



## Using FAT and SD cards with BeRTOS

Recently someone noted in the mailing list that it's not clear (to say the least) how to use the FAT file system with BeRTOS. While I think you can figure it out if you're using SD cards, in this post I'll try to summarize what it's needed to use a different memory support.

### PROVIDE LOW LEVEL ACCESS FUNCTIONS TO FATFS LIBRARY

The FAT library abstracts the physical medium on which data is stored by using a few low level interface functions. Such methods are:

- ▶ `disk_initialize()`: initializes the disk for access
- ▶ `disk_status()`: returns the disk status
- ▶ `disk_read()`: (surprisingly) read from disk
- ▶ `disk_write()`: write to disk
- ▶ `disk_ioctl()`: miscellaneous commands
- ▶ `get_fattime()`: get the current time (used to update access time)

If you use the SD card driver provided with BeRTOS, these functions are already defined, otherwise you need to define them.

### PROVIDE A COMMUNICATION CHANNEL FOR SD DRIVER

The SD driver assumes that the device is accessible through the KFile interface, so you need to open such communication channel. You can use pretty much everything that has a KFile interface, so for example SPI or Serial are fine.

Notice, of course, that the device attached to this channel must "talk" the SD protocol.

### REGISTER A WORK AREA

Now that the code is in place you are almost done. The FatFS library requires you to register a work area, which you can do with:

```
FATFS fs;
f_mount(0, &fs);
```

The first parameter of `f_mount()` is the drive number, the second is the "filesystem" to register. You won't need this variable anymore in your program.

If you use the SD card driver you are restricted to use only one drive, but it shouldn't be too difficult to add support for more drives, if you need it.

### OPEN AND USE A FILE

Finally you can open a file! Just declare a `FatFile` structure and call `fatfile_open()` on it. Then you can use the KFile interface to read/write from the file.

```
FatFile input;
```

```
fatfile_open(&input, "foo.txt", FA_READ);
//...
uint8_t buf[10];
kfile_read(&input.fd, buf, sizeof(buf));
```

The last parameter to `fatfile_open()` (ie. open mode) is a combination of flags, which are declared in `bertos/fs/fatfs/ff.h`. The most common are:

- ▶ `FA_READ`: open a file for reading;
- ▶ `FA_WRITE`: open a file for writing. Can be combined with `FA_READ`;
- ▶ `FA_OPEN_EXISTING`: open the file, fail if not present.

Return values for `fatfile_open()` are also described in `bertos/fs/fatfs/ff.h`

When everything is set up, you may need to fine tune your code. For example, in some cases the channel you're using for the SD card driver is too slow for your application. I suggest to create a device that uses DMA for transfers, then tell the SD driver to use it.

If you're concerned about ROM usage, you can consider switching off some FatFS features, such as write access, long file name support and so on. Have a look at [FatFS application note](#) for further information.

Written by **Luca Ottaviano** in [programming](#) the 09 October 2009. Tag: [FAT](#), [HowTo](#), [SD card](#), [driver](#), [tips](#).

#### DEVELOPER'S TRAC

On the developers website you will find all the information to contribute to BeRTOS development community, milestones, the possibility to open tickets to the community, the changelog and all the details on every feature. [dev.bertos.org](http://dev.bertos.org)

#### PREMIUM SUPPORT

By registering to Develer's advanced support services you will have online assistance, phone assistance and the possibility to report bugs directly to our BeRTOS engineering team. All services guarantee reply in 24 working hours. [bertos.org/supporto/premium/](http://bertos.org/supporto/premium/)

#### CERTIFIED TRAINING

Customized training, onsite teaching, embedded application debug and deploy assistance, peer review on source code, training classes for single programmers or working teams. [bertos.org/supporto/formazione/](http://bertos.org/supporto/formazione/)



BeRTOS website by [Develer](#).