SystemView : its use, bugs, and troubleshooting

12/3/2021

This document provides info for using SystemView, with the book. The following sections are:

- SEGGER_SYSVIEW_PrintfHost(): its use, and bugs
- SystemView Recorder: primary windows, and timestamps
- Upgrading the SystemView code, on the board
- SystemView troubleshooting: problems, and a bug

Document versions

- 10/18/2021 : initial version
- 11/2/2021 : added sections "SEGGER_SYSVIEW_PrintfHost()" and "Upgrading the SystemView code"
- 12/3/2021: added section "SystemView problem: Recorder reports overflow"

1 SEGGER_SYSVIEW_PrintfHost(): its use, and bugs

This section provides info needed to use `SEGGER_SYSVIEW_PrintfHost()`. This info is omitted in the *SystemView User Guide*, which is where the function is documented. Also, the function has some bugs that are likely to affect users, and they are described here.

1.1 Overview and terminology

`SEGGER_SYSVIEW_PrintfHost()` is a SystemView API that runs on the board. It generates a message that is displayed on the SystemView app, e.g., in the `Terminal` window. For brevity, the function will be referred to as `S...PrintfHost()`.

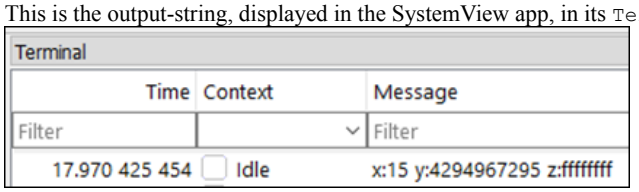
The info provided here is based on: the *SystemView Target-Sources* version 3.30 (the current version as of 10/2021), the *SystemView User Guide* version 3.20, and the SystemView Windows-app version 3.30. The book's system uses Target-Sources version 2.52h. Its `S...PrintfHost()` function appears to work the same way and have the same problems as in version 3.20.

`S...PrintfHost()` is modeled after `printf()`. However, there are significant differences between them.

This is an example of a call to `S...PrintfHost()`:

```
int x = 15;
unsigned long y = 0xffffffff;
int * z = (int *) 0xffffffff;
SEGGER_SYSVIEW_PrintfHost("x:%d y:%u z:%p", x, y, (void *)z);
```

This is the output-string, displayed in the SystemView app, in its `Terminal` window:



As with `printf()`, the arguments for `S...PrintfHost()` are: a *format-string*, followed by *format-specifiers*. The format-string can contain *format-placeholders*, which start with a "%" character. For each format-placeholder there is a corresponding format-specifier argument. The function's output is a formatted-string, and it will be referred to as the *output-string*.

The syntax and terminology for format-placeholders is shown below for `printf()`. For `S...PrintfHost()`, the syntax for its format-placeholders is not specified, but presumably it is similar.

```
%[parameter][flags][width][.precision][length]type
```

When `S...PrintfHost()` is called, it puts its arguments into a packet, and it sends the packet to the SystemView app on the host. The app uses the packet to create the output-string, and the app displays the output-string, e.g., in the Terminal window.

1.2 Using SEGGER_SYSVIEW_PrintfHost()

The *SystemView User Guide* omits info needed to use `SEGGER_SYSVIEW_PrintfHost()`. Some key parts of that omitted info are provided here. Not all of the omitted info is provided here, e.g., for format-placeholders, only the *type* field (e.g., "%d") is described here.