How to operate the text file in MiniOS7

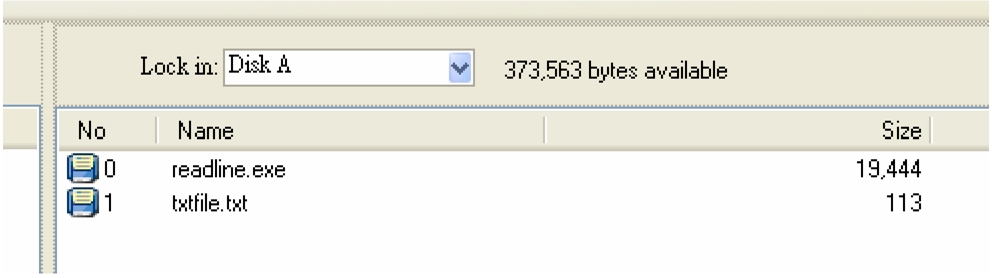
Part I: Quick start for this demo program.

Please follow steps below to run this demo program.

1. There are many applications need to use text format configure or ini file to provide parameters for program, MiniOS7 library can use on board Flash to store exe file and configure file. This demo will show how to read text files line by line. If you need to use ini file, you can refer to another INIRead demo.

You can follow steps as below to test this demo program.

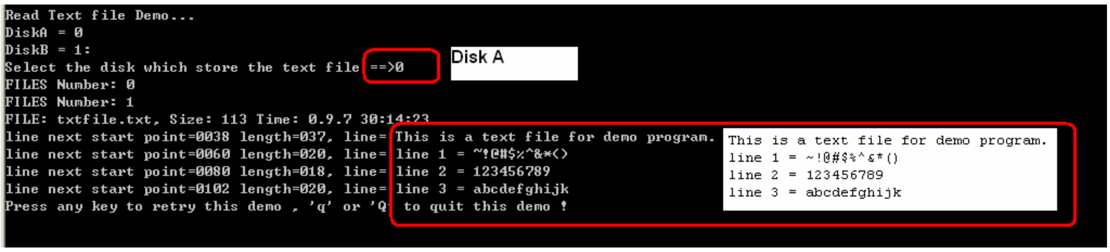
1. Load both exe and txt file in disk A.



Lock in: Disk A   373,563 bytes available

No	Name	Size
0	readline.exe	19,444
1	txtfile.txt	113

2. Run the readline.exe then select

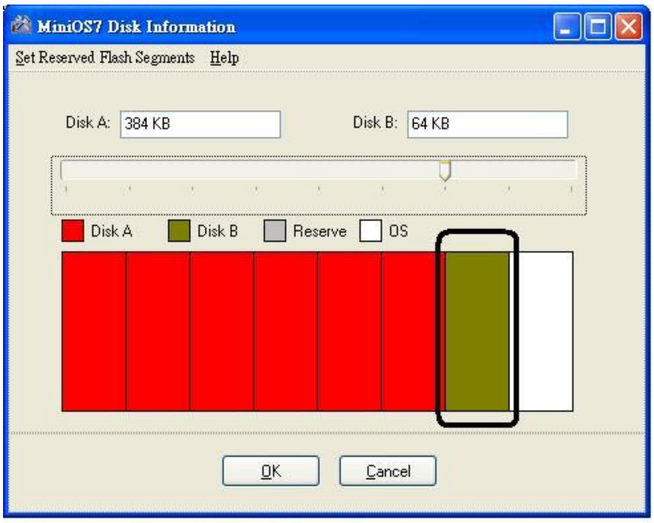


```
Read Text file Demo...
DiskA = 0
DiskB = 1:
Select the disk which store the text file ==>0
FILES Number: 0
FILES Number: 1
FILE: txtfile.txt, Size: 113 Time: 0.9.7 30:14:23
line next start point=0038 length=037, line= This is a text file for demo program.
line next start point=0060 length=020, line= line 1 = ~!@#$%^&*()
line next start point=0080 length=018, line= line 2 = 123456789
line next start point=0102 length=020, line= line 3 = abcdefghijk
Press any key to retry this demo , 'q' or 'Q' to quit this demo !
```

Disk A

```
This is a text file for demo program.
line 1 = ~!@#$%^&*()
line 2 = 123456789
line 3 = abcdefghijk
```

3. Run exe file in Disk A and read text file from Disk B. Please use MiniOS7 utility to set disk as A, and B

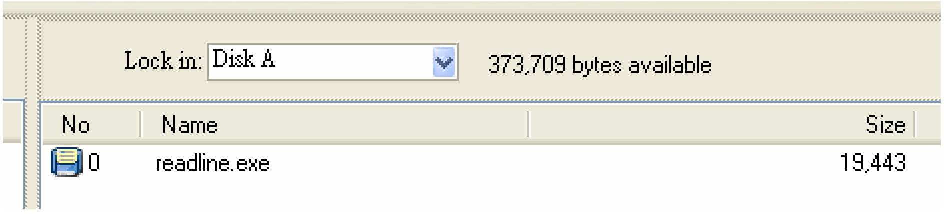


Disk A: 384 KB

Disk B: 64 KB

use Disk A to store the exe file
use Disk B to store the config file

4. Load exe file to disk A

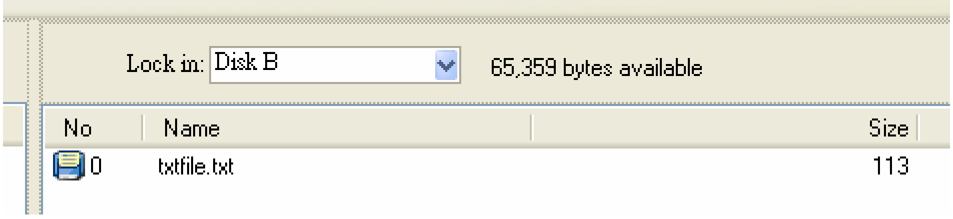


Look in: Disk A

373,709 bytes available

No	Name	Size
0	readline.exe	19,443

5. Load configure file or text file to disk B

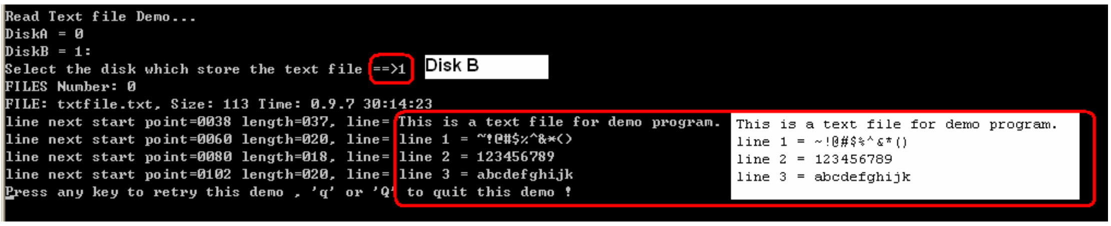


Look in: Disk B

65,359 bytes available

No	Name	Size
0	txtfile.txt	113

5. Run ReadLine.exe demo and use 7188xw.exe to check the result.



```
Read Text file Demo...
DiskA = 0
DiskB = 1:
Select the disk which store the text file ==>1
FILES Number: 0
FILE: txtfile.txt, Size: 113 Time: 0.9.7 30:14:23
line next start point=0038 length=037, line= This is a text file for demo program.
line next start point=0060 length=020, line= line 1 = ~!@#$%^&*()
line next start point=0080 length=018, line= line 2 = 123456789
line next start point=0102 length=020, line= line 3 = abcdefghijk
Press any key to retry this demo , 'q' or 'Q' to quit this demo !
```

```
This is a text file for demo program.
line 1 = ~!@#$%^&*()
line 2 = 123456789
line 3 = abcdefghijk
```

Part II: Description about API function in this demo.

Functions of MiniOS7 Library will be mark with blue color:

1. #define FILE FILE_DATA far

In MiniOS7, it use FILE_DATA as file structure, we define FILE as FILE_DATA far so that it will look like C style.

File structure in MiniOS7 system define as below.

```
typedef struct   {
   unsigned mark;     /* 0x7188 -> is file */
   unsigned char fname[12];
   unsigned char year;
   unsigned char month;
   unsigned char day;
   unsigned char hour;
   unsigned char minute;
   unsigned char sec;
   unsigned long size;
   char far *addr;
   unsigned CRC;
   unsigned CRC32;
} FILE_DATA;
```

2. int GetFileNo_AB(int disk);

To get how many files in disk A or B

Parameters:

disk: 0 diskA ; 1 diskB

return how many files in disk.

example:

```
int n, diskB=1;
n = GetFileNo_AB(diskB);
```

3. FILE_DATA far * GetFileInfoByNo_AB(int disk,int no);

To get file structure information by using disk location and file index.

Parameters:

disk: 0 diskA ; 1 diskB

no: file index in disk, file index begin at 0.

Return FILE structure

Example:

```
static FILE *fd; // We declare FILE_DATA far as FILE
int diskB=1;
fd = GetFileInfoByNo_AB(diskB,0);
```

➔ then to get file name, size and time form FILE structure.

```
Print("FILE: %s, Size: %d Time: %d.%d.%d %d:%d:%d\r\n",fd->fname,
fd->size,fd->day,fd->month,fd->year, fd->hour,fd->minute,fd->sec);
```

Note: Use GetFileNo_AB to get how many file in disk then use GetFileInfoByNo_AB to get FILE structure by file index.

4. int ReadLine (FILE *fd, char *line , int startPoint);

Parameters:

fd: FILE structure get from GetFileInfoByNo_AB.

line: char buffer for read data.

startPoint: the start point offset for ReadLine function to read data from file.

Return next start point for this ReadLine function to in this file.

Example:

```
int startPoint=0, nextPoint=0;
nextPoint =ReadLine(fd,line,startPoint);
```

5. int GetHighRamMode(void)

To check if MiniOS7 library use high RAM mode or not.

Parameter:

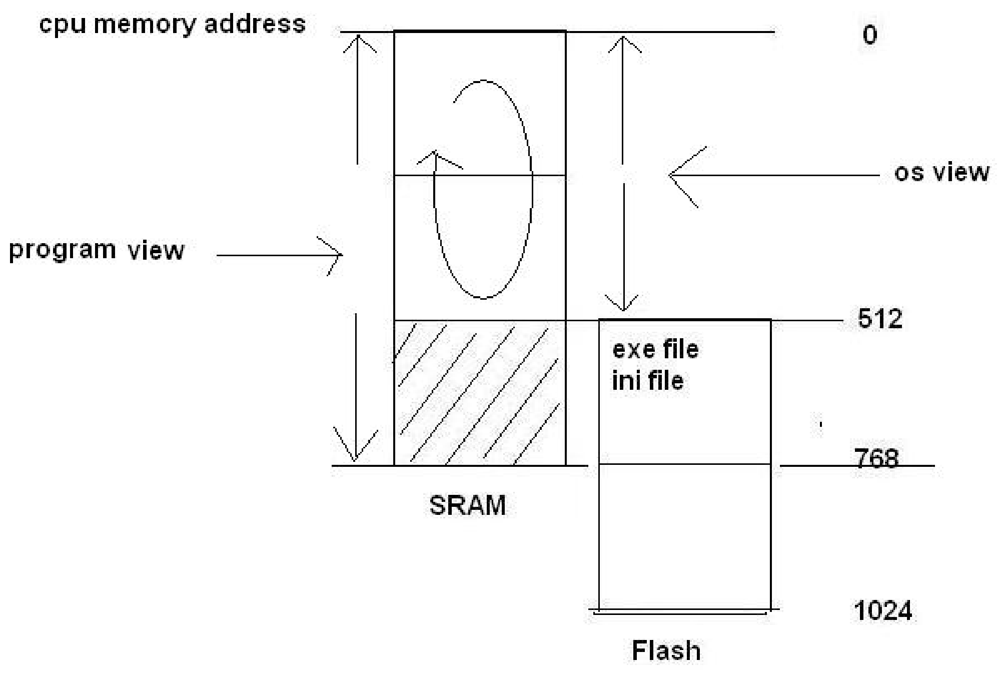
None.

Return 0 don't need to use high RAM mode, 1 need to use high RAM mode.

There are some MiniOS7 controller equipped with 768KB SRAM, for CPU memory address, there is an overlap area below, if we put configure file in this overlap area and when program runs, program can not "See" this configure file. We have to use following code to disable 768KB High SRAM flag to let program can "See" this configure file.

```
if(bNeedDisableHighRam)
{
        HighRamMode=GetHighRamMode();
        Print("High RAM Mode = %d\n",HighRamMode);
        if(HighRamMode)
            DisableHighRam();
}
```

Note: bNeedDisableHighRam is a global variable extern by MiniOS7 library.


cpu memory address
0
os view
program view
512
exe file
ini file
768
SRAM
1024
Flash