CF 系列
附 GPIB、RS232 三相控制器
操 作 手 冊

REVISION:1.1

FILE:CPC3-CF

PN:800-01023

2005/03/31

目錄：

附錄：

壹.電氣規格

本三相控制器是由數位方式產生之具頻率穩定；振幅穩定，失真率低，溫度漂移少，....等優良性能之正弦波三相信號產生器，是目前三相交流電源供應器信號來源之最佳選擇。

規格：

1. 輸入電源　：AC 110V/220V ±10％　50/60Hz。

2. 輸出振幅　：(1)SIN 1500mV(Rms)　RL=10KΩ。
   (2)穩定度：±0.02％。

3. 輸出頻率　：＿＿＿＿～＿＿＿＿ ± ＿0.01＿ Hz。

4. 波形失真　：≦0.2％。

5. 溫度係數　：100PPM。

6. 相位差　　：120 度 ± 1 度。

7. 具緩啟動裝置，電源投入時振幅由小而大，無突波現象。

8. 具 RS-232，GPIB 介面，可由電腦連線控制。

9. 連線指令請參考界面指令說明書。

注意：使用本機前，請先詳閱操作程序說明。

貳.面板說明：(請參照 FIG 2-1)

(1)本機電源開關　　：按 I 為開，按 O 為關。

(2)電源指示燈　　：當電源開關開啟時，指示燈則亮起。

(3)輸出控制按鈕　　：按一次為開,再按一次為關;開機時為關狀態。

(4)主機輸出電磁開關指示燈　　：亮為開，不亮為關。

(5)本機輸出信號振幅粗調　　：左轉到底可調至 OV，右轉到底可調至 1500mV(Rms)。

(6)本機輸出信號振幅細調　　：細部微調。

(7)“C” 相前板輸出 BNC 插座，最大輸出：SIN 1500mV(Rms) RL=10KΩ。

(8)“B” 相前板輸出 BNC 插座，最大輸出：SIN 1500mV(Rms) RL=10KΩ。

(9)“A” 相前板輸出 BNC 插座，最大輸出：SIN 1500mV(Rms) RL=10KΩ。

(10)復歸按鈕開關　　：當機器在異常情況排除後復歸按鈕 (機器異常時亮紅燈,且有警示聲)

(11)RANGE　　：輸出電壓檔位選擇(HI or LOW)

(12)輸出信號之頻率設定指撥開關

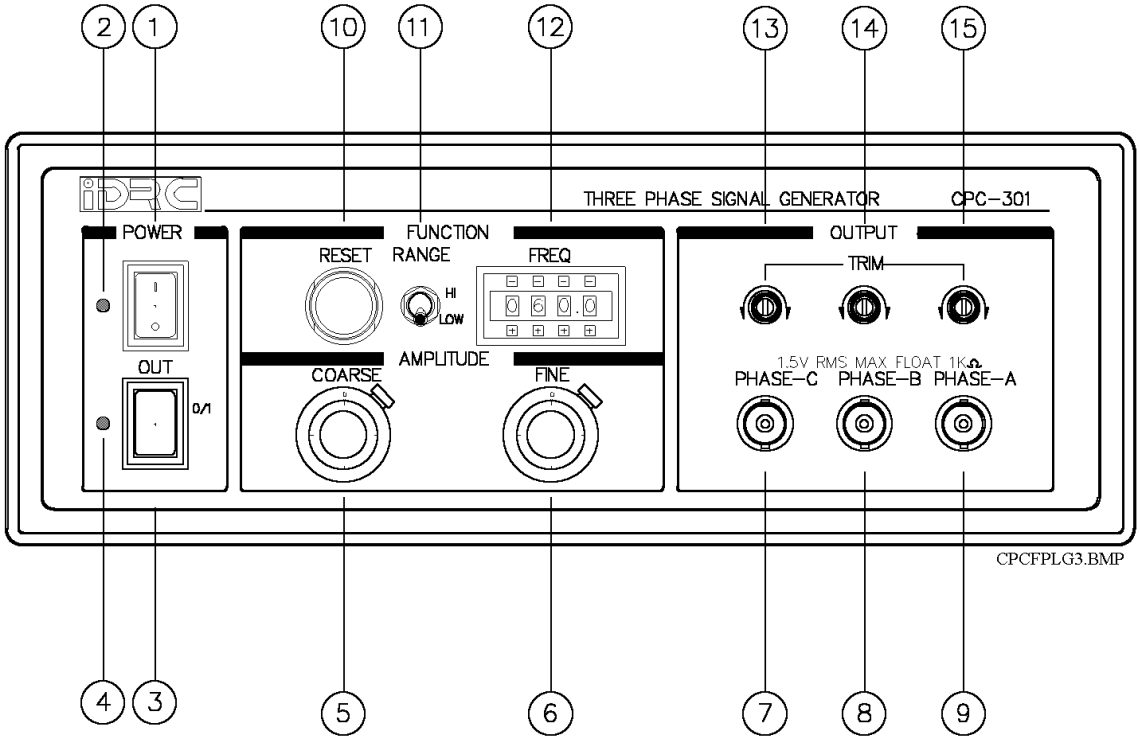
(13) “C” 相輸出振幅微調 VR，可調±20％。

(14) “B” 相輸出振幅微調 VR，可調±20％。

(15) “A” 相輸出振幅微調 VR，可調±20％。

註：(1)各相間信號相差 120 度。
(2)前板及背板 BNC 座是併連連接。

(FIG 2-1)


2
1
10
11
12
13
14
15
THREE PHASE SIGNAL GENERATOR
CPC-301
POWER
OUT
0/1
RESET
FUNCTION
RANGE
HI
LOW
FREQ
0600
AMPLITUDE
COARSE
FINE
OUTPUT
TRIM
1.5V RMS MAX FLOAT 1K
PHASE-C
PHASE-B
PHASE-A
CPCFPLG3.BMP
4
3
5
6
7
8
9

參.背板說明：(請參照 FIG 3-1)

(1)“A” 相背板輸出 BNC 插座，最大輸出：SIN 1500mV(Rms) RL=10KΩ。

(2)“B” 相背板輸出 BNC 插座，最大輸出：SIN 1500mV(Rms) RL=10KΩ。

(3)“C” 相背板輸出 BNC 插座，最大輸出：SIN 1500mV(Rms) RL=10KΩ。

(4)接地端子　　：本機外殼接地端子。

(5)本機電源保險絲座　：保險絲容量為 110V 1A、220V 0.5A 速斷型。

(6)電源電壓選擇開關　：電源電壓選擇開關。

(7)本機電源插座　　：請接至適合本機電壓電源插座。

(8)A 相控制線插座　　：9PIN D 型接頭(請勿與 RS232 混用)

(9)B 相控制線插座　　：9PIN D 型接頭(請勿與 RS232 混用)

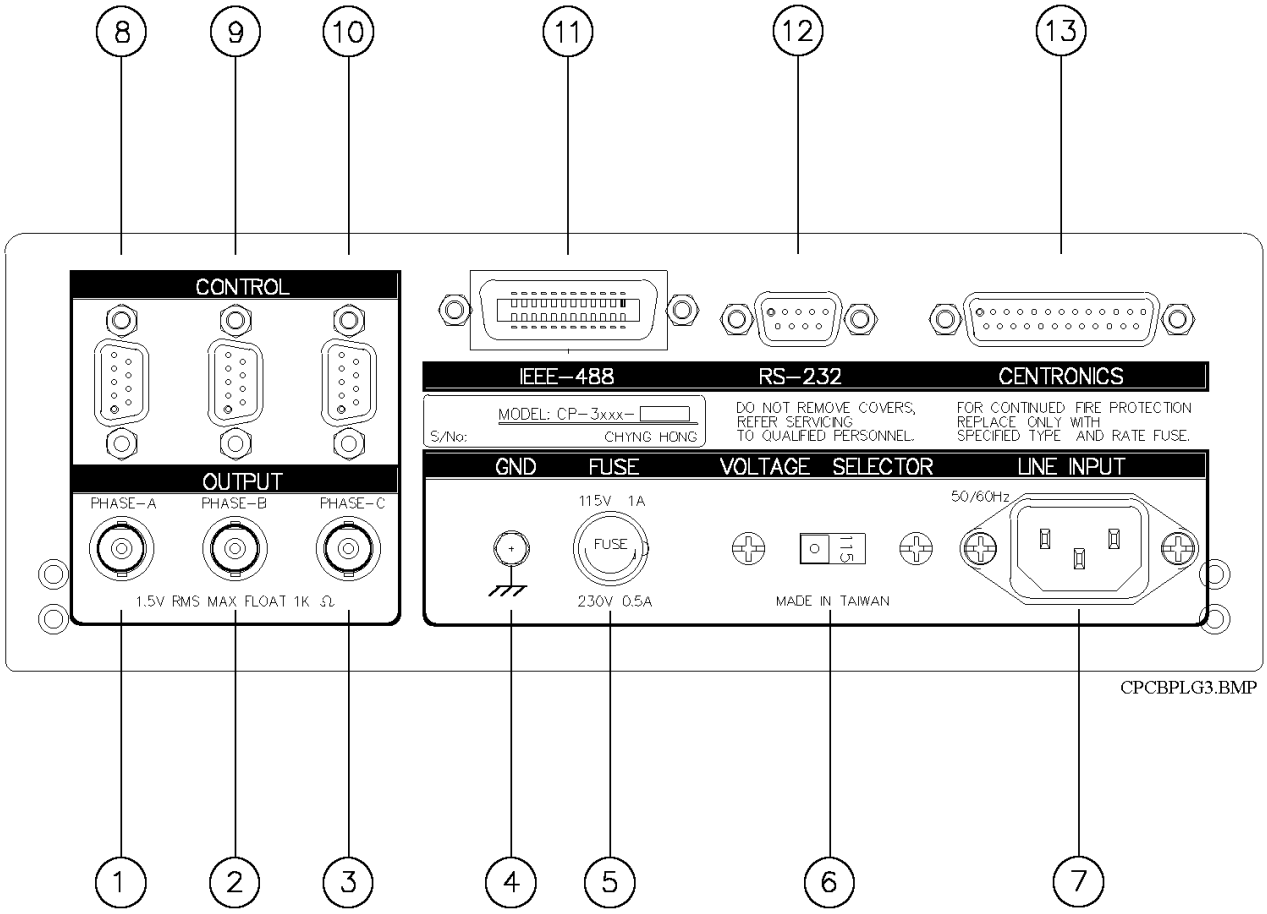
(10)C 相控制線插座　：9PIN D 型接頭(請勿與 RS232 混用)

(11) GPIB 插座　　：IEEE-488 GPIB 插座

(12) RS232 插座　　：RS232 插座(請勿與各相控制線混用)

(13) 預留插座

(FIG 3-1)



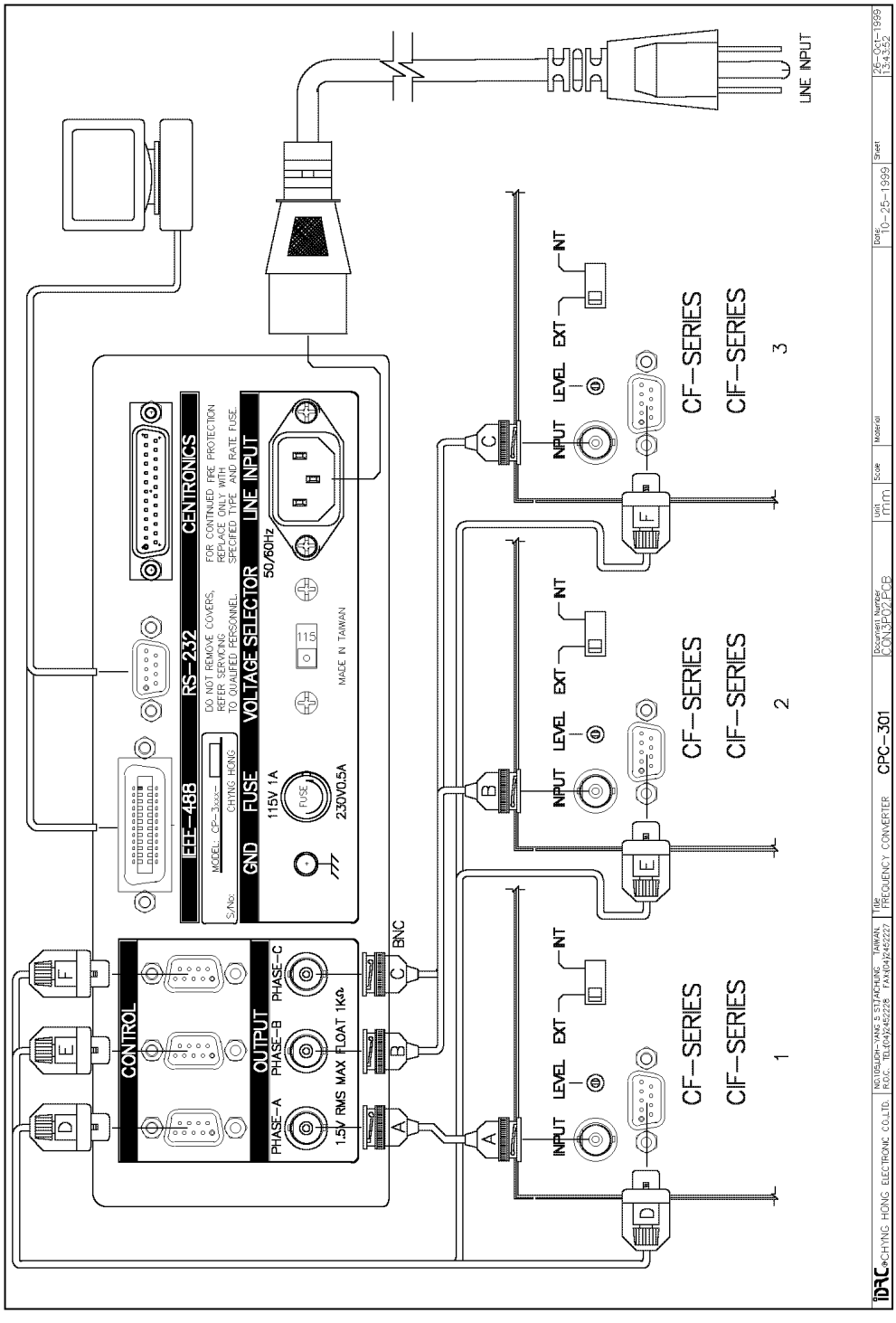
CPCBPLG3.BMP

肆.信號接線圖說明：(請參照 FIG 4-1)

(1)三相控制器下方為 AC POWER SOURCE，即交流電源供應器之後板信號輸入端部份。

(2)連接時，請將 AC POWER SOURCE 背板開關切至 EXT 處，連結之信號線請用 BNC 插頭之單隔離線。

(FIG 4-1)



伍.操作程序

1. 將所有電源及輸出開關置於 OFF 狀態。

2. 檢查 AC SOURCE 後板外部輸入信號選擇開關是否切至 EXT 上。

3. 將輸出端子台 3Ø 短路線需短路。(將 N 極全部接在一起)

4. 確認本機輸入 3Ø220V 電源輸入開關後，接上電源。

5. 開啟本機總電源輸入開關。

6. 先將 CPC-301(三相控制器)電源開啟。

7. 再開啟每一台電源無溶絲開關；而後打開電源開關，之後再開啟 OUTPUT 開關。

8. 確認每一台電壓檔位 RANGE 無誤，若大於 150V 請選擇 300V 檔。

9. 調整 CPC-301(三相控制器)V-ADJ，並檢視各 AC SOURCE 表頭顯示電壓，約過 15V 後 AC SOURCE 各表頭頻率開始顯示，此三台 AC SOURCE 顯示之頻率應一致。

10. 調至所需電壓時，若三台電壓不一致，請用小螺絲起子調整 CPC-301(三相控制器)面板 LEVEL 微調 VR，使三相輸出電壓一致。

11. 接上負載後，將每一台輸出無熔絲關打開；每單一台開啟後再開啟本機總輸出電源開關

12. 關閉時先將受測物及輸出端子座間之無熔絲開關關閉，再關掉所有機器之電源。

注意：為避免在開關各 AC SOURCE 之電源，而發生欠相問題；所以請注意主輸出無熔絲開關之開關時機。

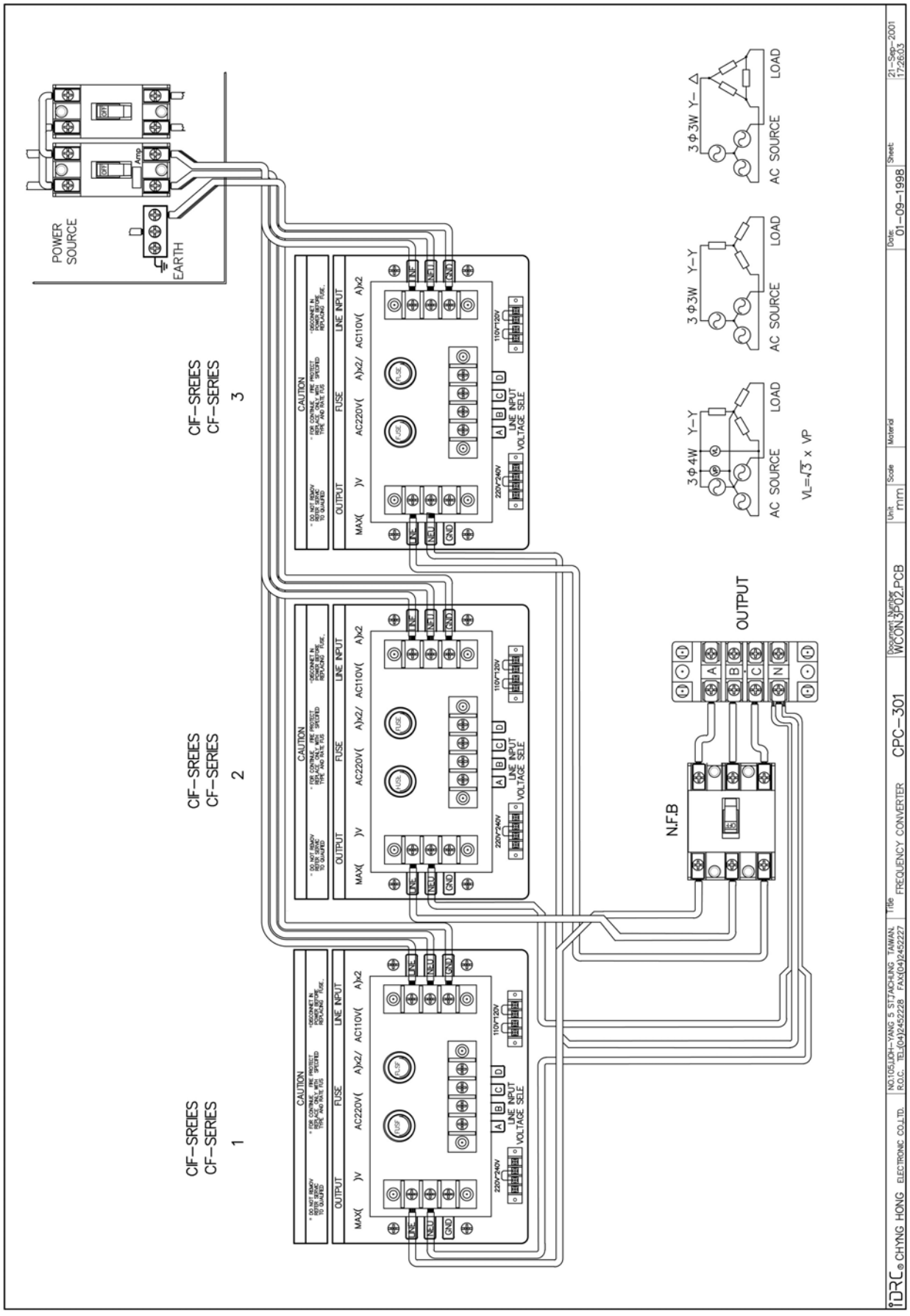
陸.三相控制器和交流電源供應器應用之電源及輸出接線圖：(請參照 FIG 6-1~FIG 6-6)

1.電源線及輸出線線徑請參考附錄壹、附錄貳配置。

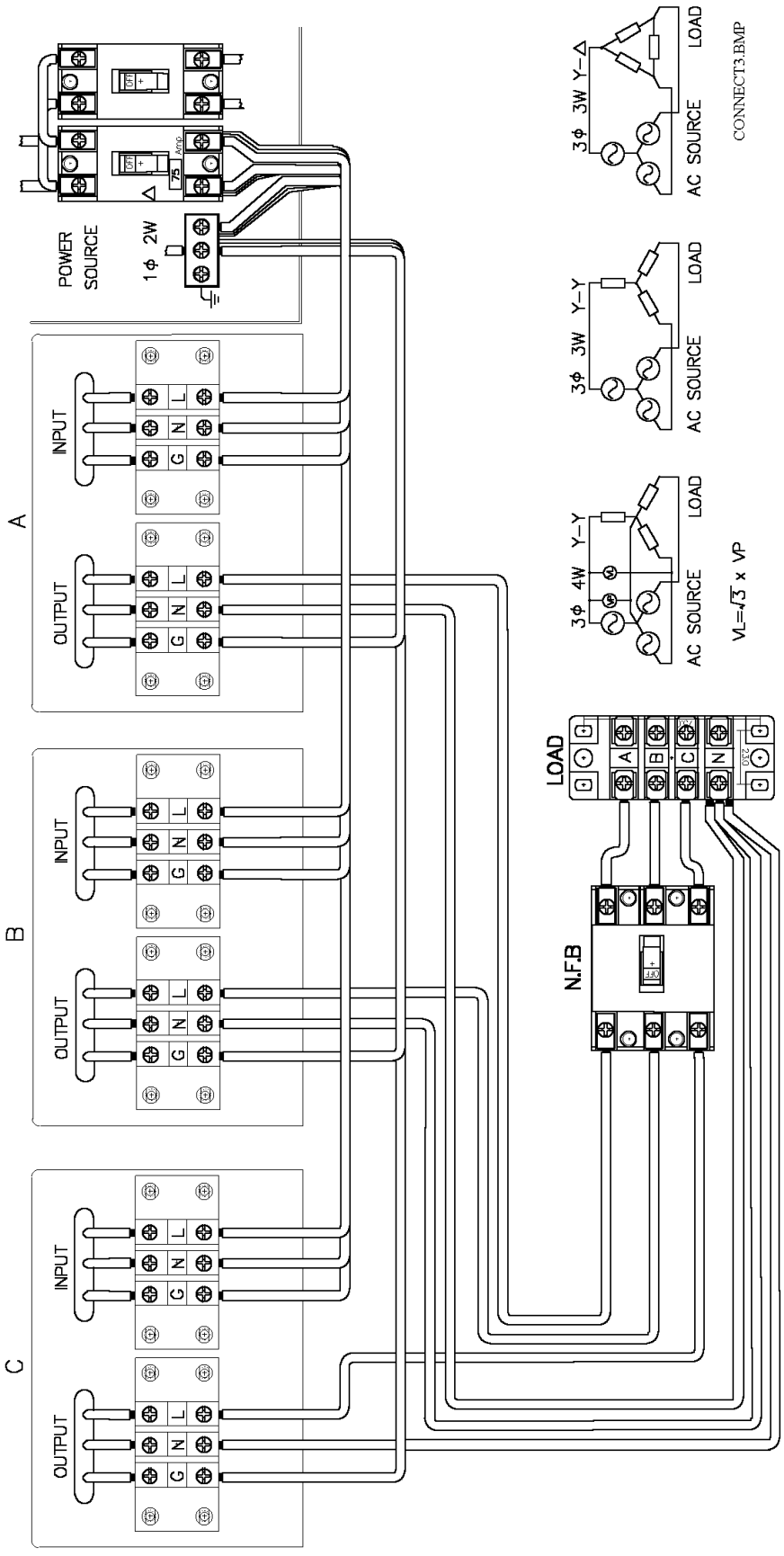
2.輸出及受測產品間應加三相無熔絲開關做測試電源之關閉或啟用（參考 □ FIG 6-1 □ FIG 6-2 □ FIG 6-3 □ FIG 6-4 □ FIG 6-5 □ FIG 6-6）。

3.確認配線無誤及電源電壓無誤後再送電。

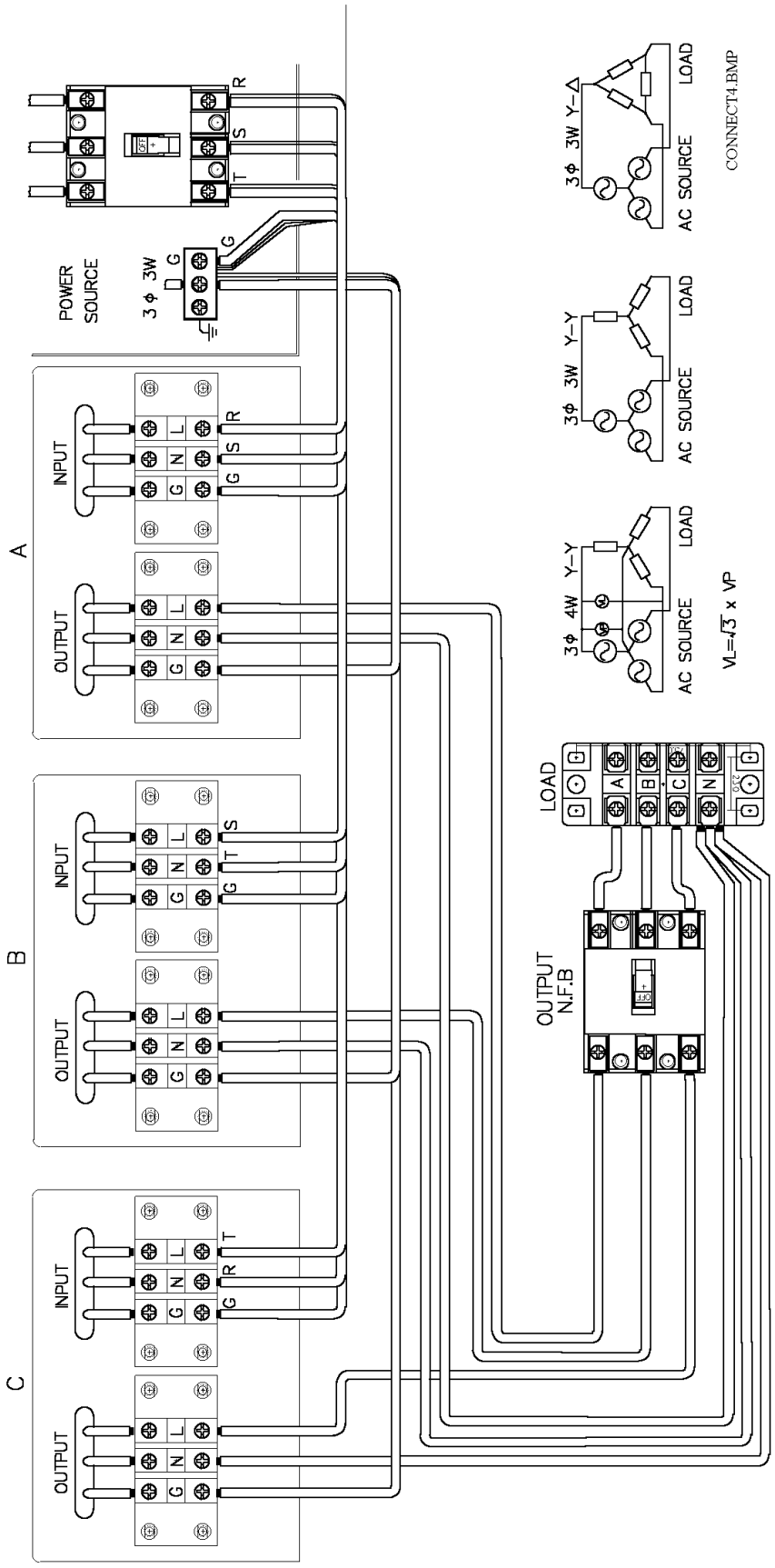
(FIG 6-1)


POWER SOURCE
EARTH
CIF-SREIES
CF-SERIES
1
CIF-SREIES
CF-SERIES
2
CIF-SREIES
CF-SERIES
3
N.F.B
OUTPUT
3 Φ 4W Y-Y
AC SOURCE
LOAD
3 Φ 3W Y-Y
AC SOURCE
LOAD
3 Φ 3W Y- △
AC SOURCE
LOAD
VL=√3 x VP
CHYNG HONG ELECTRONIC CO.,LTD.
NO.105,JUDH-YANG 5 ST.,TACHUNG TAIWAN, R.O.C. TEL:(04)2452228 FAX(04)2452227
Title FREQUENCY CONVERTER CPC-301
Document Number WCON3P02.PCB
Unit mm
Scale
Material
Date: 01-09-1998
Sheet:
21-Sep-2001 17:26:03

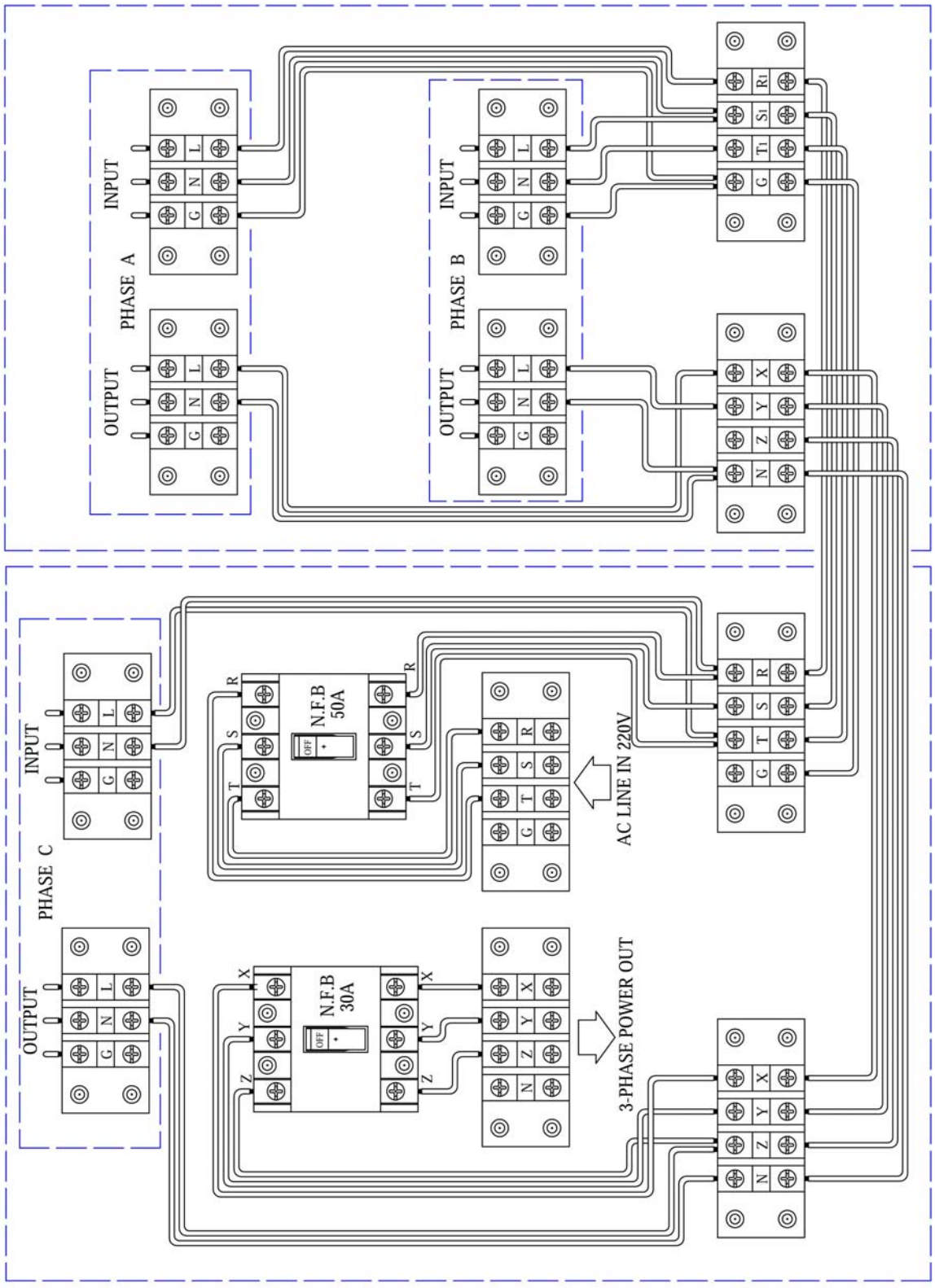
(FIG 6-2)


POWER SOURCE
1φ 2W
A
B
C
INPUT
OUTPUT
L
N
G
LOAD
A
B
C
N
N.F.B
3φ 3W Y-△
3φ 3W Y-Y
3φ 4W Y-Y
AC SOURCE
LOAD
VL=√3 x VP
CONNECT3.BMP

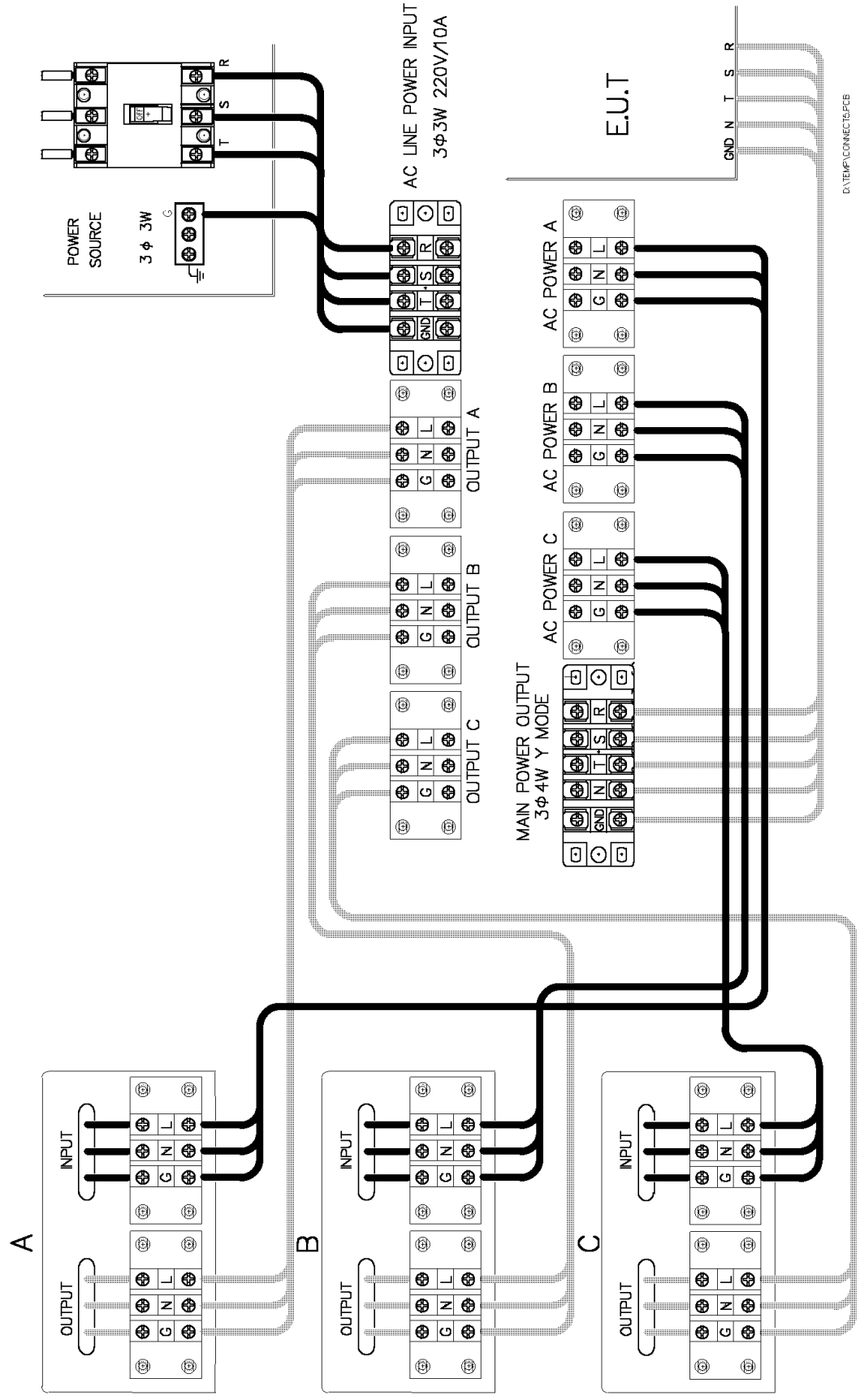
(FIG 6-3)


POWER SOURCE
3φ 3W
R
S
T
G
A
B
C
INPUT
OUTPUT
L
N
G
OUTPUT N.F.B
LOAD
A
B
C
N
3φ 3W Y-Δ
3φ 3W Y-Y
3φ 4W Y-Y
AC SOURCE
LOAD
VL=√3 x VP
CONNECT4.BMP

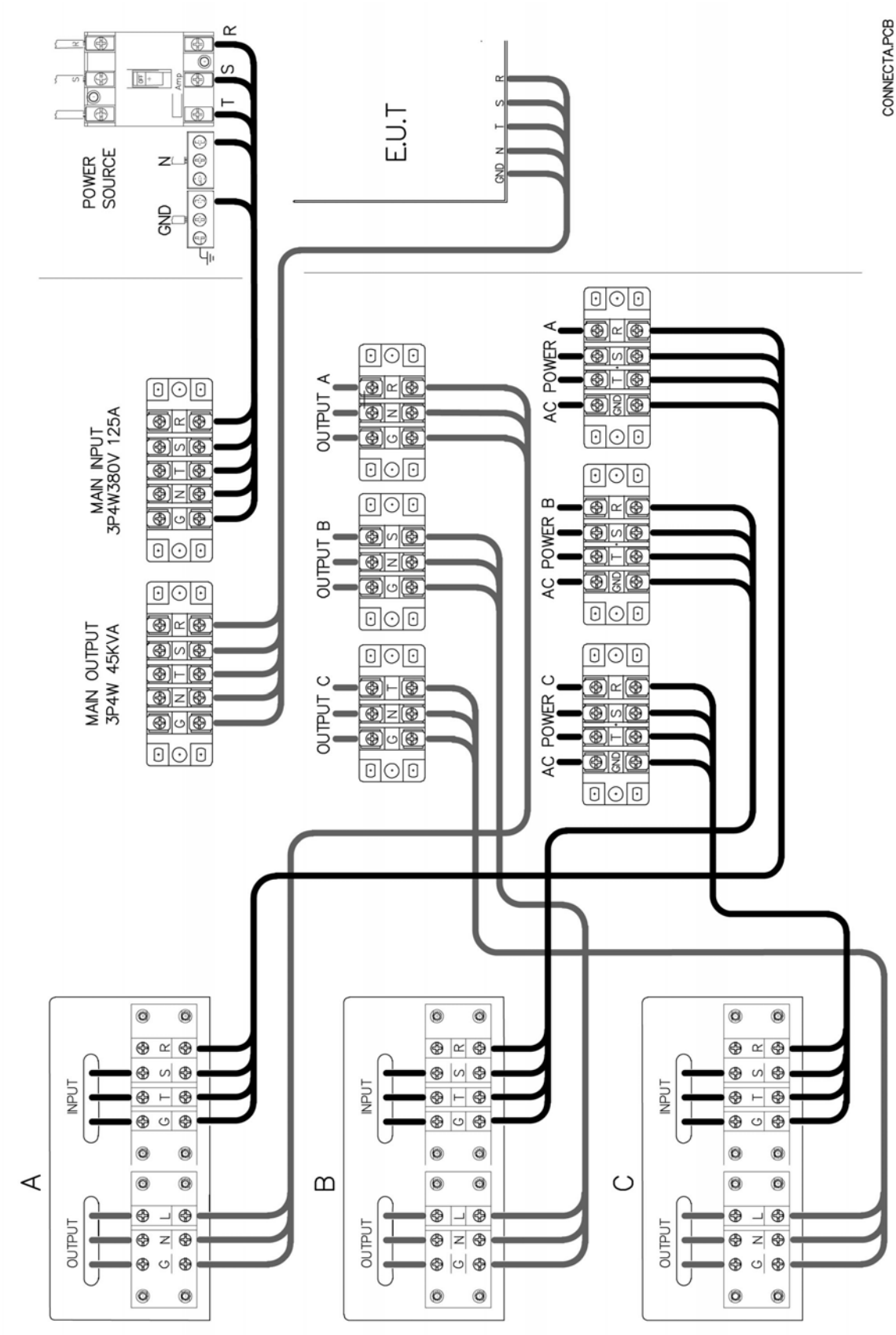
(FIG 6-4)


INPUT
PHASE A
OUTPUT
INPUT
PHASE B
OUTPUT
INPUT
PHASE C
OUTPUT
N.F.B 50A
N.F.B 30A
OFF
AC LINE IN 220V
3-PHASE POWER OUT
AUTOCAD R14\PWSWU\CPC301\CON3P21 10-29-1999

(FIG 6-5)


POWER SOURCE
3 φ 3W
R
S
T
AC LINE POWER INPUT
3φ3W 220V/10A
GND
OUTPUT A
OUTPUT B
OUTPUT C
MAIN POWER OUTPUT
3φ4W Y MODE
AC POWER C
AC POWER B
AC POWER A
E.U.T
GND N T S R
D:\TEMP\CONNECT5.PCB
A
B
C
INPUT
OUTPUT
L
N
G

(FIG 6-6)


POWER SOURCE
N
GND
R
S
T
E.U.T
GND N T S R
CONNECTA.PCB
MAIN INPUT 3P4W380V 125A
MAIN OUTPUT 3P4W 45KVA
OUTPUT A
OUTPUT B
OUTPUT C
AC POWER A
AC POWER B
AC POWER C
A
B
C
INPUT
OUTPUT
G N L
G N T S R

附錄壹：配線說明

一、CIF 系列交流電源供應器配線說明

機 型	輸入 電壓	輸入 最大電流	輸入 N.F.B	輸入 使用線徑	輸出 最大電流（ Rms ）	輸出 使用線徑
CIF-2000					0-140V 16.6A	
2KVA	1Ø2W 220V	18 A	20A	3.5mm²	0-280V 8.3A	3.5mm²
CIF-3000					0-140V 25A	
3KVA	1Ø2W 220V	27.2A	30A	5.5mm²	0-280V 12.5A	5.5mm²
CIF-50000	1Ø2W 220V	45.5 A	50A	8 mm²	0-140V 41.6A	
5KVA	3Ø3W 220V	30.3A	40A	5.5mm²	0-280V 20.8A	8.0mm²
CIF-7500	1Ø2W 220V	68A	75A	14mm²	0-140V 62.5A	
7.5KVA	3Ø3W 220V	45.3 A	50A	8 mm²	0-280V 31.2A	14mm²
CIF-1030	1Ø2W 220V	90 A	100 A	22mm²	0-140V 83.3A	
10KVA	3Ø3W 220V	60 A	75A	14mm²	0-280V 41.6A	22mm²
CIF-1530					0-140V 125A	
15KVA	3Ø3W 220V	90A	100 A	22mm²	0-280V 67.5A	38mm²
CIF-2030					0-140V 167A	
20KVA	3Ø3W 220V	121.2 A	125 A	38mm²	0-280V 83.5A	60mm²
CIF-3030					0-140V 250A	
30KVA	3Ø3W 220V	181 A	200 A	38mm²X2	0-280V 125A	60mm²

註：(1)配線建議採用多心絞線。

(2)配線時，導線應對絞。若導線超過 3 公尺時應再加粗一級，例：原 3.5mm²改為 5.5mm² 。

(3)N.F.B.：無熔絲斷路器。

(4)線材及無熔絲斷路器，請使用優良品牌。

(5)若輸出常需 0-150V 或 0-300V 切換使用時，導線應使用 0-150V 設定之導線線徑。

(6)2000VA 以上輸入電源固定為 AC 220V，如需改為 AC 110V 時，請另行訂製。

二、CF 系列交流電源供應器配線說明

機　　型	輸　　入				輸　　出	
	電源	最大電流	建議使用無熔絲開關	建議使用線徑	最大電流 (Rms)	建議使用線徑
CF-500A 500VA	1Ø2W 110V	15A	15A	2.0mm²	0-150V 4.2A 0-300V 2.1A	2.0mm²
	1Ø2W 220V	7.5A	7.5A	2.0mm²		
CF-1000A 1KVA	1Ø2W 110V	30A	30A	5.5mm²	0-150V 8.4A 0-300V 4.2A	2.0mm²
	1Ø2W 220V	15A	15A	2.0mm²		
CF-2000A 2KVA	1Ø2W 220V	30A	30A	5.5mm²	0-150V 16.8A 0-300V 8.4A	2.0mm²
CF-3000A 3KVA	1Ø2W 220V	45A	50A	8mm²	0-150V 25.2A	3.5mm²
					0-300V 12.6A	2.0mm²
CF-5000A 5KVA	1Ø2W 220V	75A	75A	14mm²	0-150V 43A	8mm²
					0-300V 21.5A	3.5mm²

註：(1)配線建議採用多心絞線。

(2)配線時，導線應對絞。若導線超過 3 公尺時應再加粗一級，例：原 3.5mm²改為 5.5mm² 。

(3)N.F.B.：無熔絲斷路器。

(4)線材及無熔絲斷路器，請使用優良品牌。

(5)若輸出常需 0-150V 或 0-300V 切換使用時，導線應使用 0-150V 設定之導線線徑。

(6)2000VA 以上輸入電源固定為 AC 220V，如需改為 AC 110V 時，請另行訂製。

附錄貳：導線線徑與電流規格表

※請注意！線材規格請依下列表格，方能正常使用。

使用 CD-SERIES 直流電源供應器或 CF、CIF 交流電源供應器時，需特別注意輸入與輸出導線之線徑問題，以防止因電流太大引起過熱，而造成意外，下列表格為導線在不同溫度下之線徑與電流規格表。

AWG NO	線徑 (約略值)	銅線溫度			
		60℃	75℃	85℃	90℃
		電流(A)			
14	2mm²	20	20	25	25
12	3.5mm²	25	25	30	30
10	5.5mm²	30	35	40	40
8	8mm²	40	50	55	55
6	14mm²	55	65	70	75
4	22mm²	70	85	95	95
3	30mm²	85	100	110	110
2	38mm²	95	115	125	130
1	50mm²	110	130	145	150
0	60mm²	125	150	165	170
00	70mm²	145	175	190	195
000	80mm²	165	200	215	225
0000	100mm²	195	230	250	260

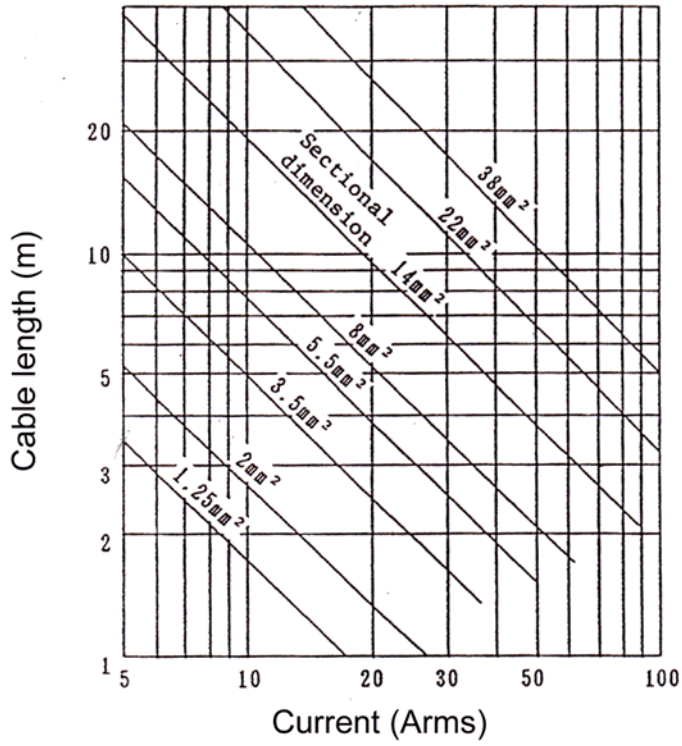
導線的阻抗與其長度成正比與線徑成反比，並且直接影響 CD-SERIES 直流電源供應器或 CF、CIF 交流電源供應器的輸出特性；所以，往往在輸出端子上所量測出來的電壓不同於負載上的電壓，一般而言，這個電位差不得大於 0.5V(參考圖 A3-1)。**備註：當電位差大於 0.5V 時，可將線徑加粗 1 倍或 2 倍甚至 3 倍。**

下列表格為導線線徑每公尺所產生之阻抗規格表。

AWG NO	Resistance in mΩ/M (at 20℃)
22	52.8
20	33.5
18	20.96
16	13.19
14	8.30
12	5.22
10	3.277

FIG A3-1 (Cable length and voltage drop)

Condition：Voltage drop 0.5V　　JIS C3307 IV Cable



附錄參：GPIB、RS-232 界面指令說明

1. IEEE488.2 Interface

- Specification : Standard IEE488.2
- Function :
  - 1.SH1: Full Source Handshake
  - 2.Ah1: Full Acceptor Handshake
  - 3.T6: Basic Talker
  - 4.L4: Basic Listener
  - 5.SR0: Without Service Request
  - 6.RL1: Remote/Local Change
  - 7.PR0: Without Parallel Polling
  - 8.DC1: Device Clear
  - 9.DT0: Without Device Trigger
  - 10.C0: Without Controller function
- Command :

*CLS	Clear the status byte summary register and all event registers .
*ESE	<Enable Value> Enable bits in the standard Event status enable register.
*ESE?	Query standard Event enable register.
*ESR?	Query standard Event register.
*IDN	Identification query.
*OPC	Sets the “operation complete” bit(bit 0) in the Standard Event.
*OPC	Return “1” to the output buffer after the command is executed.
*RST	Reset the instrucment to its power-on configuration.
*SRE	<Enable Value> Enable bits in the Status Byte enable register.
*SRE?	Query the Status Byte enable register.
*STB?	Read status byte query.
*TST?	Perform a complete self-test of instrument . Return “0” if the self-test is successful , or “1” if it test fails.

2.SCPI Command

- OUTPut

  :OUTPut {ON/OFF/1/0}

  :OUTPut:STATe {ON/OFF/1/0}

  Sets the output of the AC source.

- SOURce

  :SOURce:FREQuency <NRf>

  Sets or query the output signal frequency.

  :SOURce:RANGe {HIGH|LOW|AUTO}

  Sets or query the output range.

  :SOURce:SPPHase <NRf>

  Sets or query the stop phase.

  :SOURce:STPHase <NRf>

  Sets or query Start phase.

  :SOURce:TURN {ON/OFF/1/0}

  Sets or Query the output signal status.

  :SOURce:VOLTage <NRf>

  Sets or query the output voltage.

- STATus

  :PRESent

  Clears the Questionable status register.both the event and enable registers are cleared.

  :QUEStionable:CONDition?

  Query the contents of condition register of Questionable status register group.

  :QUEStionable:ENABle

  Sets or queries the enable register of Questionable status register group.

  :QUEStionable[:EVENt]?

  Query the contents of event register of Questionable status register group.

- SYSTem

  :SYSTem:BEEPer

  :SYSTem:BEEPer:IMMediate

  :SYSTem:BEEPer:STATe {ON/OFF/1/0}

  Sets the beeper to ON/OFF,queries the current setting.

  :SYSTem:ERRor? (Query Only)

  Queries the occurred error code and message.

  :SYSTem:KLOCk

  Sets or queries whether the front-panel keys are locked.

:SYSTem:PRESet(No Query)

Resets to the default state.

:SYSTem:VERSion

Queries the value corresponding to the SCPI version to which the instrucment complies

3.RS-232 Interface

- Mode : Start stop synchronization
- Baud rate : 1200.2400.4800.9600bps
- Command :

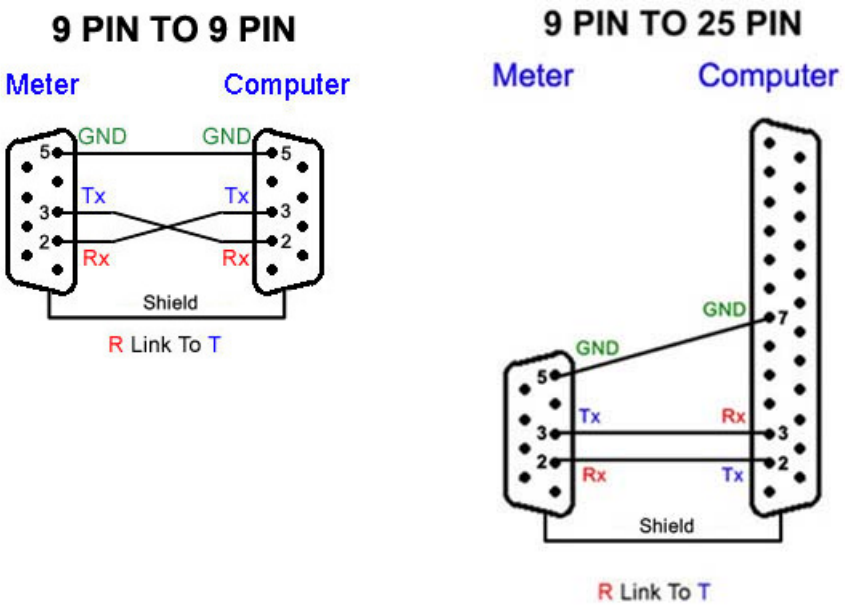
:SYSTem:LOCal(No Query)

Sets the system to local control.

:SYSTem:REMote(No Query)

Sets the system to remote control.

- RS-232 PC To CF/CIF AC SOURCE Connection



4.SCPI Status System

- Questionable Data

Event Register .or. Enable Register

Bit	Condition	
10	FAULT	:System Fault.
9	FUSE	:Fuse Break.
8	ODP	: Over Drive Protection.
3	OTP	:Over Temperature Protection.
1	OLP	:Over Load Protection.

• Standard Event
Event Register .or. Enable Register

Bit	Condition
7	Not used
6	Not used
5	Command Error
4	Not used
3	Not used
2	Query Error
1	Not used
0	Operation Complete

• Status Byte
Summary Register .or. Enable Register

Bit	Conditon
7	Operation Data
6	Request Service
5	Standard Event
4	Message Avilable
3	Questionable Data
2	
1	
0	

附錄肆.錯誤碼對照表

Code	Message	Code	Message
0	No Error		
-100	Command error	-330	Self-test failed
-101	Invalid character	-350	Queue overflow
-102	Syntax error	-400	Query errors
-103	Invalid separator	-410	Query INTERRUPTED
-104	Data type error	-420	Query UNTERMINATED
-105	GET not allowed	-430	Query DEADLOCKED
-108	Parameter not allowed	-440	Query UNTERMINATED after indefinite response
-109	Missing parameter		
-112	Program mnemonic too long		
		60	ROM FAILED"
-113	Undefined header	61	RAM FAILED"
-121	Invalid character in number	62	EEPROM R/W FAILED"
-123	Exponent too large	63	USER SETTING LOST"
-124	too many digits	64	Command allowed only with RS-232
-128	Numeric data not allowed	65	Command not allowed in local
-131	Invaild suffix	100	FAULT
-138	Suffix not allowed	101	OLP
-140	Character data error	102	OTP
-141	Invalid character data	103	ODP
-144	Character data too long	104	FUSE
-148	Character data not allowed		
-150	String data error		
-151	Invalid string data		
-158	String data not allowed		
-160	Block data error		
-161	Invalid block data		
-168	Block data not allowed		
-170	Expression error		
-171	Invalid expression		
-178	Expression data not allowed		
-200	Execution errors		
-211	Trigger ignored		
-213	Init ignored		
-221	Setting conflict		
-222	Data out of rang		
-223	Too much data		
-224	Illegal parameter valve		
-230	Data corrupt or stale		
-241	Hardware missing		
-310	System error		
-311	Memory error		
-313	Calibration memory lost		

附錄伍.RS232 鮑率與 GPIB 位址設定法

1.關閉電源開關(POWER)

2.頻率指撥開關撥於 0069

3.按著 RESET 開關同時開啟電源開關(POWER)

4.以上步驟若操作正確則 REMOT 燈亮，表示已進入設定模式

5.RS232 鮑率設定

a.指撥開關與鮑率關係如下：

0100：1200

0101：2400

0102：4800

0103：9600

b.依上列所示選定所需鮑率來設定指撥開關，如：9600 為 0103

c.按 RESET 鍵完成輸入動作

d.撥指撥開關於 0300 位置

e.按 RESET 鍵完成儲存動作，並跳出設定模式進入正常操作模式

f.將指撥開關撥於所需輸出電源頻率，如：50.00

6.GPIB 位址設定：

a.指撥開關與位址關係如下：

0200：00

0201：01

0231：31

b.依上列所示選定所需位址，來設定指撥開關，如 0206 表示設定位址為 06

c.按 RESET 鍵完成輸入動作

d.調指撥開關於 0300 位置

e.按 RESET 鍵完成儲存動作，並跳出設定模式進入正常操作模式

f.將指撥開關設於所需輸出電源頻率，如：50.00

註：GPIB 內定 Address 為 06

RS232 內定鮑率為 9600

附錄陸.程式範例

範例 1：GPIB demo Program

```
REM Example Program using NI-488 board level functions
REM QBDECL.BAS contains constants, declarations, and subroutine prototypes.
REM $INCLUDE: 'qbdecl.bas'
REM GPIBERR is an error subroutine that is called when a NI-488 function fails.
REM DVMERR is an error subroutine that is called when the IDRC CP-600 POWER
REM ANALYZER does not have valid data to send.
    DECLARE SUB gpiberr (msg$)
    DECLARE SUB dvmerr (msg$, rd$)
    CLS
    LOCATE 2, 3
    PRINT "CFxx Series demo Program ....."
    PRINT
    bdname$ = "GPIB0"
    CALL ibfind(bdname$, brd0%)
    IF brd0% < 0 THEN CALL gpiberr("Ibfind Error")
REM Send the Interface Clear (IFC) message.
    CALL ibsic(brd0%)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibsic Error")
REM Turn on the Remote Enable (REN) signal.
    CALL ibsre(brd0%, 1)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibsre Error")
REM Inhibit front panel control with the Local Lockout (LLO) command (hex 11).
REM Place the IDRC CP-600 in remote mode by addressing it to listen (ASCII "&").
REM Send the Device Clear (DCL) message to clear device (hex 14).
REM Address the GPIB interface board to talk (hex 40 or ASCII "@").
    cmd$ = CHR$(&H11) + "&" + CHR$(&H14) + "@"
    CALL ibcmd(brd0%, cmd$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibcmd Error")
REM set range to 150V
    wrt$ = ":SOUR:RANG LOW;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
REM set voltage to 100V
    wrt$ = ":SOUR:VOLT 100;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
REM set frequency to 60Hz
    wrt$ = ":SOUR:FREQ 60;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
REM set start phase
    wrt$ = ":SOUR:STPH   0;"
    CALL ibwrt(brd0%, wrt$)
```

```
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
REM set stop phase
    wrt$ = ":SOUR:SPPH    0;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
REM output relay on
    wrt$ = ":OUTP ON;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
REM turn on signal
    wrt$ = ":SOUR:TURN ON;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
    SLEEP (3)
REM set voltage to 120V
    wrt$ = ":SOUR:VOLT 120;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
   SLEEP (3)
REM set frequency to 50Hz
    wrt$ = ":SOUR:FREQ 50;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
    SLEEP (3)
REM set voltage to 90V
    wrt$ = ":SOUR:VOLT 90;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
    SLEEP (3)
REM turn on signal
    wrt$ = ":SOUR:TURN OFF;"
    CALL ibwrt(brd0%, wrt$)
    IF (ibsta% AND EERR) THEN CALL gpiberr("Ibwrt Error")
    SLEEP (3)
REM Call the IBONL function to disable the hardware and software.
    CALL ibonl(brd0%, 0)
END

SUB dvmerr (msg$, rd$) STATIC
    PRINT msg$
    PRINT "Status Byte = "; rd$
REM Call the IBONL function to disable the hardware and software.
    CALL ibonl(brd0%, 0)
    STOP
END SUB

SUB gpiberr (msg$) STATIC
    PRINT msg$
    PRINT "ibsta = &H"; HEX$(ibsta%); " <";
```

```
 IF ibsta% AND EERR THEN PRINT " ERR";
    IF ibsta% AND TIMO THEN PRINT " TIMO";
    IF ibsta% AND EEND THEN PRINT " END";
    IF ibsta% AND SRQI THEN PRINT " SRQI";
    IF ibsta% AND RQS THEN PRINT " RQS";
    IF ibsta% AND SPOLL THEN PRINT " SPOLL";
    IF ibsta% AND EEVENT THEN PRINT " EVENT";
    IF ibsta% AND CMPL THEN PRINT " CMPL";
    IF ibsta% AND LOK THEN PRINT " LOK";
    IF ibsta% AND RREM THEN PRINT " REM";
    IF ibsta% AND CIC THEN PRINT " CIC";
    IF ibsta% AND AATN THEN PRINT " ATN";
    IF ibsta% AND TACS THEN PRINT " TACS";
    IF ibsta% AND LACS THEN PRINT " LACS";
    IF ibsta% AND DTAS THEN PRINT " DTAS";
    IF ibsta% AND DCAS THEN PRINT " DCAS";
    PRINT " >"
    PRINT "iberr = "; iberr%;
    IF iberr% = EDVR THEN PRINT " EDVR <DOS Error>"
    IF iberr% = ECIC THEN PRINT " ECIC <Not CIC>"
    IF iberr% = ENOL THEN PRINT " ENOL <No Listener>"
    IF iberr% = EADR THEN PRINT " EADR <Address error>"
    IF iberr% = EARG THEN PRINT " EARG <Invalid argument>"
    IF iberr% = ESAC THEN PRINT " ESAC <Not Sys Ctrlr>"
    IF iberr% = EABO THEN PRINT " EABO <Op. aborted>"
    IF iberr% = ENEB THEN PRINT " ENEB <No GPIB board>"
    IF iberr% = EOIP THEN PRINT " EOIP <Async I/O in prg>"
    IF iberr% = ECAP THEN PRINT " ECAP <No capability>"
    IF iberr% = EFSO THEN PRINT " EFSO <File sys. error>"
    IF iberr% = EBUS THEN PRINT " EBUS <Command error>"
    IF iberr% = ESTB THEN PRINT " ESTB <Status byte lost>"
    IF iberr% = ESRQ THEN PRINT " ESRQ <SRQ stuck on>"
    IF iberr% = ETAB THEN PRINT " ETAB <Table Overflow>"
    PRINT "ibcnt = "; ibcnt%
REM Call the IBONL function to disable the hardware and software.
    CALL ibonl(brd0%, 0)
    STOP
END SUB
```

範例 2：GPIB demo Program

```
#include <string.h>
#include "decl.h"

void gpiberr(char *msg);
void dvmerr(char *msg, char *rd);

   char        read[512];                  /* read data buffer                */
   int         bd,                         /* board or device number           */
               i;                          /* FOR loop counter                 */

void main() {

      clrscr();

      gotoxy(3,2);
      printf("CFxx Series demo Program .....");

      bd = ibfind ("GPIB0");
      if (bd < 0) gpiberr("ibfind Error");

/*    Send the Interface Clear (IFC) message.        */

      ibsic (bd);
      if (ibsta & ERR) gpiberr("ibsic Error");

/*    Turn on the Remote Enable (REN) signal.        */

      ibsre (bd,1);
      if (ibsta & ERR) gpiberr("ibsre Error");

/*
 *    Inhibit front panel control with the Local Lockout (LLO) command
 *    (hex 11).    Place the IDRC CP-600 in remote mode by addressing it to listen
 *    (hex 26 or ASCII "&").    Send the Device Clear (DCL) message to clear
 *    internal device functions (hex 14).    Address the GPIB interface board to
 *    talk (hex 40 or ASCII "@").
 */

      ibcmd (bd,"\021&\024@",4L);
      if (ibsta & ERR) gpiberr("ibcmd Error");

      /* set range to 150V                */
      ibwrt (bd,":SOUR:RANG LOW;", 15L);
      if (ibsta & ERR) gpiberr("ibwrt Error");
```

```
        /* set voltage to 100V              */
        ibwrt (bd,":SOUR:VOLT 100;", 15L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        /* set frequency to 60Hz            */
        ibwrt (bd,":SOUR:FREQ 60;", 14L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        /* set start phase                  */
        ibwrt (bd,":SOUR:STPH   0;", 14L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        /* set stop phase                   */
        ibwrt (bd,":SOUR:SPPH   0;", 14L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        /* output relay on                  */
        ibwrt (bd,":OUTP ON;", 9L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        /* turn on signal                   */
        ibwrt (bd,":SOUR:TURN ON;", 14L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        sleep(3);
        /* set voltage to 120V              */
        ibwrt (bd,":SOUR:VOLT 120;", 15L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        sleep(3);
        /* set frequency to 50Hz            */
        ibwrt (bd,":SOUR:FREQ 50;", 14L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        sleep(3);
        /* set voltage to 90V               */
        ibwrt (bd,":SOUR:VOLT 90;", 14L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        sleep(3);
        /* turn on signal                   */
        ibwrt (bd,":SOUR:TURN OFF;", 15L);
        if (ibsta & ERR) gpiberr("ibwrt Error");

        sleep(3);

    /*  Call the ibonl function to disable the hardware and software.       */
```

```
        ibonl (bd,0);

}

void gpiberr(char *msg) {

        printf ("%s\n", msg);

        printf ("ibsta = &H%x    <", ibsta);
        if (ibsta & ERR )    printf (" ERR");
        if (ibsta & TIMO)    printf (" TIMO");
        if (ibsta & END )    printf (" END");
        if (ibsta & SRQI)    printf (" SRQI");
        if (ibsta & RQS )    printf (" RQS");
        if (ibsta & SPOLL) printf (" SPOLL");
        if (ibsta & EVENT) printf (" EVENT");
        if (ibsta & CMPL)    printf (" CMPL");
        if (ibsta & LOK )    printf (" LOK");
        if (ibsta & REM )    printf (" REM");
        if (ibsta & CIC )    printf (" CIC");
        if (ibsta & ATN )    printf (" ATN");
        if (ibsta & TACS)    printf (" TACS");
        if (ibsta & LACS)    printf (" LACS");
        if (ibsta & DTAS)    printf (" DTAS");
        if (ibsta & DCAS)    printf (" DCAS");
        printf (" >\n");

        printf ("iberr = %d", iberr);
        if (iberr == EDVR) printf (" EDVR <DOS Error>\n");
        if (iberr == ECIC) printf (" ECIC <Not CIC>\n");
        if (iberr == ENOL) printf (" ENOL <No Listener>\n");
        if (iberr == EADR) printf (" EADR <Address error>\n");
        if (iberr == EARG) printf (" EARG <Invalid argument>\n");
        if (iberr == ESAC) printf (" ESAC <Not Sys Ctrlr>\n");
        if (iberr == EABO) printf (" EABO <Op. aborted>\n");
        if (iberr == ENEB) printf (" ENEB <No GPIB board>\n");
        if (iberr == EOIP) printf (" EOIP <Async I/O in prg>\n");
        if (iberr == ECAP) printf (" ECAP <No capability>\n");
        if (iberr == EFSO) printf (" EFSO <File sys. error>\n");
        if (iberr == EBUS) printf (" EBUS <Command error>\n");
        if (iberr == ESTB) printf (" ESTB <Status byte lost>\n");
        if (iberr == ESRQ) printf (" ESRQ <SRQ stuck on>\n");
        if (iberr == ETAB) printf (" ETAB <Table Overflow>\n");

        printf ("ibcnt = %d\n", ibcnt);
        printf ("\n");
```

```
/*    Call the ibonl function to disable the hardware and software.    */

      ibonl (bd,0);

      exit(1);

}

void dvmerr(char *msg,char *rd) {

      printf ("%s\n", msg);
      printf("Status byte = %x\n", rd[0]);

/*    Call the ibonl function to disable the hardware and software.    */

      ibonl (bd,0);

      exit(1);
}
```

範例 3：RS232 demo Program

```
CLS
LOCATE 2, 3
PRINT "CFxx Series demo Program ....."
OPEN "COM1:9600,N,8,1,RS,CS0,DS0,CD0" FOR RANDOM AS #1
COM(1) ON

PRINT #1, ":SYST:REM;"              ' set device to remote mode
PRINT #1, ":SOUR:RANG LOW;"      ' set range to 150V
PRINT #1, ":SOUR:VOLT 100;"      ' set voltage to 100V
PRINT #1, ":SOUR:FREQ 60;"       ' set frequency to 60Hz
PRINT #1, ":SOUR:STPH   0;"      ' set start phase
PRINT #1, ":SOUR:SPPH   0;"      ' set stop phase
PRINT #1, ":OUTP ON;"               ' output relay on
PRINT #1, ":SOUR:TURN ON;"       ' turn on signal

SLEEP (3)
PRINT #1, ":SOUR:VOLT 120;"      ' set voltage to 120V

SLEEP (3)
PRINT #1, ":SOUR:FREQ 50;"       ' set frequency to 50Hz

SLEEP (3)
PRINT #1, ":SOUR:VOLT 90;"       ' set voltage to 90V

SLEEP (3)
PRINT #1, ":SOUR:TURN OFF;"      ' turn off signal

SLEEP (3)
PRINT #1, ":SYSTEM:LOC;"            ' set device to local mode
END
```

範例 4：RS232 demo Program

```
#include "bios.h"
#include "conio.h"
#include "stdio.h"
#include "dos.h"

char err_no=0;

main()
{
        unsigned char readbuf[100],*p;
        int i;

        clrscr();
        gotoxy(3,2);
        printf("CFxx Series demo Program .....");

        initial_sys();                  /* initial RS-232 */

        send(":SYSTEM:REM;");           /* set device to remote mode   */


        send(":SOUR:RANG LOW;");        /* set range to 150V              */
        send(":SOUR:VOLT 100;");        /* set voltage to 100V            */
        send(":SOUR:FREQ 60;");         /* set frequency to 60Hz          */
        send(":SOUR:STPH   0;");        /* set start phase                */
        send(":SOUR:SPPH   0;");        /* set stop phase                 */
        send(":OUTP ON;");              /* output relay on                */
        send(":SOUR:TURN ON;");         /* turn on signal                 */

        sleep(3);
        send(":SOUR:VOLT 120;");        /* set voltage to 120V            */

        sleep(3);
        send(":SOUR:FREQ 50;");         /* set frequency to 50Hz          */

        sleep(3);
        send(":SOUR:VOLT 90;");         /* set voltage to 90V             */


        sleep(3);
        send(":SOUR:TURN OFF;");        /* turn off signal                */

        sleep(3);
        send(":SYSTEM:LOC;");           /* set device to local mode     */
```

```
 }

initial_sys()
{
        outportb(0x3fb,0x80);
        outportb(0x3f8,0x0c);
        outportb(0x3f9,0x00);
        outportb(0x3fb,0x07);
        outportb(0x3fb,0x03);
};

send(unsigned char *p)
{
        unsigned status;
        int     i,j,k;

        j=strlen(p);
        for(i=0;i<j;i++)
                {
                for(k=0;k<10000;k++)
                        {
                         status=inportb(0x3fd);
                         if(status & 0x20)
                                {
                                 outportb(0x3f8,*p);
                                 p++;
                                 break;
                                 }
                        }

                if(k==10000)
                        {
                         err_no=1;
                         gotoxy(3,22);
                         printf("RS232 sending timeout.....");
                         break;
                         }
                }
}

read(unsigned char *p)
{
        unsigned char status,over;
        int     i,j;
        long int    k;

        over=0;
        while(1)
```

```
                    {
                    for(k=0;k<2000000;k++)
                            {
                             status=inportb(0x3fd);
                             if(status & 0x01)
                                    {
                                     *p=inportb(0x3f8);
                                     if(*p==0x0a)
                                         {
                                          *++p=0;
                                          over=1;
                                          }

                                     p++;
                                     break;
                                     }
                            }
                    if(over==1)
                            break;
                    if(k==2000000)
                            {
                             err_no=1;
                             gotoxy(3,22);
                             printf("RS232 reading timeout.....\n");
                             break;
                             }
                    }
}
```