Trace debug:

In the MSCC environment

for release 0.1
April 5th 2017

Table of Contents

1 Trace

Trace messages are implemented on a module basis. The trace messages can be put into a number of disjunct groups, and the trace level can be configured per group within that module at run time.

1.1 Introduction

In order to have access to debug commands in general, the command

```
# platform debug allow
```

must be give. For trace the command is

```
# debug trace module level <tab>
```

If tab is pressed, as suggested above, the possible modules are listed. As an example we can choose `mstp`, i.e.

```
# debug trace module level mstp <tab>
```

If again tab is pressed, then the groups within the `mstp` module is shown. In this case it is

```
control
default
interface
```

So lets say

```
# debug trace module level mstp default
```

Giving this command, the status for the group `default` in module `mstp` will be shown. One of the elements in the status is the level. The level will most likely be `warning`, which means that messages at the level `warning` will be printed. This level can be changed. If we say

```
# debug trace module level mstp default <tab>
```

the possible levels will be shown, and we can choose one of them.

1.2 Find the Trace implementation

In the demo code `demo_trace.c` it is illustrated how the trace infrastructure is organized for a module. The macro

```
VTSS_TRACE_REGISTER(...)
```

register the trace structure. Therefore, if we want to find out details about the trace for a module we can look for the structures similar to the one show in `demo_trace.c`.

For example for mstp, if we did not have any idea of where the trace definitions was, we could say

```
cd .../vtss_appl
grep VTSS_TRACE_MODULE_ID -r * | grep "mstp"
```

and see if we can spot the right place. In this case it is `vtss_appl/mstp/platform/mstp.c`. If we search for `vtss_trace_grp_t` the groups `control`, `default` and `interface` will show up. All these groups have the default debug level of `warning` as given by `VTSS_TRACE_LVL_WARNING`.

The method above to locating the trace implementation in a module may not always succeed, since in the example above this requires that `VTSS_TRACE_MODULE_ID` and then name of the module is on the same line. But then we just have to be more clever about it, or do some more manual searching.

1.3 Trace messages

In the `vtss_appl/util/vtss_trace_api.h` file, we can see the macros that should be used for tracing. It is easy to see, that there is a set of macros for each debug level. If we focus on the debug group, the most interesting macros are `T_D(...)` and `T_DG(...)`. What should be noted by the first one, is, that it apply to the group `VTSS_TRACE_GRP_DEFAULT`. Therefore, if we want to use that one, the group name `VTSS_TRACE_GRP_DEFAULT` should be defined. Normally a module has a group called `default`, and this number is associated with it.

Of course a module do not need to have a default group, but if we want to use `T_D(...)` then we should. The second macro `T_DG(grp,...)` is similar to `T_D(...)` except we can specify the group.

So looking in `mstp.c`, if we say

```
T_DG(VTSS_TRACE_GRP_CONTROL, "Something");
```

then this statement is not printed by default since the default levelt is `warning`; but if we say

```
debug trace module level mstp control debug
```

then it will.