proc file system

trace/buffer

|  |  |  |
| --- | --- | --- |
| File | Read | Write |
| buffer | Print based on offset associated with open file | Write into buffer at “head” |
| control |  | reset, print=, mode= |
| level |  | NAME=|VAL |
| map |  |  |
| raw |  |  |
| versions |  |  |
|  |  |  |
| mode |  | |1, &~1 |
| reset |  |  |
| print |  |  |
|  |  |  |
|  |  |  |

Replace “control” with “mode”, “reset”, and “print”

Start with NextWriteIdx=0, Used=0

# Successes from previous Version

standard trace preprocessor include directives:

# Changes from previous Version

last archived version of TRACE was for a patch to the 2.6.30.5 or 2.6.18-164.6.1 kernels (SLF5)

Standard preprocessor directives should be documented in trace.h

No Performance Measurement Counter (PMC) support; this was never really done right/well and never really intended to be a feature.

Separate userspace header.

## Kernel Code

snprintf (lib/vsprintf.c) now in the kernel (i.e. 2.6.32)

Compile via standard kernel module build env:

tar xzf TRACE3.0.tgz

make [KSRC=/lib/modules/`uname –r`/build] [MSG\_MAX=80] [PARM\_MAX=8]

as root:

make insert [MEM=4%] [TID\_MAX=200]

32 bit system example:

Entry size =

“Trace System” via shared mem vs. proc filesystem???

Need access to “mode” and bit lvl data array.

Need to get TID.

# “Features”

The code does not need to know the msg\_max or parm\_max.

Code compiled with header will work on non-trace system.

Init per module vs. Init per exe???????