFF",
X"FFFF",X"BFFF",X"AFFF",X"ABFF",X"AAFF",X"AABF",X"AAAF",X"AAAB",
X"AAAA",X"2AAA",X"0AAA",X"02AA",X"00AA",X"002A",X"000A",X"0002",
X"0000",X"8001",X"C003",X"E007",X"F00F",X"F81F",X"FC3F",X"FE7F",
X"FFFF",X"FE7F",X"FC3F",X"F81F",X"F00F",X"E007",X"C003",X"8001",
X"0000",X"00FF",X"FF00",X"00FF",X"FF00",X"00FF",X"FF00",X"00FF",
X"FF00",X"0000",X"FFFF",X"0000",X"FFFF",X"0000",X"FFFF",X"0000",
X"FFFF",X"F00F",X"0FF0",X"F00F",X"0FF0",X"F00F",X"0FF0",X"F00F",
X"0FF0",X"0000",X"0001",X"0003",X"0007",X"000F",X"001F",X"003F",
X"007F",X"00FF",X"01FF",X"03FF",X"07FF",X"0FFF",X"1FFF",X"3FFF",
X"7FFF",X"FFFF",X"0FFF",X"00FF",X"000F",X"0000",X"FFFF",X"0000"
        );
    signal LED_T_p:integer range 0 to 127;          --LED_指標
    signal LED_case:std_logic_vector(1 downto 0);  --執行選項

    signal PWM_reset:std_logic;                     --PWM 開關
    type PWM_T is array(0 to 15) of integer range 0 to 31;
    --制定 PWM 格式
    signal LED_PWM:PWM_T;
    signal LED_PWM_data,not_LED:std_logic_vector(15 downto 0);
    --PWM,反相控制

    signal speeds:integer range 0 to 3;    --速度選擇
    signal speed:std_logic;                --執行 clk
```

```
begin

    --LED 輸出運算:   原表格資料        反相       PWM
    LED16<=(LED_Tdata(LED_T_p) xor not_LED) or LED_PWM_data;
    --速度選項
    speed<= FD(19) when speeds=0 else   --47.7Hz
            FD(20) when speeds=1 else   --23.8Hz
            FD(21) when speeds=2 else   --11.9Hz
            FD(22);                     --6Hz

--LED_P 主控器--
LED_P:process (speed,rstP99)
    variable N:integer range 0 to 511; --執行次數
    variable LED_T_ps,LED_T_pe:integer range 0 to 127;
    --起點,終點
    variable dir_LR:std_logic;          --PWM 轉動方向
begin
    if rstP99='0' then                  --系統重置
        N:=0;                           --行選項已結束
        LED_case<="00";                 --執行選項預設
        speeds<=0;                      --速度 0
    elsif rising_edge(speed) then
        if N=0 then                     --執行選項已結束
            LED_case<=LED_case+1;       --執行選項調整
            case LED_case is            --執行選項預設值
                when "00"=>
                    N:=1;               --執行選項
                    LED_T_p<=0;         --指標由 0 開始
                    LED_T_ps:=0;        --由 0 開始
                    LED_T_pe:=127;      --由 127 結束
                    not_LED<=(others=>'0');--不反相
                    PWM_reset<='0';     --PWM off
                when "01"=>
                    N:=1;               --執行選項
                    LED_T_p<=0;         --指標由 0 開始
                    LED_T_ps:=0;        --由 0 開始
                    LED_T_pe:=127;      --由 127 結束
                    not_LED<=(others=>'1');--反相
                    PWM_reset<='0';     --PWM off
                when "10"=>
                    N:=1;               --執行選項
                    LED_T_p<=127;       --指標由 127 開始
                    LED_T_ps:=127;      --由 127 開始
                    LED_T_pe:=0;        --由 0 結束
                    not_LED<=(others=>'0');--不反相
                    PWM_reset<='0';     --PWM off
                when "11"=>
                    N:=127;             --執行選項
                    LED_T_p<=0;         --指標由 0 開始
                    LED_T_ps:=0;        --由 0 開始
                    LED_T_pe:=0;        --由 0 結束
                    not_LED<=(others=>'0');--不反相
                    dir_LR:='0';        --PWM 轉動方向
```

```
        LED_PWM<=(0,0,1,2,3,4,5,7,10,12,15,17,20,24,28,31);
                    --PWM 預設值
                    PWM_reset<='1';     --PWM on
                    speeds<=speeds+1;   --速度調整
            end case;
        else    --執行選項
            if LED_T_ps=LED_T_pe then  --PWM 預設值移位
                if dir_LR='0' then
                    for i in 0 to 14 loop
                        LED_PWM(i)<=LED_PWM(i+1);
                    end loop;
                    LED_PWM(15)<=LED_PWM(0);
                else
                    for i in 0 to 14 loop
                        LED_PWM(i+1)<=LED_PWM(i);
                    end loop;
                    LED_PWM(0)<=LED_PWM(15);
                end if;
                if (N mod 16)=0 then
                    dir_LR:=not dir_LR; --調整 PWM 轉動方向
                end if;
                N:=N-1;                 --次數-1
            elsif LED_T_ps<LED_T_pe then
                LED_T_p<=LED_T_p+1;     --遞增
                if (LED_T_p+1)=LED_T_pe then
                    N:=0;               --結束
                end if;
            else
                LED_T_p<=LED_T_p-1;     --遞減
                if (LED_T_p-1)=LED_T_pe then
                    N:=0;               --結束
                end if;
            end if;
        end if;
    end if;
end process LED_P;

--PWM_P--
PWM_P:process(FD(0))
    variable PWMc:integer range 0 to 31;--PWM 計數器
begin
    if PWM_reset='0' then
        PWMc:=0;                        --PWM 計數器
        LED_PWM_data<=(others=>'0');    --all on
    elsif rising_edge(FD(0)) then
        for i in 0 to 15 loop
            if LED_PWM(i)>PWMc then
                LED_PWM_data(i)<='0';   --on
            else
                LED_PWM_data(i)<='1';   --off
            end if;
        end loop;
        PWMc:= PWMc+1;                  --PWM 計數器上數+1
```

```
    end if;
end process PWM_P;

--除頻器--
Freq_Div:process(gckP31)                    --系統頻率 gckP31:50MHz
begin
    if rstP99='0' then                      --系統重置
        FD<=(others=>'0');                  --除頻器:歸零
    elsif rising_edge(gckP31) then          --50MHz
        FD<=FD+1;       --除頻器:2 進制上數(+1)計數器
    end if;
end process Freq_Div;

end Albert;
```

CH2_LED_2.vhd

```
--LED 霹靂燈 2:強生計數器演算法
--EP3C16Q240C8 50MHz LEs:15,408 PINs:161 ,gckP31 ,rstP99

Library IEEE;                           --連結零件庫
Use IEEE.std_logic_1164.all;            --引用套件
Use IEEE.std_logic_unsigned.all;        --引用套件

entity CH2_LED_2 is
port(gckP31,rstP99:in std_logic;    -- 系統時脈、系統重置
     LEDs:buffer std_logic_vector(15 downto 0)      --LED
     -- 87,93,95,94,100,101,102,103
     -- 106,107,108,110,111,112,113,114
    );
end entity CH2_LED_2;

architecture Albert of CH2_LED_2 is
    signal FD:std_logic_vector(24 downto 0);--除頻器

begin

--LED_P 主控器--
LED_P:process (FD(16))
    variable N:integer range 0 to 127;          --執行次數
    variable LED_point:integer range 0 to 15;  --LED_指標
    variable dir_LR,set10,incDec:std_logic;
    --dir_LR:資料移動方向(0:右移、1:左移),資料移動方向與 LED 移動方向相反
    --set10:全設值,incDec:LED_指標_遞增遞減
begin
    if rstP99='0' then                      --系統重置
        N:=64;                              --次數由 64 開始
        LED_point:=0;                       --LED_指標由 0 開始
        dir_LR:='0';                        --資料移動方向:右移
```

```
        set10:='0';                             --全設 0
        incDec:='1';                            --遞增
        LEDs<=(others=>'0');                    --LED 全亮
    elsif rising_edge(FD(21)) then              --約 12Hz
        if N=0 then                             --次數已結束
            if LEDs/=(LEDs'range=>set10) then  --恢復原狀
                if dir_LR='0' then              --資料方向右移
                    LEDs<=set10 & LEDs(15 downto 1);
                else                            --資料方向左移
                    LEDs<=LEDs(14 downto 0) & set10;
                end if;
            else                                --重設參數
                N:=64;                          --次數由 64 開始
                if LED_point=0 and incDec='0' then
                    dir_LR:=not dir_LR;         --改變資料方向
                    incDec:='1';                --指標遞增
                    set10:=set10 xor dir_LR;--全設:0<-->1
                elsif  LED_point=15 and incDec='1' then
                    incDec:='0';                --指標遞減
                elsif incDec='1' then           --遞增
                    LED_point:=LED_point+1;--LED_指標遞增
                else                            --遞減
                    LED_point:=LED_point-1;--LED_指標遞減
                end if;
                LEDs<=(others=>set10);          --LED 全亮
            end if;
        else                                    --次數未結束
            if dir_LR='0' then                  --資料方向右移
                LEDs<=not LEDs(LED_point) & LEDs(15 downto 1);
            else                                --資料方向左移
                LEDs<=LEDs(14 downto 0) & not LEDs(LED_point);
            end if;
            N:=N-1;                             --次數-1
        end if;
    end if;
end process LED_P;

--除頻器--
Freq_Div:process(gckP31)                    --系統頻率 gckP31:50MHz
begin
    if rstP99='0' then                      --系統重置
        FD<=(others=>'0');                  --計數器歸零
    elsif rising_edge(gckP31) then          --50MHz
        FD<=FD+1;                           --以除 2 計數器為基礎的除頻器
    end if;
end process Freq_Div;

end Albert;
```