

Hệ điều hành LINUX
PHIÊN BẢN FEDORA CORE 5

ỨNG DỤNG LINUX

(PIC16F877A và giao tiếp RS-232 với PC)

Auth: Nguyễn Quang Phú

Rev: gnuPICuC-FC5-200607

Đây là một tài liệu miễn phí. Bạn hoàn toàn có thể phân phối nó lại cho những người sử dụng khác hoặc có thể chỉnh sửa lại cho phù hợp nhưng phải tuân theo những yêu cầu trong giấy phép bản quyền GNU (phiên bản 2.0 hay các phiên bản khác).

*Tài liệu này được phát hành với hy vọng rằng nó sẽ trở nên hữu ích, nhưng nó **KHÔNG KÈM THEO BẤT KỲ SỰ ĐẢM BẢO NÀO**, ngay cả những đảm bảo **NGẦM HIỂU VỀ VIỆC THƯƠNG MẠI HOÁ** hay **PHẢI PHÙ HỢP VỚI MỘT MỤC ĐÍCH CỤ THỂ NÀO ĐÓ**. (Vấn đề này bạn có thể tham khảo giấy phép **GNU General Public License** để biết thêm chi tiết).*

*Thông thường bạn sẽ nhận được một bản sao của giấy phép **GNU General Public License** kèm theo tài liệu này; nếu chưa có bạn có thể gửi thư đến địa chỉ sau **Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA** để có một bản giấy phép.*

Mục lục

- 1. Giới thiệu**
- 2. Tạo file asm bằng chương trình soạn thảo gedit**
- 3. Công cụ vẽ mạch (EDA)**
- 4. Các chương trình biên dịch và nạp chip**
- 5. Quá trình biên dịch và nạp chip**
- 6. Kiểm tra chương trình**
- 7. Mã nguồn các chương trình**

Tài liệu này được biên soạn với hy vọng khuyến khích mọi người tôn trọng bản quyền sở hữu trí tuệ và không sử dụng các phần mềm bị bẻ khoá trên các máy tính cá nhân. Bạn có thể liên hệ với người viết theo địa chỉ opentdoors@yahoo.com nếu bạn cảm thấy tài liệu này có thiếu sót, chưa đầy đủ cũng như nếu nó hữu ích đối với bạn. Hoặc bạn cũng có thể liên hệ với người viết tại các web site (<http://www.dientuvietnam.net>) tại các chủ đề liên quan đến các ứng dụng của Linux trong điện tử.

1. Giới thiệu:

Đây là tài liệu giới thiệu về các công cụ lập trình và nạp chương trình cho vi điều khiển họ 16FxxA (16F84a, 16F876a và 16F877a) của Microchip trên hệ điều hành Linux phiên bản Fedora Core 5.

Nội dung tài liệu này bao gồm :

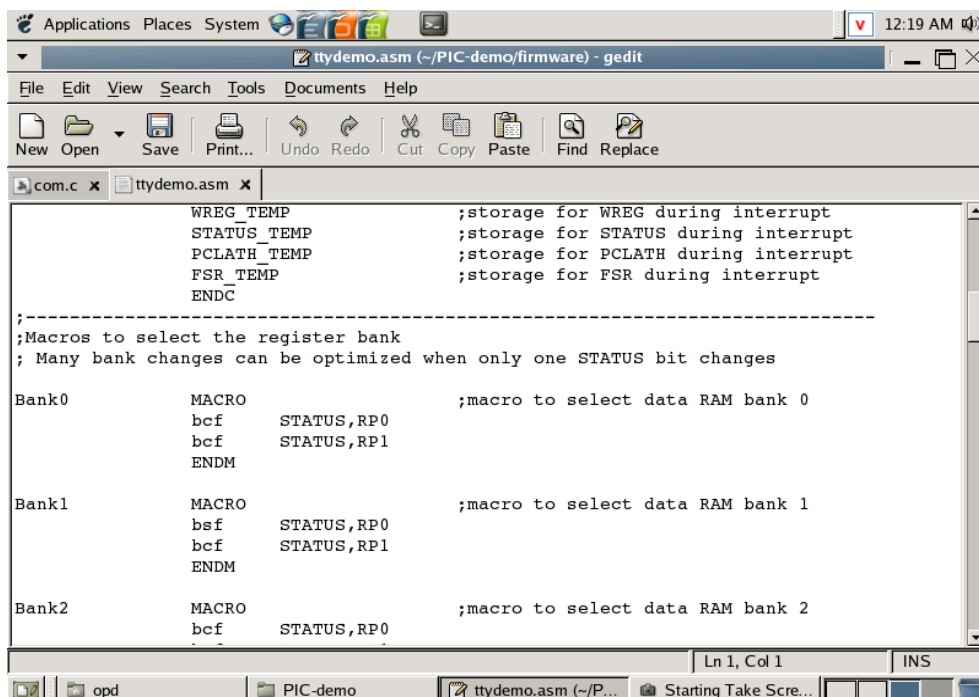
- Viết chương trình.
- Biên dịch
- Nạp chương trình vào vi điều khiển
- Kiểm tra hoạt động của vi điều khiển thông qua giao tiếp RS-232.

Những nội dung của tài liệu này được viết thông qua việc thực hiện mạch và đã được kiểm tra trên máy tính của người viết.

2. Tạo file asm bằng chương trình soạn thảo gedit

Trên các phiên bản Linux của Red Hat, các chương trình soạn thảo luôn là gEdit hoặc kEdit. Một số người khác thích sử dụng chương trình vim, hay emacs để soạn thảo các source code từ dòng lệnh, tuy nhiên cách này chỉ dành cho những ai thành thạo Unix và Linux.

Chương trình gEdit tương tự như Notepad trên Windows, nhưng trong các trường hợp viết chương trình C/C++, nó tỏ ra tiện ích hơn do nó có phân biệt các hằng, hàm hay biến được định nghĩa trong ngôn ngữ C/C++.



```
WREG_TEMP           ;storage for WREG during interrupt
STATUS_TEMP         ;storage for STATUS during interrupt
PCLATH_TEMP         ;storage for PCLATH during interrupt
FSR_TEMP            ;storage for FSR during interrupt
ENDC

;-----
;Macros to select the register bank
; Many bank changes can be optimized when only one STATUS bit changes

Bank0                MACRO                               ;macro to select data RAM bank 0
                    bcf     STATUS,RP0
                    bcf     STATUS,RP1
                    ENDM

Bank1                MACRO                               ;macro to select data RAM bank 1
                    bsf     STATUS,RP0
                    bcf     STATUS,RP1
                    ENDM

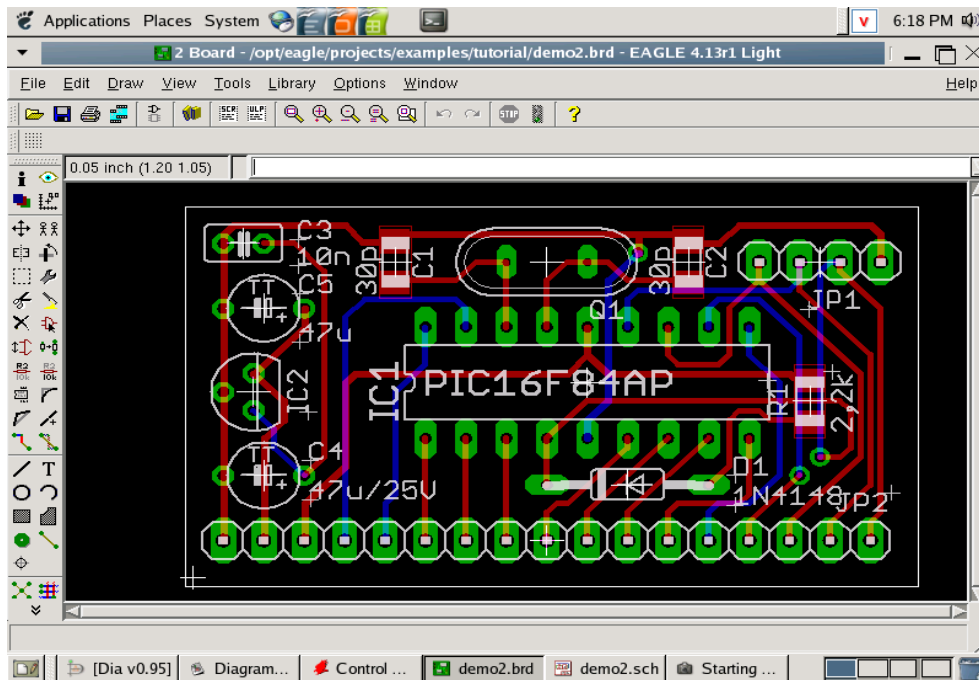
Bank2                MACRO                               ;macro to select data RAM bank 2
                    bcf     STATUS,RP0
```

Một chương trình viết cho vi điều khiển PIC16 trên linux luôn tuân theo các cú pháp viết

trên Windows hay MPLAB.

3. Công cụ vẽ mạch (EDA)

Có nhiều công cụ GNU hỗ trợ việc thiết kế mạch, layout mạch in trên Linux. Tuy nhiên ở đây giới thiệu Eagles, một phiên bản miễn phí trên các phiên bản Linux của Red Hat (RH8.0 cho đến FC5), do nó quen thuộc với những người sử dụng Eagles trên Windows, thư viện phong phú với các chip của Microchip, Atmel ... và không có giới hạn nào trong việc thiết kế mạch với phiên bản miễn phí này.



4. Các chương trình biên dịch và nạp chip

- GNU PIC Utilities: Trình biên dịch gpasm, một phần của bộ chương trình gputils. Nó hỗ trợ tất cả các họ PIC.
- Picprog của Jaakko Hyvätti . Nạp chip qua cổng nối tiếp, bằng JDM.

Việc sử dụng 2 công cụ này rất đơn giản. Tải các gói gputils và picprog dưới dạng các file nén gzip, giải nén các gói này vào thư mục nào đó. Sử dụng chương trình giải nén được tích hợp sẵn trong FC 5, bằng cách kích đúp hoặc kích chuột phải để bung ra, hoặc sử dụng lệnh :

```
$ tar-xzvf "tên đầy đủ".tar.gz
```

Biên dịch, và cài đặt 2 công cụ này như sau (theo các file hướng dẫn cài đặt INSTALL trong gputils và README trong picprog):

1. Logon với tư cách root.
2. "cd" vào thư mục đã được bung ra của **gputils**.
3. Đánh các lệnh

```
#!/configure
# make
# make install
# make clean
```

1. Logon với tư cách root.
2. “cd” vào thư mục đã được bung nén của **picprog**.
3. Đánh các lệnh

```
# make dep
# make
# make install
```

5. Quá trình biên dịch và nạp chip

Ở đây chúng ta sử dụng trình soạn thảo gedit tạo ra file ttydemo.asm, mã nguồn được trình bày cuối tài liệu này. Biên dịch mã sử dụng gpasm như sau:

```
[opd@localhost firmware]$ gpasm ttydemo.asm
ttydemo.asm:17:Warning [230] found lower case match for include filename
[opd@localhost firmware]$ dir
main.asm main.cod main.lst ttydemo.asm ttydemo.cod ttydemo.lst
```

Sau đó chúng ta gắn chip vào JDM và gắn JDM vào cổng COM2. Thực hiện các lệnh sau:

```
[opd@localhost firmware]$ su
Password:
[root@localhost firmware]# picprog --erase --burn --input ttydemo.hex --pic /dev/ttyS1
CPU clock speed: 1818 MHz
/dev/ttyS1: id 0x0e27: detected pic16f877a version 0x07
Device pic16f877a, program memory: 8192, data memory: 256.
Erased and removed code protection.
Burning program memory, 113 locations,
burning data memory, 0 locations,
burning id words, 0 locations,
burning fuses, 1 locations,
done.
[root@localhost firmware]#exit
```

6. Kiểm tra chương trình

Do đây là một chương trình giao tiếp USART của 16F877A, nên chúng ta kiểm tra nó giao tiếp với máy tính thông qua cổng COM1.

Sử dụng chương trình ttydevinit của tác giả Guido Socher. Dùng trình soạn thảo tạo một file ttydevinit.c (mã nguồn trình bày cuối tài liệu). Sau đó thực hiện các lệnh sau:

```
[opd@localhost firmware]$ gcc ttydevinit.c -o ttydevinit
```

```
[opd@localhost firmware]$ su
```

```
Password:
```

```
[root@localhost firmware]#.ttydevinit /dev/ttyS0
```

Sau đó chúng ta mở 2 màn hình terminal để kiểm tra chương trình. Một terminal dùng để hiển thị các ký tự nhận được từ 16F877A,

```
[root@localhost opd]# cat /dev/ttyS0
```

```
PIC-Demo v1.0
```

```
Opentdoors at Linux systems
```

```
Opentdoors
```

```
nguyen quang phu
```

```
linux serial testing
```

và một terminal thứ hai được dùng để gửi các ký tự gõ từ bàn phím đến 16F877A.

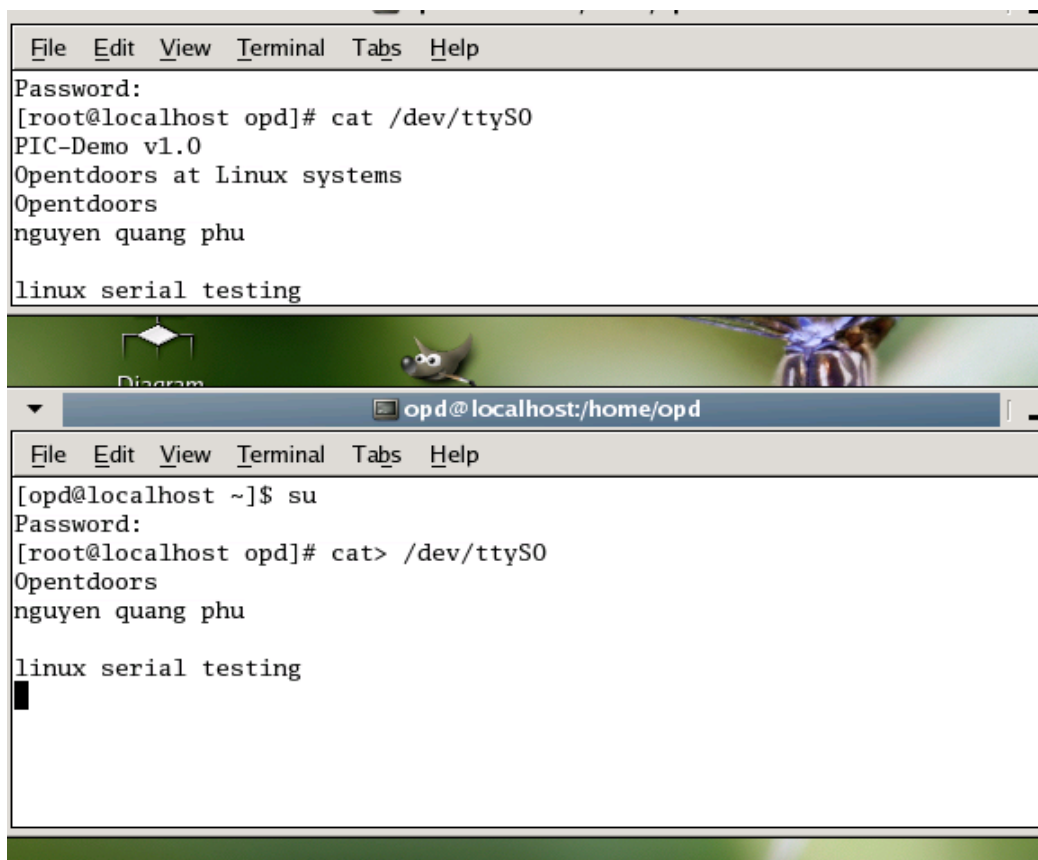
```
[opd@localhost ~]$ su
```

```
Password:
```

```
[root@localhost opd]# cat> /dev/ttyS0 Opentdoors
```

```
nguyen quang phu
```

```
linux serial testing
```




```

00398 LIST
00017 errorlevel -302 ;suppress "not in bank 0" message
002007 3F72 00018 __CONFIG_PWRTE_ON&_HS_OSC&_LVP_OFF&_WDT_OFF
00019
00020 ;-----
00021 ;Constants
00022
00000081 00023 SPBRG_VAL EQU .129 ;set baud rate 9600 for 20Mhz clock
00000020 00024 dptr EQU 0x20
00025 ;-----
00026 ;Variables
00027
00028
00029 CBLOCK 0x70
00030 WREG_TEMP ;storage for WREG during interrupt
00031 STATUS_TEMP ;storage for STATUS during interrupt
00032 PCLATH_TEMP ;storage for PCLATH during interrupt
00033 FSR_TEMP ;storage for FSR during interrupt
00034 ENDC
00035 ;-----
00036 ;Macros to select the register bank
00037 ; Many bank changes can be optimized when only one STATUS bit changes
00038
00039 Bank0 MACRO ;macro to select data RAM bank 0
00040 bcf STATUS,RP0
00041 bcf STATUS,RP1
00042 ENDM
00043
00044 Bank1 MACRO ;macro to select data RAM bank 1
00045 bsf STATUS,RP0
00046 bcf STATUS,RP1
00047 ENDM
00048
00049 Bank2 MACRO ;macro to select data RAM bank 2
00050 bcf STATUS,RP0
gpasm-0.13.3 beta ttydemo.asm 7-5-2006 00:40:35 PAGE 2

```

```

LOC OBJECT CODE LINE SOURCE TEXT
VALUE

```

```

00051 bsf STATUS,RP1
00052 ENDM
00053
00054 Bank3 MACRO ;macro to select data RAM bank 3
00055 bsf STATUS,RP0

```

```

00056      bsf  STATUS,RP1
00057      ENDM
00058
00059 ;-----
0000      00060      ORG  0x0000      ;place code at reset vector
00061
0000 018A  00062 ResetCode:  clrf  PCLATH      ;select program memory page 0
0001 2818  00063      goto  Main      ;go to beginning of program
0004      00064      ORG  0x004      ; interrupt vector location
00065
0004 00F0  00066 InterruptCode: movwf WREG_TEMP      ;save WREG
0005 0803  00067      movf  STATUS,W      ;store STATUS in WREG
0006 0183  00068      clrf  STATUS      ;select file register bank0
0007 00F1  00069      movwf STATUS_TEMP      ;save STATUS value
0008 080A  00070      movf  PCLATH,W      ;store PCLATH in WREG
0009 00F2  00071      movwf PCLATH_TEMP      ;save PCLATH value
000A 018A  00072      clrf  PCLATH      ;select program memory page0
000B 0804  00073      movf  FSR,W      ;store FSR in WREG
000C 00F3  00074      movwf FSR_TEMP      ;save FSR value
00075 ;-----
00076 ;End of interrupt routine restores context
000D      00077 EndInt
00078      Bank0      ;select bank 0
000D 1283  M      bcf  STATUS,RP0
000E 1303  M      bcf  STATUS,RP1
000F 0873  00079      movf  FSR_TEMP,W      ;get saved FSR value
0010 0084  00080      movwf FSR      ;restore FSR
0011 0872  00081      movf  PCLATH_TEMP,W      ;get saved PCLATH value
0012 008A  00082      movwf PCLATH      ;restore PCLATH
0013 0871  00083      movf  STATUS_TEMP,W      ;get saved STATUS value
0014 0083  00084      movwf STATUS      ;restore STATUS
0015 0EF0  00085      swapf WREG_TEMP,F      ;prepare WREG to be restored
0016 0E70  00086      swapf WREG_TEMP,W      ;restore WREG without affecting STATUS
0017 0009  00087      retfie      ;return from interrupt
00088
00089 ;-----
0018      00090 Main:
0018 2020  00091      call  SetupSerial
0019 3000  00092      movlw m0-tit-1
001A 2036  00093      call  Putstr
001B 3010  00094      movlw m1-tit-1
001C 2036  00095      call  Putstr
001D      00096 MainLoop:
001D 2031  00097      call  Getchr
001E 202D  00098      call  Putchr
00099

```

```

001F 281D 00100      goto  MainLoop      ;repeat main loop to check for data
00101 ;-----
00102 ;   SetupSerial:Set up serial port and buffers.
gpasm-0.13.3 beta      ttydemo.asm 7-5-2006 00:40:35      PAGE 3

```

```

LOC OBJECT CODE  LINE SOURCE TEXT
VALUE

```

```

00103 ;   Author:      Mike Garbutt
00104 ;   Company:     Microchip Technology Inc.
00105 ;-----
0020      00106 SetupSerial: Bank1      ;select bank 1
0020 1683      M      bsf  STATUS,RP0
0021 1303      M      bcf  STATUS,RP1
0022 30C0 00107      movlw 0xc0      ;set tris bits for TX and RX
0023 0487 00108      iorwf TRISC,F
0024 3081 00109      movlw SPBRG_VAL ;set baud rate
0025 0099 00110      movwf SPBRG
0026 3024 00111      movlw 0x24      ;enable transmission and high baud rate
0027 0098 00112      movwf TXSTA
00113      Bank0      ;select bank 0
0028 1283      M      bcf  STATUS,RP0
0029 1303      M      bcf  STATUS,RP1
002A 3090 00114      movlw 0x90      ;enable serial port and reception
002B 0098 00115      movwf RCSTA
002C 0008 00116      return
00117 ;-----
00118 ;   Function : Putchr
00119 ;   Hàm gửi một byte từ thanh ghi W ra cổng RS-232
00120 ;-----
002D      00121 Putchr:
002D 1E0C 00122      btfss PIR1,TXIF ;check that buffer is empty
002E 282D 00123      goto  $-1
002F 0099 00124      movwf TXREG      ;transmit byte
0030 0008 00125      return
00126 ;-----
00127 ;   Function : Getchr
00128 ;   Hàm nhận một byte từ cổng RS-232 và lưu vào thanh ghi W
00129 ;-----
00130
0031      00131 Getchr:
0031 0000 00132      nop
0032 1E8C 00133      btfss  PIR1,RCIF ; check for received data
0033 2831 00134      goto  Getchr
0034 081A 00135      movf  RCREG,w

```

```

0035 0008 00136 return
      00137 ;-----
00138 ; Function : Putstr
00139 ;      Hàm gọi một chuỗi các ký tự trở bởi thanh ghi W ra cổng RS-232.
      00140 ;-----
00141
0036      00142 Putstr
0036 00A0 00143 movwf dptr      ; Point at first letter
0037      00144 putstr1
0037 3000 00145 movlw HIGH tit
0038 008A 00146 movwf PCLATH
0039 0820 00147 movf dptr,W      ; Get pointer into W
003A 2044 00148 call tit
003B 018A 00149 clrf PCLATH      ; Get character to output
003C 3800 00150 iorlw 0      ; At the End of the Message?
003D 1903 00151 btfs STATUS, Z      ; Skip if not at end
003E 2842 00152 goto putstr2
gpasm-0.13.3 beta      ttydemo.asm 7-5-2006 00:40:35      PAGE 4

```

```

LOC OBJECT CODE  LINE SOURCE TEXT
VALUE

```

```

003F 202D 00153 call Putchr
0040 0AA0 00154 incf dptr,F
0041 2837 00155 goto putstr1
0042      00156 putstr2
0042 018A 00157 clrf PCLATH
0043 0008 00158 return
      00159 ;-----
0044 0782 00160 tit addwf PCL,F
0045 3450 3449 00161 m0 dt "PIC-Demo v1.0",0x0a,0x0d,0
0047 3443 342D
0049 3444 3465
004B 346D 346F
004D 3420 3476
004F 3431 342E
0051 3430 340A
0053 340D 3400
0055 344F 3470 00162 m1 dt "Opentdoors at Linux systems",0x0a,0x0d,0
0057 3465 346E
0059 3474 3464
005B 346F 346F
005D 3472 3473
005F 3420 3461
0061 3474 3420

```

0063 344C 3469
0065 346E 3475
0067 3478 3420
0069 3473 3479
006B 3473 3474
006D 3465 346D
006F 3473 340A
0071 340D 3400

00163 ;-----

gpasm-0.13.3 beta

ttydemo.asm 7-5-2006 00:40:35

PAGE 5

SYMBOL TABLE

LABEL	VALUE
ACKDT	00000005
ACKEN	00000004
ACKSTAT	00000006
ADCON0	0000001F
ADCON1	0000009F
ADCS0	00000006
ADCS1	00000007
ADDEN	00000003
ADFM	00000007
ADIE	00000006
ADIF	00000006
ADON	00000000
ADRESH	0000001E
ADRESL	0000009E
BCLIE	00000003
BCLIF	00000003
BF	00000000
BRGH	00000002
C	00000000
C1INV	00000004
C1OUT	00000006
C2INV	00000005
C2OUT	00000007
CCP1CON	00000017
CCP1IE	00000002
CCP1IF	00000002
CCP1M0	00000000
CCP1M1	00000001
CCP1M2	00000002
CCP1M3	00000003
CCP1X	00000005

CCP1Y	00000004
CCP2CON	0000001D
CCP2IE	00000000
CCP2IF	00000000
CCP2M0	00000000
CCP2M1	00000001
CCP2M2	00000002
CCP2M3	00000003
CCP2X	00000005
CCP2Y	00000004
CCPR1H	00000016
CCPR1L	00000015
CCPR2H	0000001C
CCPR2L	0000001B
CHS0	00000003
CHS1	00000004
CHS2	00000005
CIS	00000003
CKE	00000006
CKP	00000004
CM0	00000000
CM1	00000001

gpasm-0.13.3 beta

ttydemo.asm 7-5-2006 00:40:35

PAGE 6

LOC	OBJECT CODE	LINE	SOURCE	TEXT
VALUE				

CM2	00000002
CMCON	0000009C
CMIE	00000006
CMIF	00000006
CREN	00000004
CSRC	00000007
CVR0	00000000
CVR1	00000001
CVR2	00000002
CVR3	00000003
CVRCON	0000009D
CVREN	00000007
CVROE	00000006
CVRR	00000005
D	00000005
DATA_ADDRESS	00000005
DC	00000001
D_A	00000005

EEADR	0000010D
EEADRH	0000010F
EECON1	0000018C
EECON2	0000018D
EEDATA	0000010C
EEDATH	0000010E
EEIE	00000004
EEIF	00000004
EEPGD	00000007
EndInt	0000000D
F	00000001
FERR	00000002
FSR	00000004
FSR_TEMP	00000073
GCEN	00000007
GIE	00000007
GO	00000002
GO_DONE	00000002
Getchr	00000031
I2C_DATA	00000005
I2C_READ	00000002
I2C_START	00000003
I2C_STOP	00000004
IBF	00000007
IBOV	00000005
INDF	00000000
INTCON	0000000B
INTE	00000004
INTEDG	00000006
INTF	00000001
IRP	00000007
InterruptCode	00000004
Main	00000018
MainLoop	0000001D
NOT_A	00000005
NOT_ADDRESS	00000005

LOC	OBJECT CODE	LINE	SOURCE	TEXT
-----	-------------	------	--------	------

NOT_BO	00000000
NOT_BOR	00000000
NOT_DONE	00000002
NOT_PD	00000003

NOT_POR	00000001
NOT_RBPU	00000007
NOT_RC8	00000006
NOT_T1SYNC	00000002
NOT_TO	00000004
NOT_TX8	00000006
NOT_W	00000002
NOT_WRITE	00000002
OBF	00000006
OERR	00000001
OPTION_REG	00000081
P	00000004
PCFG0	00000000
PCFG1	00000001
PCFG2	00000002
PCFG3	00000003
PCL	00000002
PCLATH	0000000A
PCLATH_TEMP	00000072
PCON	0000008E
PEIE	00000006
PEN	00000002
PIE1	0000008C
PIE2	0000008D
PIR1	0000000C
PIR2	0000000D
PORTA	00000005
PORTB	00000006
PORTC	00000007
PORTD	00000008
PORTE	00000009
PR2	00000092
PS0	00000000
PS1	00000001
PS2	00000002
PSA	00000003
PSPIE	00000007
PSPIF	00000007
PSPMODE	00000004
Putchr	0000002D
Putstr	00000036
R	00000002
RBIE	00000003
RBIF	00000000
RC8_9	00000006
RC9	00000006

RCD8 00000000
RCEN 00000003
RCIE 00000005
RCIF 00000005

gpasm-0.13.3 beta

ttydemo.asm 7-5-2006 00:40:35

PAGE 8

LOC OBJECT CODE LINE SOURCE TEXT
VALUE

RCREG 0000001A
RCSTA 00000018
RD 00000000
READ_WRITE 00000002
RP0 00000005
RP1 00000006
RSEN 00000001
RX9 00000006
RX9D 00000000
R_W 00000002
ResetCode 00000000
S 00000003
SEN 00000000
SMP 00000007
SPBRG 00000099
SPBRG_VAL 00000081
SPEN 00000007
SREN 00000005
SSPADD 00000093
SSPBUF 00000013
SSPCON 00000014
SSPCON2 00000091
SSPEN 00000005
SSPIE 00000003
SSPIF 00000003
SSPM0 00000000
SSPM1 00000001
SSPM2 00000002
SSPM3 00000003
SSPOV 00000006
SSPSTAT 00000094
STATUS 00000003
STATUS_TEMP 00000071
SYNC 00000004
SetupSerial 00000020
T0CS 00000005

T0IE	00000005
T0IF	00000002
T0SE	00000004
T1CKPS0	00000004
T1CKPS1	00000005
T1CON	00000010
T1INSYNC	00000002
T1OSCEN	00000003
T1SYNC	00000002
T2CKPS0	00000000
T2CKPS1	00000001
T2CON	00000012
TMR0	00000001
TMR0IE	00000005
TMR0IF	00000002
TMR1CS	00000001
TMR1H	0000000F
TMR1IE	00000000

gpasm-0.13.3 beta

ttydemo.asm 7-5-2006 00:40:35

PAGE 9

LOC	OBJECT CODE	LINE	SOURCE	TEXT
VALUE				

TMR1IF	00000000
TMR1L	0000000E
TMR1ON	00000000
TMR2	00000011
TMR2IE	00000001
TMR2IF	00000001
TMR2ON	00000002
TOUTPS0	00000003
TOUTPS1	00000004
TOUTPS2	00000005
TOUTPS3	00000006
TRISA	00000085
TRISB	00000086
TRISC	00000087
TRISD	00000088
TRISE	00000089
TRISE0	00000000
TRISE1	00000001
TRISE2	00000002
TRMT	00000001
TX8_9	00000006
TX9	00000006

TX9D	00000000
TXD8	00000000
TXEN	00000005
TXIE	00000004
TXIF	00000004
TXREG	00000019
TXSTA	00000098
UA	00000001
W	00000000
WCOL	00000007
WR	00000001
WREG_TEMP	00000070
WREN	00000002
WRERR	00000003
Z	00000002
_BODEN_OFF	00003FBF
_BODEN_ON	00003FFF
_CPD_OFF	00003FFF
_CPD_ON	00003EFF
_CP_ALL	00001FFF
_CP_OFF	00003FFF
_DEBUG_OFF	00003FFF
_DEBUG_ON	000037FF
_HS_OSC	00003FFE
_LP_OSC	00003FFC
_LVP_OFF	00003F7F
_LVP_ON	00003FFF
_PWRTE_OFF	00003FFF
_PWRTE_ON	00003FF7
_RC_OSC	00003FFF
_WDT_OFF	00003FFB
_WDT_ON	00003FFF
gpasm-0.13.3 beta	ttydemo.asm 7-5-2006 00:40:35
	PAGE 10

LOC	OBJECT CODE	LINE	SOURCE	TEXT
VALUE				

_WRT_1FOURTH	00003BFF
_WRT_256	00003DFF
_WRT_HALF	000039FF
_WRT_OFF	00003FFF
_XT_OSC	00003FFD
__16F877A	00000001
dptr	00000020
m0	00000045

m1 00000055
putstr1 00000037
putstr2 00000042
tit 00000044

MEMORY USAGE MAP ('X' = Used, '-' = Unused)

00000000 : XX--XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
00000040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXX-----
00002000 : -----X----- -----

All other memory blocks unused.

Program Memory Words Used: 114

Errors : 0
Warnings : 1 reported, 0 suppressed
Messages : 0 reported, 3 suppressed

ttydevinit.c source code

```
/* vim: set sw=8 ts=8 si: */
/* Linux software to set the speed on the serial line
 * Written by Guido Socher
 * run this program like this:
 * ttydevinit /dev/ttyS0 (for com1) and then use
 * cat > /dev/ttyS0 to write or cat /dev/ttyS0 to read commands
 * to/from the linuxlcdpanel */

#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <termios.h>
int main(int argc, char *argv[])
{
    struct termios portset;
    char *device;
    int fd;

    if (argc != 2){
        printf("USAGE: ttydevinit /dev/ttyS1\n");
        exit(0);
    }
    device=argv[1];

    /* Set up io port correctly, and open it... */
    fd = open(device, O_RDWR | O_NOCTTY | O_NDELAY);
    if (fd == -1) {
        fprintf(stderr, "ERROR: open for %s failed.\n",device)
;
        exit(1);
    }
    tcgetattr(fd, &portset);
    cfmakeraw(&portset);
    cfsetospeed(&portset, B9600); /* speed */
    tcsetattr(fd, TCSANOW, &portset);
    close(fd);
    return(0);
}
```