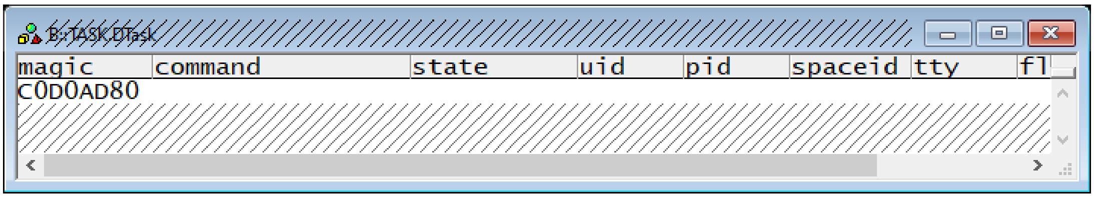
[Linux] Troubleshooting an awareness problem

2023-10-24 - Comments (0) - OS-aware debugging

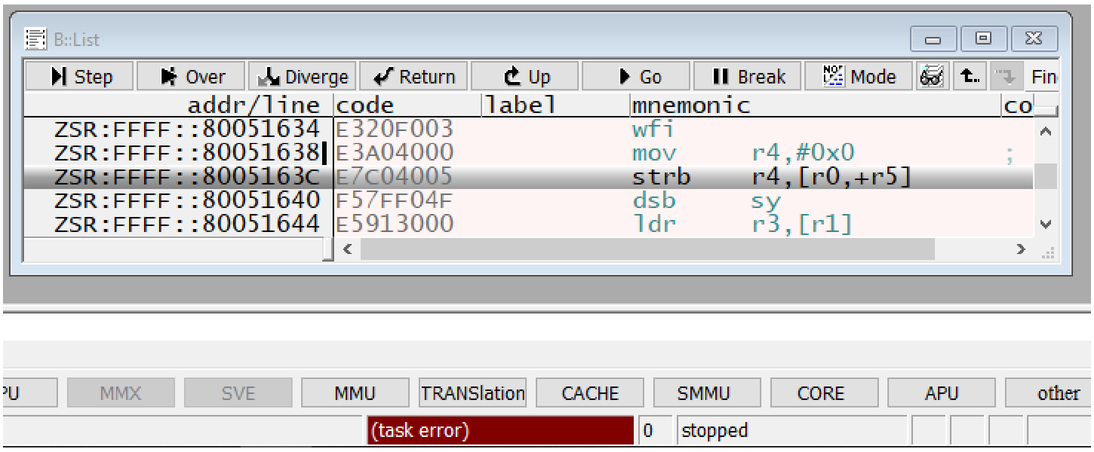
There is a problem related to the setup of the Linux awareness if:

- All or some awareness windows (e.g. **TASK.DTask**, **TASK.Process**) show errors or are hatched.



```
magic     command            state     uid    pid    spaceid tty    fl
C0D0AD80
```

- Current task cannot be displayed after Linux has booted, and the target has been stopped: "(task error)" or "(other)". The space-id 0xFFFF is displayed in the **List** window



```
          addr/line code     label    mnemonic                co
ZSR:FFFF::80051634 E320F003           wfi
ZSR:FFFF::80051638 E3A04000           mov      r4,#0x0        ;
ZSR:FFFF::8005163C E7C04005           strb     r4,[r0,+r5]
ZSR:FFFF::80051640 F57FF04F           dsb      sy
ZSR:FFFF::80051644 E5913000           ldr      r3,[r1]
```

```
(task error) 0 stopped
```

1. Check the kernel configuration. The problem is e.g. often caused a missing kernel configuration “Compile the kernel with debug info” (CONFIG_DEBUG_INFO) or by the kernel configuration "Reduce debugging information" (CONFIG_DEBUG_REDUCED). Refer for more information to the chapter “Kernel Configuration” in Training Linux Debugging.
2. Try to disable the translation with **TRANSlation.OFF**. If you get better results, then the problem is related to wrong translation settings. Auto-detection scripts are available for Arm and Arm64 under:

demo\arm64\kernel\linux\board\generic-template\detect_translation.cmm
demo\arm\kernel\linux\board\generic-template\detect_translation.cmm

Please read the script header carefully.

Contact the technical support by opening a new ticket if this script returns an error.

1. Check if the loaded vmlinux matches the executed kernel binary:

    - Get the target Linux banner by execution the command **cat /proc/version** in the terminal window
    - Load the vmlinux file including code into the debugger virtual memory with **Data.LOAD.Elf vmlinux AVM:0 /NoSymbol**
    - Dump the **linux_banner** from loaded vmlinux: **Data AVM:linux_banner /NoHex /NoOrient**
    - Compare both strings including timestamps

Note: in some cases the symbols may not be matching even if the Linux banner comparison shows the same string. A typical example is caused by the fact that OpenEmbedded / bitbake may compile multiple kernel variants in the same folder. Each result file is moved to a deploy folder - typically the build step will add a postfix if there are multiple variants of the same file. e.g. `Image-5.10-minimal`, `Image-5.10-xen`,... This is however not done for the `vmlinux` file of the respective build run. What can happen for instance is that the kernel is build once and later on the `initrd` is embedded.

Another pitfall that typically applies to all kernel symbols is - with Linux 5.9 the `0x80000` offset got removed from the kernel. But some bootloaders still start the kernel with an `0x80000` offset. This will trigger a relocation of the kernel to the next properly aligned address up/down.

Refer for more information to Training Linux Debugging.