# Question: "What are the differences between other functional vector files and special scan vector files?"
## Markus Seuring

**Question:**

Often vectors of scan tests are stored in a different format compared to other functional tests, although basically scan test is like any other functional test: Vector by vector certain values are forced at the inputs and certain values are expected at the outputs. Nevertheless, a special scan test vector format exists – what is the difference?

**Answer:**

Basically, scan test vectors can be stored like other functional test vectors, so that there is no difference regarding the stimulus applied to the DUT when the test is executed.

However, scan vectors can be stored in a special format that benefits from the characteristics of scan test and requires less memory for storage.
In the following this format is explained, using the scan pattern shown in Figure 5 of the second part of the introduction to scan test as an example.

Like for any functional test, the vectors of this scan pattern are shown on the left side:



During load/unload, all non-scan pins (any pin but "scan_in" and "scan_out") do not change values. This is valid for any scan pattern and can be used to describe the pattern in a more efficient way, as shown on the right side.

The vectors 2-5 and 10-13 are called "serial" vectors, all other vectors are "parallel" vectors.
So the special scan vector format is about usage of serial vectors for load/unload, i.e. for load/unload only the values of scan pins are stored, while values of all other pins are constant.
Scan vector files similar to the one described on the left side, i.e. the vector file contains only parallel vectors, are often called flattened scan vector files.

In the example above the size of scan chains is very small (just four scan cells), however, often scan chains contain thousands of scan cells, resulting in thousands of shift cycles for load/unload. So in practice the special scan format can save a lot of the memory that is required to store values for non-scan pins. Furthermore, this format facilitates reporting of scan fails in the pattern/chain/bit format, since load/unload cycles are implicitly marked.

Finally, below is shown the differences of (scan) vector formats for the Standard Test Interface Language (STIL), the widely used pattern format for scan tests defined in IEEE Std 1450.

Regular functional format / flattened format:

```
...
Pattern scan_example {
W WavTblScan;
V {clock=0; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=1; scan_out=X;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=1; scan_out=X;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
V {clock=0; reset=0; scan_en=0; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=0; incrmnt=1; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=0; incrmnt=1; overflow=X; scan_in=0; scan_out=X;}
V {clock=0; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=1; scan_out=H;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=1; scan_out=L;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=1; scan_out=L;}
V {clock=P; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=1; scan_out=L;}
}
...
```

Special scan format:

```
...
MacroDefs { "serial_shift" {
W WavTblScan;
Shift { V {scan_in=#; scan_out=#; clock=P;}}
}}
...
Pattern scan_example {
W WavTblScan;
V {clock=0; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
Macro "serial_shift" {
"scan_in" =0110;
"scan_out"=XXXX;
}
V {clock=0; reset=0; scan_en=0; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=0; incrmnt=1; overflow=X; scan_in=0; scan_out=X;}
V {clock=P; reset=0; scan_en=0; incrmnt=1; overflow=X; scan_in=0; scan_out=X;}
V {clock=0; reset=0; scan_en=1; incrmnt=0; overflow=X; scan_in=0; scan_out=X;}
Macro "serial_shift" {
"scan_in" =1111;
"scan_out"=HLLL;
}
}
```

...