Operating Systems And Applications For Embedded Systems

Building the kernel

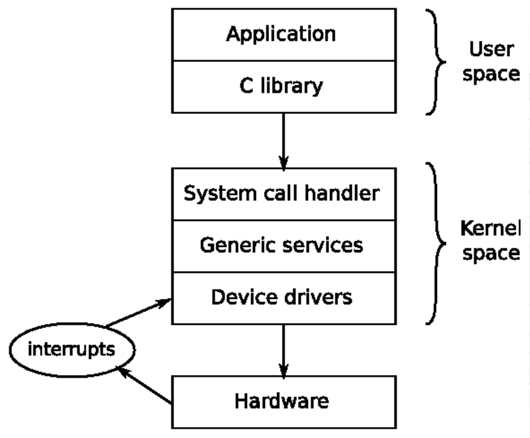
Katedra Inżynierii Komputerowej

Plan

- Toolchain
  - Toolchain
  - Main component of GNU toolchain
  - C library
  - Finding a toolchain
- crosstool-NG
  - crosstool-NG
  - Installing
  - Anatomy of a toolchain
  - Information about cross-compiler
  - Configruation
  - Most interesting features
  - Sysroot
  - Other tools
  - POSIX functions AP

Main Features

The kernel has three main jobs: to manage resources, to interface with hardware, and to provide an API that offers a useful level of abstraction to user space programs.



Getting the source

1. git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linuxstable.git linux
2. cd linux

Main directories

- arch: This contains architecture-specific files. There is one subdirectory per architecture.
- Documentation: This contains kernel documentation. Always look here first if you want to find more information about an aspect of Linux.
- drivers: This contains device drivers, thousands of them. There is a subdirectory for each type of driver.
- fs: This contains filesystem code.
- include: This contains kernel header files, including those required when building the toolchain.
- init: This contains the kernel start-up code.
- kernel: This contains core functions, including scheduling, locking, timers, power management, and debug/trace code.
- mm: This contains memory management.
- net: This contains network protocols.
- scripts: This contains many useful scripts including the device tree compiler, dtc, which I described in Chapter 3, All About Bootloaders.
- tools: This contains many useful tools, including the Linux performance counters tool, perf, which I will describe in Chapter 13, Profiling and Tracing.

Kernel configration I

This configuration item, along with all the others, is stored in a file named .config.
The variable names stored in .config are prefixed with CONFIG_, so if DEVMEM is enabled, the line reads:
CONFIG_DEVMEM=y

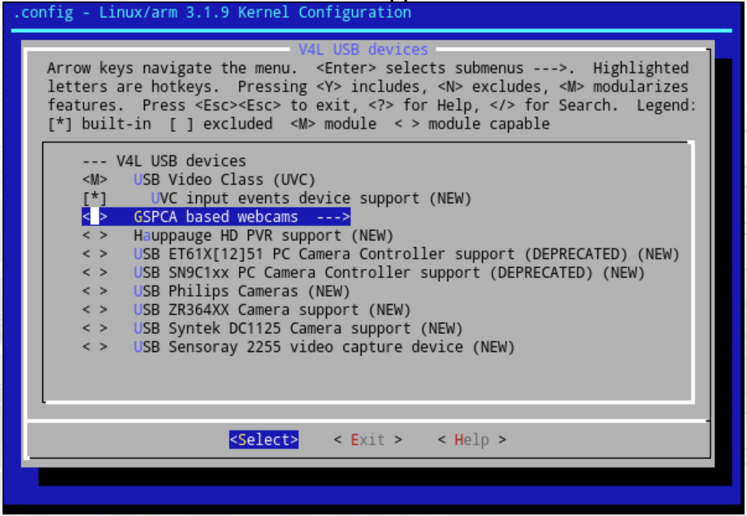
- bool: This is either y or not defined.
- tristate: This is used where a feature can be built as a kernel module or built into the main kernel image. The values are m for a module, y to be built in, and not defined if the feature is not enabled.
- int: This is an integer value written using decimal notation.
- hex: This is an unsigned integer value written using hexadecimal notation.
- string: This is a string value.

Kernel configration II

make ARCH=arm menuconfig

The star (*) to the left of an item means that it is selected (="y") or, if it is an M, that it has been selected to be built as a kernel module.

Kernel configration III



```
.config - Linux/arm 3.1.9 Kernel Configuration

                         V4L USB devices
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted
  letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes
  features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend:
  [*] built-in  [ ] excluded  <M> module  < > module capable

      --- V4L USB devices
      <M>   USB Video Class (UVC)
      [*]     UVC input events device support (NEW)
      < >   GSPCA based webcams  --->
      < >   Hauppauge HD PVR support (NEW)
      < >   USB ET61X[12]51 PC Camera Controller support (DEPRECATED) (NEW)
      < >   USB SN9C1xx PC Camera Controller support (DEPRECATED) (NEW)
      < >   USB Philips Cameras (NEW)
      < >   USB ZR364XX Camera support (NEW)
      < >   USB Syntek DC1125 Camera support (NEW)
      < >   USB Sensoray 2255 video capture device (NEW)

                 <Select>    < Exit >    < Help >
```

Compiling the kernel image

To build a kernel image, you need to know what your bootloader expects. This is a rough guide:

- U-Boot: Traditionally U-Boot has required a uImage, but newer versions can load a zImage file using the bootz command
- x86 targets: It requires a bzImage file
- Most other bootloaders: It requires a zImage file

```
make -j 4 ARCH=arm CROSS_COMPILE=arm-none-gnueabihf LOADADDR=0x80008000
uImage
```

The -j 4 option tells make how many jobs to run in parallel, which reduces the time taken to build. A rough guide is to run as many jobs as you have CPU cores.

Compiling device trees

```
make ARCH=arm dtbs
```

Compiling modules

```
make -j4 ARCH=arm CROSS_COMPILE=arm-cortex_a8-linux-gnueabihf
INSTALL_MOD_PATH=$HOME/rootfs modules_install
```

Kernel modules are put into the directory /lib/modules/[kernel version], relative to the root of the filesystem.

Cleaning kernel sources

- clean: removes object files and most intermediates.
- mrproper: removes all intermediate files, including the .config file.
- distclean: This is the same as mrproper but also deletes editor backup files, patch leftover files, and other artifacts of software development.

QEMU

```
QEMU_AUDIO_DRV=none qemu-system-arm -m 256M -nographic -M vexpress-a9 -kernel
zImage -dtb vexpress-v2p-ca9.dtb -append "console=ttyAMA0"
```

Kernel messages

Level	Value	Meaning
KERN_EMERG	0	The system is unusable
KERN_ALERT	1	Action must be taken immediately
KERN_CRIT	2	Critical conditions
KERN_ERR	3	Error conditions
KERN_WARNING	4	Warning conditions
KERN_NOTICE	5	Normal but significant conditions
KERN_INFO	6	Informational
KERN_DEBUG	7	Debug-level messages

Kernel command line I

- debug
  Sets the console log level to the highest level, eight, to ensure thatyou see all the kernel messages on the console.
- init=
  The init program to run from a mounted root filesystem, which defaults to /sbin/init.
- lpj=
  Sets the loops_per_jiffy to a given constant, see the following paragraph.
- panic=
  Behavior when the kernel panics: if it is greater than zero, it gives the number of seconds before rebooting; if it is zero, it waits forever (this is the default); or if it is less than zero, it reboots without any delay.

Kernel command line II

- quiet
  Sets the console log level to one, suppressing all but emergency messages. Since most devices have a serial console, it takes time to output all those strings. Consequently, reducing the number of messages using this option reduces boot time.
- rdinit=
  The init program to run from a ramdisk, it defaults to /init.
- ro
  Mounts the root device as read-only. Has no effect on a ramdisk which is always read/write.
- root=
  Device to mount the root filesystem.
- rootdelay=
  The number of seconds to wait before trying to mount the root device, defaults to zero. Useful if the device takes time to probe the hardware, but also see rootwait.

Kernel command line III

- rootfstype=
  The filesystem type for the root device. In many cases, it is autodetected during mount, but it is required for jffs2 filesystems.
- rootwait
  Waits indefinitely for the root device to be detected. Usually necessary with mmc devices.
- rw
  Mounts the root device as read-write (default)

Additional reading

- Linux Kernel Newbies, kernelnewbies.org
- Linux Weekly News, www.lwn.net

References

C. Simmonds.
Mastering Embedded Linux Programming.
Packt Publishing, 2015.

The End