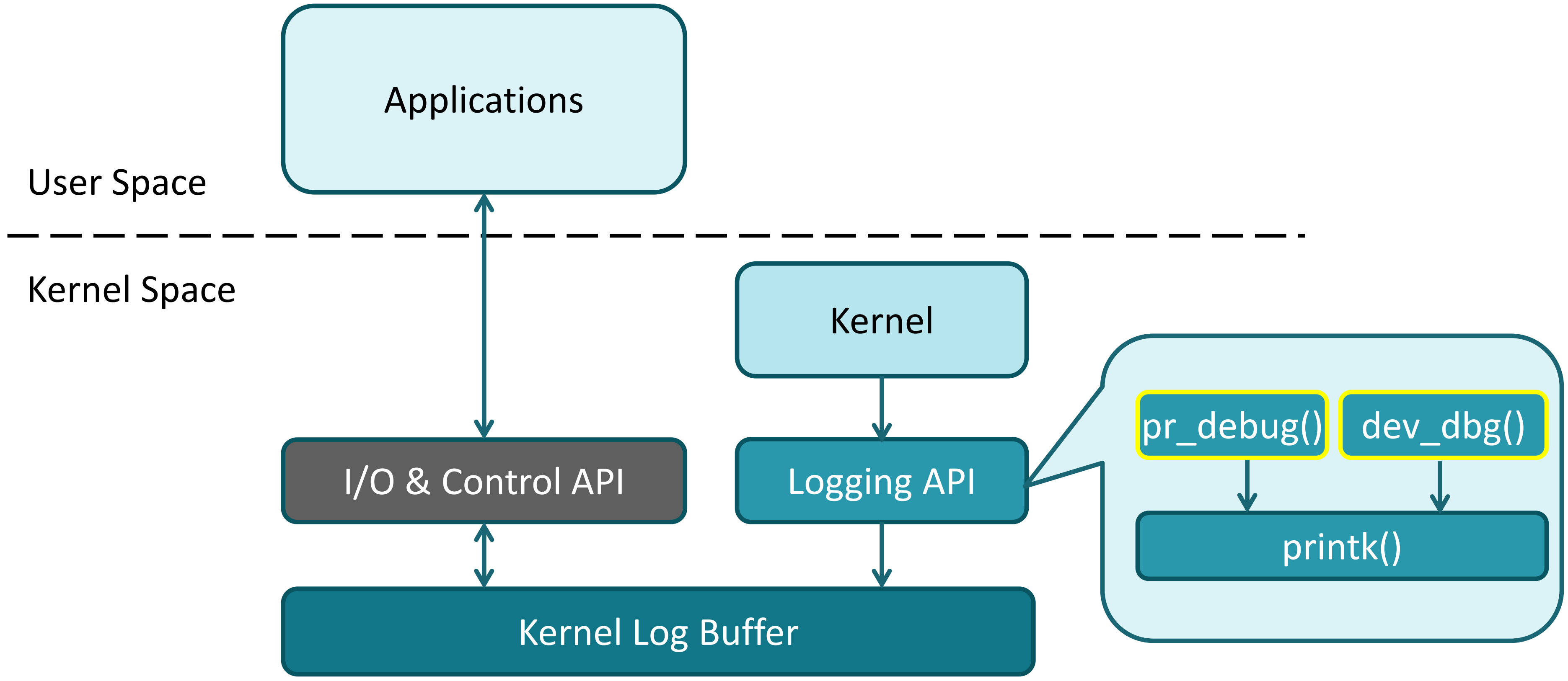
Debugging Embedded Linux Systems: Dynamic Debug

Debugging Embedded Linux Training Series [Part 4]

Debugging Embedded Linux Training Series

- Part 1: Linux/Kernel Overview
- Part 2: Kernel Logging System Overview
- Part 3: printk and Variations
- **Part 4: Dynamic Debug**
- Part 5: Locate Device Driver Source Code
- Part 6: Understand Kernel Oops Logs

Kernel logging system architecture



Agenda

- Introduction
- Control interface
- Case study: Debug USB xHCI

What is dynamic debug?

Dynamically enable/disable kernel debug code at runtime to obtain kernel debug log:

- pr_debug()/dev_dbg()
- print_hex_dump_debug()/print_hex_dump_bytes()

Why dynamic debug?

Benefits:

- Almost no overhead when log code is not enabled.
- Turn on/off debug log at runtime.
- No need to recompile the kernel.

Control interface overview

- Control methods:
  - Line Number or Range
  - Function Name
  - Filename
  - Module Name

- Control interface:
  - debugfs
  - u-boot bootargs

debugfs control example

```
# mount -t debugfs none /sys/kernel/debug/
# cd /sys/kernel/debug/dynamic_debug/
# echo “file xxx.c +p” > control
# echo “file svcsock.c line 1603 +p” > control
# echo “file drivers/usb/core/* +p” > control
# echo “file xxx.c -p” > control
```

debugfs control interface

- `# echo “<matches> <ops><flags>” > <debugfs>/dynamic_debug/control`

debugfs control interface

- `# echo “<matches> <ops><flags>” > <debugfs>/dynamic_debug/control`
- matches:
  - ‘**file**’ string
  - ‘**func**’ string
  - ‘**line**’ line-range
  - ‘**module**’ string (seen in lsmod)
  - supports wildcard (* ?)

debugfs control interface

- `# echo “<matches> <ops><flags>” > <debugfs>/dynamic_debug/control`
- matches:
  - ‘**file**’ string
  - ‘**func**’ string
  - ‘**line**’ line-range
  - ‘**module**’ string (seen in lsmod)
  - supports wildcard (* ?)
- ops:

-	remove the given flags
+	add the given flags
=	set to the given flags

debugfs control interface

- `# echo “<matches> <ops><flags>” > <debugfs>/dynamic_debug/control`
- matches:
  - ‘**file**’ string
  - ‘**func**’ string
  - ‘**line**’ line-range
  - ‘**module**’ string (seen in lsmod)
  - supports wildcard (* ?)
- ops:

-	remove the given flags
+	add the given flags
=	set to the given flags

- flags:

p	print the log message
f	include the function name
l	include the line number
m	include the module name
t	include the thread ID
_	no flags are set

Enable debug messages during boot process

- This allows debugging of core code or built-in modules during the boot process.
- uboot bootargs
  - dyndbg="*QUERY*" <-- *for kernel*
  - module.dyndbg="*QUERY*" < -- *for module*

- Example:

```
dyndbg="file ec.c +p"
```

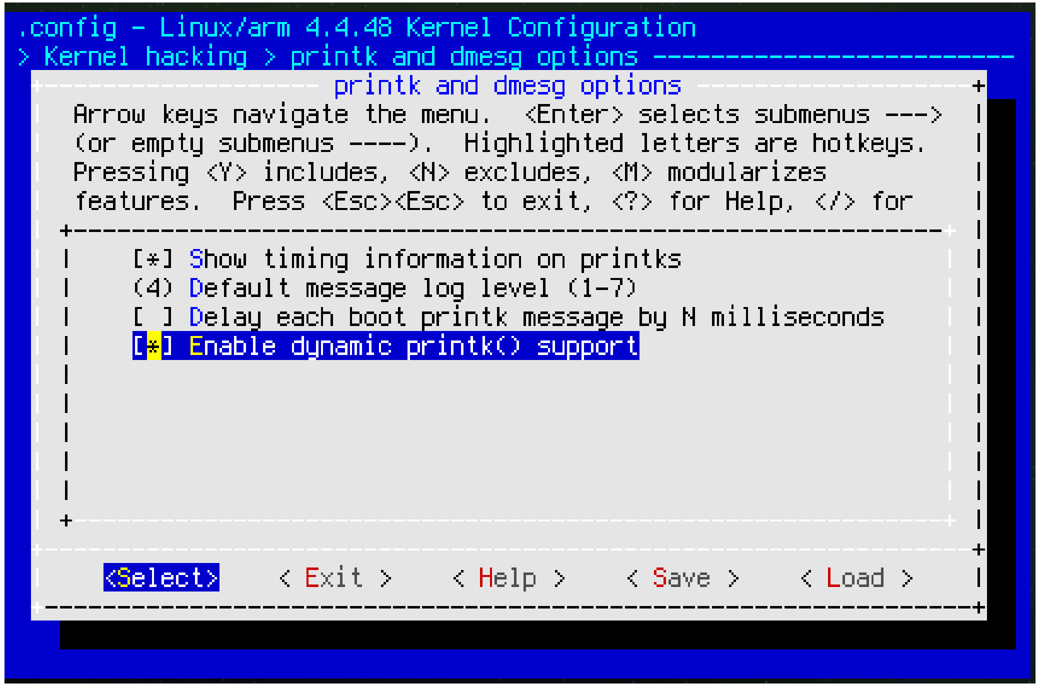
Enable dynamic debug

- CONFIG_DYNAMIC_DEBUG=y
- menuconfig (v4.4, v4.9):

  Kernel hacking --->

  printk and dmesg options --->

  [*] Enable dynamic printk() support



```
.config - Linux/arm 4.4.48 Kernel Configuration
> Kernel hacking > printk and dmesg options ---------------------------
 +------------------- printk and dmesg options --------------------------+
 |  Arrow keys navigate the menu.  <Enter> selects submenus --->         |
 |  (or empty submenus ----).  Highlighted letters are hotkeys.          |
 |  Pressing <Y> includes, <N> excludes, <M> modularizes                 |
 |  features.  Press <Esc><Esc> to exit, <?> for Help, </> for           |
 | +-------------------------------------------------------------------+ |
 | |     [*] Show timing information on printks                        | |
 | |     (4) Default message log level (1-7)                           | |
 | |     [ ] Delay each boot printk message by N milliseconds          | |
 | |     [*] Enable dynamic printk() support                           | |
 | |                                                                   | |
 | |                                                                   | |
 | |                                                                   | |
 | |                                                                   | |
 | |                                                                   | |
 | +-------------------------------------------------------------------+ |
 +-----------------------------------------------------------------------+
 |      <Select>    < Exit >    < Help >    < Save >    < Load >         |
 +-----------------------------------------------------------------------+
```

Case study: Debug USB xHCI (1/2)

- Boot the AM57x EVM.
- `# dmesg -C`
- `# echo 'module xhci_hcd =p' > /sys/kernel/debug/dynamic_debug/control`
- Plug a USB device into the USB host port.
- `# dmesg`

Case study: Debug USB xHCI (2/2)

```
[1119724.004734] xhci-hcd xhci-hcd.0.auto: // Ding dong!
[1119724.004770] xhci-hcd xhci-hcd.0.auto: Successful setup context command
[1119724.004779] xhci-hcd xhci-hcd.0.auto: Op regs DCBAA ptr = 0x000000fe866000
[1119724.004788] xhci-hcd xhci-hcd.0.auto: Slot ID 3 dcbaa entry @f2658018 = 0x000000fe877000
[1119724.004796] xhci-hcd xhci-hcd.0.auto: Output Context DMA address = 0xfe877000
[1119724.004804] xhci-hcd xhci-hcd.0.auto: Slot ID 3 Input Context:
[1119724.004812] xhci-hcd xhci-hcd.0.auto: @f26d8000 (virt) @fe87c000 (dma) 0x000000 - drop flags
[1119724.004820] xhci-hcd xhci-hcd.0.auto: @f26d8004 (virt) @fe87c004 (dma) 0x000003 - add flags
[1119724.004828] xhci-hcd xhci-hcd.0.auto: @f26d8008 (virt) @fe87c008 (dma) 0x000000 - rsvd2[0]
...
[1119724.004907] xhci-hcd xhci-hcd.0.auto: Slot Context:
[1119724.004915] xhci-hcd xhci-hcd.0.auto: @f26d8040 (virt) @fe87c040 (dma) 0x8300001 - dev_info
[1119724.004923] xhci-hcd xhci-hcd.0.auto: @f26d8044 (virt) @fe87c044 (dma) 0x010000 - dev_info2
[1119724.004931] xhci-hcd xhci-hcd.0.auto: @f26d8048 (virt) @fe87c048 (dma) 0x000000 - tt_info
[1119724.004938] xhci-hcd xhci-hcd.0.auto: @f26d804c (virt) @fe87c04c (dma) 0x000000 - dev_state
[1119724.004946] xhci-hcd xhci-hcd.0.auto: @f26d8050 (virt) @fe87c050 (dma) 0x000000 - rsvd[0]
...
[1119724.005008] xhci-hcd xhci-hcd.0.auto: IN Endpoint 00 Context (ep_index 00):
[1119724.005016] xhci-hcd xhci-hcd.0.auto: @f26d8080 (virt) @fe87c080 (dma) 0x000000 - ep_info
[1119724.005024] xhci-hcd xhci-hcd.0.auto: @f26d8084 (virt) @fe87c084 (dma) 0x400026 - ep_info2
```

Summary

- Enable/disable debug log messages at runtime. There is no need to recompile kernel.
- Control interface

  `/sys/kernel/debug/dynamic_debug/control`

- Uboot
  - dyndbg="*QUERY*"
  - module.dyndbg="*QUERY*"

For more information

- Processor SDK Training Series: http://training.ti.com/processor-sdk-training-series
- Debugging Embedded Linux Training Series: http://training.ti.com/debug-embedded-linux-training-series
- Processor SDK Linux Getting Started Guide: http://processors.wiki.ti.com/index.php/Processor_SDK_Linux_Getting_Started_Guide
- Download Processor SDK Linux for Embedded Processors: http://www.ti.com/processorsdk
- For questions about this training, refer to the E2E Embedded Linux Community Forum: http://e2e.ti.com/support/embedded/linux/f/354