DM13A_Driver_RGB.vhd

```
--DM13A 驅動器
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 DM13A_Driver_RGB is
    port(--DM13A_Driver_RGB 操作頻率,重置
         DM13ACLK,DM13A_RESET:in std_logic;
         -- ALE 控制,OE 控制,方向控制,反相控制
         DM13ALE,DM13AOE,BIT_R_L,not01:in std_logic;
         sbit:in integer range 0 to 15;     --開始操作位元
         MaskRGB:in std_logic_vector(5 downto 0);
         --罩蓋操作位元
         --mask (5):0:disable 1:enable
         --(4..3)00:load,01:xor:10:or,11:and RGB

         LED_R,LED_G,LED_B:in std_logic_vector(15 downto 0);
         --R G B 圖形位元
         DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go,DM13ASDI_Bo:out std_logic;
         DM13ALEo,DM13AOEo:out std_logic;--DM13A 硬體操作位元
         DM13A_Sendok:out std_logic);--DM13A_Driver_RGB 完成操作位元
end DM13A_Driver_RGB;

architecture Albert of DM13A_Driver_RGB is
    signal DM13A_CLK:std_logic;         --DM13A CLK 內部操作位元
    signal i:integer range 0 to 31;         --輸出位元數控制
    constant databitN:integer range 0 to 31:=16;
    --輸出位元數參數:16 bit
    signal sbits:integer range 0 to 15;--內部開始操作位元控制
    signal R,G,B:std_logic;             --內部圖形位元取出

begin

--R,G,B 圖形位元取出罩蓋運算
R<= LED_R(sbits)when MaskRGB(5)='0' else   --nop
    MaskRGB(2)  when BIT_R_L='1' and sbits>sbit
                    and MaskRGB(4 downto 3)="00" else--load
    MaskRGB(2)  when BIT_R_L='0' and sbits<sbit
                    and MaskRGB(4 downto 3)="00" else--load
    MaskRGB(2) xor LED_R(sbits)when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="01" else--xor
    MaskRGB(2) xor LED_R(sbits)when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="01" else--xor
    MaskRGB(2) or  LED_R(sbits)when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="10" else--or
    MaskRGB(2) or  LED_R(sbits)when BIT_R_L='0' and
                    sbits<sbit and
```

```
                    MaskRGB(4 downto 3)="10" else--or
    MaskRGB(2) and LED_R(sbits)when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="11" else--and
    MaskRGB(2) and LED_R(sbits)when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="11" else--and
    LED_R(sbits);

G<= LED_G(sbits)when MaskRGB(5)='0' else   --nop
    MaskRGB(1)  when BIT_R_L='1' and sbits>sbit
                    and MaskRGB(4 downto 3)="00" else--load
    MaskRGB(1)  when BIT_R_L='0' and sbits<sbit
                    and MaskRGB(4 downto 3)="00" else--load
    MaskRGB(1) xor LED_G(sbits) when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="01" else  --xor
    MaskRGB(1) xor LED_G(sbits) when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="01" else  --xor
    MaskRGB(1) or  LED_G(sbits)  when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="10" else  --or
    MaskRGB(1) or  LED_G(sbits)  when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="10" else  --or
    MaskRGB(1) and LED_G(sbits) when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="11" else  --and
    MaskRGB(1) and LED_G(sbits) when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="11" else  --and
    LED_G(sbits);

B<= LED_B(sbits)when MaskRGB(5)='0' else   --nop
    MaskRGB(0)  when BIT_R_L='1' and sbits>sbit
                    and MaskRGB(4 downto 3)="00" else--load
    MaskRGB(0)  when BIT_R_L='0' and sbits<sbit
                    and MaskRGB(4 downto 3)="00" else--load
    MaskRGB(0) xor LED_B(sbits) when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="01" else  --xor
    MaskRGB(0) xor LED_B(sbits) when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="01" else  --xor
    MaskRGB(0) or  LED_B(sbits)  when BIT_R_L='1' and
                    sbits>sbit and
                    MaskRGB(4 downto 3)="10" else  --or
    MaskRGB(0) or  LED_B(sbits)  when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="10" else  --or
    MaskRGB(0) and LED_B(sbits) when BIT_R_L='1' and
                    sbits>sbit and
```

```
                    MaskRGB(4 downto 3)="11" else  --and
    MaskRGB(0) and LED_B(sbits) when BIT_R_L='0' and
                    sbits<sbit and
                    MaskRGB(4 downto 3)="11" else  --and
    LED_B(sbits);

--DM13A 硬體操作位元輸出
DM13ASDI_Ro<=R xor not01;   --R SDI 輸出運算(反相控制)
DM13ASDI_Go<=G xor not01;   --G SDI 輸出運算(反相控制)
DM13ASDI_Bo<=B xor not01;   --B SDI 輸出運算(反相控制)
DM13ACLKo<=DM13A_CLK;       --CLK
DM13ALEo<=DM13ALE;          --ALE
DM13AOEo<=DM13AOE;          --OE

DM13A_Send:process(DM13ACLK,DM13A_RESET)
begin
    if DM13A_RESET='0' then     --重置
        i<=0;                   --輸出位元數個數預設 0
        sbits<=sbit;            --載入開始操作位元
        DM13A_CLK<='0';         --預設 Low
        DM13A_Sendok<='0';      --預設未完成
    elsif rising_edge(DM13ACLK) then
        if i=databitN then      --判斷輸出位元數是否完成
            DM13A_Sendok<='1';  --完成
        else
            if DM13A_CLK='0' then
                DM13A_CLK<='1'; --啟動載入 CLK
            else
                i<=i+1;         --輸出位元數完成 1 個
                DM13A_CLK<='0'; --預備載入 CLK
                if BIT_R_L='1' then --取樣方向
                    sbits<=sbits-1; --向低位元
                else
                    sbits<=sbits+1; --向高位元
                end if;
            end if;
        end if;
    end if;
end process DM13A_Send;

end Albert;
```

CH4_RGB16x16_1.vhd

```
--RGB16x16 廣告看板
--106.12.30 版
--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 CH4_RGB16x16_1 is
port(gckp31,rstP99:in std_logic;    --系統頻率,系統 reset
     --DM13A 輸出
     DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go,DM13ASDI_Bo:out std_logic;
     DM13ALEo,DM13AOEo:out std_logic;
     --186,187,189,194,188,185
     --Scan 輸出
     Scan_DCBAo:buffer std_logic_vector(3 downto 0)
     --198,197,196,195
    );
end entity CH4_RGB16x16_1;

architecture Albert of CH4_RGB16x16_1 is
    component DM13A_Driver_RGB is
    port(--DM13A_Driver_RGB 操作頻率,重置,ALE 控制,
         DM13ACLK,DM13A_RESET,DM13ALE:in std_logic;
         -- OE 控制,方向控制,反相控制
         DM13AOE,BIT_R_L,not01:in std_logic;
         sbit:in integer range 0 to 15;         --開始操作位元
         maskRGB:in std_logic_vector(5 downto 0);--罩蓋操作位元
         --mask (5):0:disable 1:enable,
         -- (4..3)00:load,01:xor:10:or,11:and RGB
         LED_R,LED_G,LED_B:in std_logic_vector(15 downto 0);
         --R G B 圖形位元
         DM13ACLKo:out std_logic;
         DM13ASDI_Ro,DM13ASDI_Go,DM13ASDI_Bo:out std_logic;
         DM13ALEo,DM13AOEo:out std_logic;--DM13A 硬體操作位元
         DM13A_Sendok:out std_logic);
        --DM13A_Driver_RGB 完成操作位元
    end component;
            --DM13A_Driver_RGB 操作頻率,重置
    signal DM13ACLK,DM13A_RESET:std_logic;
            --ALE 控制,OE 控制,方向控制,反相控制
    signal DM13ALE,DM13AOE,BIT_R_L,not01:std_logic;
    signal sbit:integer range 0 to 15; --開始操作位元
    signal maskRGB:std_logic_vector(5 downto 0):="000000";
            --罩蓋操作位元
    signal LED_R,LED_G,LED_B:std_logic_vector(15 downto 0);
            --R G B 圖形位元
    signal DM13A_Sendok:std_logic;--DM13A_Driver_RGB 完成操作位元

    signal FD:std_logic_vector(24 downto 0);   --系統除頻器
    signal cn:std_logic_vector(10 downto 0);   --變換計時
    signal T:integer range 0 to 2047;          --顯示計時
    signal color:integer range 0 to 15;            --圖形取樣指標
    signal S:std_logic_vector(4 downto 0);     --操作模式
    signal LED_R1,LED_G1,LED_B1:std_logic_vector(15 downto 0);
            --R,G,B 樣板圖形
    signal LED_R2,LED_G2,LED_B2:std_logic_vector(15 downto 0);
            --R,G,B 捲動圖形
```

```
begin

----DM13A_Driver_RGB
 DM13ACLK<=FD(2);
 U1: DM13A_Driver_RGB
    port map(   DM13ACLK,DM13A_RESET,DM13ALE,DM13AOE,BIT_R_L,
                not01,sbit,maskRGB,LED_R,LED_G,LED_B,
                DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go,DM13ASDI_Bo,
                DM13ALEo,DM13AOEo,DM13A_Sendok);

--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;

BIT_R_L<=S(1);  --方向變換
not01<=S(2);    --反相變換

main:process(FD(0),rstP99) --主控器
begin
if rstP99='0' then
    Scan_DCBAo<="0000"; --掃瞄預設
    DM13A_RESET<='0';   --重置 DM13A_Driver_RGB
    DM13ALE<='0';       --無更新資料預設
    DM13AOE<='1';       --DM13A off
    cn<=(others=>'0');  --計時計次預設
    S<=(others=>'0');   --操作模式預設
    color<=0;           --圖形取樣重新開始
    sbit<=15;           --從 15 位元開始
elsif rising_edge(FD(0)) then           --操作頻率 25MHz
    if DM13ALE='0' and DM13AOE='1' then--無更新資料且顯示已關閉
        if DM13A_RESET='0' then         --尚未啟動 DM13A_Driver_RGB
            DM13A_RESET<='1';           --啟動 DM13A_Driver_RGB
            Scan_DCBAo<=Scan_DCBAo-1;   --調整掃瞄
        elsif DM13A_Sendok='1' then     --傳送完成
            DM13A_RESET<='0';           --重置 DM13A_Driver_RGB
            DM13ALE<='1';               --更新顯示資料
            LED_R2<=LED_R2(14 downto 0)&LED_R2(15);--R 圖形捲動 1 位元
            LED_G2<=LED_G2(14 downto 0)&LED_G2(15);--G 圖形捲動 1 位元
            LED_B2<=LED_B2(14 downto 0)&LED_B2(15);--B 圖形捲動 1 位元
            if Scan_DCBAo=0 then        --已完成一個畫面
                cn<=cn+1;               --計時
                if cn=1800 then         --計時到
                    cn<=(others=>'0');  --計時歸零
                    color<=color+1;     --圖形取樣遞增
                    if color=13 then    --最後一個了
                        S<=S+1;         --改變操作模式
```

```
                        color<=0;       --圖形取樣重新開始
                    end if;
                    LED_R2<=LED_R1;     --載入 R 預設圖形
                    LED_G2<=LED_G1;     --載入 G 預設圖形
                    LED_B2<=LED_B1;     --載入 B 預設圖形
                elsif cn(7 downto 0)=0 and S(3)='1' then
                    --計時到 & 允許圖形捲動
                    LED_R2<=LED_R2(13 downto 0)&LED_R2(15 downto 14);
                    --R 圖形捲動 2 位元
                    LED_G2<=LED_G2(13 downto 0)&LED_G2(15 downto 14);
                    --G 圖形捲動 2 位元
                    LED_B2<=LED_B2(13 downto 0)&LED_B2(15 downto 14);
                    --B 圖形捲動 2 位元
                end if;
                if S(4)='1' then            --允許控制開始位元
                    if cn(7 downto 0)=0 then--計時到
                        sbit<=sbit+1;       --往高位元移動開始
                    end if;
                else
                    sbit<=15;               --固定從 15 位元開始
                end if;
            end if;
        end if;
        T<=0;               --顯示計時歸零
    else
        DM13ALE<='0';       --顯示資料不更新
        DM13AOE<='0';       --顯示
        T<=T+1;             --顯示計時
        if T=100 then       --顯示計時到
            DM13AOE<='1';   --不顯示
        end if;
    end if;
end if;
end process;

LED_R<=LED_R1 when S(0)='0' else LED_R2;   --R 圖案選擇
LED_G<=LED_G1 when S(0)='0' else LED_G2;   --G 圖案選擇
LED_B<=LED_B1 when S(0)='0' else LED_B2;   --B 圖案選擇

--圖形樣板取樣
with color select
LED_R1<="0000000000000000" when 0, --暗
        "1111111111111111" when 1, --R
        "0000000000000000" when 2, --G
        "0000000000000000" when 3, --B
        "1111111111111111" when 4, --RG
        "1111111111111111" when 5, --RB
        "0000000000000000" when 6, --GB
        "1111111111111111" when 7, --RGB
        "0011000000110000" when 8, --00RRGGBB00RRGGBB
        "0000111100001111" when 9, --R0000111100001111
        "0101010101010101" when 10,--G0101010101010101
        "0101010101010101" when 11,--B0101010101010101
```

```
        "0011000011110011" when 12,--0011000011110011
        "1010101010101010" when others;

with color select
LED_G1<="0000000000000000" when 0, --暗
        "0000000000000000" when 1, --R
        "1111111111111111" when 2, --G
        "0000000000000000" when 3, --B
        "1111111111111111" when 4, --RG
        "0000000000000000" when 5, --RB
        "1111111111111111" when 6, --GB
        "1111111111111111" when 7, --RGB
        "0000110000001100" when 8, --00RRGGBB00RRGGBB
        "0101010101010101" when 9, --R0101010101010101
        "0000111100001111" when 10,--G0000111100001111
        "0011001100110011" when 11,--B0011001100110011
        "0000110011001111" when 12,--0000110011001111
        "0101010101010101" when others;

with color select
LED_B1<="0000000000000000" when 0, --暗
        "0000000000000000" when 1, --R
        "0000000000000000" when 2, --G
        "1111111111111111" when 3, --B
        "0000000000000000" when 4, --RG
        "1111111111111111" when 5, --RB
        "1111111111111111" when 6, --GB
        "1111111111111111" when 7, --RGB
        "0000001100000011" when 8, --00RRGGBB00RRGGBB
        "0011001100110011" when 9, --R0011001100110011
        "0011001100110011" when 10,--G0011001100110011
        "0000111100001111" when 11,--B0000111100001111
        "0000001100111111" when 12,--0000001100111111
        "1100110011001100" when others;

end Albert;
```

CH4_RGB16x16_2.vhd

```
--RGB16x16 人行道小綠人
--106.12.30 版
--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 CH4_RGB16x16_2 is
port(gckp31,rstP99:in std_logic;    --系統頻率,系統 reset
```

```
     --DM13A 輸出
     DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go:out std_logic;
     --186,187,189
     DM13ASDI_Bo,DM13ALEo,DM13AOEo:out std_logic;
     --194,188,185
     --Scan 輸出
     Scan_DCBAo:buffer std_logic_vector(3 downto 0)
     --198,197,196,195
    );
end entity CH4_RGB16x16_2;

architecture Albert of CH4_RGB16x16_2 is
    component DM13A_Driver_RGB is
    port(--DM13A_Driver_RGB 操作頻率,重置,ALE 控制
         DM13ACLK,DM13A_RESET,DM13ALE:in std_logic;
         --OE 控制,方向控制,反相控制
         DM13AOE,BIT_R_L,not01:in std_logic;
         startbit:in integer range 0 to 15; --開始操作位元
         maskRGB:in std_logic_vector(5 downto 0);
         --罩蓋操作位元
         --(5):0:disable 1:enable,
         --(4..3)00:load,01:xor:10:or,11:and RGB
         LED_R,LED_G,LED_B:in std_logic_vector(15 downto 0);
         --R G B 圖形位元
         DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go:out std_logic;
         DM13ASDI_Bo,DM13ALEo,DM13AOEo:out std_logic;
         --DM13A 硬體操作位元
         DM13A_Sendok:out std_logic);
         --DM13A_Driver_RGB 完成操作位元
    end component;
         --DM13A_Driver_RGB 操作頻率,重置,ALE 控制
    signal DM13ACLK,DM13A_RESET,DM13ALE:std_logic;
         -- OE 控制,方向控制,反相控制
    signal DM13AOE,BIT_R_L,not01:std_logic;
    signal startbit:integer range 0 to 15; --開始操作位元
    signal maskRGB:std_logic_vector(5 downto 0):="000000";
         --罩蓋操作位元
    signal LED_R,LED_G,LED_B:std_logic_vector(15 downto 0);
         --R G B 圖形位元
    signal DM13A_Sendok:std_logic;
         --DM13A_Driver_RGB 完成操作位元

    signal FD:std_logic_vector(24 downto 0);--系統除頻器
    signal G_step:integer range 0 to 3;         --圖形取樣指標
    signal RGB_point:integer range 0 to 15;--圖形取樣指標
    signal RG,RGB16X16_SCAN_reset,scan_1T,:std_logic;
    signal RGB16X16_TP_clk,RGB16X16_P_clk:std_logic;
    signal RGB16X16_SCAN_p_clk:std_logic;
    signal T_runstep:integer range 0 to 7; --執行階段
    type RGB16x16_T1 is array(0 to 15) of std_logic_vector(15 downto 0);
    --自定資料型態
    signal RGB16x16_R,RGB16x16_G:RGB16x16_T1;          --1 維陣列
    --圖像
```

```
    type RGB16x16_T2 is array(0 to 3) of RGB16x16_T1;  --2 維陣列
    --圖像:請參考小綠人編碼.doc
    constant RGB16x16_GD:RGB16x16_T2:=(
        (X"0000",X"0000",X"0000",X"0000",X"0003",X"060F",
         X"6F3F",X"DFF9",X"DFF0",X"DFFF",X"DF3E",X"6606",
         X"4004",X"0000",X"0000",X"0000"),          --小綠人 1
        (X"0000",X"0000",X"0000",X"0180",X"03E3",X"676F",
         X"D63F",X"DFF9",X"DFF8",X"DFBD",X"69CF",
         X"40C7",X"40C1",X"0080",X"0000",X"0000"), --小綠人 2
        (X"0004",X"018C",X"03CE",X"07E6",X"0E47",X"0C0E",
         X"6C3C",X"DFF8",X"DFF8",X"DFB9",X"D39B",
         X"61CF",X"40ED",X"41C1",X"0180",X"0080"), --小綠人 3
        (X"0000",X"0000",X"0000",X"0180",X"03E3",X"676F",
         X"D63F",X"DFF9",X"DFF8",X"DFBD",X"69CF",
         X"40C7",X"40C1",X"0080",X"0000",X"0000") );--小綠人 4

begin

----DM13A_Driver_RGB
 DM13ACLK<=FD(2);
 U1: DM13A_Driver_RGB port map(
        DM13ACLK,DM13A_RESET,DM13ALE,DM13AOE,BIT_R_L,
        not01,startbit,maskRGB,LED_R,LED_G,LED_B,
        DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go,DM13ASDI_Bo,
        DM13ALEo,DM13AOEo,DM13A_Sendok);

--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;

RGB16X16_TP_clk<=FD(22);              --約 6Hz ,0.167s

--時間配置管理器
RGB16X16_TP:process(RGB16X16_TP_clk,rstP99)
variable TT:integer range 0 to 511;     --階段計時器
variable T_step:integer range 0 to 7;  --階段
begin
if rstP99='0' then
    RG<='1';                        --選圖來源 1
    TT:=40;                         --階段時間設定
    T_runstep<=0;                   --R 階段預設 0
    T_step:=0;                      --R 階段
elsif rising_edge(RGB16X16_TP_clk) then
     TT:=TT-1;                      --階段時間倒數
     if TT=0 then                   --階段時間到
        if T_step=6 then            --已完成最後階段
            T_step:=0;              --階段重新開始
```

```
        else
            T_step:=T_step+1;   --下一階段
        end if;
        T_runstep<=T_step;      --交付執行階段
        case T_step is          --階段參數設定
            when 0=>            --R
                TT:=40;         --階段時間設定
            when 1=>            --R->G,G 靜置
                TT:=25;         --階段時間設定
            when 2=>            --gGgG:正常步行
                RG<='0';        --選圖來源 0
                TT:=120;        --階段時間設定
            when 3=>            --gGgG:快步行
                TT:=30;         --階段時間設定
            when 4=>            --gGgG:急步行
                TT:=30;         --階段時間設定
            when 5=>            --G 靜置
                RG<='1';        --選圖來源 1
                TT:=15;         --階段時間設定
            when others=>       --6:G->R,R 靜置
                TT:=15;         --階段時間設定
        end case;
    end if;
end if;
end process RGB16X16_TP;

--RGB16X16_P 執行速度變換---------------------------
RGB16X16_P_clk<= FD(7) when T_runstep=4 else  --急步行速度(195.3kHz)
                 FD(8) when T_runstep=3 else  --快步行速度(97.7kHz)
                 FD(9);                       --正常步行速度(48.8kHz)

RGB16X16_P:process(RGB16X16_P_clk,rstP99)
variable frames:integer range 0 to 31;     --停留時間控制
variable i:integer range 0 to 31;          --紅轉綠次數
begin
if rstP99='0' then
    RGB16X16_SCAN_reset<='0';--掃瞄 off
    --靜止圖形預設:請參考小紅人編碼.doc
    RGB16x16_R<=(X"0000",X"0000",X"0001",X"07C1",  --靜態小紅人
                 X"0FF3",X"6FEF",X"F81F",X"DAB8",X"DAB8",
                 X"F81F",X"6FEF",X"0FF3",X"07C1",X"0001",
                 X"0000",X"0000");
    RGB16x16_G<=((others=>'0'),(others=>'0'),(others=>'0'), --清空
                 (others=>'0'),(others=>'0'),(others=>'0'),
                 (others=>'0'),(others=>'0'),(others=>'0'),
                 (others=>'0'),(others=>'0'),(others=>'0'),
                 (others=>'0'),(others=>'0'),(others=>'0'),
                 (others=>'0'));
elsif rising_edge(RGB16X16_P_clk) then
    RGB16X16_SCAN_reset<='1';   --啟動掃瞄
    case T_runstep is           --階段執行
        when 0=>                --紅靜置
            i:=16;              --紅轉綠次數預設
```

```
            frames:=3;          --停留時間預設
        when 1=>                --紅轉綠
            if i=0 then         --綠靜置
                G_step<=0;      --步行圖 0
                frames:=10;     --停留時間預設
            elsif scan_1T='1' then --RGB16X16_SCAN_p 回信號
                frames:=frames-1;   --停留時間次數遞減
                if frames=0 then    --frame 停留時間到
                    RGB16x16_R(i-1)<=RGB16x16_G(i-1);--左至右轉換
                    RGB16x16_G(i-1)<=RGB16x16_R(i-1);
                    frames:=3;          --停留時間預設
                    i:=i-1;             --紅轉綠次數遞減
                end if;
            end if;
        when 2|3|4=>                    --動態
            if scan_1T='1' then         --RGB16X16_SCAN_p 回信號
                frames:=frames-1;       --掃瞄 frame 次數遞減
                if frames=0 then        --frame 停留時間到
                    G_step<=G_step+1;   --調整步行圖 0123
                    if T_runstep=2 then     --正常步行
                        frames:=10;         --停留時間預設
                    elsif T_runstep=3 then  --快步行
                        frames:=10;         --停留時間預設
                    else                    --急步行
                        frames:=10;         --停留時間預設
                    end if;
                end if;
            end if;
        when 5=>            --綠靜置
            frames:=3;      --停留時間預設
        when others=>       --綠轉紅
            if i=16 then    --紅靜置
                null;
            elsif scan_1T='1' then --RGB16X16_SCAN_p 回信號
                frames:=frames-1;   --停留時間次數遞減
                if frames=0 then    --frame 停留時間到
                    RGB16x16_R(i+1)<=RGB16x16_G(i+1);--右至左轉換
                    RGB16x16_G(i+1)<=RGB16x16_R(i+1);
                    frames:=3;      --停留時間預設
                    i:=i+1;         --綠轉紅次數遞增
                end if;
            end if;
    end case;
end if;
end process;

BIT_R_L<='0';       --方向變換
not01<='0';         --反相變換
startbit<=0;        --從 15 位元開始
maskRGB<="000000";  --直接輸出

RGB_point<=conv_integer(Scan_DCBAo);    --轉換圖形取樣指標
LED_G<= RGB16x16_G(RGB_point) when RG='1'
```

```
        else RGB16x16_GD(G_step)(RGB_point);--G 圖案選擇取圖
LED_R<= RGB16x16_R(RGB_point) when RG='1'
        else (others=>'0');     --R 圖案選擇取圖
LED_B<=(others=>'0');           --B 圖案選擇取圖

--RGB16X16_SCAN_p 執行速度變換-----------
RGB16X16_SCAN_p_clk<=FD(6) when T_runstep=4 else--急步行速度
                     FD(7) when T_runstep=3 else--快步行速度
                     FD(8);                     --正常步行速度

RGB16X16_SCAN_p:process(RGB16X16_SCAN_p_clk,RGB16X16_SCAN_reset)
variable frame:integer range 0 to 15;  --15~0:1 frame
variable T:integer range 0 to 255;     --每一掃瞄停留時間計時器
begin
if RGB16X16_SCAN_reset='0' then
    Scan_DCBAo<="0000";     --掃瞄預設
    DM13A_RESET<='0';       --重置 DM13A_Driver_RGB
    DM13ALE<='0';           --無更新資料預設
    DM13AOE<='1';           --DM13A off
    frame:=0;               --frame 數預設 0
    scan_1T<='0';           --未完成 1 次掃瞄
elsif rising_edge(RGB16X16_SCAN_p_clk) then
    if DM13ALE='0' and DM13AOE='1' then--無更新資料且顯示已關閉
        if DM13A_RESET='0' then         --尚未啟動 DM13A_Driver_RGB
            DM13A_RESET<='1';           --啟動 DM13A_Driver_RGB
            Scan_DCBAo<=Scan_DCBAo-1;   --調整掃瞄
            scan_1T<='0';
        elsif DM13A_Sendok='1' then     --傳送完成
            DM13A_RESET<='0';           --重置 DM13A_Driver_RGB
            DM13ALE<='1';               --更新顯示資料
        end if;
        T:=0;                           --顯示計時歸零
    else
        DM13ALE<='0';                   --顯示資料不更新
        DM13AOE<='0';                   --顯示
        T:=T+1;                         --顯示計時
        if T=50 then                    --顯示計時到
            DM13AOE<='1';               --不顯示
        elsif T=49 then
            if Scan_DCBAo=0 then        --完成 15~0 掃瞄
                if frame=4 then
                    scan_1T<='1';       --完成 5frame
                    frame:=0;           --重新數 frame
                else
                    frame:=frame+1;     --完成 1frame
                end if;
            end if;
        end if;
    end if;
end if;
end process RGB16X16_SCAN_p;

end Albert;
```

CH4_RGB16x16_3.vhd

```
--RGB16x16 跑馬燈(大家恭喜)
--106.12.30 版
--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 CH4_RGB16x16_3 is
port(gckp31,rstP99:in std_logic;       --系統頻率,系統 reset
     --DM13A 輸出
     DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go,:out std_logic;
     --186,187,189
     DM13ASDI_Bo,DM13ALEo,DM13AOEo:out std_logic;
     --194,188,185
     --Scan 輸出
     Scan_DCBAo:buffer std_logic_vector(3 downto 0)
     --198,197,196,195
    );
end entity CH4_RGB16x16_3;

architecture Albert of CH4_RGB16x16_3 is
    component DM13A_Driver_RGB is
    port(--DM13A_Driver_RGB 操作頻率,重置,ALE 控制
         DM13ACLK,DM13A_RESET,DM13ALE:in std_logic;
         --OE 控制,方向控制,反相控制
         DM13AOE,BIT_R_L,not01:in std_logic;
         startbit:in integer range 0 to 15; --開始操作位元
         maskRGB:in std_logic_vector(5 downto 0);
         --罩蓋操作位元
         --(5):0:disable 1:enable
         --(4..3)00:load,01:xor:10:or,11:and RGB
         LED_R,LED_G,LED_B:in std_logic_vector(15 downto 0);
         --R G B 圖形位元
         DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go:out std_logic;
         DM13ASDI_Bo,DM13ALEo,DM13AOEo:out std_logic;
         --DM13A 硬體操作位元
         DM13A_Sendok:out std_logic);
         --DM13A_Driver_RGB 完成操作位元
    end component;
         --DM13A_Driver_RGB 操作頻率,重置,ALE 控制
    signal DM13ACLK,DM13A_RESET,DM13ALE:std_logic;
         --OE 控制,方向控制,反相控制
    signal DM13AOE,BIT_R_L,not01:std_logic;
    signal startbit:integer range 0 to 15; --開始操作位元
    signal maskRGB:std_logic_vector(5 downto 0):="000000";
```

```
    --罩蓋操作位元
    signal LED_R,LED_G,LED_B:std_logic_vector(15 downto 0);
    --R G B 圖形位元
    signal DM13A_Sendok:std_logic;
    --DM13A_Driver_RGB 完成操作位元

    signal FD:std_logic_vector(24 downto 0);   --系統除頻器
    signal Gspeed:integer range 0 to 3;        --圖形取樣速度
    signal RGB_point1:integer range 0 to 15; --圖形取樣指標(掃瞄範圍)
    signal RGB_point0:integer range 0 to 127;  --圖形取樣指標(起點)
    signal RGB16X16_SCAN_p_clk:std_logic;      --clk
    type RGB16x16_T1 is array(0 to 127) of std_logic_vector(15 downto 0);
    --圖像格式
    --圖像:請參考 R.docx,G.docx,B.docx  8 個字或圖
    constant RGB16x16_RD:RGB16x16_T1:=(
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",--大
        X"0000",X"0000",X"0000",X"0000",X"0000",X"0000",
        X"0000",X"0000",X"0000",X"0000",X"0000",
        X"0000",X"0000",X"0000",X"0000",X"0000",--家
        X"FBBF",X"DBB7",X"DB67",X"DACF",X"01FB",X"5BF9",
        X"DB03",X"DB7F",X"DBEF",X"03F3",X"59DF",
        X"DAEF",X"9B67",X"D33F",X"FB7F",X"FFFF",--恭
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",--喜
        X"FFFF",X"FFFF",X"C003",X"C003",X"C003",X"C7E3",
        X"C7E3",X"C663",X"C663",X"C7E3",X"C7E3",
        X"C003",X"C003",X"C003",X"FFFF",X"FFFF",
        X"0000",X"77DC",X"745C",X"729C",X"0920",X"6440",
        X"72C8",X"79D8",X"73C8",X"67C0",X"0820",
        X"701C",X"501C",X"701C",X"0000",X"0000",
        X"FFFE",X"BFFE",X"7FFE",X"A0FE",X"FFFE",X"A0FE",
        X"7FFE",X"A1FE",X"FFFE",X"A77E",X"7FFE",
        X"B87E",X"FFFE",X"FFFE",X"FFFE",X"FFFE",
        X"0000",X"7FFC",X"7FFC",X"7FFC",X"7FFC",X"7FFC",
        X"7FFC",X"7FFC",X"7FFC",X"7FFC",X"7FFC",
        X"7FFC",X"7FFC",X"7FFC",X"7FFC",X"0000");

    constant RGB16x16_GD:RGB16x16_T1:=(
        X"F7FD",X"F7FD",X"F7FB",X"F7F7",X"F7EF",X"F79F",
        X"F67F",X"00FF",X"777F",X"F79F",X"F7EF",
        X"F7F7",X"F7FB",X"E7F9",X"F7FB",X"FFFF",--大
        X"F7D7",X"8ED7",X"D6B7",X"D5AF",X"D56B",X"52DB",
        X"94B9",X"D703",X"D77F",X"D6BF",X"D5DF",
        X"DDDF",X"D7EF",X"8FE7",X"DFEF",X"FFFF",--家
        X"0440",X"2448",X"2498",X"2530",X"FE04",X"A406",
        X"24FC",X"2480",X"2410",X"FC0C",X"A620",
        X"2510",X"6498",X"2CC0",X"0480",X"0000",--恭
        X"FFBF",X"BFBF",X"AFBF",X"A8A1",X"AAAB",X"AA2B",
        X"AAAB",X"0AAB",X"AAAB",X"AA2B",X"AAAB",
        X"A8A1",X"AFBF",X"BF3F",X"FFBF",X"FFFF",--喜
```

```
        X"0000",X"7FFE",X"7FFE",X"6006",X"6FF6",X"6816",
        X"6BD6",X"6A56",X"6A56",X"6BD6",X"6816",
        X"6FF6",X"6006",X"7FFE",X"7FFE",X"0000",
        X"0000",X"0000",X"2380",X"0100",X"0000",X"638C",
        X"551C",X"4E3C",X"551C",X"638C",X"0100",
        X"0280",X"2448",X"07C0",X"0000",X"0000",
        X"C001",X"C001",X"DF7D",X"DF3D",X"DF0D",X"DF0D",
        X"DF0D",X"DE1D",X"DC7D",X"D8FD",X"D3FD",
        X"C7FD",X"C001",X"8001",X"3FFF",X"7FFF",
        X"FFFF",X"8003",X"8003",X"BFDB",X"B39B",X"A19B",
        X"A01B",X"BC3B",X"B83B",X"B01B",X"B19B",
        X"B3DB",X"BBDB",X"8003",X"8003",X"FFFF");

    constant RGB16x16_BD:RGB16x16_T1:=(
        X"0000",X"0000",X"0000",X"0000",X"0000",X"0000",
        X"0000",X"0000",X"0000",X"0000",X"0000",
        X"0000",X"0000",X"0000",X"0000",X"0000",--大
        X"0828",X"7128",X"2948",X"2A50",X"2A94",X"AD24",
        X"6B46",X"28FC",X"2880",X"2940",X"2A20",
        X"2220",X"2810",X"7018",X"2010",X"0000",--家
        X"FBBF",X"DBB7",X"DB67",X"DACF",X"01FB",X"5BF9",
        X"DB03",X"DB7F",X"DBEF",X"03F3",X"59DF",
        X"DAEF",X"9B67",X"D33F",X"FB7F",X"FFFF",--恭
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",'
        X"FFFF",X"FFFF",X"FFFF",X"FFFF",X"FFFF",--喜
        X"0000",X"0000",X"0000",X"1FF8",X"1FF8",X"1FF8",
        X"1FF8",X"1E78",X"1E78",X"1FF8",X"1FF8",
        X"1FF8",X"1FF8",X"0000",X"0000",X"0000",
        X"0000",X"07C0",X"0448",X"0280",X"0100",X"038C",
        X"0154",X"00E4",X"0054",X"000C",X"0000",
        X"0100",X"2388",X"0000",X"0000",X"0000",
        X"FFFF",X"BFFF",X"7F83",X"A0C3",X"FFF3",X"A0F3",
        X"7FF3",X"A1E3",X"FF83",X"A703",X"7F83",
        X"B803",X"FFFF",X"FFFF",X"FFFF",X"FFFF",
        X"FFFC",X"FFFC",X"FFFC",X"FFE4",X"FFE4",X"FFE4",
        X"FFE4",X"FFC4",X"FFC4",X"FFE4",X"FFE4",
        X"FFE4",X"FFE4",X"FFFC",X"FFFC",X"FFFC");

begin
----DM13A_Driver_RGB
 DM13ACLK<=FD(2);
 U1: DM13A_Driver_RGB port map(
        DM13ACLK,DM13A_RESET,DM13ALE,DM13AOE,BIT_R_L,
        not01,startbit,maskRGB,LED_R,LED_G,LED_B,
        DM13ACLKo,DM13ASDI_Ro,DM13ASDI_Go,DM13ASDI_Bo,
        DM13ALEo,DM13AOEo,  DM13A_Sendok);

--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;

BIT_R_L<='0';           --方向變換
startbit<=0;            --從 15 位元開始
maskRGB<="000000";      --直接輸出

RGB_point1<=15-conv_integer(Scan_DCBAo);
--轉換圖形取樣指標
LED_R<=RGB16x16_RD(RGB_point1+RGB_point0); --R 圖案選擇取圖
LED_G<=RGB16x16_GD(RGB_point1+RGB_point0); --G 圖案選擇取圖
LED_B<=RGB16x16_BD(RGB_point1+RGB_point0); --B 圖案選擇取圖

--RGB16X16_SCAN_p 執行速度變換---------------
RGB16X16_SCAN_p_clk<=FD(8) when Gspeed=0 else
                     FD(7) when Gspeed=1 else
                     FD(6) when Gspeed=2 else
                     FD(5);

RGB16X16_SCAN_p:process(RGB16X16_SCAN_p_clk,rstP99)
variable frame:integer range 0 to 31;--15~0:1 frame
variable T:integer range 0 to 255; --每一掃瞄停留時間計時器
begin
if rstP99='0' then
    Scan_DCBAo<="0000"; --掃瞄預設
    RGB_point0<=0;      --移位 0
    not01<='0';         --反相變換
    DM13A_RESET<='0';   --重置 DM13A_Driver_RGB
    DM13ALE<='0';       --無更新資料預設
    DM13AOE<='1';       --DM13A off
    Gspeed<=0;          --速度預設
    frame:=0;           --frame 數預設 0
elsif rising_edge(RGB16X16_SCAN_p_clk) then
    if DM13ALE='0' and DM13AOE='1' then--無更新資料且顯示已關閉
        if DM13A_RESET='0' then     --尚未啟動 DM13A_Driver_RGB
            DM13A_RESET<='1';       --啟動 DM13A_Driver_RGB
            Scan_DCBAo<=Scan_DCBAo-1;   --調整掃瞄
        elsif DM13A_Sendok='1' then     --傳送完成
            DM13A_RESET<='0';           --重置 DM13A_Driver_RGB
            DM13ALE<='1';               --更新顯示資料
        end if;
        T:=0;                           --顯示計時歸零
    else
        DM13ALE<='0';               --顯示資料不更新
        DM13AOE<='0';               --顯示
        T:=T+1;                     --顯示計時
        if T=50 then                --顯示計時到
            DM13AOE<='1';           --不顯示
            if Scan_DCBAo=0 then    --完成 15~0 掃瞄
                if frame=15 then
                    frame:=0;       --重新數 frame
```

```
                    RGB_point0<=RGB_point0+1;--移位
                    if RGB_point0=127 then
                        not01<=not not01;   --反相變換
                        if not01='1' then
                            Gspeed<=Gspeed+1;--執行速度
                        end if;
                    end if;
                else
                    frame:=frame+1; --完成 1frame
                end if;
            end if;
        end if;
    end if;
end if;
end process RGB16X16_SCAN_p;

end Albert;
```