Navigation: Sampling data > Sampling configuration > Resolution >

Data file types

Spike2 is optimised to use and sample data to the 64-bit SON64 filing system. It can also use and sample data into the original 32-bit SON filing system. We would recommend that you use the new format unless you have pressing reasons for using the old. Not all sampling features are supported by the old filing system and at some point we will remove support for sampling to it. You can read the file format version with the `FileInfo()` script command.

The 32-bit file system (*.smr)

The original Spike2 data format was first released as version 1 around 1988 and the original design has been extended several times since; we are currently at version 9. Each revision has added new data types and/or new filing system features. Changes were incremental; older versions of Spike2 would read newer files as long as they did not use new data types or features that the older versions did not know about.

The original design was based on the idea that all items in the file were located in time at an integer multiple of a basic clock period and that there are two fundamental types of data: waveforms (equally spaced in time data samples) and events (items at defined times with other data attached). There is some crossover between these types; waveforms can have gaps and events can have waveforms attached to them. You can download a manual explaining this library plus interface software from our web site.

32-bit original format

There are two limits to the size of a 32-bit Spike2 data file: the number of clock ticks in the file (limited to 2147483647) and the physical size of the file. The physical size before version 7 was limited to 2147483647 bytes, or 2 GB. These limits are related to the maximum size of a signed 32-bit number. For many users these limits are not a problem. However, it is possible to run out of disk space before you run out of clock ticks. For example, if you set the Waveform sample rate to be one clock tick and sample one channel, each sample point uses 2 bytes of data. If you set Burst mode sampling in the Resolution tab with n channels, each taking one sample per clock tick, then each clock tick is using 2n data bytes. This means that your maximum possible run time is limited to 2147483647/2n clock ticks.

Before Spike2 version 7, if you were sampling 32 channels of data at 31.25 kHz each, in non-burst mode you would need to set the clock tick to 1 microsecond, giving a clock tick limit of some 35 minutes of sampling. However, the file size limit would be hit in around 17 minutes. If you changed to burst mode sampling, the clock could be run at 32 microseconds per tick, which extends the clock limit to 19 hours, but the file size limit is still 17 minutes.

32-bit big files (Spike2 version 7 onwards)

If you set the file type to 32-bit big, the maximum file size is increased by a factor of 512, and with the example given above (32 channels at 31.25 kHz each in burst mode), you could sample for the full 19 hours allowed by the clock tick period. However, your timing resolution is now 32 microseconds. This would generate a huge file (128 GB) which would be slow to navigate.

32-bit limitations

The 32-bit system served us well, but has limitations due to the 32-bit nature of the format:

1. Times are stored as 32-bit integers, limiting us to a maximum of 2 billion clock ticks per file. At a resolution of 1 µs (microsecond) the maximum duration is 35 minutes and 47 seconds.
2. The file size was originally limited to 2GB. This was later extended to 1TB, but at the cost of making data recovery harder.
3. It can take a long time to locate data in the middle of a file. Finding data was speeded up by including lookup tables, but the methods used were limited by the requirement for backwards compatibility.
4. Waveform data with gaps is not stored efficiently as each gap starts a new disk block, so short waveform fragments use a lot of disk space.
5. There are limits on the number of characters used for channel titles (9), units (5) and comments (71) and file comments (79). Further, these are 8-bit characters, so if you take advantage of character codes outside the ASCII range, these will take two or more UTF-8 characters to store. This can be a problem, particularly for the Units. For example, if you decided to replace `"uVolt"` with `"µVolt"`, and saved this to a 32-bit file, it would be truncated to `"µVol"` as the code for the Unicode character µ is U+03BC which is coded in UTF-8 as 0xce, 0xbc.

The 64-bit filing system (*.smrx)

The new 64-bit filing system was designed to be logically compatible with the 32-bit system; by this we mean that it can hold the same types of data as the original without information loss, though some of these types are extended. It is also likely that we will add further data types, as required, in the future. Features include:

1. Times are stored as 64-bit integers. At a time resolution of 1 ns (nanosecond, 10^{-9} seconds), the maximum duration is around 256 years.

2. The file size is limited only by the capabilities of the operating system and by the size of a file that you can manage conveniently for archival. As I write this disk drives have a maximum size of a few TB. The file format is designed to make recovery of damaged files relatively straightforward.
3. The files have a built-in data lookup system designed to minimise the number of disk reads required to locate data on any channel.
4. The overhead for severely fragmented waveform data has been reduced to a few bytes per fragment.

We have removed many of the limits on things like the number of channels in a file and the length of channels comments and units. Before Spike2 version 8.03 the program enforced the original limits on the length of strings (but treated as Unicode characters). From version 8.03 onwards, a 64-bit `smrx` file in Spike2 can have 20 Unicode characters of channel title, 10 of channel units and comments can be up to 100 characters. The underlying file format does not impose a fixed limit, but practical considerations make it useful to define them.

How older versions of Spike2 cope with 64-bit (.smrx) files

Spike2 version 7.11c onwards can read (but not modify) the new 64-bit file system if the `son64.dll` file is in the Spike2 version 7 folder. Of course, if the file is longer (in clock ticks) than the old file system can read, only the start of the file will be visible. No other older versions of Spike2 can read them.

How older versions of Spike2 cope with 32-bit (.smr) files

Spike2 version 8 can read files with more than 400 channels, but ignores channels 401 upwards. Spike2 5.15 onwards can read files with up to 400 channels. Versions from 5.00 to 5.14 read files with up to 256 channels. Version 4.03 onwards reads files with up to 100 data channels. Versions before 4.03 can read files with up to 32 channels. You can use the File menu Export command to write channels from a data file to a new file with a suitable maximum channel limit so that it can be read by older versions of Spike2. If you check the Big file box you will not be able to open any generated file with versions of Spike2 before 6.11. Spike2 version 6 revisions that can open the file will treat it as read only.