

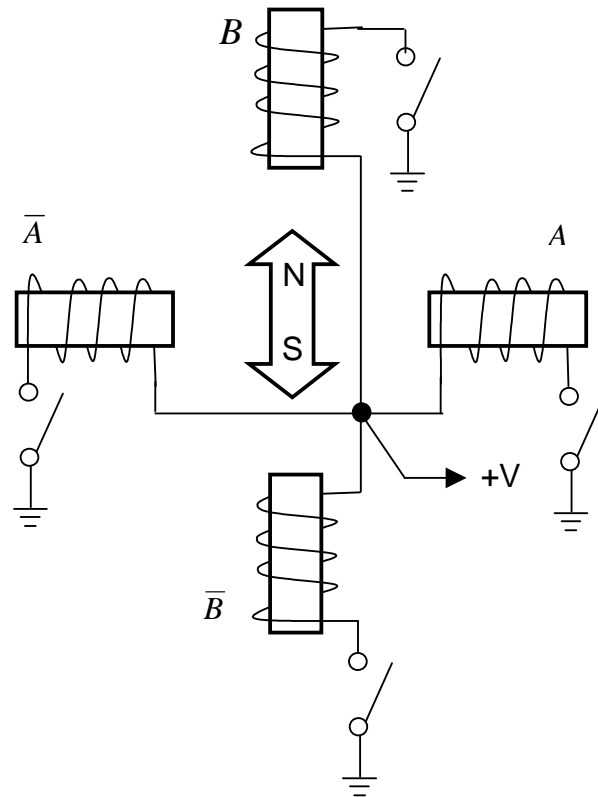


# 步進馬達控制實習

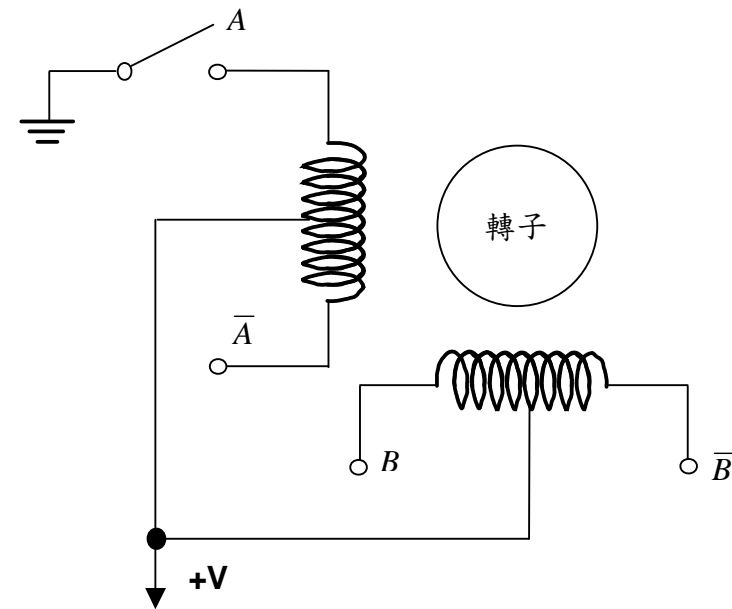
# 步進馬達全步控制（一相激磁）實習

## ■ 相關知識

□ 四相式步進馬達驅動示意圖



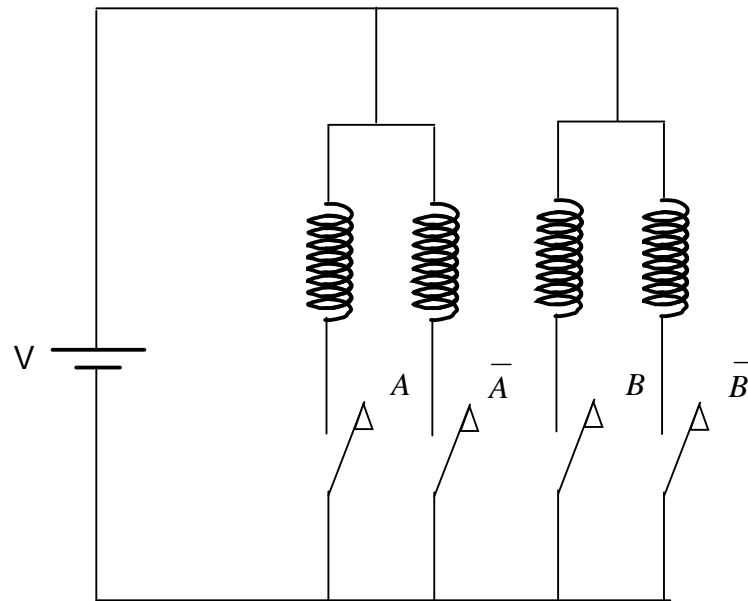
□ 4相6線式步進馬達



## 步進馬達全步控制（一相激磁）實習

### ■ 相關知識(續)

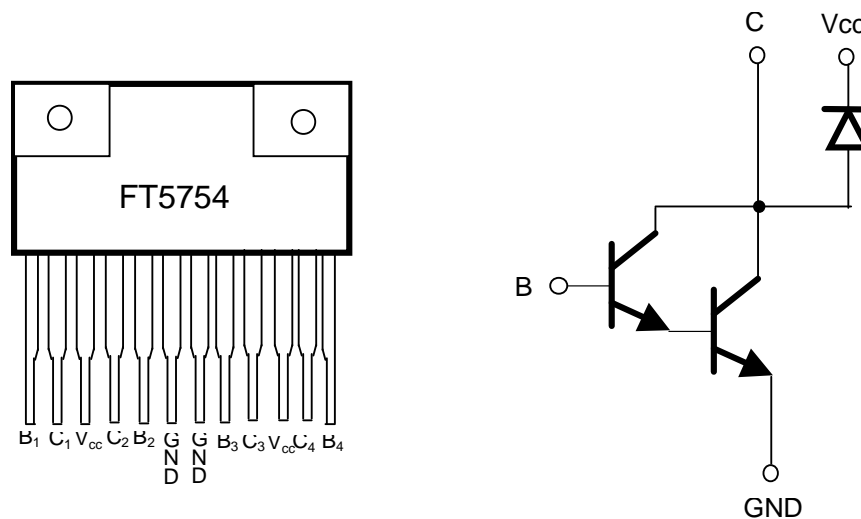
- 一個四相步進馬達的工作方式如下圖所示；其中線圈接點所連接的開關是以脈波控制電晶體的飽和或截止加以實現。



# 步進馬達全步控制（一相激磁）實習

## ■ 相關知識(續)

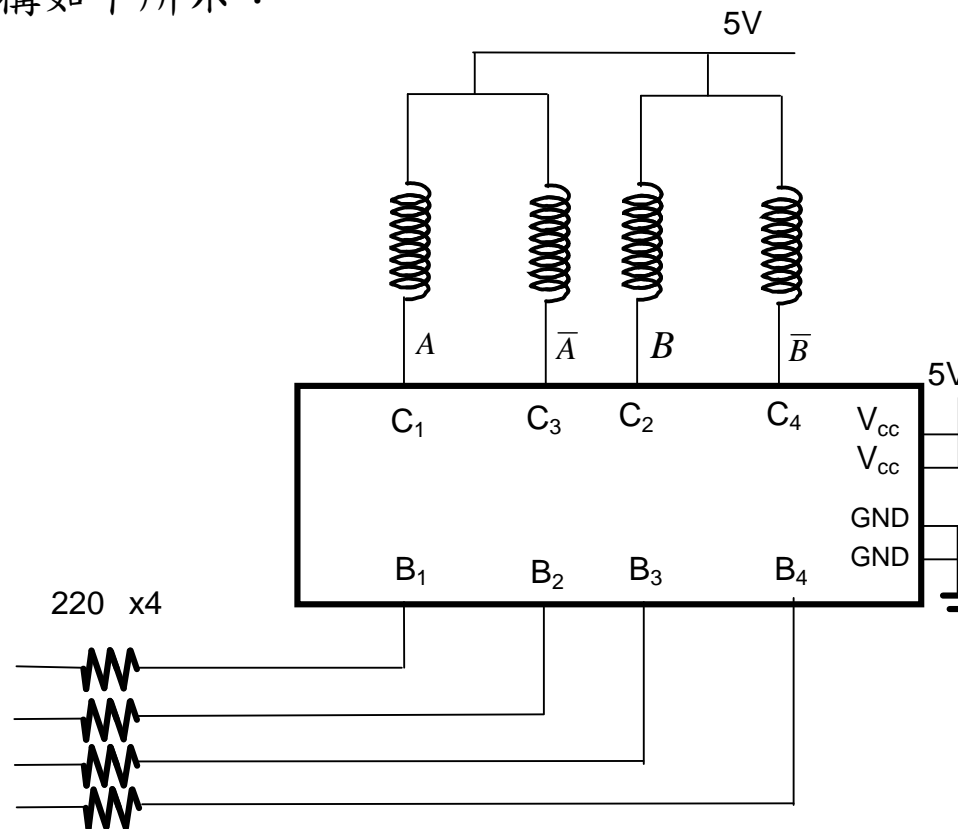
- 而在一般的實驗中，我們常常拿內含四組NPN達靈頓電晶體的FT5754元件來驅動步進馬達，達靈頓電晶體在B極輸入正電壓而導通飽和時，可以放大C極端從線圈往下流通的電流，電源能否提供足夠的電流來驅動步進馬達會影響到步進馬達的轉速與力矩。FT5754外觀及內部結構如下所示：



# 步進馬達全步控制（一相激磁）實習

## ■ 相關知識(續)

- 當採用FT5754的達靈頓對電晶體來作為連接步進馬達的開關時，其電路的結構如下所示：

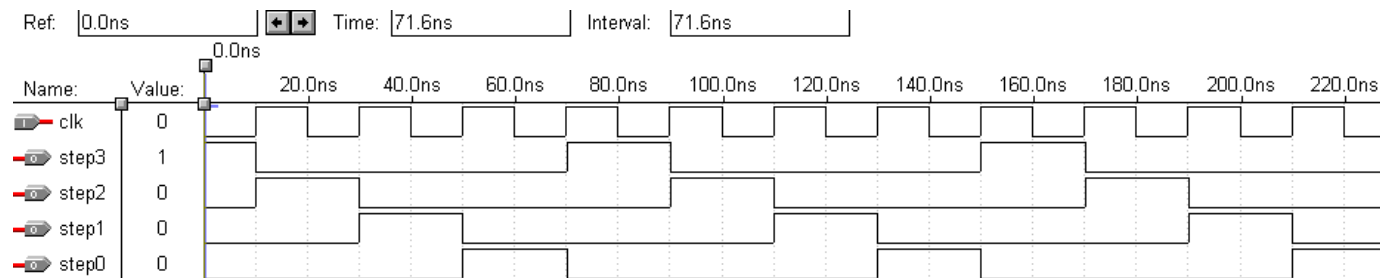


# 步進馬達全步控制（一相激磁）實習

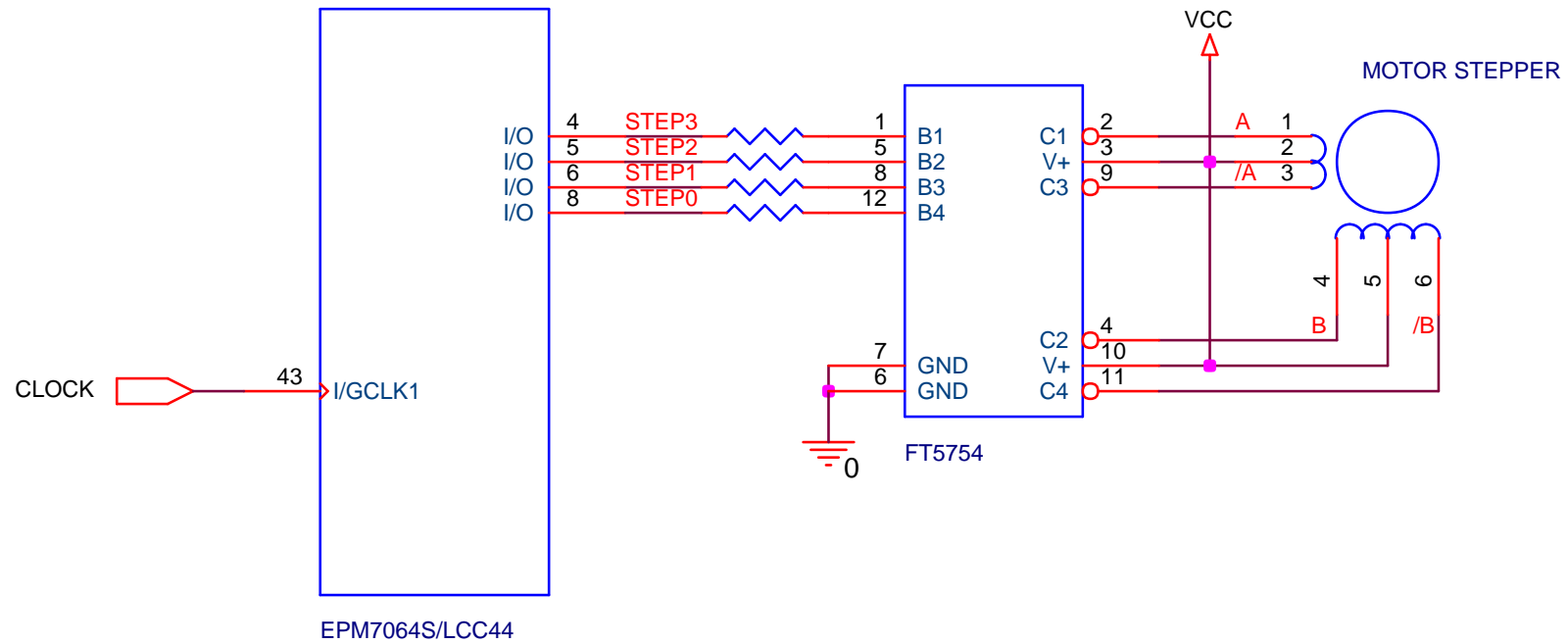
## ■ 相關知識(續)

- 一相激磁的作法是控制每次只有一個線圈有電流通過，當四個線圈輪流動作時，步進馬達便能開始轉動。一相激磁的控制順序如下表所列：

	線圈控制狀態				轉動方向	
	A	B	$\bar{A}$	$\bar{B}$	順時針	逆時針
1	1	0	0	0	↓	↑
2	0	1	0	0		
3	0	0	1	0		
4	0	0	0	1		



# 實驗電路圖



# 步進馬達全步控制（一相激磁）實習

## ■ 程式與說明

### □ 步進馬達全步控制（一相激磁）程式碼

```
11 entity stepmotor_fulla is
12 port(
13     clk : in std_logic;
14     step: out std_logic_vector(3 downto 0)
15 );
16 end stepmotor_fulla;
17
18 architecture arch of stepmotor_fulla is
19     signal cnt : std_logic_vector(1 downto 0);
20 begin
21
22 ----- process -----
23     process(clk)
24     begin
25         if clk'event and clk='1' then
26             cnt <= cnt+1;
27         end if;
28     end process;
29     step <= "1000" when cnt=0 else
30             "0100" when cnt=1 else
31             "0010" when cnt=2 else
32             "0001";
33 end arch;
```



# 步進馬達全步控制（一相激磁）實習

## ■ 程式與說明

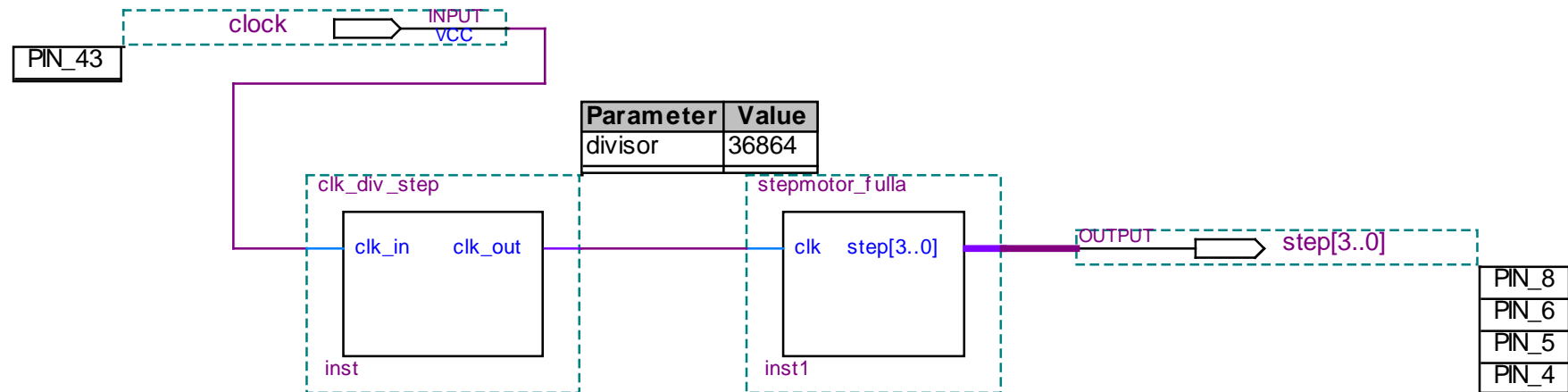
□ 除頻器程式碼（將25MHz除頻產生50Hz輸出）：

```
13 entity clk_div_step is
14   generic(divisor:integer:=500000);
15   port(
16     clk_in : in std_logic;
17     clk_out: out std_logic
18   );
19 end clk_div_step;
20
21 architecture arch of clk_div_step is
22   signal cnt2 : std_logic;
23   begin
24     ----- clk divider -----
25     process(clk_in)
26     variable cnt1,divisor2 : integer range 0 to divisor;
27     begin
28       divisor2:=divisor/2;
29       ----- up counter -----
30       if (clk_in'event and clk_in='1') then
31         if cnt1 = divisor then
32           cnt1 := 1;
33         else
34           cnt1 := cnt1 + 1;
35         end if;
36       end if;
37       ----- clk_out register clk generator -----
38       if (clk_in'event and clk_in='1') then
39         if (( cnt1 = divisor2) or (cnt1 = divisor))then
40           cnt2 <= not cnt2 ;
41         end if;
42       end if;
43       clk_out <= cnt2 ;
44     end process;
45 end arch;
46
```

# 步進馬達全步控制（一相激磁）實習

## ■ 程式與說明

□ 主程式之電路連線圖：

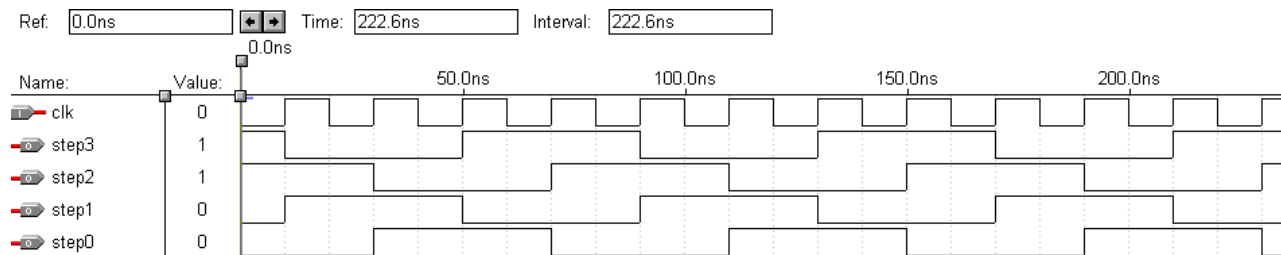


# 步進馬達全步控制（二相激磁）實習

## ■ 相關知識

- 二相激磁的控制方法是每次都讓兩個線圈通過電流，每一個步驟可以移動一個步進角：

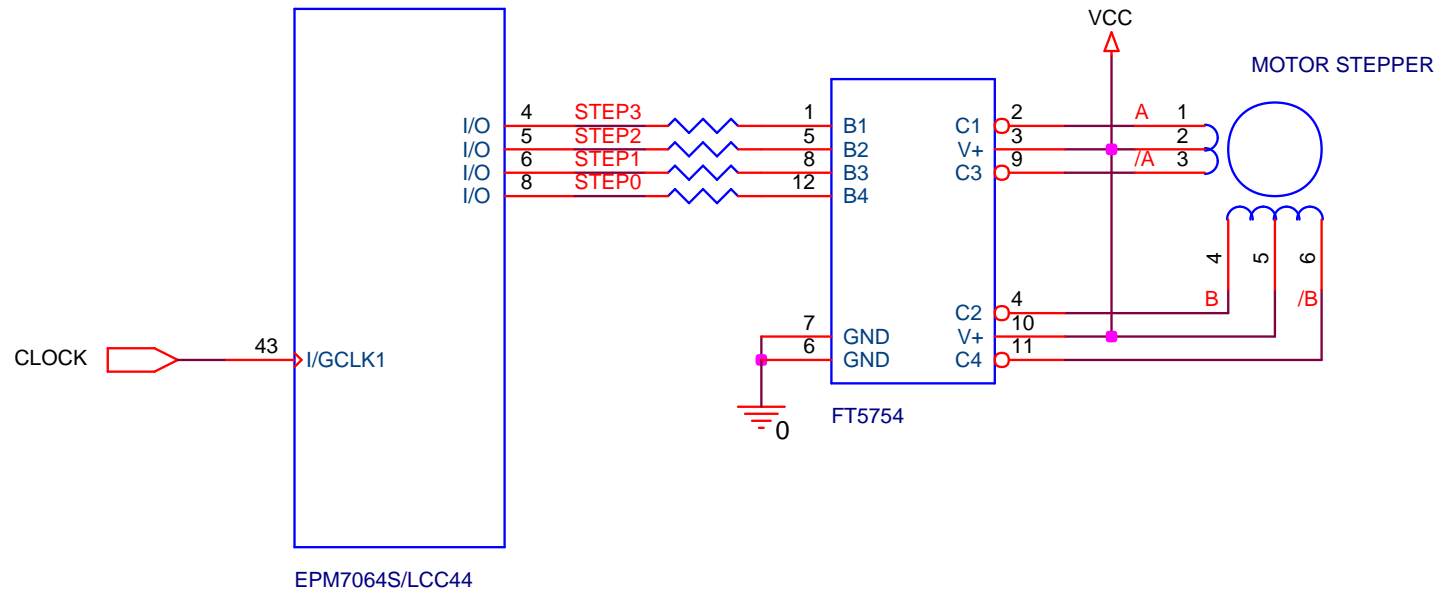
	線圈控制狀態				轉動方向	
	A	B	$\bar{A}$	$\bar{B}$	順時針	逆時針
1	1	1	0	0	↓	↑
2	0	1	1	0		
3	0	0	1	1		
4	1	0	0	1		



# 功能模擬與CPLD下載驗證

輸入	腳位	輸出	腳位
clock	28	Step(3)	5
		Step(2)	4
		Step(1)	2
		Step(0)	224

# 實驗電路圖



# 步進馬達全步控制（二相激磁）實習

## ■ 程式與說明

### □ 步進馬達全步控制(二相激磁) 程式碼

```
12 entity stepmotor_fullb is
13 port(
14 clk : in std_logic;
15 step: out std_logic_vector(3 downto 0)
16 );
17 end stepmotor_fullb;
18
19 architecture arch of stepmotor_fullb is
20 signal cnt : std_logic_vector
    (1 downto 0);
21 begin
22
```

```
23 ----- process -----
24 process(clk)
25 begin
26   if clk'event and clk='1' then
27     cnt <= cnt+1;
28   end if;
29 end process;
30
31 step <= "1100" when cnt=0 else
32         "0110" when cnt=1 else
33         "0011" when cnt=2 else
34         "1001";
35 end arch;
```

# 步進馬達全步控制（二相激磁）實習

## ■ 程式與說明

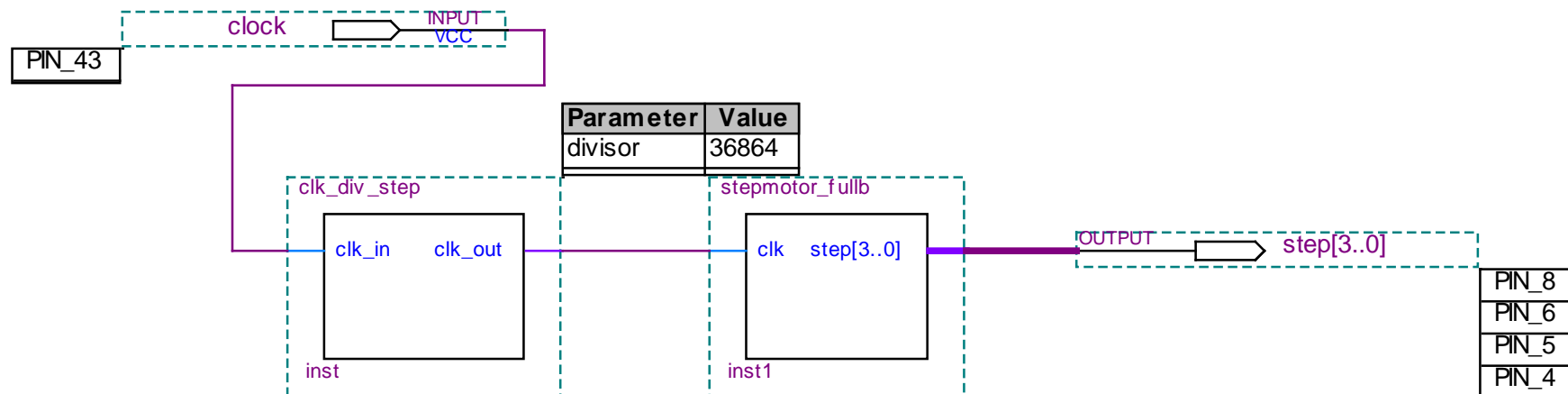
□ 除頻器程式碼（將25MHz除頻產生50Hz輸出）：

```
13 entity clk_div_step is
14   generic(divisor:integer:= 500000);
15   port(
16     clk_in : in std_logic;
17     clk_out: out std_logic
18   );
19 end clk_div_step;
20
21 architecture arch of clk_div_step is
22   signal cnt2 : std_logic;
23   begin
24     ----- clk divider -----
25     process(clk_in)
26     variable cnt1,divisor2 : integer range 0 to divisor;
27     begin
28       divisor2:=divisor/2;
29       ----- up counter -----
30       if (clk_in'event and clk_in='1') then
31         if cnt1 = divisor then
32           cnt1 := 1;
33         else
34           cnt1 := cnt1 + 1;
35         end if;
36       end if;
37       ----- clk_out register clk generator -----
38       if (clk_in'event and clk_in='1') then
39         if (( cnt1 = divisor2) or (cnt1 = divisor))then
40           cnt2 <= not cnt2 ;
41         end if;
42       end if;
43       clk_out <= cnt2 ;
44     end process;
45 end arch;
46
```

# 步進馬達全步控制（二相激磁）實習

## ■ 程式與說明

- 主程式之電路連線圖：





# 步進馬達全步控制（二相激磁）實習

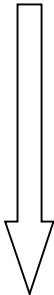
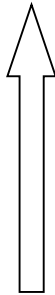
## ■ 功能模擬與CPLD下載驗證

輸入	腳位	輸出	腳位
clock	28	Step(3)	5
		Step(2)	4
		Step(1)	2
		Step(0)	224

# 步進馬達半步控制（一二相激磁）實習

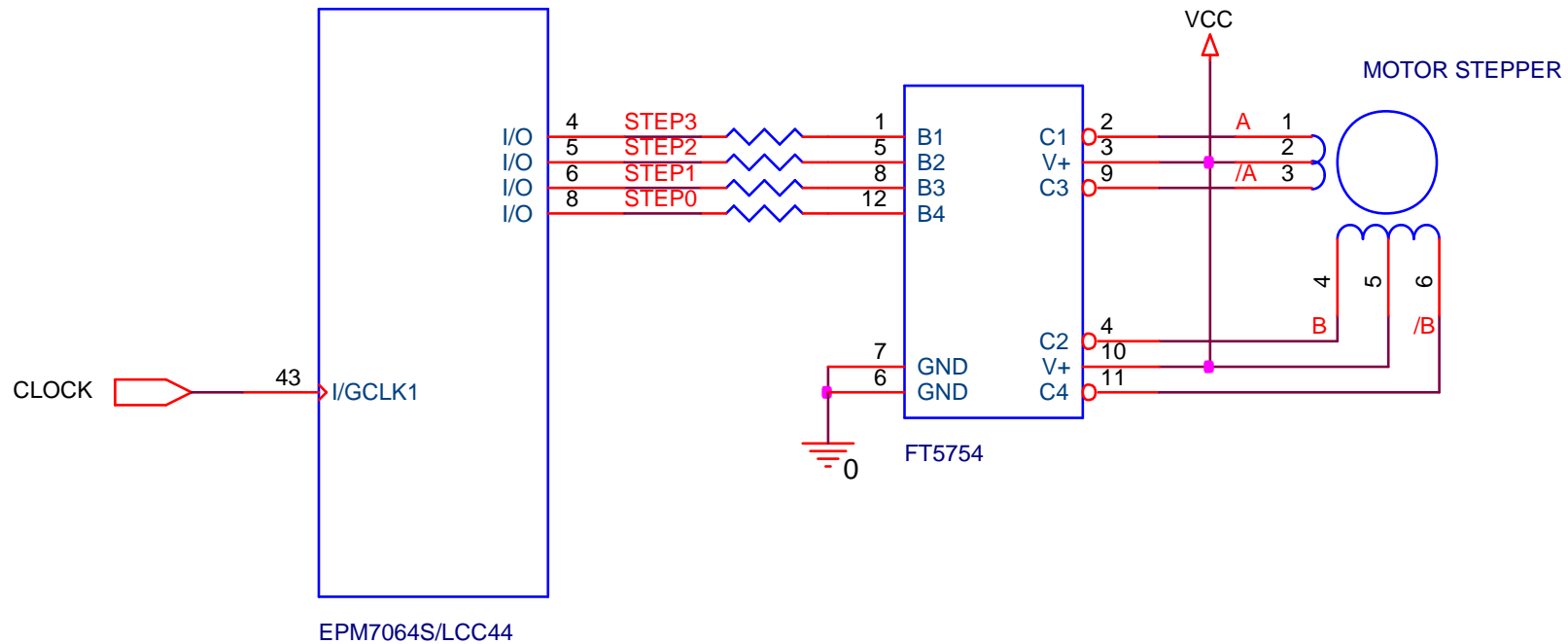
## ■ 相關知識

- 一二相激磁的作法是以一相激磁和二相激磁來交替控制，在定子繞組切換激磁線圈電壓時，仍有一個線圈上有電流通過，可保持一定的力矩以維持運轉上的平穩，而其耗電量又比二相激磁來得節省。一二相激磁每次可以移動半個步進角，其激磁的控制順序如下：

	線圈控制狀態				轉動方向	
	$A$	$B$	$\bar{A}$	$\bar{B}$	順時針	逆時針
1	1	0	0	0		
2	1	1	0	0		
3	0	1	0	0		
4	0	1	1	0		
5	0	0	1	0		
6	0	0	1	1		
7	0	0	0	1		
8	1	0	0	1		

# 步進馬達半步控制（一二相激磁）實習

## ■ 實驗電路圖



# 步進馬達半步控制（一二相激磁）實習

## ■ 程式與說明

### □ 步進馬達半步控制（一二相激磁）程式碼

```
12 entity stepmotor_half is
13 port(
14     clk : in std_logic;
15     step: out std_logic_vector(3 downto 0)
16 );
17 end stepmotor_half;
18
19 architecture arch of stepmotor_half is
20     signal cnt : std_logic_vector(2 downto 0);
21 begin
22
23     ----- process -----
24     process(clk)
25     begin
26         if clk'event and clk='1' then
27             cnt <= cnt+1;
28         end if;
29     end process;
30
31     step <= "1000" when cnt=0 else
32         "1100" when cnt=1 else
33         "0100" when cnt=2 else
34         "0110" when cnt=3 else
35         "0010" when cnt=4 else
36         "0011" when cnt=5 else
37         "0001" when cnt=6 else
38         "1001";
39 end arch;
```

# 步進馬達半步控制（一二相激磁）實習

## ■ 程式與說明

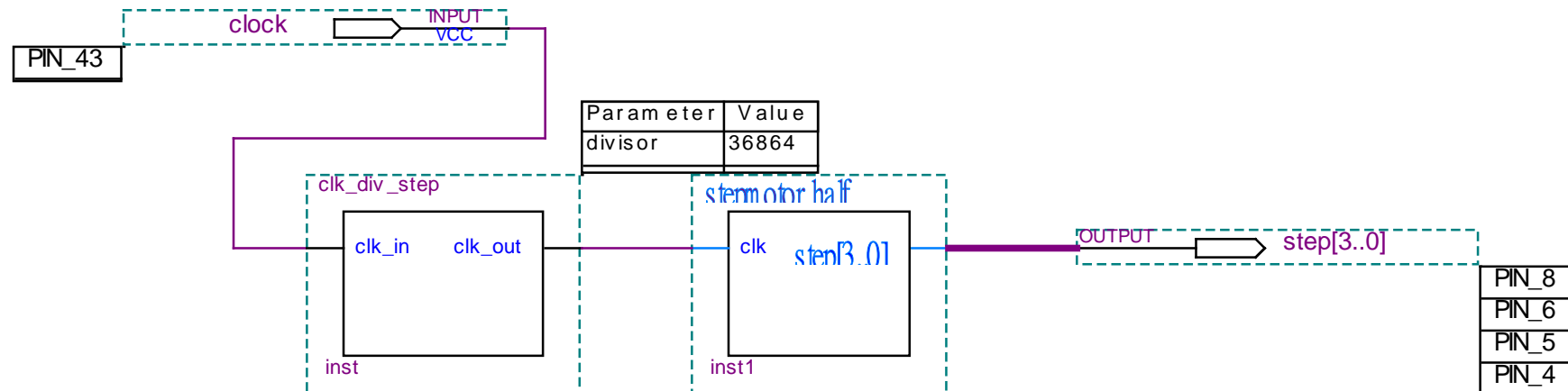
□ 除頻器程式碼（將25MHz除頻產生50Hz輸出）：

```
13 entity clk_div_step is
14   generic(divisor:integer:= 500000);
15   port(
16     clk_in : in std_logic;
17     clk_out: out std_logic
18   );
19 end clk_div_step;
20
21 architecture arch of clk_div_step is
22   signal cnt2 : std_logic;
23   begin
24     ----- clk divider -----
25     process(clk_in)
26     variable cnt1,divisor2 : integer range 0 to divisor;
27     begin
28       divisor2:=divisor/2;
29       ----- up counter -----
30       if (clk_in'event and clk_in='1') then
31         if cnt1 = divisor then
32           cnt1 := 1;
33         else
34           cnt1 := cnt1 + 1;
35         end if;
36       end if;
37       ----- clk_out register clk generator -----
38       if (clk_in'event and clk_in='1') then
39         if (( cnt1 = divisor2) or (cnt1 = divisor))then
40           cnt2 <= not cnt2 ;
41         end if;
42       end if;
43       clk_out <= cnt2 ;
44     end process;
45   end arch;
46
```

# 步進馬達半步控制（一二相激磁）實習

## ■ 程式與說明

□ 主程式之電路連線圖：



# 步進馬達半步控制（一二相激磁）實習

- 功能模擬與CPLD下載驗證

輸入	腳位	輸出	腳位
clk	28	Step(3)	8
		Step(2)	5
		Step(1)	2
		Step(0)	224

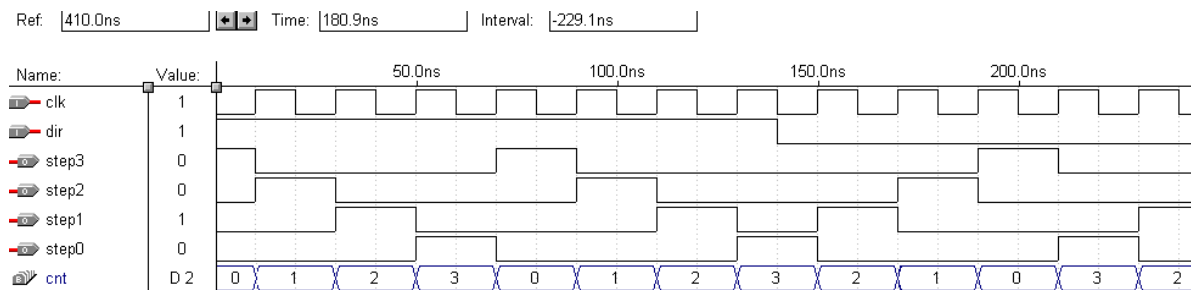
# 步進馬達轉向控制實習

## ■ 相關知識

- 本實驗中，我們將設計一個可以控制上、下數計數功能的DIR鍵來控制步進馬達一相激磁之正反轉

- 上、下數動作之控制程式碼的寫法如下：

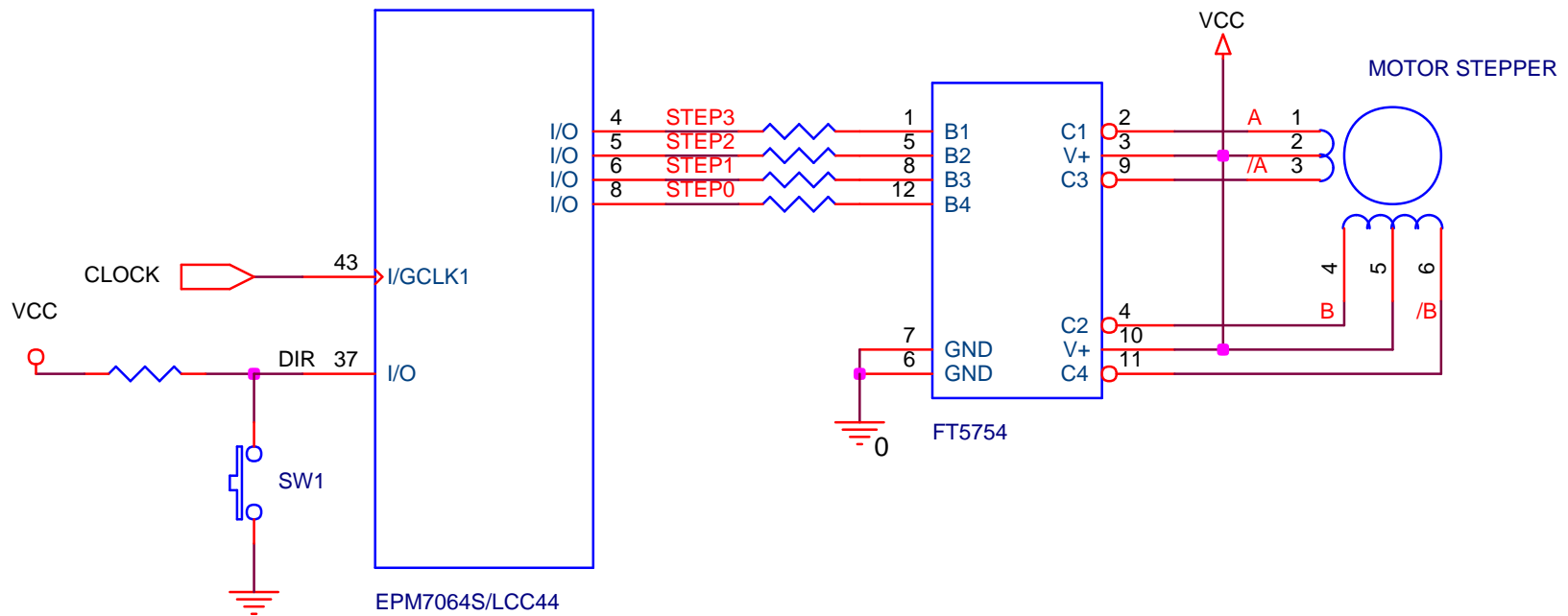
```
if clk'event and clk='1' then
  if dir='1' then
    cnt <= cnt+1;
  else
    cnt <= cnt-1;
  end if;
end if;
```





# 步進馬達轉向控制實習

## ■ 實驗電路圖



# 步進馬達轉向控制實習

## ■ 程式與說明

### □ 步進馬達轉向控制程式碼

```
11 entity stepmotor_dir is
12 port(
13     clk : in std_logic;
14     dir : in std_logic;
15     step: out std_logic_vector(3 downto 0)
16 );
17 end stepmotor_dir;
18
19 architecture arch of stepmotor_dir is
20 signal cnt : std_logic_vector(1 downto 0);
21 begin
22
23 ----- process -----
24 process(clk)
25 begin
```

```
26     if clk'event and clk='1' then
27         if dir='1' then
28             cnt <= cnt+1;
29         else
30             cnt <= cnt-1;
31         end if;
32     end if;
33 end process;
34
35 step <= "1000" when cnt=0 else
36         "0100" when cnt=1 else
37         "0010" when cnt=2 else
38         "0001";
39 end arch;
```

# 步進馬達轉向控制實習

## ■ 程式與說明

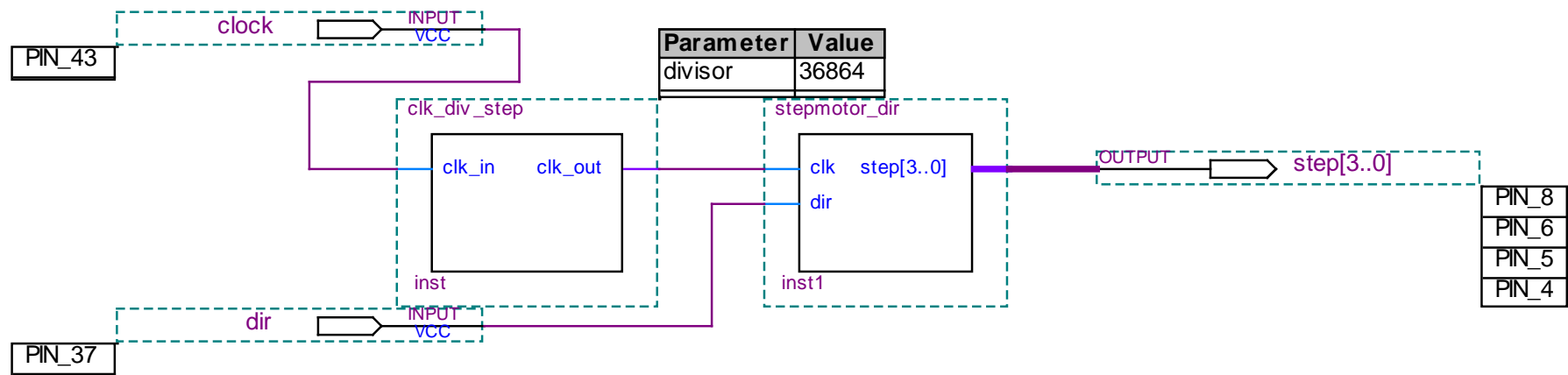
□ 除頻器程式碼（將25MHz除頻產生50Hz輸出）：

```
13 entity clk_div_step is
14   generic(divisor:integer:= 500000);
15   port(
16     clk_in : in std_logic;
17     clk_out: out std_logic
18   );
19 end clk_div_step;
20
21 architecture arch of clk_div_step is
22   signal cnt2 : std_logic;
23   begin
24     ----- clk divider -----
25     process(clk_in)
26     variable cnt1,divisor2 : integer range 0 to divisor;
27     begin
28       divisor2:=divisor/2;
29       ----- up counter -----
30       if (clk_in'event and clk_in='1') then
31         if cnt1 = divisor then
32           cnt1 := 1;
33         else
34           cnt1 := cnt1 + 1;
35         end if;
36       end if;
37       ----- clk_out register clk generator -----
38       if (clk_in'event and clk_in='1') then
39         if (( cnt1 = divisor2) or (cnt1 = divisor))then
40           cnt2 <= not cnt2 ;
41         end if;
42       end if;
43       clk_out <= cnt2 ;
44     end process;
45 end arch;
46
```

# 步進馬達轉向控制實習

## ■ 程式與說明

□ 主程式之電路連線圖：



# 步進馬達轉向控制實習

## ■ 功能模擬與CPLD下載驗證

輸入	腳位	輸出	腳位
clock	28	Step(3)	5
dir	222	Step(2)	4
		Step(1)	2
		Step(0)	224