Verix Developer Training

Files

The File Manager

§ File system implemented in non-volatile RAM and Flash

§ Files may be accessed in RAM or Flash using a drive prefix

- *I:* RAM File System   (default)
- *F:* Flash File System

§ Files are assigned to groups (0-15) with access control

§ Groups can span both RAM and Flash

§ Files are added to the directory in the order they are created

- Downloading
- Creating a new file

VeriFone.

The Verix File Manager

§ Groups are not specified as part of filenames

§ Group 1 tasks are allowed to change their effective group
  - `set_group()` Allows access to files in other groups
  - `get_group()` Retrieves the current group setting

§ Any task can change its group to 15

§ Group 15 files can be accessed by a slash(/) prefix to the filename
  - `/batch.dat` Access "batch.dat" in Group 15 RAM
  - `F:/cardlist` Access "cardlist" in Group 15 Flash

VeriFone.

Files in Flash

§ Limit the use of the Flash file system to files that rarely or never change

- *.out files
- Font files
- Message files

Files and Memory

§ Memory manager attempts to recover unused memory space

§ Design file management routines to minimize the need to increase or decrease the file size

- Reuse record space to improve performance

Files and Memory

§ File extension feature allows specification of maximum file size

- Called "Memory Padding"
- Declare maximum size (in bytes) of file v/s writing blank records and overwriting
- Required space is immediately claimed
- Future operations on the file do not disturb other files in the system
- `get_file_max()` â `get_file_attributes()`
- `set_file_max()` â `set_file_attributes()`
- `get_file_size()` â `reset_file_attributes()`

VeriFone.

File Naming & Handles

§ File Naming

- 32 characters, maximum
- Must be NULL terminated
- Not case sensitive

§ File Handles

- 30 file handles available
- Up to 30 files may be open at any one time
- System files are not counted in the 30 file limit
- A single file may have multiple file handles
- Can change the maximum number of file handles
    - *FILE in config.sys (Group 1)
- Only one flash file can be opened with write access at any time

VeriFone.

File Types

§ Binary (Generic)

§ Variable Length Record (VLR)

§ Compressed Variable Length Record (CVLR)

§ Keyed

Generic Files

§ May contain any type of data

§ Data accessed by byte address within the file

§ Normally each record is of fixed length defined by the size of a data structure

Variable Length Record (VLR) Files

§ Implemented as a counted string - transparent to user

§ Allows data to be stored and accessed as records

§ Data is accessed by record number

§ Requires the file be processed from the beginning

§ Suited for chronologically sequenced records with little or no random access requirements

§ Limited to 254-byte records

VeriFone.

Compressed Variable Length Record (CVLR) Files

§ Same as VLR...Plus...

§ File system compresses/decompresses records

§ Compression technique:

- Numeric characters compressed to 1/2 byte
- Lower case characters converted to upper case equivalent.
- Byte values greater than 0x5F cannot be correctly translated -- should not be used.
- Byte boundaries ignored

§ File space saving noticeable

Keyed Files

- § Paired File
- § Double CVLR file
  - Uses two records for each record
    - A key
    - Associated data value
- § Slowest
  - Requires two reads
  - Comparison of the key
- § Not index sequential
- § Ordered at time of creation
- § Config.Sys is an example

VeriFone.

Opening/Closing Files

§ Opening a File: `hFile = open(filename, attributes);`

§ Attributes:
- `O_RDONLY`   Read access only
- `O_WRONLY`   Write access only
- `O_RDWR`   Read/write access
- `O_APPEND`   Opens with file pointer at EOF
- `O_CREAT`   Creates a new file
- `O_TRUNC` Opens file & truncates length to 0
- `O_EXCL`   Error value for `O_CREAT`

§ Once a file has been opened, set the seek pointer to the appropriate location before attempting to read or write

§ Closing a File: *status = close(hFile);*

VeriFone.

Suggested Way of Opening Files

§ To prevent overwriting data in an existing file, instead of `O_CREAT | O_RDWR` use

```
while((FileHdl = open("data.bat", O_RDWR)) < 0)
{
        hFile = open("data.bat", O_CREAT);
        close(hFile);
}
```

§ Another method

```
hFile = open("trans.dat", O_WRONLY|O_APPEND);
if (hFile < 0)
{
        hFile = open("trans.dat", O_CREAT);
        close (hFile);
}
```

VeriFone.

File Positioning

§ Setting the read/write pointer:

```
                    lseek
    position =  seek_vlr(hFile, offset, origin);
                    seek_cvlr
```

§ Offset:

- Long integer

§ Origin is where to begin searching:

- Beginning of file: **SEEK_SET**
- Current point location: **SEEK_CURRENT**
- End of file: **SEEK_END**

File Positioning

Beginning of File

SEEK_SET

SEEK_CURRENT

SEEK_END

End of File

Read / Write

§ Reading from a file:

```
                 read
   bytes_read =  read_vlr(hFile, buffer, len);
                 read_cvlr
```

§ Writing to a file:

```
                  write
   bytes_written = write_vlr(hFile, buffer,
len);
                  write_cvlr
```

Insert / Delete

§ Inserting a record to a file:

```
                    insert
    bytes_read =  insert_vlr(hFile, buffer,
len);
                    insert_cvlr
```

§ Deleting a record from a file:

```
                    delete
    bytes_written = delete_vlr(hFile, buffer,
len);
                    delete_cvlr
```

File Utilities

§ Deleting a file:

```
status = _remove("filename");
```

- File must be closed before it can be removed
- All file handles must be released

§ Renaming a file:

```
status = _rename("old_name", "new_name");
```

§ Lock/Unlock a file:

```
status = lock(hFile, 0, 0);
status = unlock(hFile, 0, 0);
```

VeriFone.

File Utilities

§ Calculate file checksum and compare to file header
- `SVC_CHECKFILE()`

§ Get the checksum for a file
- `dir_get_checksum()`

§ Date of last file modification
- `get_file_date()`
- `dir_get_file_date()`

§ Attach a date to a file (downloaded using SVC_ZONTALK())
- `dir_put_file_date()`

File Utilities

§ `dir_get_file_size()`

- Returns the size of the file

§ `dir_get_file_sz()`

- Returns the number of data bytes in the named file

§ `dir_get_attributes()`

- Access file attribute bits associated with a file

§ `dir_set_attributes()`

- Turns on one or more attribute bits for a specified file

§ `dir_reset_attributes()`

- Turns off one or more attribute bits for a specified file

VeriFone.

File Utilities

§ `SVC_RAM_SIZE()`

- Amount of RAM memory installed in the device

§ `SVC_FLASH_SIZE()`

- Amount of Flash memory installed in the device

§ `dir_get_sizes()`

- Number of files in the directory
- Amount of memory used by the file system
- Amount of free memory remaining

VeriFone.

File Utilities

§ `dir_flash_coalesce_size()`
- Returns the number of bytes to reclaim with a coalesce

§ `dir_flash_coalesce()`
- Erase all files marked for deletion and reclaim unused memory

§ `dir_get_first()`
- Returns the name of the 1st file in the directory

§ `dir_get_next()`
- Used after dir_get_first()
- Provides the name of the next file in the directory

Config.sys

§ Contains system options and parameters

§ Configure the system environment

§ Double CVLR file maintained as a "keyed" file.

§ Entries stored in pairs of variable length records: key=data

- key $\leq$ 32 bytes   (7 bytes if Zontalk)
- data $\leq$ 128 bytes

§ Values of keys preceded with * or # are preserved during full downloads.

* System variables only

#Application variables

Examples:

*GO=myapp.out

*SMDL=1

#MNAME=MY IMPORT SHOP

Config.sys

§ There can be up to 15 config.sys files

§ One for each file group in use

§ System properties are maintained in file group 1

- Apps in other file groups can not change the system properties

VeriFone.

§ Access config.sys in three ways:
- – System Mode
- – Download
- – Application

§ `int get_env(key, data, num_bytes);`

Retrieves the value of an environment value

§ `int put_env(key, data, num_bytes);`

Stores and environment variable and data in config.sys

VeriFone.

LAB9：File Access

Get parameter's value from config.sys and update the value

For example:

Give a entry to update parameter #PRNLOGO

#PRNLOGO=1 :print out logo

#PRNLOGO=0 :don't print out logo

VeriFone.