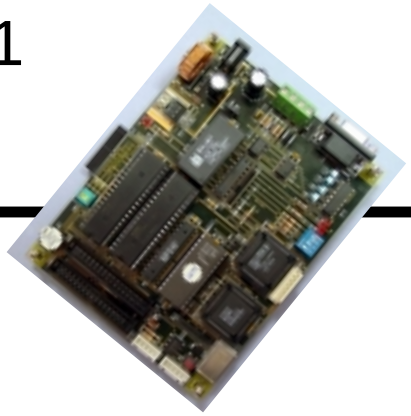
CHIPWARE51 使用說明書



前言

謝謝您聰明的選擇購買旗威科技有限公司的產品，對於您購買的產品，本公司提供一年的技術與維修服務，在這段期間若本產品故障或損壞時，請將其包裝妥當並註明故障原因，寄到高雄市三民區昌裕街18-1號，若是一般的故障我們只收取100元的郵件處理費，您可以在收到維修品後再用等值的郵票寄給我們，若是嚴重損壞時，我們會與您取得連繫並告知正確的維修費，經您同意後才進行維修。如果您對於本產品與產品相關的資訊還有任何問題時，請直接打電話(07-395-5152)或E-mail (service@chipware.com.tw)與我們連絡，您也可以隨時連上旗威科技(www.chipware.com.tw)的網站，立即獲得最新的產品與技術資料。

購買旗威科技的任何產品30天內，如果您已檢視過或使用過旗威科技的產品後，很可惜地發現本產品並不適合您的應用，或是其他理由無法再度使用該產品時，不論是否有無拆封或曾經安裝使用過，您可以不說明任何理由將本產品及購買時開立的發票退回旗威科技有限公司，我們會在收到包裹後的一週內將全額的購買費用退還給您。

旗威科技公司特別聲明:
爲了使本產品更安全及穩定，本公司有權利修改產品的硬體規格與軟體程式的內容，恕不另行通知，但使用者可透過www.chipware.com.tw取得最新版的控制程式與功能說明。

Printing History
本使用手册已經經過詳細的校對與檢查。若本使用手册的經過再版時，將包含了些微的修改和更新及一些已發行的相同資料。
二版---Feb, 2004

◎光碟內容 Contents

本轉換盒所附的光碟片內容如下:

1. 旗威網站www.chipware.com.tw的所有內容
2. CHIPWARE51下載程式
3. 本說明書的PDF說明檔
4. CHIPWARE51程式範例(Demo C)
5. 旗威科技在RS485的設備發展預定表Road Map (PDF檔)

◎請先檢查Check here

本控制板內包括以下硬體與附件，請在打開本包裝後立即檢查，若有缺少時請立刻與旗威科技公司(07-395-5152)連絡，我們會在最短的時間內將短缺的零件補寄給您。

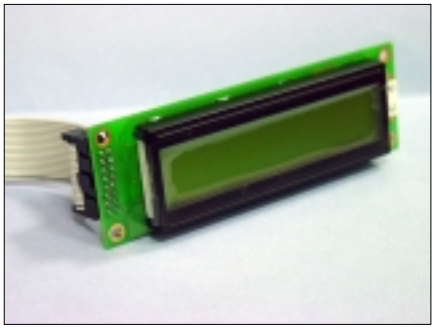
1. CHIPWARE51控制板
2. CHIPWARE51使用手册
3. CHIPWARE51使用者光碟

◎從這裡開始Quick start

1. 如果您有（20x2）的文字型LCD模組（選購，一組NT$600，含排線），
   請先接上CHIPWARE51
2. 開啓PC，安裝CHIPWARE51控制板下載程式
3. 啓動CHIPWARE51下載程式，並點選正確的ComPort
4. 將CHIPWARE51控制板與電腦連線（使用RS232一對一傳輸線），並通上DC7~12V電源
5. 檢視下載程式是否有回應值，如果沒有，請按CHIPWARE51上RESET鍵再行確認
6. 如有LCD模組，通電後應可顯示本公司資訊，若無則省略此步驟
7. 將自行撰寫之程式下載至CHIPWARE51進行驗證

LCD模組的接線方式：

如果您自己有LCD模組，配線時如圖所示，必須將排針焊接在LCD模組的背面，否則LCD會損壞唷！



請注意照片左邊16P排線是由LCD螢幕後方連接上的

◎Chipware51特點 Features

1.採用RISC-8051 CPU速度超快

2.使用CPLD2032做位址解碼，可以依實際需要修改IO位址

3.ROM區可達64K Bytes，採用27512 EPROM一枚

4.RAM區可達64K Bytes，採用62256 SRAM兩枚

5.兩枚8255可程式化週邊IC，提供48點數位輸出入埠

6.DS12C887 Real Time Clock提供系統時間及額外120 Bytes儲存空間

7.提供LCM模組驅動線路，可接20x2 LCM模組

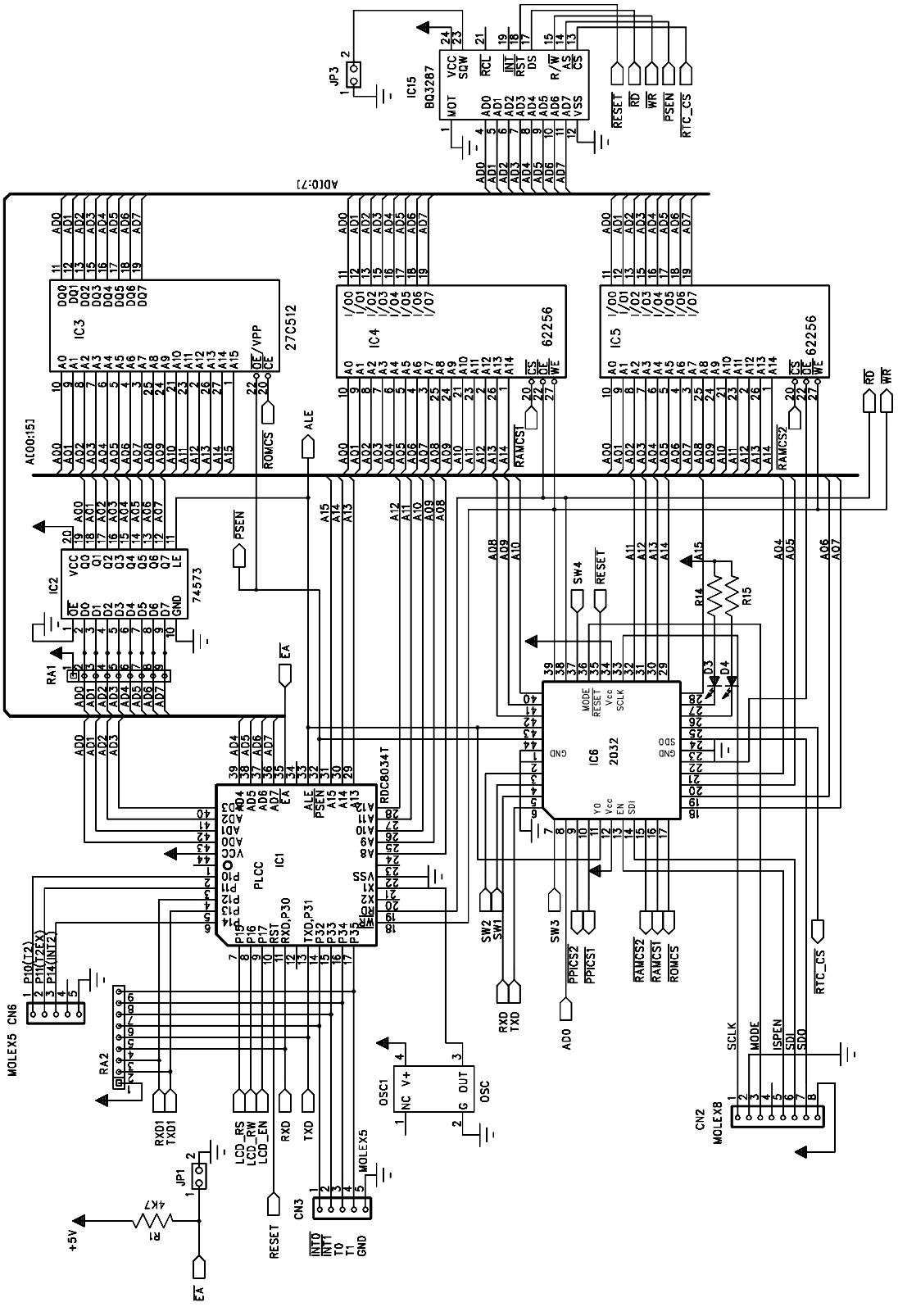
8.提供RS232與RS485串列通訊界面

9.輸入直流電源電壓可由DC7V到DC32V

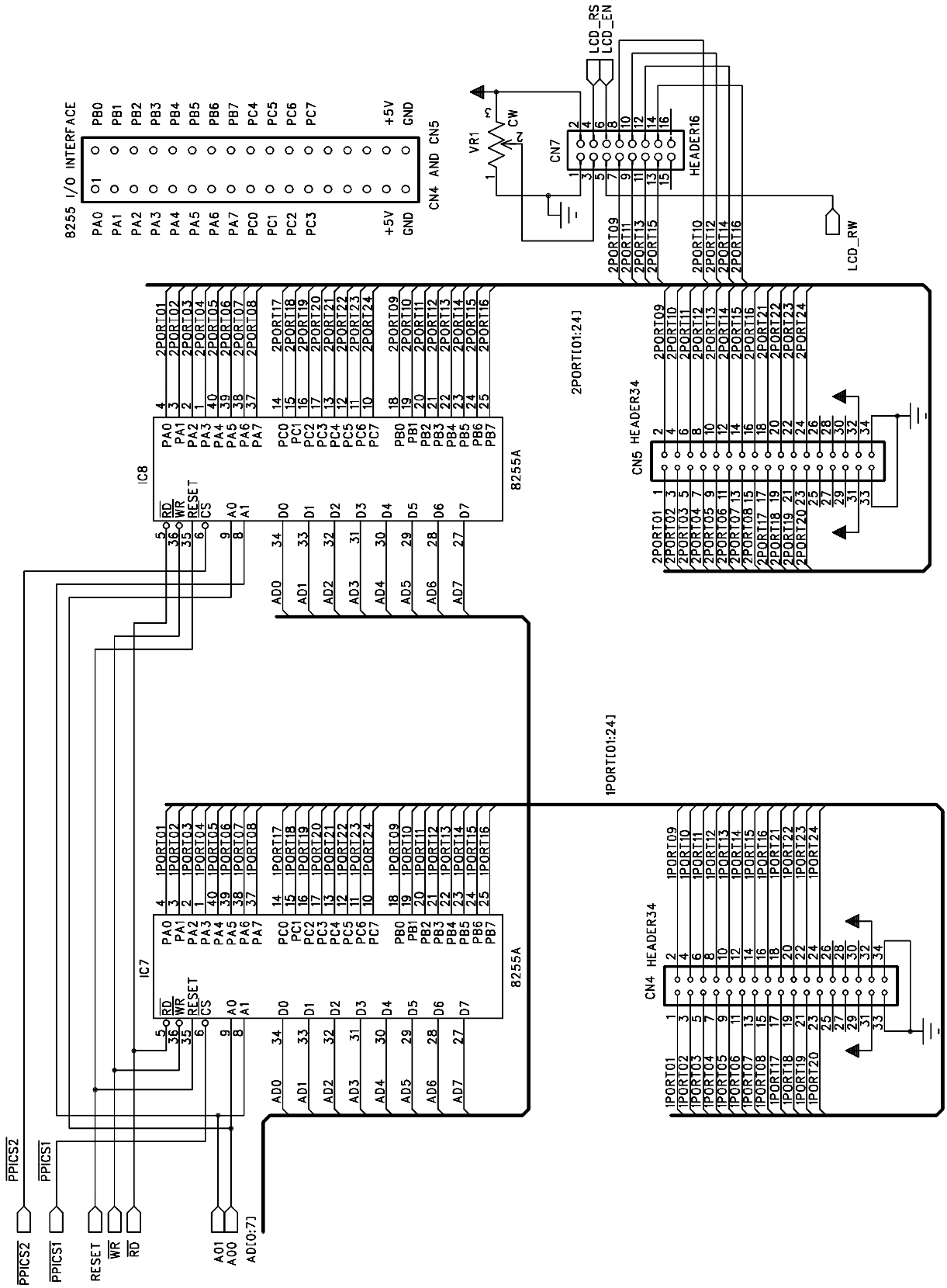
10.提供Download程式界面，使用者可以將程式傳到SRAM區進行模擬

◎Chipware51線路圖

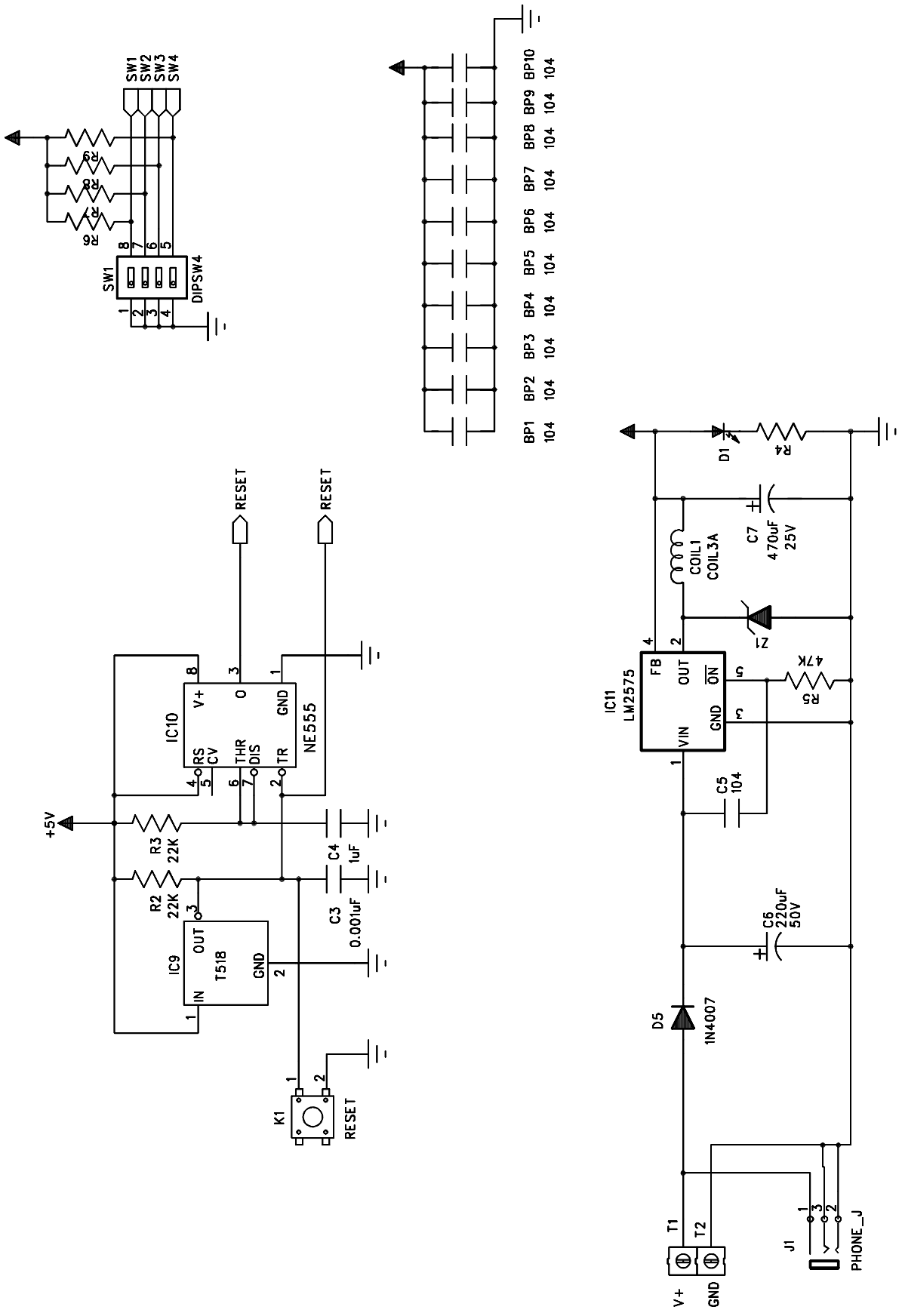
CPU與MEMORY線路圖



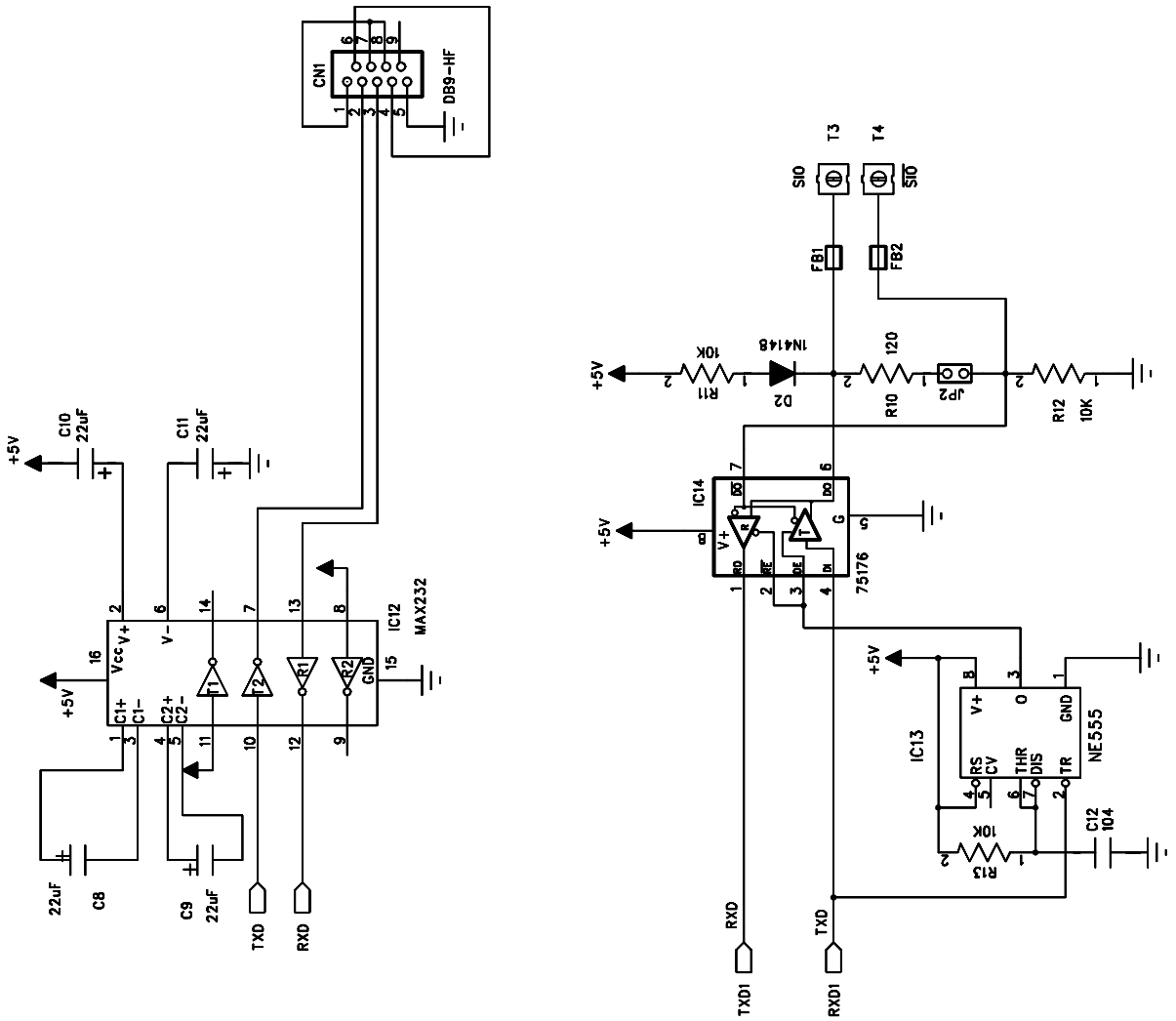
8255與IO連接器配置



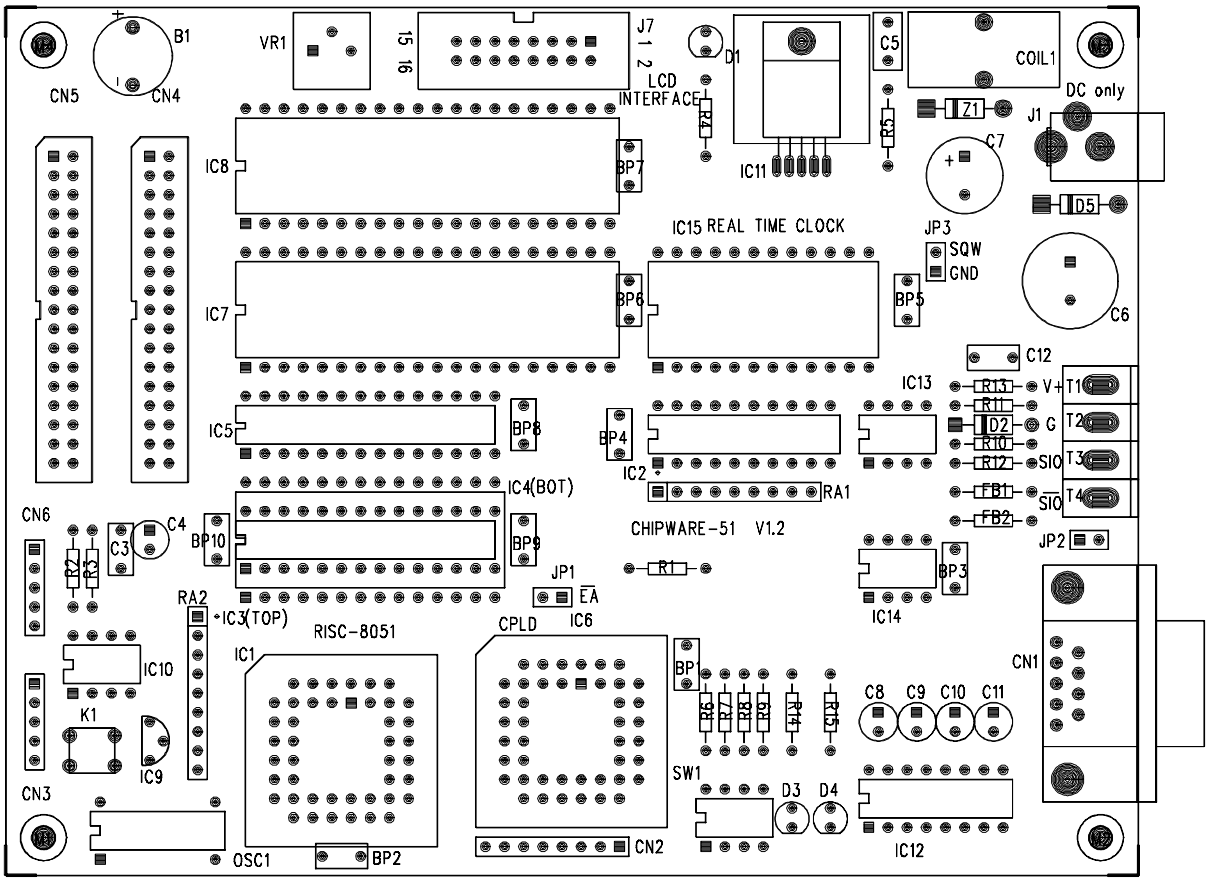
電源輸入與RESET線路圖

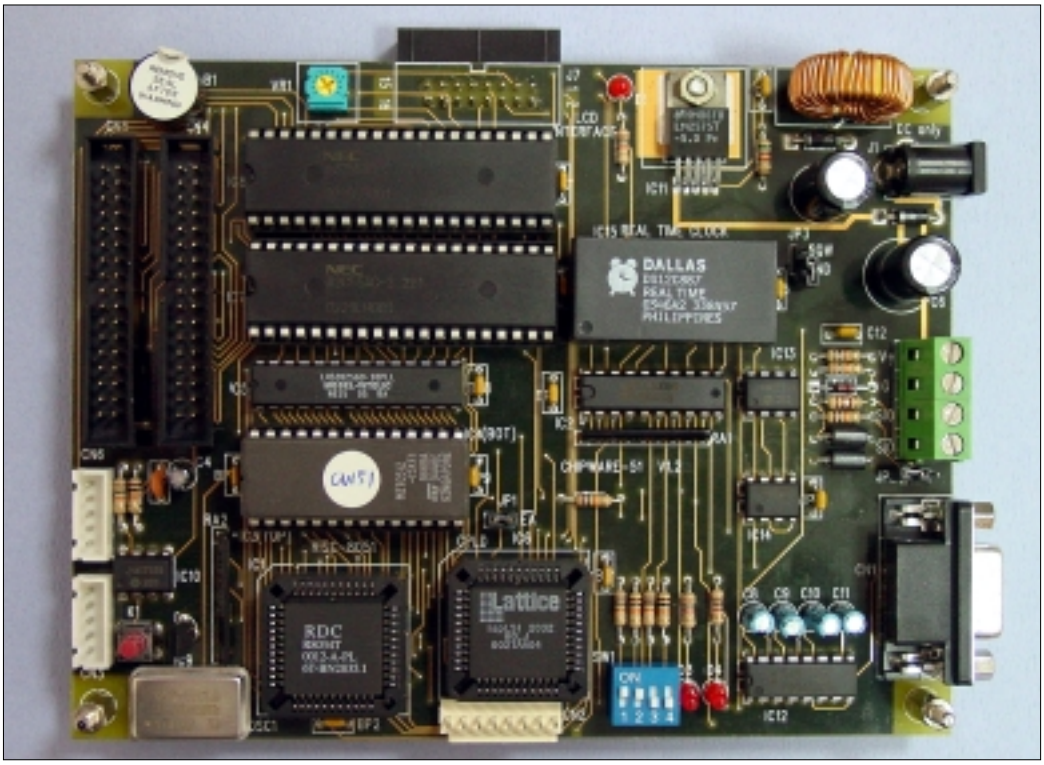


串列通訊線路圖


CN1
DB9-HF
C10
22uF
C11
22uF
+5V
IC12
MAX232
Vcc
V+
V-
C1+
C1-
C2+
C2-
T1
T2
R1
R2
GND
C8
22uF
C9
22uF
TXD
RXD
T3
T4
SI0
SI0
FB1
FB2
1N4148
D2
R11
10K
R10
120
JP2
R12
10K
IC14
75176
IC13
NE555
RS
CV
THR
DIS
TR
R13
10K
C12
104
TXD1
RXD1

◎ CHIPWARE51 零件位置



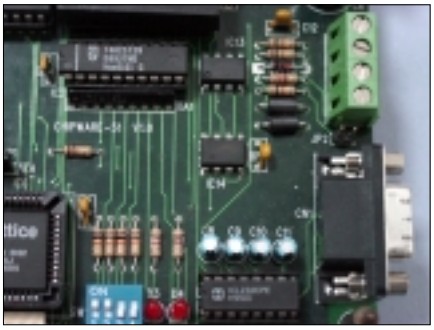


Chipware51的實體照片

兩枚8255提供48點IO

RTC DS12C887時間IC的厚度較高

RDC8034T是一枚RISC8051



兩種串列通訊界面

◎Chipware51 Memory與IO位置分配

正常模式Normal Mode(DIPSW4 OFF)

Program Memory
0000H-FFFFH IC3 27512

Data Memory
0000H-7FFFH IC4 62256
8000H-FF4FH IC5 62256
FF60H-FF63H IC7 8255
FF70H-FF73H IC8 8255
FF80H-FFFFH IC15 DS12C887

模擬模式Emulate Mode(DIPSW4 ON)

Program Memory
0000H-7FFFH IC5 62256(Read Only)

Data Memory
0000H-7FFFH IC4 62256
FF60H-FF63H IC7 8255
FF70H-FF73H IC8 8255
FF80H-FFFFH IC15 DS12C887

假設，您正在除錯階段，且程式空間小於32K Bytes時，可以先借助系統程式將程式暫時Download到SRAM上(DIPSW4 OFF)，然後再把DIPSW4撥到ON的位置，即可進行全速的模擬。如果您的程式已經除錯完畢，可以將程式碼燒錄到EPROM或FLASH內，此時必須要將DIPSW4撥到OFF位置。

◎RDC資料重點

CHIPWARE51的核心CPU是RDC8034T，只要4個CLOCK就執行完一個機械周期，相較於8051的架構，它的效能可以說遠遠超越很多。以下是RDC8034T相較於8051有所不同的一些基本特性：

1. 高速運作：每個周期只要4個CLOCK。內建乘法器，只需一個機械周期便可完成。
2. 電源管理支援閒置模式(IDLE)，省電模式(POWER DOWN)，及甦醒模式(WAKE UP)。
3. 兩個串列埠。(SCON、SBUF、SCON1、SBUF1)
4. 兩組 WATCHDOG TIMER。
5. 兩組 DPTR。(DPTR、DPTR1)
6. 3組 16bits Timer。(Timer0、Timer1、Timer2)
7. 內建1KByte SRAM

由於架構上與原8051有些變化，因此在使用上有一些需要特別注意的地方，我們將其整理如下：

1. 串列埠使用限制：

RDC8034T有兩個全雙工的串列埠，但CHIPWARE51的設計，將這兩個串列埠規劃成一個可全雙工的RS232埠(SCON、SBUF)，及一個半雙工的RS485埠(SCON1、SBUF1)，因此如需使用RS485的功能時，請記得使用第二個串列埠。

第一個串列埠的用法與設定，跟原來的8051相同，不過值的注意的是，它的傳送與接收資料的速度可設定成不同的鮑率，其鮑率的設定，可分別利用TIMER1及TIMER2來控制，列表如下。

RCLK Bit	TCLCK Bit	Receiver Baud Rate Generator	Transmitter Baud Rate Generator
0	0	Timer1	Timer1
0	1	Timer1	Timer2
1	0	Timer2	Timer1
1	1	Timer2	Timer2

第二個串列埠的用法，大致跟原來的串列埠相同，不同的是需以SCON1、SBUF1做爲設定與傳送的SFR。而傳送及接收的鮑率設定都由Timer1決定。

2. Watchdog Timer的使用方式：

RDC8034T有兩個Watchdog Timer可以用，第一個就是我們所熟悉的WDT，它是一個22bits的計數器，爲外部振盪器的振盪次數，藉由WDT暫存器中的WD0、WD1、WD2來設定Watchdog Timer要產生溢位所需的次數設定，列表如下。

WD2	WD1	WD0	Counting
0	0	0	1×2^{19}
0	0	1	$2x\ 2^{19}$
0	1	0	3×2^{19}
0	1	1	4×2^{19}
1	0	0	5×2^{19}
1	0	1	6×2^{19}
1	1	0	7×2^{19}
1	1	1	8×2^{19}

另一個Watchdog Timer是WD1，它可以透過特別的暫存器CKCON(8EH)裡WD1-1及WD1-0兩個位元，設定溢位計數，列表如下。

WD1-1	WD1-0	Counting
0	0	2^{17}
0	1	2^{20}
1	0	2^{23}
1	1	2^{26}

更詳細的資料，可以參考RDC8034T的DATA SHEET。

3. SRAM的用法：

在RDC8034T中內含1 Kbyte的SRAM，它會佔用外部記憶體64Kbyte的其中1Kbyte。但在高速運算下，如果外部RAM的運作跟不上CPU的運作頻率時，內建的SRAM便會派上用場。

4. 乘法器的時序問題：

在原8051的架構下，乘法與除法運算所花的時間是相同的(48 CLOCKS)，但在RDC8034T的規劃，乘法運算會比除法運算快上三倍，因爲乘法運算只需一個周期(4 CLOCKS)即可完成，而除法需三個周期(12 CLOCKS)，這是要特別留意的地方。

◎Chipware51使用注意事項

電源供應

Chipware51有兩種電源供應的方式，一是Adaptor的電源輸入，請先確定此Adaptor已經是直流電壓輸出，且輸出的電壓至少是+7VDC以上，Chipware51對輸入的電壓會另行穩壓成系統要的+5VDC。另一種方式是以交換式電源做輸入，可以選擇+12V或+24VDC的輸出，Chipware51的耗電量不到200mA，一般的交換式電源供應器50W或100W應該都能適用。

LCD模組的使用

爲了使系統更便捷，我們強烈建議您將LCD模組方面的程式納入。LCD模組的硬體佔用IC8 8255的PB7-PB0以及RISC8051的P15、P16、P17共11 bits。請在程式當中避開這些bits，否則會造成不可預期的程式錯誤。

RTC的使用

DS12C887是一枚常見的Real Time Clock時間IC，它本身就是非常省電的裝置，內含小鋰電池，資料約可撐10年左右。不過RTC出廠前會將供電的控制bit關閉，若直接讀取RTC位址時，是不能得到正確的年月日時分秒的，您的程式要在RESET後把該bit打開後，才能進行時間的設定與讀取, 詳細設定方法請參考DS12C887的資料。

RISC8051 CPU的使用

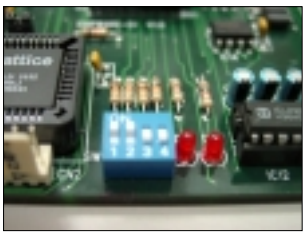
RDC8034T的執行速度是傳統8051的三倍以上，所以您原有用軟體產生的delay程式要重新確認時間，RDC8051還有許多新增的中斷輸入點，這一部份請參考RDC8034T的Datasheet。

◎如何Download

Chipware51可以用下載程式的方法，減少燒錄EPROM程碼的次數，以下就是正確的步驟:

Step 1

Chipware51控制板上裝有標示"chipware"的EPROM，並且20x2的LCM模組也連接在CN7上，此時請確認DIPSW4是在OFF的位置



DIPSW4**在**OFF**位置**

Step 2

開啓Chipware51的電源，幾秒鐘後應該可以在LCD幕上看到"READY TO DOWNLOAD"等字樣

確定EPROM**是**Download**專用的**

開機後顯示的畫面

Step 3

請把您的8051程式預先轉成純二進位檔，即我們通稱的TSK檔。如果您的編譯程式無法轉出Binary檔時，請自行到網路上找尋HEX2BIN.EXE檔案，即可進行轉換。

Download**完會自動算出**Checksum

Step 4

執行本手册所附的DOS或Windows下載程式，由PC的COM埠將資料送到Chipware51的CN1埠上,Download完畢後，LCD幕上會自動顯示"checksum=XXXXH"，代表資料已經進入Chipware51的SRAM區中。

將DIPSW4**切到**ON,**開始進模擬**

Step 5

請把DIPSW4撥到ON，並按下RESET鍵。此時Chipware51會從SRAM區位址0000H處啓動，亦即開始執行使用者的程式。

Step 6

如果發現程式的動作不正確時，可以再度把DIPSW4撥到OFF位置，然後按下RESET鍵，讓系統程式重新啓動。

Step 7

重覆step1-step6的動作，直到所有程式的動作都正確爲止

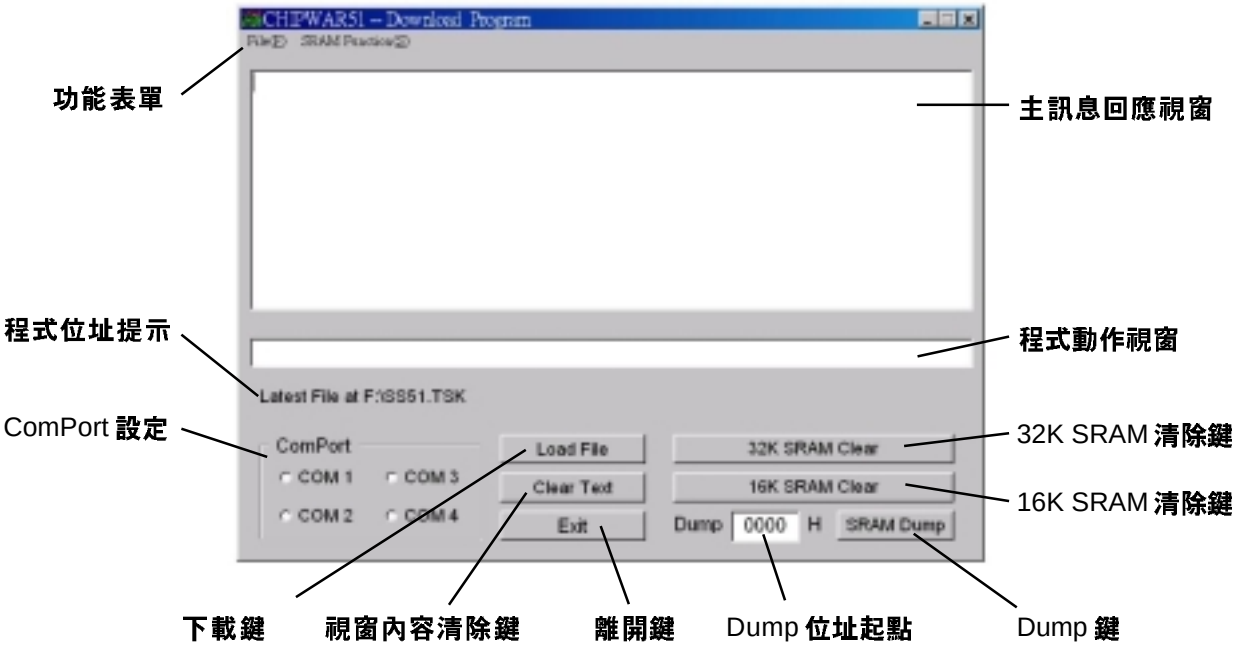
Step 8

把最後的程式碼燒入27512 EPROM中，開啓電源前請確認DIPSW4是OFF的，然後再開啓電源

◎Chipware51與FLAG51的比較

	Chipware51	FLAG51
CPU型號	RDC8051	i8051
CPU包裝	PLCC44	DIP40
CPU內部SRAM	1 KBytes	0 KBytes
CPU執行速度	4 MIPS	1 MIPS
Clock速度	11.0592MHz	11.0592MHz
Clock (最快)	66MHz	12MHz
程式空間	64K Bytes	32K Bytes
RAM空間	64K Bytes	8K Bytes
解碼方式	CPLD2032	GAL/PEEL
8255數量	2 (48bits IO)	2 (48bits IO)
RS232界面	Yes	Yes
RS485界面	Yes	No
RTC時間	Yes	No
LCD界面	Yes	No
電源輸入範圍	7-32VDC	5VDC only
DOWNLOAD	Yes	Yes
User程式開始位址	0000H	8000H
開發日期	2003/09	1992

◎ CHIPWARE51 下載程式使用說明：



安裝CHIPWARE51 下載程式完成後，執行該程式會出現上面的畫面，其功能說明如下：

1. 功能表單：下載程式的基本指令，畫面上皆有相對應的按鍵。
2. 主訊息回應視窗：此視窗會顯示所有來自 CHIPWARE51 控制板的回應訊息。
3. 程式動作視窗：顯示下載程式現在正在執行的動作狀態。
4. 程式位址提示：顯示前一次下載資料的檔名及位址。
5. ComPort 設定：設定連接埠的視窗，一定要先設定才能動，無預設值。
6. 下載鍵：將自行撰寫的程式下載至 CHIPWARE51 的功能鍵。
7. 視窗內容清除鍵：當主訊息視窗回應過多的資料時，可利用此鍵清除。
8. 32K SRAM 清除鍵：將 32K SRAM 所儲存的資料清爲 0 。
9. 16K SRAM 清除鍵：將 16K SRAM 所儲存的資料清爲 0 。
10. Dump 鍵：顯示 SRAM 內所儲存的資料，請先在 Dump 位址起點輸入四位數 16 進位碼。
11. 離開鍵：離開程式。

◎Demo C程式

```
/* chipware.c for NEW 8051 KIT */

#include <stdio.h>
#include <io51.h>
#include <ctype.h>
#include "lcd.h"

#define INTERVAL    9217/5  /* 2mS FOR 11.0592MHz CLOCK */
#define BAUD_RATE 0xfd    /*  9600 BPS */
#define TMH             (65536-INTERVAL) / 256
#define TML             (65536-INTERVAL) % 256


#define  SRAM1        0x0000          /* 0000H-7FFFH */
#define  SRAM2        0x8000          /* 8000H-FF5FH */

#define  P1_PA        0xff60          /* output port */
#define  P1_PB        P1_PA+1
#define  P1_PC        P1_PA+2
#define  P1_CNTL      P1_PA+3

#define  P2_PA        0xff70          /* output port */
#define  P2_PB        P2_PA+1         /* PA,PB,PC output */
#define  P2_PC        P2_PA+2
#define  P2_CNTL      P2_PA+3

#define  LCD_DATA  P2_PB          /* 8255 Latch */
#define  LCD_CNTL  P2_CNTL        /* 8255 Latch */
#define  LINE1         1
#define  LINE2         2

#define  RTC           0xff80
#define  SECOND     RTC+0x00
#define  MINUTE     RTC+0x02
#define  HOUR       RTC+0x04
#define  WEEK       RTC+0x06
#define  DATE       RTC+0x07
#define  MONTH       RTC+0x08
#define  YEAR       RTC+0x09

#define  REG_A         RTC+0x0a
#define  REG_B         RTC+0x0b
#define  REG_C      RTC+0x0c
#define  REG_D      RTC+0x0d

#define  EN_A          (read_bit(P3_2_bit))
#define  EN_B          (read_bit(P3_3_bit))
#define  LOCK_KEY (read_bit(P3_4_bit))
#define  LOCAL_KEY               (read_bit(P3_5_bit))

#define  SELECT_KEY(read_bit(P1_6_bit))
#define  ON_KEY        (read_bit(P1_7_bit))

int lcd_clear_line(int line);
int lcd_cmd(int code);
int lcd_str(char *q);
int lcd_char(int q);

void enable_all();
void t0_int();

int lcd_4d(int v);
int lcd_ms(int v);
int lcd_2d(int v);
int lcd_hex(int v);
int delay(int t);
int sdelay(char t);
char read_stm(char addr);
int write_stm(char addr,char data);
char check_error(char x);
int show_out(char enable);

const char tel[20]     =                          { "886-7-3955152      " };
const char chipware[20]=          { "Chipware Systems Inc" };
const char RAM_TEST[20]=     { "SRAM TEST START...."  };
const char RAM_OK[20]  =          { "SRAM 0000H-FF5FH OK"  };
const char RAM_ERR[20] =          { "SRAM ERR AT:" };
const char PPI_TEST[20]=          { "8255 TEST START...."  };
const char PPI_OK[20]  =          { "8255 SELFTEST OK   "  };
const char PPI_ERR[20] =          { "8255 IO ERR AT:" };

char intr_count;
char ph_a,ph_a_old,ph_b,ph_b_old;
char local,lock,unlock_cnt;
char buf[12],sin_cnt;
char out[10],out_cnt;
int  width,width_old;

main()
{
 unsigned int m;
 unsigned char n,k;

    output(P3,0xff);
    for (m=0;m<8;m++) delay(10);
    write_XDATA(P1_CNTL,0x9b);
    write_XDATA(P2_CNTL,0x80);

    init_8255(); lcd_init();
    for (m=0;m<0x1000;m++) /* read 4096 bytes */
    { n=read_CODE(m+0x2000);
      write_XDATA(m,n);
    }

    lcd_cmd(DISP_L2);
    lcd_str(tel);
    lcd_cmd(DISP_L1);
    lcd_str(chipware);
    for (m=0;m<8;m++) delay(100);
/*
    lcd_clear_line(1); lcd_clear_line(2);
    lcd_cmd(DISP_L1);
    lcd_str(RAM_TEST);

    lcd_cmd(DISP_L2);
    for (m=0x3000,n=0;m<0xff60;m++,n++)
    {
     write_XDATA(m,n);
     k=read_XDATA(m);
     if (k!=n) break;
     if (m%4096==4095) lcd_char('P');
    }
    lcd_clear_line(2);
    lcd_cmd(DISP_L2);
    if (m==0xff60) lcd_str(RAM_OK);
    else
    { lcd_str(RAM_ERR);
      lcd_hex(m/256);
      lcd_hex(m%256);
      lcd_char('H');
    }
    for (m=0;m<20;m++) delay(100);

    lcd_clear_line(1); lcd_clear_line(2);
    lcd_cmd(DISP_L1);
    lcd_str(PPI_TEST);
    write_XDATA(P2_CNTL,0x80);
    write_XDATA(P1_CNTL,0x9b);

    for (m=0,n=0;m<256;m++,n++)
    {
     write_XDATA(P2_PA,n);
     write_XDATA(P2_PB,n);
```

```
    write_XDATA(P2_PC,n);
    k=read_XDATA(P1_PA);
    if (k!=n) break;
    k=read_XDATA(P1_PB);
    if (k!=n) break;
    k=read_XDATA(P1_PC);
    if (k!=n) break;
   }

   lcd_cmd(DISP_L2);
   if (m==256) lcd_str(PPI_OK);
   else
   { lcd_str(PPI_ERR);
     lcd_hex(m);
   }
*/
   enable_bq3287();
   init_rtc();
   for (m=0;m<20;m++) delay(100);
/*  enable_interrupt();  */

   while (1)
   { lcd_cmd(DISP_L2);
     show_ymd();
     delay(100);
   }
}

int init_8255()
{
      write_XDATA(P1_CNTL,0x9b);    /* PA,PB,PC all input */
      write_XDATA(P1_PA,0xff);
      write_XDATA(P1_PB,0xff);
      write_XDATA(P1_PC,0xff);

      write_XDATA(P2_CNTL,0x80);     /* PA & PB & PC output */
      write_XDATA(P2_PA,0x00);
      write_XDATA(P2_PB,0x00);
      write_XDATA(P2_PC,0xff);
}


int clear_buffer()
{
 char x;
      for (x=0;x<12;x++) buf[x]=0;
      sin_cnt=0;
}

int serial_int()
{
 char s_code;

      if (read_bit(RI_bit))
      { s_code=input(SBUF);
        clear_bit(RI_bit);
/*      if (s_code>=1 && s_code<=9) clear_buffer();    */
        buf[sin_cnt]=s_code;
        sin_cnt++;
        if (s_code==0x0a)
        {
          clear_buffer();
        }
      }

      if (read_bit(TI_bit))
      { out_cnt++;
        clear_bit(TI_bit);
        s_code=out[out_cnt];
        if (out_cnt<9) output(SBUF,s_code);
        else          out_cnt=0;
      }
}

void t0_int() /* 10 mS interrupt */
{
 char s_code;

      output(TH0,TMH);     /* timer0 mode1 interrupt */
      output(TL0,TML);

      ph_a=EN_A;
      ph_b=EN_B;


     if (intr_count<100) intr_count++;
     else
      { intr_count=0;
      }
      ph_a_old=ph_a;
      ph_b_old=ph_b;

}

int enable_interrupt()
{
       output(TMOD,0x21);     /* timer1 is mode 2,time0 is mode 1 */
       output(TCON,0x51);     /* TR1 and TR0 start counter */
       output(TH0,TMH);       /* timer0 mode1 interrupt for time*/
       output(TL0,TML);
       output(SCON,0x50);     /* mode 1 */
       output(TH1,BAUD_RATE); /* timer1 mode2 interrupt for BAUD_RATE*/
       set_bit(IT0_bit);
       set_bit(IT1_bit);
       output(IE,0x92);       /* enabled ES+ET0 */
       clear_bit(TI_bit);
       clear_bit(RI_bit);
}


int delay(int t)
{
 int d1,d2;

       for (d1=0;d1<t;d1++)
       { for (d2=0;d2<55;d2++) {} }
}

int sdelay(char d)
{
 char x,y;
       clear_bit(EA_bit);
       for (x=0;x<d;x++)
       { for (y=0;y<25;y++) {}; }
       set_bit(EA_bit);
}


int lcd_init()
{
   lcd_cmd(LCD_INIT);
   delay(100);
   lcd_cmd(LCD_INIT);
   delay(100);
   lcd_cmd(LCD_INIT);
   delay(10);
   lcd_cmd(LCD_CLEAR);
   delay(10);
   lcd_cmd(LCD_CURSOR_HOME);
   delay(10);
   lcd_cmd(DISP_INC_SHIFT_OFF);
   delay(10);
   lcd_cmd(DISP_ON_CURSOR_ON_BLINK_ON);
   delay(10);
}

int lcd_clear_line(int line)
{
```

```
 char x;
      if (line==1) lcd_cmd(DISP_L1);
      else         lcd_cmd(DISP_L2);
      delay(1);
      for (x=0;x<20;x++) lcd_char(' ');
}

int lcd_cmd(int code)
{
      write_XDATA(LCD_DATA,code);        /* command setting */
      LCD_RS0();       /* RS=0,command */
      LCD_WR0();        /* wr=0 */
      LCD_EN1();       /* e=1  */
      LCD_EN0();       /* e=0  */
      LCD_WR1();        /* wr=1 */
      LCD_RS1();       /* RS=1,command */
}

int lcd_data(int code)
{
 char m;
      write_XDATA(LCD_DATA,code);    /* character output */
      LCD_RS1();       /* RS=1,data */
      LCD_WR0();        /* wr=0 */
      LCD_EN1();       /* e=1  */
      m=m;
      LCD_EN0();       /* e=0  */
      LCD_WR1();        /* wr=1 */
      LCD_RS1();       /* RS=1,data */
}

int lcd_str(char *q)
{
 char m;

/*     m=0;
      while (q[m]!=0 && q[m]!='$')
      {
       lcd_char(q[m]);
       m++;
      }
*/
      while (*q!=0)
      { lcd_char(*q);
        q++;
      }
}

int lcd_char(int q)
{
      lcd_data(q);
}

int LCD_RS0() { clear_bit(P1_5_bit); }
int LCD_RS1() {   set_bit(P1_5_bit); }

int LCD_WR0() { clear_bit(P1_6_bit); }
int LCD_WR1() {   set_bit(P1_6_bit); }

int LCD_EN0() { clear_bit(P1_7_bit); }
int LCD_EN1() {   set_bit(P1_7_bit); }

int lcd_2d(int v)
{
 char d1,d2;

        d1=v/10;
        d2=v%10;
        lcd_char(d1+'0');
        lcd_char(d2+'0');
}

int lcd_4d(int v)
{
 char d1,d2,d3,d4;
 int  x;

        d1=v/1000; x=v%1000;
        d2=x/100;  x=v%100;
        d3=x/10;
        d4=x%10;

        lcd_char(d1+'0');
        lcd_char(d2+'0');
        lcd_char(d3+'0');
        lcd_char(d4+'0');
}

int lcd_hex(int v)
{
 char d1,d2;

        d1=v/16;
        d2=v%16;

        if (d1<10) lcd_char(d1+'0');
        else       lcd_char(d1-10+'A');

        if (d2<10) lcd_char(d2+'0');
        else       lcd_char(d2-10+'A');
}

int enable_bq3287()
{
 char a;
/*     a=read_XDATA(REG_A);
       if (a!=0x20) write_XDATA(REG_A,0x20); /* clock re-start */
*/
       write_XDATA(REG_A,0x2f);
       write_XDATA(REG_B,0x0e);            /* SQW out+binary+24H format */
}

int show_ymd()
{
 int ty,tm,td;

        ty=read_XDATA(YEAR);
        if (ty<50) ty=ty+2000;
        else       ty=ty+1900;
        lcd_4d(ty); lcd_char(' ');

        td=read_XDATA(MONTH);
        lcd_2d(td); lcd_char (' ');
        td=read_XDATA(DATE);
        lcd_2d(td); lcd_char (' ');
        ty=read_XDATA(HOUR);
        tm=read_XDATA(MINUTE);
        td=read_XDATA(SECOND);

        lcd_2d(ty); lcd_char (':');
        lcd_2d(tm); lcd_char (':');
        lcd_2d(td);
}

int init_rtc()
{
       if (read_XDATA(YEAR)!=0x03)
       {
        write_XDATA(YEAR,3);
        write_XDATA(MONTH,12);
        write_XDATA(DATE,27);
        write_XDATA(HOUR,23);
        write_XDATA(MINUTE,20);
        write_XDATA(SECOND,00);
       }
}
```

◎Chipware51控制板選購指南

RS485介面轉換產品

RS232-RS485轉接盒

型號：RR485
NT $ 1, 299

外接電源12V~24V
兩線半雙工傳輸模式
傳輸速率9600

USB-RS485轉接盒(for USB1.1)

型號：UR485
NT $ 1,299

不需外加電源
兩線半雙工傳輸模式
傳輸速率可達19200bps

USB-RS485轉接盒(for USB2.0)

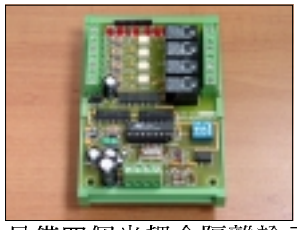
型號：UIR485
NT $ 2,499

不需外加電源
兩線半雙工傳輸模式
最高傳輸速率超過19200bps

備註
◆以上商品價格皆爲含稅價

RS485介面擴充產品

工業用數位輸出入控制板



型號：DIO-I
NT $ 4,200

具備四個光耦合隔離輸入點與
四個RELAY輸出點
標準DC24V電壓輸入

工業用數位轉類比控制板

型號：DA-I
NT $ 4,500

2CH類比輸出,輸出範圍0-10V
採用10bit DAC
具備七段顯示器可顯示類比電壓輸出值
標準DC24V電壓輸入

工業用類比轉數位控制板

型號：AD-I
NT $ 4,800

採用20bit ADC,輸入範圍DC 0-10V
顯示單位V與mV
四位數顯示輸入類比電壓值
DC24V電壓輸入

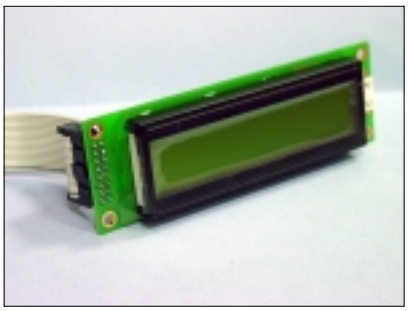
溫濕度控制板

型號：TH2040
NT $ 4,725

高穩定度的溫度傳送器
溫度攝氏0-60度,濕度顯示範圍10%-90%(未結露)
供電範圍DC8-30V

8255介面相關擴充產品

20x2 文字型LCD背光模組(含排



型號：CW51-LCD
NT $ 600 (*)

標示 * 之商品，若搭配 CHIPWARE51
控制板一起購買，可減價 NT$100

其 它 選 購 產 品

RS485專用隔離雙絞線材(20米)

型號：CB485-20
NT $ 400

DC24V交換式電源供應器(50瓦)

型號：PS24V-50W
NT $ 700

備註
◆以上商品價格皆爲含稅價