line, just before you press the Return key. It indicates to the shell that the command is not yet complete and more text will follow.

Continued over page...

Because you are redirecting the output from gzip to a file, it is your responsibility to supply the file name and the '.gz' extension. You must also include the hyphen after the '-9' in the gzip command. This tells gzip to write its output to the console which, in this case, has been redirected to a file named Rpi_8gb_backup.img.gz.

Splitting the Backup

As mentioned previously, some file systems cannot cope with files larger than a specific size. FAT32, for example, has a 4Gb limit and 32-bit operating systems can also only cope with 4Gb for each individual file. If you intend to create or copy your backups with these types of systems or devices, then you must split the image files into appropriate sized chunks.

Once again, a pipe comes to the rescue. The split utility in Linux is designed to allow a large file to be split into a number of smaller files, each of a given size. If we wish to create a compressed backup of an SD card, ready for burning onto one or more CDs or DVDs, then the following command will backup the card, compress the backup on the fly and then split the compressed image into a number of 2Gb files ready to be burned onto DVDs.

```
$ dd if=/dev/mmcblk0 bs=2M | \
gzip -9 - | \
split --bytes=2G - \
Rpi_8gb_backup.img.gz.part_
```

Remember to press the Return key immediately after typing the "\" on each line.

The split command, similar to gzip, requires a hyphen as the input file name. In the above there is no input file name as we are reading from a pipe, so the hyphen tells split to read the input from the console which, in this case, is the piped output from gzip.

The final parameter to split defines the root part of the output file names. Each one will be called 'Rpi_8gb_backup.img.gz.part_xx'. The 'xx' part will be 'aa', 'ab', 'ac' and so on.

It is possible that compressing the SD image will make the file small enough to fit within the file system limits of your chosen output device. In this case, you may rename the single part file back to the original name.

Restoring Backup Files

Having the backup image compressed or split into a number of sections means that restoring the backup takes a little more work. Once again the process is relatively simple and involves piping a number of commands together to form a chain of utilities, with the final output going into the dd command. This then writes the data back to the SD card.

The simplest case is when the image file has been compressed but not split. To restore this directly to the SD card, without having to create a full sized uncompressed image, type the following command:

```
$ gunzip Rpi_8gb_backup.img.gz -c | \
dd of=/dev/mmcblk0 bs=2M
```

The '-c' in the gunzip command above is required to tell gunzip to write the decompresssed data to the console. The dd command reads its input, unless otherwise specified, from the console so the output from gunzip passes straight into dd and out to the SD card.

If the backup is split into chunks, then we need to concatenate these together in the correct order and pipe that to the above command pipeline, as follows:

```
$ cat Rpi_8gb_backup.img.gz.part_* | \
gunzip -c | \
dd of=/dev/mmcblk0 bs=2M
```

The list of files to be joined back together again must be specified in the correct order. However, using a wild card to the cat command causes them to be read in alphbetical order, which is exactly how we want them to be.

Coming in Part 2

In part 2 of this article, I will show you how you can check that the backup was successful and how you can use the backup