FPGA系統設計\_第十一章\_電子鐘專題

**10867024 廖育賢**

# **objective- THE PROBLEM AND PURPOSE**

透過LED數字顯示的控制，逐步建立一個數字式的電子鐘。

# **procedure – DESIGN methods**

## 外部電路配置:

LED鍵盤模組、基本測試模組。

1. XA(3~0) : 連接四位數字的4位元掃描控制，其中XA3最高位。
2. XB(7~0) : 連接七段式LED數字顯示的8位元控制。
3. YC(3~0) : 四位元的按鍵輸入，預備來做數字設定及測試工作。

## 內部電路設計:

1. 除10計數電路。
2. 除6計數電路。
3. 自由計數電路。
4. 矩陣掃描電路。
5. 談跳消除電路。
6. 按鍵微分電路。
7. 碼表控制電路。
8. 數字選擇電路。
9. 七段查表電路。

# **simulation results**

## program codes

### **ch011.vhd**

-The IEEE standard 1164 package, declares std\_logic, rising\_edge(), etc.

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_arith.all;

use IEEE.std\_logic\_unsigned.all;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- DIGIT6 : one digit control with divide-by-6

-- designed by Pei-Chong Tang, Feb. 1999

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

entity DIGIT6 is

port

( CLK : in STD\_LOGIC; --system clock

DB : out STD\_LOGIC\_VECTOR (3 downto 0); --digit bus

ENB : in STD\_LOGIC; --output enable

CLR : in STD\_LOGIC; --clear signal

EC : in STD\_LOGIC; --carry in

CY : out STD\_LOGIC --carry out

);

end DIGIT6;

architecture DIGIT6\_ARCH of DIGIT6 is

signal Q : STD\_LOGIC\_VECTOR (3 downto 0);

signal RST,DLY : STD\_LOGIC;

begin

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

-- one digit control with divide-by-10

--

process (CLK,RST)

begin

if RST='1' then --reset control

Q <= "0000"; --reset to 0

elsif CLK'event and CLK='1' then --clock rising

DLY <= Q(2); --delay

if EC='1' then --clock enable

Q <= Q+1; --counter+1

end if;

end if;

end process;

CY <= not Q(2) and DLY; --carry output

RST <= '1' when Q=6 or CLR='1' else --reset when Q=6

'0';

DB <= Q when ENB='1' else --output control

"ZZZZ";

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- end of architechture

--

end DIGIT6\_ARCH;

--The IEEE standard 1164 package, declares std\_logic, rising\_edge(), etc.

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_arith.all;

use IEEE.std\_logic\_unsigned.all;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- DIGIT10 : one digit control with divide-by-10

-- designed by Pei-Chong Tang, Feb. 1999

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

entity DIGIT10 is

port

( CLK : in STD\_LOGIC; --system clock

DB : out STD\_LOGIC\_VECTOR (3 downto 0); --digit bus

ENB : in STD\_LOGIC; --output enable

CLR : in STD\_LOGIC; --clear signal

EC : in STD\_LOGIC; --carry in

CY : out STD\_LOGIC --carry out

);

end DIGIT10;

architecture DIGIT10\_ARCH of DIGIT10 is

signal Q : STD\_LOGIC\_VECTOR (3 downto 0);

signal RST,DLY : STD\_LOGIC;

begin

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

-- one digit control with divide-by-10

--

process (CLK,RST)

begin

if RST='1' then --reset control

Q <= "0000"; --reset to 0

elsif CLK'event and CLK='1' then --clock rising

DLY <= Q(3); --delay

if EC='1' then --clock enable

Q <= Q+1; --counter+1

end if;

end if;

end process;

CY <= not Q(3) and DLY; --carry output

RST <= '1' when Q=10 or CLR='1' else --reset when Q=10

'0';

DB <= Q when ENB='1' else --output control

"ZZZZ";

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- end of architechture

--

end DIGIT10\_ARCH;

--The IEEE standard 1164 package, declares std\_logic, rising\_edge(), etc.

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_arith.all;

use IEEE.std\_logic\_unsigned.all;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- CH11C : digital clock with 7-segment LED control (test #C)

-- designed by Pei-Chong Tang, Jan. 1999

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

entity CH11C is

port

( CLK : in STD\_LOGIC; --4MHz clock

XA : out STD\_LOGIC\_VECTOR (3 downto 0); --output port

XB : out STD\_LOGIC\_VECTOR (7 downto 0); --input/output

XC : in STD\_LOGIC\_VECTOR (3 downto 0) --input port

);

end CH11C;

architecture CH11C\_ARCH of CH11C is

component DIGIT10 --\*\*\*\*\* one digit with divide-by-10 \*\*\*\*\*

port

( CLK : in STD\_LOGIC; --system clock

DB : out STD\_LOGIC\_VECTOR (3 downto 0); --digit bus

ENB : in STD\_LOGIC; --output enable

CLR : in STD\_LOGIC; --clear signal

EC : in STD\_LOGIC; --carry in

CY : out STD\_LOGIC --carry out

);

end component;

component DIGIT6 --\*\*\*\*\* one digit with divide-by-6 \*\*\*\*\*

port

( CLK : in STD\_LOGIC; --system clock

DB : out STD\_LOGIC\_VECTOR (3 downto 0); --digit bus

ENB : in STD\_LOGIC; --output enable

CLR : in STD\_LOGIC; --clear signal

EC : in STD\_LOGIC; --carry in

CY : out STD\_LOGIC --carry out

);

end component;

--\*\*\*\*\* global signals \*\*\*\*\*

signal DB : STD\_LOGIC\_VECTOR (3 downto 0);

signal SEG : STD\_LOGIC\_VECTOR (6 downto 0);

signal ENB : STD\_LOGIC\_VECTOR (3 downto 0);

signal HZ,SEC,CLR,HOLD : STD\_LOGIC;

signal SAMPLE,MATCH : STD\_LOGIC;

signal KEY,FLT,DIF : STD\_LOGIC;

signal SEL : STD\_LOGIC\_VECTOR (2 downto 0);

begin

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

-- system connection for the experiment

--

SYSTEM\_CONNECT : block

signal EC, EC1,EC2,EC3,EC4 : STD\_LOGIC;

signal CY0,CY1,CY2,CY3,CY4 : STD\_LOGIC;

begin

U1: DIGIT10 port map (CLK,DB,ENB(0),CLR,EC1,CY1);

U2: DIGIT6 port map (CLK,DB,ENB(1),CLR,EC2,CY2);

U3: DIGIT10 port map (CLK,DB,ENB(2),CLR,EC3,CY3);

U4: DIGIT6 port map (CLK,DB,ENB(3),CLR,EC4,CY4);

CLR <= not XC(3);

EC <= HZ and not SEL(2) and not XC(0);

CY0 <= HZ and not HOLD;

EC1 <= (CY0 and SEL(2)) or (EC and not SEL(1) and not SEL(0));

EC2 <= (CY1 and SEL(2)) or (EC and not SEL(1) and SEL(0));

EC3 <= (CY2 and SEL(2)) or (EC and SEL(1) and not SEL(0));

EC4 <= (CY3 and SEL(2)) or (EC and SEL(1) and SEL(0));

KEY <= not XC(0) or not XC(1) or not XC(2) or not XC(3);

XA <= not ENB;

XB(7) <= SEC and ENB(2) and SEL(2) and not HOLD;

GEN: for I in 0 to 6 generate

XB(I) <= SEG(I) and (SEC or not MATCH or SEL(2));

end generate;

end block SYSTEM\_CONNECT;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- 24-bit free counter

--

FREE\_COUNTER : block

signal Q : STD\_LOGIC\_VECTOR (23 downto 0);

signal D : STD\_LOGIC\_VECTOR (1 downto 0);

signal DLY1,DLY2 : STD\_LOGIC;

begin

process (CLK)

begin

if CLK'event and CLK='1' then

DLY1 <= Q(23);

DLY2 <= Q(15);

Q <= Q+1;

end if;

end process;

SEC <= Q(23);

HZ <= Q(23) and not DLY1;

SAMPLE <= Q(15) and not DLY2;

D <= Q(17 downto 16);

ENB <= "0001" when D=0 else

"0010" when D=1 else

"0100" when D=2 else

"1000" when D=3 else

"0000";

MATCH <= '1' when D=SEL(1 downto 0) else '0';

end block FREE\_COUNTER;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

-- keyboard debouncing

--

DEBOUNCING : block

signal D0,D1 : STD\_LOGIC;

begin

process (CLK)

begin

if CLK'event and CLK='1' then --CLK rising

if SAMPLE='1' then --CLK enable

D1<=D0; D0<=KEY; --delay

FLT <= ((D0 and D1) or FLT) and (D0 or D1); --RS-F/F

end if;

end if;

end process;

end block DEBOUNCING;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--

-- differential signal

--

DIFFERENTIAL : block

signal D0,D1 : STD\_LOGIC;

begin

process (CLK)

begin

if CLK'event and CLK='1' then --CLK rising

D1<=D0; D0<=FLT; --delay

end if;

end process;

DIF <= D0 and not D1; --differential

end block DIFFERENTIAL;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- timer control

--

TIMER\_CONTROL : block

signal SET,EC : STD\_LOGIC;

begin

process (CLK)

begin

if SET='1' then

HOLD <= '1';

elsif CLK'event and CLK='1' then

if EC='1' then

HOLD <= not HOLD;

end if;

end if;

end process;

SET <= not XC(3);

EC <= not XC(2) and DIF;

end block TIMER\_CONTROL;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- digit select

--

DIGIT\_SELECT : block

signal Q : STD\_LOGIC\_VECTOR (2 downto 0);

signal SET,EC : STD\_LOGIC;

begin

process (CLK)

begin

if SET='1' then

Q <= "100";

elsif CLK'event and CLK='1' then

if EC='1' then

Q <= Q-1;

end if;

end if;

end process;

SET <= '1' when Q=7 else '0';

EC <= DIF and not XC(1);

SEL <= Q;

end block DIGIT\_SELECT;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- seven-segment LED transformation

--

SEVEN\_SEGMENT : block

begin

--GFEDCBA

SEG <= "0111111" when DB=0 else

"0000110" when DB=1 else

"1011011" when DB=2 else

"1001111" when DB=3 else

"1100110" when DB=4 else

"1101101" when DB=5 else

"1111101" when DB=6 else

"0000111" when DB=7 else

"1111111" when DB=8 else

"1101111" when DB=9 else

"0000000";

end block SEVEN\_SEGMENT;

--

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- end of architechture

--

end CH11C\_ARCH;

### **CH010.UCF**

##############################################

# I/O mapping for USB 3s1000 FPGA board #

##############################################

NET CLK LOC = B12;

NET XA<0> LOC = N2;

NET XA<1> LOC = T1;

NET XA<2> LOC = T2;

NET XA<3> LOC = U2;

NET XB<0> LOC = M3;

NET XB<1> LOC = M4;

NET XB<2> LOC = W3;

NET XB<3> LOC = W4;

NET XB<4> LOC = Y3;

NET XB<5> LOC = Y4;

NET XB<6> LOC = W6;

NET XB<7> LOC = Y6;

NET XC<0> LOC = V1;

NET XC<1> LOC = V2;

NET XC<2> LOC = W1;

NET XC<3> LOC = W2;

## observations

### 歸零

### 

### 2.選擇位置

### 