Original Writer：Tim Tsai
Last Editer：Vic Tsai, 9/21/2007

MiniOS7

MiniOs7 is designed by ICPDAS for i-7188/i-8000 series embedded controllers. Support the following functions:

(1) File download and execute.

(2) System and Hardware diagnose function.

(3) MiniOS7 Update function.

(4) Execute the EXE, COM or BAT of DOS executable files

(5) Support AUTOEXEC.BAT for program auto execution.

Note:

- It can execute some of DOS-executable file, not all of the DOS-executable files.
- Support the memory management functions of INT 21H.
- Doesn't support the file I/O functions of INT 21H. The library provides functions to read file.
- When one program is running, it cannot run another program.(Therefore, system() of C's function can not be used.)
- Except the library for C language, it will also support library(unit) for Turbo PASCAL.

Using MiniOS7 Utility (IDE Mode)

MiniOS7 upgrade, file load/delete operation, file list and part commands of MiniOS7 commands are integrated into MiniOS7 Utility.

Location: http://www.icpdas.com/download/minios7.htm

Using 7188xw (Console Mode)

Using 7188xw.exe to communicate with the module built-in MiniOS7

7188xw.exe Location:

1. http://ftp.icpdas.com.tw/pub/cd/8000cd/napdos/minios7/utility/ (Included 7188xw document.)
2. CD:\napdos\minios7\utility\ (Included 7188xw document.)

Command line prompt

I. For i-7188XA: **“i7188XA>”**
II. For i-7188XB/7522A/7524/7527: **“i-7188XB>”**
III. For i-7188XC/7521/7522/7523: **“i7188XC>”**
IV. For i-7188: **“i-7188>”**
V. For i-7188E: **“i7188E>”**
VI. For i-8000: **“i8000>”**

Note:
There is a small difference for the prompt according to different MiniOS7 version.

Command is case-insensitive

MiniOS7 Commands

Some commands are supported by some special modules only, please refer to the note of command.

MiniOS7 Commands		
	COMMAND	**DESCRIPTION**
1	**LED ON | OFF**	**ON**: Turn on the red LED. (Note: On I-8000 is to light the LED "L1".) **OFF**: Turn off the red LED. (Note: On I-8000 is to light the LED "L1".)
2	**LED5 pos value**	Show digit on chosen position of 5DigitLed.

	(see “Deep Description” below)	
3	**USE NVRAM** (see “Deep Description” below)	Go into the sub-function for read/write NVRAM. (The function works only with RTC hardware.)
4	**USE EEPROM** (see “Deep Description” below)	Go into the sub-function for read/write EEPROM.
5	**USE FLASH** (see “Deep Description” below)	Go into the sub-function for read/write/erase FLASH memory. (Also can read SRAM.)
6	**USE COM2 /option** **USE COM0 /option** (For I-8000 use "use com0" to test 87K board.) (see “Deep Description” below)	Go into the sub-function for COM2(RS-485).(Like the function of "echo485.exe" on I-7188.) **option** can be as follows: **a**xx : Set the start address of auto scan. **b**xxxx : Set baud rate. **c**xxxx : Send command directly to COM2. **d**x : Set data bits, 7 or 8. **e**xx : Set the end address of auto scan. **p**x : Set Parity.(0:NONE, 1:EVEN, 2:ODD) **q**: Auto quit after send command. **t**xx : Set Timeout period(ms).
7	**DATE [new_date]**	Set/Read system date.(work with RTC.) **[new_date]**: (month)/(day)/(year) For example: i7188>date 12/11/2006
8	**TIME [new_time]**	Set/Read system time.(work with RTC.) **[new_time]**: (hour)/(minute)/(second) For example: i7188>time 5:30:30
9	**MCB**	Display the amount and status of SRAM.
10	**UPLOAD**	The 1st step for updating MiniOS7. (Save the image file to SRAM.)
11	**BIOS1**	The 2nd step for updating MiniOS7. (Write the image file to FLASH memory.)(After updating, the module will be reset.)
12	**LOAD** (see “Deep Description” below)	Download file to FLASH memory.
13	**DIR [/crc]**	Display information about files saved in Flash

	(see “Deep Description” below)	memory. Using the option **/crc** will also do CRC-16 check.
14	**RUN fileno**	Execute the file that file index is "**fileno**". (only .BAT .COM .EXE files can be executed.) File index begins from 0, use command **DIR** will show the file index.
15	**file name**	Run that file. For example: i7188>echo485 (if not key-in the full filename, MiniOS7 will search ".BAT" first, then ".COM" and final ".EXE". If the filename is the same as command of MiniOS7, please use full filename.(For example: "led.exe")
16	**DELETE [/y]** **DEL [/y]**	DELETE all files saved in Flash memory.(it will ask user to press 'y' then DELETE the files.) Option : **/y**. Delete files directly. For example: “i7188>del /y” or “i7188>del”
17	**RESET**	Reset the system. The command "RESET" is used to software RESET, the hardware signal /RESET will not be send out. If want the hardware signal /RESET to be send out, please use **DIAG WDT** or use turn POWER off to reset system.
18	**DIAG [option]**	options are as follows: **ram** : check SRAM **eeprom** : check EEPROM **nvram** : Check NVRAM **led** : check LED **led5** : check 5DigitLed **flash** : check FLASH MEMORY **wdt** : check WatchDog Timer **clock** : (check)display system TIMER ticks. **All** : Check all items of above. **ON** : Set to show the debug message. When the file is running, there will be debug messages to show from COM1. (i-7188xa/i-7188 is COM4)

		OFF : Set to hide the debug message. If no option will show the INIT* status, SRAM/FLASH size, system reset status,and so on.
19	**BAUD baudrate**	Set new BAUDRATE(300-115200). (There will be messages to be sent from COM1 (i-7188xa/ i-7188 is COM4) to show baud rate information.) @BAUD baudrate: The same function, but the messages won't be sent. (For use on autoexec.bat)
20	**TYPE filename [/b]**	Display the contents of the specified file. use **/b** will show hex code.
21	**REP [/#] COMMAND**	Repeat the COMMAND [#] times. If the repeat times don't be specified, it will repeat infinitely.
22	**RESERVE [n]**	If the parameter n is used, it will reserve n sectors of Flash Memory for user to store data. If the command "**reserve**" only be type, it will show the reserved sector number of Flash Memory.
23	**LOADR** (see "Deep Description" below)	The same function as "**LOAD**", but the file is stored to SRAM. Only one file can be stored in SRAM. Use LOADR again will overwrite the last file. Use the command "**runr**" to run the program stored in SRAM.
24	**RUNR [param1 [param2...]]**	Run the program stored in SRAM.(only one file, need not key-in the filename.)
25	**I/INP port** **IW/INPW port**	Read data from Input PORT. **port** is the port to read, in HEX. I/INP read one BYTE(8-BITS mode). IW/INPW read one WORD(16-BITS mode). For example: i7188>inp 10 or i7188>i 10
26	**O/OUTP port value** **OW/OUTPW port value**	Output data to the Output PORT. **port** is the output port, in HEX. **value** is the output data, in HEX. O/OUTP output one BYTE(8-BITS mode). OW/OUTPW output one WORD(16-BITS mode).

27	**DI**	Read the status of DI1/2/3 pins. 0:LOW,1:HIGH.
28	**DO1 0/1**	Set the output value of DO1 pin.**[Note 1]** 0 : output LOW,1 : output HIGH.
29	**DO2 0/1**	Set the output value of DO2 pin.**[Note 1]** 0 : output LOW,1 : output HIGH.
30	**DO3 0/1**	Set the output value of DO3 pin.**[Note 1]** 0 : output LOW,1 : output HIGH.
31	**OUTWAVE port freq hi lo**	Set TO_0/TO_1 to output DUTY CYCLE waveform.**[Note 2], [Note 3]** 1. port : 0, set output from TO_0 pin. 1, set output from TO_1 pin. 2. freq : frequency. When CPU clock=40M Hz, Maximun frequency is 5M Hz,Minimun frequency is 77Hz. When CPU clock=20M Hz, Maximun frequency is 2.5M Hz, Minimun frequency is 77Hz. 3. hi/lo : Set the high/low ratio. (Use HI:LO=30:70 is the same as HI:LO=3:7 or HI:LO=6:14)
32	**STOPWAVE port**	Stop to output DUTY CYCLE waveform from TO_0/TO_1. **[Note 3]** port : 0, stop output from TO_0 pin. 1, stop output from TO_1 pin.
33	**SETDO pin 0/1**	Set output status for I/O BUS.(only when the PIO pin is set on output mode will work. The PIO pins that user can use please refer to "SETPIO".) 0 : output LOW, 1: output HIGH. **[Note 3]**
34	**SETPIO pin [mode]**	(Please use it carefully.) **[Note 3]** Set the mode of PIO pin. 1. pin : AM188ES PIO pin number. (Please refer to User'S Manual of AM188ES.) The following pins can be used by user: (pin used will be different between modules.)

		a. TO_0(pin=10). b. TO_1(pin=1). c. TI_0(pin=11). d. TI_1(pin=0). e. DIO4(pin=4). f. DIO9(pin=9). g. DIO14(pin=14). 2. mode = 0 : NORMAL, 1 : input with pull up/down, 2 : output, 3 : input without pull up/down. If the parameter "mode" is not typed, it will show the current mode of the PIO pin.
35	**DISKSIZE [A size] [B size] [Reserve]**	Divide FLASH ROM into three parts which are diskA, diskB and Reserve. **FLASH 256K:** three segments can be used. One segment is 64K bytes. **FLASH 512K:** seven segments can be used. One segment is 64K bytes. For example(512K): i7188>disksize 2 2 3 or i7188>disksize 7
36	**DELB [/y]**	It is the same as DEL [/y]. The only difference is DEL [/y] is diskA, DELB [/y] is diskB.
37	**LOADB** (see “Deep Description” below)	It is the same as LOAD. The only difference is LOAD is diskA, LOADB is diskB.

Note 1 : The output pins(DO1/2/3) of 7188XC series are Open Collector, user maybe add pull up resister to work well.

Note 2 : The DUTY CYCLE wave form output is not for any ratio of HI:LO, and not for any frequency.
The output frequency is equal to **CPU CLOCK/4/(high+low)**, and **high+low <= 65536**, and minimum value of high/low is 1.

Note3: Please refer to “I/O Expansion Bus for 7188X/7188E User’s Manual” to know TO_0/TO_1/TI_0/TI_1/DIO4/DIO9/DIO14.
Location: http://ftp.icpdas.com/pub/cd/8000cd/napdos/7188xabc/xboard/document/

Deep Description

led5

command --> **led5 position[.] value**

- position = **from 1 to 5**, the **.** will show the decimal-point.
- value = **from 0 to 17**. The **Hxxxx** can be used for Hex value.
- **led5 time on** will show the time information.
- **led5 time off** will stop the showing of time information.
  (By default the MiniOs7 will show the system operating time on 5-DigitLed.)

```
Step1:i7188>led5 time off --> stop the showing of time information
Step2:i7188>led5 1 7
Step3:i7188>led5 2 1
Step4:i7188>led5 3 8
Step5:i7188>led5 4. 8
Step6:i7188>led5 5 13 --> Now the‘d’will be shown in the 5-digit LED.
Step7:i7188>led5 time on --> enable the showing of time information
```

use nvram

- input **address(0-30)** & Enter can read the value of NVRAM.
- input **address value** & Enter can write the value to the NVRAM.
- input **address w** & Enter will show the 16-bit value.
- input **address w value** & Enter can write 16-bit data to NVRAM.
- **quit or exit** can return to the original mode.

```
i7188>use nvram
[NVRAM:]0
```

```
[0]=100(64)
[NVRAM:]1
[1]=200(C8)
[NVRAM:]2
[2]=152(98)
[NVRAM:]3
[3]=255(FF)
[NVRAM:]0 123
Write 123 to [0]
[NVRAM:]1 122
Write 122 to [1]
[NVRAM:]2 w 65535
write -1(FFFF) to Word[02]
[NVRAM:]0
[0]=123(7B)
[NVRAM:]1
[1]=122(7A)
[NVRAM:]2
[2]=255(FF)
[NVRAM:]3
[3]=255(FF)
[NVRAM:]exit
i7188>
```

use eeprom

- **BLOCK（0-7）** --> Select the active block
- **d[b|w|L] address** --> Dump the continuous 64 bytes data start from address.
  b: Unit=byte, w: Unit=word, L: Unit=long word.
- **E[b|w|L] address** --> Write continuous 64 bytes data start from address.
  b: Unit=byte, w: Unit=word, L: Unit=long word.
  It will show the old value first and then wait for new value.
  After inputting the new value, the next address value will be shown and waited for input. The character '.' can stop this command.
- **Quit or exit** --> return to the original mode.

```
[EEPROM:]block 4
```

```
Set Current Block to 4
[EEPROM:]d 0
Block 4
B(0):004[04]. 005[05]. 006[06]. 007[07]. 008[08]. 009[09]. 010[0A]. 011[0B].
B(8):012[0C]. 013[0D]. 014[0E]. 015[0F]. 016[10]. 017[11]. 018[12]. 019[13].
B(10):020[14]. 021[15]. 022[16]. 023[17]. 024[18]. 025[19]. 026[1A]. 027[1B].
B(18):028[1C]. 029[1D]. 030[1E]. 031[1F]. 032[20]  033[21]! 034[22]" 035[23]#
B(20):036[24]$ 037[25]% 038[26]& 039[27]' 040[28]( 041[29]) 042[2A]* 043[2B]+
B(28):044[2C], 045[2D]- 046[2E]. 047[2F]/ 048[30]0 049[31]1 050[32]2 051[33]3
B(30):052[34]4 053[35]5 054[36]6 055[37]7 056[38]8 057[39]9 058[3A]: 059[3B];
B(38):060[3C]< 061[3D]= 062[3E]> 063[3F]? 064[40]@ 065[41]A 066[42]B 067[43]C
[EEPROM:]e 0
Block 4
B(0)=4 --> 0
B(1)=5 --> 1
B(2)=6 --> 2
B(3)=7 --> 3
B(4)=8 --> .
[EEPROM:]ew 4
Block 4
Word(4)=2312 --> 123
Word(6)=2826 --> 124
Word(8)=3340 --> 1255
Word(10)=3854 --> .
[EEPROM:]dw 4
Block 4
W(004):000123[007B] 000124[007C] 001255[04E7] 003854[0F0E]
W(012):004368[1110] 004882[1312] 005396[1514] 005910[1716]
W(020):006424[1918] 006938[1B1A] 007452[1D1C] 007966[1F1E]
W(028):008480[2120] 008994[2322] 009508[2524] 010022[2726]
W(036):010536[2928] 011050[2B2A] 011564[2D2C] 012078[2F2E]
W(044):012592[3130] 013106[3332] 013620[3534] 014134[3736]
W(052):014648[3938] 015162[3B3A] 015676[3D3C] 016190[3F3E]
W(060):016704[4140] 017218[4342] 017732[4544] 018246[4746]
[EEPROM:]dl 0
Block 4
L(000[0]):00050462976[00000100] 00008126587[0000007B]
L(008[8]):00252576999[000004E7] 00319951120[00001110]
```

```
L(016[10]):00387323156[00001514] 00454695192[00001918]
L(024[18]):00522067228[00001D1C] 00589439264[00002120]
L(032[20]):00656811300[00002524] 00724183336[00002928]
L(040[28]):00791555372[00002D2C] 00858927408[00003130]
L(048[30]):00926299444[00003534] 00993671480[00003938]
L(056[38]):01061043516[00003D3C] 01128415552[00004140]
[EEPROM:]exit
i7188>
```

use flash

The unit of EEPROM is BLOCK, the unit of Flash-ROM is SEGMENT. ADDRESS is offset in the active SEGMENT.

After type command **"USE FLASH"** can use the following commands:

1. **segment nnnn:** Set new segment(HEX value), range of nnnn is 0-F000.(Both the range of Flash or SRAM can be used.)
   SRAM address: Starting address is 0x0000. Ending address is 0x#000 (#=(SRAM size/64K)-1)
   FLASH address: 0xC000~0xE000(256K), 0x8000~0xE000(512K)
2. **d[b|w|L] address：**
   Dump 64 bytes data from the address.
   b: Use byte as dump unit. **w**: use word as dump unit, **L**: use long(dword) as dump unit.
3. **E[b|w|L] address：**
   Input new data to FLASH from the address of current block. (Only when current segment is on FLASH, it will work.)
   b: Use byte as dump unit. **w**: Use word as dump unit, **L**: Use long(dword) as dump unit.
   It will display current value first and wait user to key-in the new value. Then displaying next address value and wait for input. User can type next value or just type character '.' to end input.
4. **erase segment:** Erase the specified segment of Flash memory.(all 64K data will become 0XFF.)
5. **quit / exit :** Exit "USE FLASH".

Command Examples:

```
i7188>use flash
Manufacturer=C2(MXIC) ID=B0(size=256K) Protect mode=x0000000
[Flash:]segment c000
Set Current Segment to C000
[Flash:]d
Segment C000
0000: 88 71 75 61 72 74 2E 68 00 00 00 00 00 00 14 0C | equart.h     ?♀
0010: 1F 12 18 11 B1 05 00 00 00 00 02 C0 0D 38 FF FF | ▼?↑???    ?└.8??
0020: 0D 0A 23 64 65 66 69 6E 65 20 54 78 62 75 66 09 | ..#define Txbuf○
0030: 20 20 30 78 30 30 09 20 20 20 2F 2A 20 74 78 20 |   0x00○   /* tx
0040: 62 75 66 66 65 72 20 2A 2F 0D 0A 23 64 65 66 69 | buffer */..#defi
0050: 6E 65 20 52 78 62 75 66 09 20 20 30 78 30 30 09 | ne Rxbuf○  0x00○
0060: 20 20 20 2F 2A 20 72 78 20 62 75 66 66 65 72 20 |    /* rx buffer
0070: 2A 2F 0D 0A 23 64 65 66 69 6E 65 20 44 6C 6C 09 | */..#define Dll○

[Flash:]erase e000
Segment E000 is being erased....
Erase Segment E000
success
Loop=-31511
[Flash:]segment e000
Set Current Segment to E000
[Flash:]d
Segment E000
0080: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0090: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
00A0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
00B0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
00C0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
00D0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
00E0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
00F0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |

[Flash:]e 0
Segment E000
B(0000)=FF --> 30
B(0001)=FF --> 31
B(0002)=FF --> 32
```

```
B(0003)=FF --> 33
B(0004)=FF --> 34
B(0005)=FF --> 35
B(0006)=FF --> 36
B(0007)=FF --> 37
B(0008)=FF --> .
[Flash:]d 0
Segment E000
0000: 30 31 32 33 34 35 36 37 FF FF FF FF FF FF FF FF | 01234567
0010: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0020: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0030: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0040: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0050: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0060: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |
0070: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |

[Flash:]ew 10
Segment E000
Word(000A)=-1 --> 0123
Word(000C)=-1 --> 4567
Word(000E)=-1 --> 89AB
Word(0010)=-1 --> CDEF
Word(0012)=-1 --> .
[Flash:]d 0
Segment E000
0000: 3130 3332 3534 3736 FFFF 0123 4567 89AB
0010: CDEF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0020: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0030: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0040: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0050: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0060: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0070: FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF

[Flash:]dl 0
Segment E000
0000: 33323130 37363534 0123FFFF 89AB4567
```

```
0010: FFFFCDEF FFFFFFFF FFFFFFFF FFFFFFFF
0020: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
0030: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
0040: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
0050: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
0060: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
0070: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF

[Flash:]exit

i7188>
```

use com2

use com2 [/option]
The valid values of option are given as follows:
a#--> Set the starting address of auto scan
b#--> Set baud rate.
c#--> Send command directly to COM.
d#--> set the data-bit, 7 or 8.
e#--> Set the ending address of auto scan.
p#--> Set the parity-bit. (0:NONE, 1: EVEN, 2:ODD)
q --> Auto quit after send command.
t#--> Set Timeout period(ms).
The valid commands after type **use com2** are given as following:
/q: quit
/scan: Auto scan i-7000 modules.（Default searching address is from 00 to FF）
/s0: Checksum OFF
/s1: Checksum ON
/start [address]: Set the starting-address of scan-function.
/end [address]: Set the ending-address of the scan-function
/baud [baudrate]: Change baud rate
????? : Directly send **????** to COM2. For example, **$01M，$012**
/timeout [timeout]: Set new timeout. Unit is ms.

Example1:

```
“i7188>use com2 /b9600” or “i7188>use com2 /a1” or
“i7188>use com2 /d7”
```

Example2:

```
i7188>use com2
/? : show command help
Current setting : baudrate=9600 --> timeout=4 ms
StartAddr=0 EndAddr=10
[COM2:]/baud 115200
Change baudrate to 115200
[COM2:]/start 0
StartAddr=0(00)
EndAddr=10(0A)
[COM2:]/end 10
StartAddr=0(00)
EndAddr=10(0A)
[COM2:]/scan
Scan 01!017060{1ms}
Scan 0A
[COM2:]$01M
Com2Echo]!017060{2ms}
[COM2:]$012
Com2Echo]!01400A01{2ms}
[COM2:]$02M
Com2Echo]!!! Timeout !!!
[COM2:]/q
i7188>
```

load/dir

Input **load** first and there will be a message to ask you to press **ALT_E**. Then input **filename** & Enter to download this file into Flash-Memory. Now the command **“dir”** can show all information about this file.

Example1:

```
i7188>dir
( 0)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C000:001C-C321:000A
```

```
( 1)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C321:0026-C643:0004
( 2)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C643:0020-C964:000E
Total File number is 3 Free space=158102 bytes
i7188>load
File will save to C964:000E
StartAddr-->C000:964D
Press ALT_E to download file!

FileName:echo485.exe
Size=12544
Download 12798 Bytes OK.
Transfer time is: 2.197802 seconds

Test count=72309
Back to Terminal mode
i7188>dir
 0)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C000:001C-C321:000A
 1)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C321:0026-C643:0004
 2)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C643:0020-C964:000E
 3)echo485.exe 12/20/1999 15:49:38 12798(0x031FE) C964:002A-CC86:0008
Total File number is 4 Free space=145276 bytes
i7188>
```

Example2:

```
i7188>dir
 0)demo6.exe 12/18/1999 11:56:21 10670[029AE]C002:0000-C29C:000E
 1)echo485.exe 02/23/2000 14:20:00 13088[03320]C29E:000E-C5D0:000E
 2)echo485.exe 02/23/2000 14:20:00 13088[03320]C5D2:000E-C904:000E
 3)echo485.exe 02/23/2000 14:20:00 13088[03320]C906:000E-CC38:000E
 4)echo485.exe 02/23/2000 14:20:00 13088[03320]CC3A:000E-CF6C:000E
Total File number is 5 Free space=133394 bytes

i7188>dir /crc
 0)demo6.exe 12/18/1999 11:56:21 10670[029AE]C002:0000-C29C:000E{CRC OK}
 1)echo485.exe 02/23/2000 14:20:00 13088[03320]C29E:000E-C5D0:000E{CRC OK}
 2)echo485.exe 02/23/2000 14:20:00 13088[03320]C5D2:000E-C904:000E{CRC OK}
 3)echo485.exe 02/23/2000 14:20:00 13088[03320]C906:000E-CC38:000E{CRC OK}
 4)echo485.exe 02/23/2000 14:20:00 13088[03320]CC3A:000E-CF6C:000E{CRC OK}
```

```
Total File number is 5 Free space=133394 bytes

i7188>
```

loadr

Command **“LOADR”** is used to download file to SRAM of MiniOS7 system.

1. After type command **“LOADR”**, MiniOS7 will show message to ask user press ALT_E. After pressing ALT_E, user must type the file name.

2. After the file is downloaded into the system. Using command dir will don't show the message of the information of this file because the file is saved on SRAM, not Flash.

3. If the file isn't downloaded successfully, please try use any of the following methods to download again:
   I. Please use command **“reset”** to reset system and try again.
   II. Use command **“baud”** for set to lower communication speed. For example 57600, then try again.
   III. Press enter key some times to ensure no more other data in the input buffer of the system.
   IV. Don't press ALT_E too early, it must wait the screen show the message and then press ALT_E.

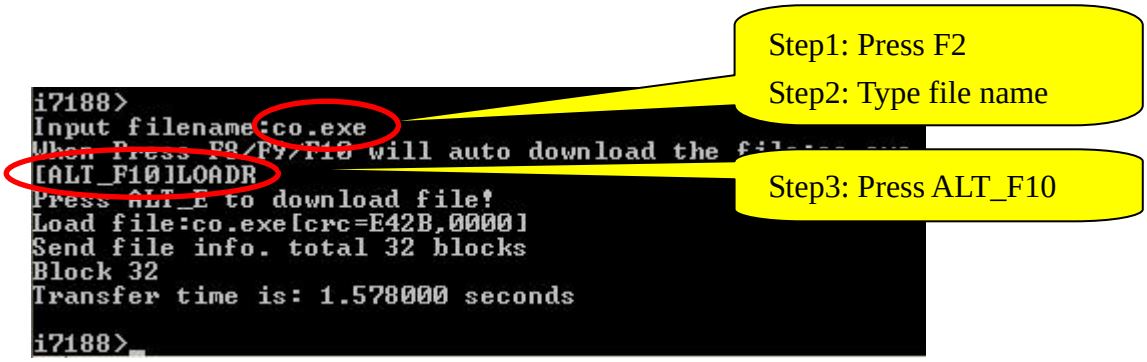
4. The file had already been downloaded to SRAM. If the system is powered off or reset, the file will disappear. Besides, downloading another new file to the system, the old will be replaced by new because only one file can be downloaded to SRAM. If the size of the file is larger, it isn't executed on SRAM because the memory isn't enough.

5. Use command **“LOADR”** to download the file to SRAM. Only the command **“RUNR”** can execute this file.

```
i7188e>dir

Total File number is 0  Free space=458720 bytes
i7188e>loadr
Press ALT_E to download file!
Input filename:vcom3007.exe
Load file:vcom3007.exe
Send file info. total 367 blocks
Block 367
Transfer time is: 12.168000 seconds

i7188e>dir

Total File number is 0  Free space=458720 bytes
i7188e>runr
```

Press the F2 to set the file for auto download process, then there are two choice:

(1) Press ALT+F10. It will auto download the file to SRAM. Type command **“runr”** to execute file.



(2) Press F10. It means the file will be downloaded and run automatically right away.