

Kconfig介绍

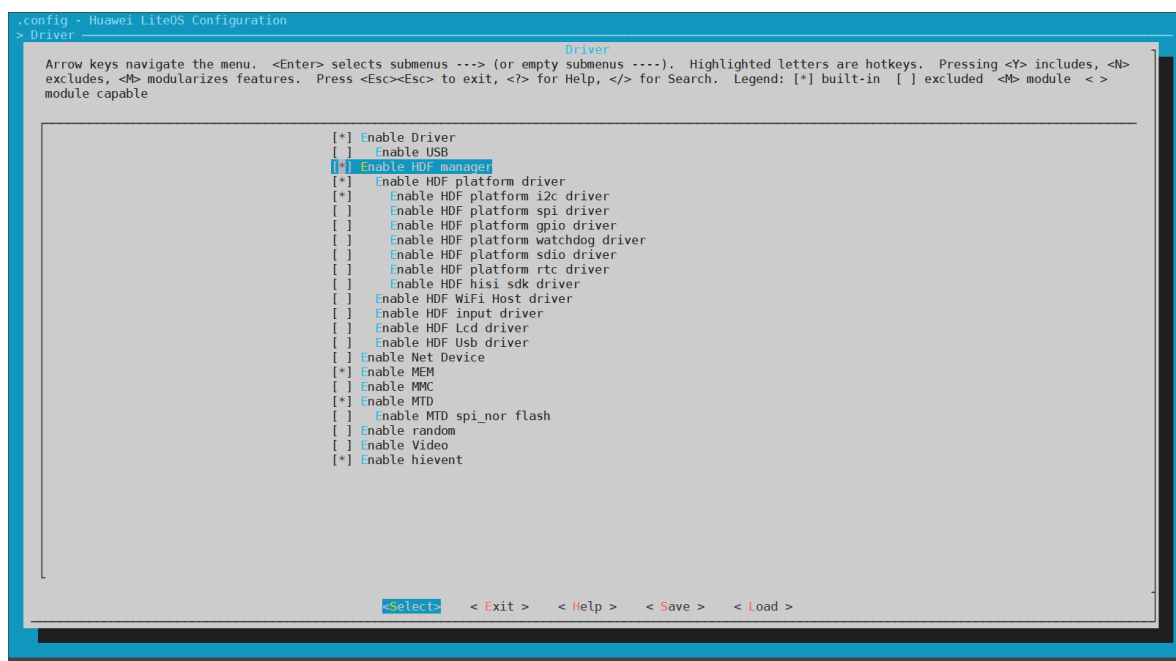
参考文档:

任一Linux内核的Documentation\kbuild\kconfig-language.rst

<https://www.rt-thread.org/document/site/programming-manual/kconfig/kconfig/>

对于各类内核，只要支持menuconfig配置界面，都是使用Kconfig。
在配置界面中，可以选择、设置选项，这些设置会保存在.config文件里。
Makefile会包含.config，根据里面的值决定编译哪些文件、怎么编译文件。

1.1 配置界面示例



问题:

- 这个界面里，各个配置项来自哪里
- 这个界面里，这些配置项是怎么组织的
- 这个界面里，我们的选择、设置，结果保存在哪里

1.2 配置结果的保存

1.2.1 示例

在配置界面中操作的结果保存在.config文件中，示例如下:

```
# LOSCFG_COMPILER_HIMIX_32 is not set
LOSCFG_COMPILER_CLANG_LLVM=y

#
# Platform
#
LOSCFG_PLATFORM="stm32mp157"
# LOSCFG_PLATFORM_HI3516DV300 is not set
```

```
# LOSCFG_PLATFORM_HI3518EV300 is not set
LOSCFG_PLATFORM_STM32MP157=y
# LOSCFG_PLATFORM_IMX6ULL is not set
LOSCFG_PLATFORM_BSP_GIC_V2=y
LOSCFG_ARCH_ARM=y
LOSCFG_ARCH_ARM_AARCH32=y
LOSCFG_ARCH_ARM_V7A=y
LOSCFG_ARCH_ARM_VER="armv7-a"
LOSCFG_ARCH_FPU_VFP_V4=y
LOSCFG_ARCH_FPU_VFP_D32=y
LOSCFG_ARCH_FPU_VFP_NEON=y
LOSCFG_ARCH_FPU="neon-vfpv4"
LOSCFG_ARCH_CORTEX_A7=y
LOSCFG_ARCH_CPU="cortex-a7"
```

Makefile会包含.config文件，它会根据里面的变量比如 `LOSCFG_PLATFORM_STM32MP157` 选择单板相关的文件。

1.2.2 配置项的前缀

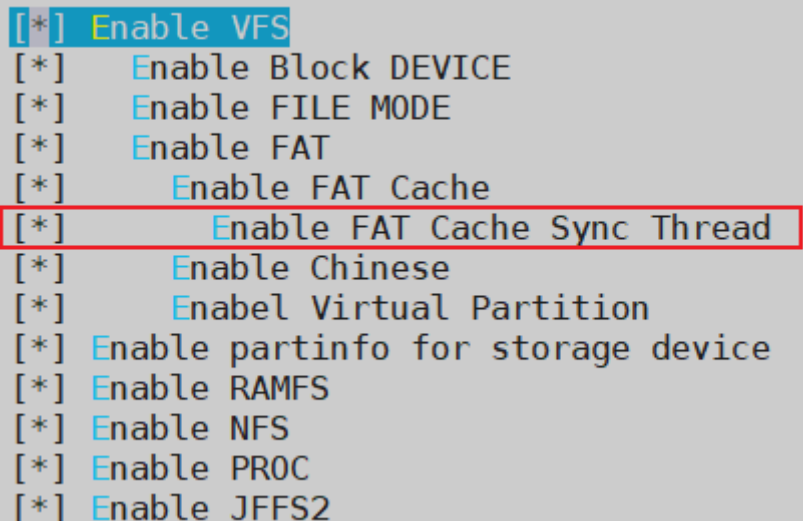
在Kconfig文件中，假设配置项的名字是XXX，在.config文件中：

- 默认情况下，它对应的变量名为 `CONFIG_XXX`
- 如果设置了环境变量 `CONFIG_=ABC`，则对应的变量名为 `ABC_XXX`
- 在Liteos-a中的Makefile中 `export CONFIG_=LOSCFG_`，所以对应的变量名为 `LOSCFG_XXX`

1.3 描述单个配置项config

1.3.1 示例

在 `make menuconfig` 界面，可以看到这个配置项：



```
[*] Enable VFS
[*] Enable Block DEVICE
[*] Enable FILE MODE
[*] Enable FAT
[*] Enable FAT Cache
[*] Enable FAT Cache Sync Thread
[*] Enable Chinese
[*] Enabel Virtual Partition
[*] Enable partinfo for storage device
[*] Enable RAMFS
[*] Enable NFS
[*] Enable PROC
[*] Enable JFFS2
```

在配置界面，使用方向箭头游走到 `Enable FAT Cache Sync Thread` 后，可以：

- 输入Y，选择配置项，在.config中对应 `LOSCFG_FS_FAT_CACHE_SYNC_THREAD=y`
- 输入N，不选择配置项，在.config中对应 `# LOSCFG_FS_FAT_CACHE_SYNC_THREAD is not set`

上图中的配置项怎么实现的？

在Kconfig文件中，它对应下列代码：

```
config FS_FAT_CACHE_SYNC_THREAD 配置项的名字，一般会加上前缀，比如：
类型 bool "Enable FAT Cache Sync Thread"  LOSCFG_FS_FAT_CACHE_SYNC_THREAD
default n 默认值  prompt, 提示信息
depends on FS_FAT_CACHE 依赖：只有FS_FAT_CACHE被选中时，才可以选择本配置项
help 帮助信息
    Answer Y to enable LiteOS fat filesystem support cache sync thread. // 缩进的是帮助信息
// 可多行
```

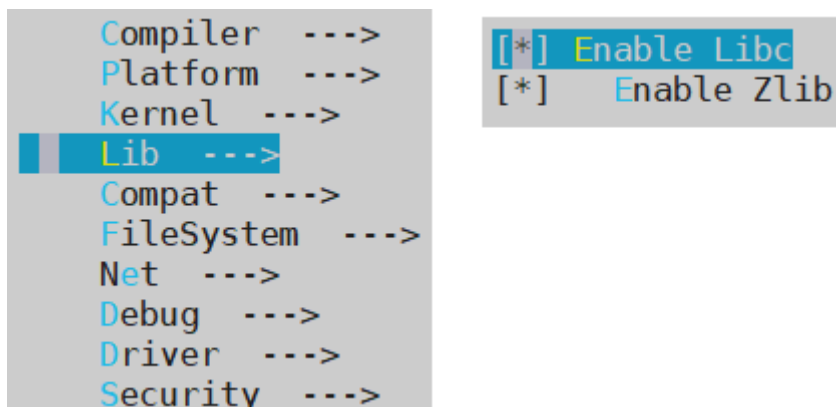
1.3.2 语法

解释如下：

- config
表示 config option，这是Kconfig的基本entry；其他entry是用来管理config的。
config 表示一个配置选项的开始，紧跟着的 FS_FAT_CACHE_SYNC_THREAD 是配置选项的名称。
config 下面几行定义了该配置选项的属性。
属性可以是该配置选项的：类型、输入提示、依赖关系、默认值、帮助信息。
 - bool 表示配置选项的类型，每个 config 菜单项都要有类型定义，变量有5种类型
 - bool 布尔类型
 - tristate 三态类型
 - string 字符串
 - hex 十六进制
 - int 整型
 - "Enable FAT Cache Sync Thread"：提示信息
 - depends on：表示依赖关系，只有FS_FAT_CACHE被选中，才可以选择 FS_FAT_CACHE_SYNC_THREAD
 - select XXX：表示反向依赖关系，即当前配置选项被选中后，xxx 选项就会被选中。
 - default 表示配置选项的默认值，bool 类型的默认值可以是 y/n。
 - help 帮助信息，在 make menuconfig 界面输入H键时，就会提示帮助信息。

1.4 实现菜单menu/endmenu

1.4.1 示例



在Kconfig中，代码如下：

```
menu "Lib"
config LIB_LIBC
    bool "Enable Libc"
```

```

    default y
    help
        Answer Y to enable libc for full code.

config LIB_ZLIB
    bool "Enable Zlib"
    default y
    depends on LIB_LIBC
    help
        Answer Y to enable LiteOS support compress file library.
endmenu

```

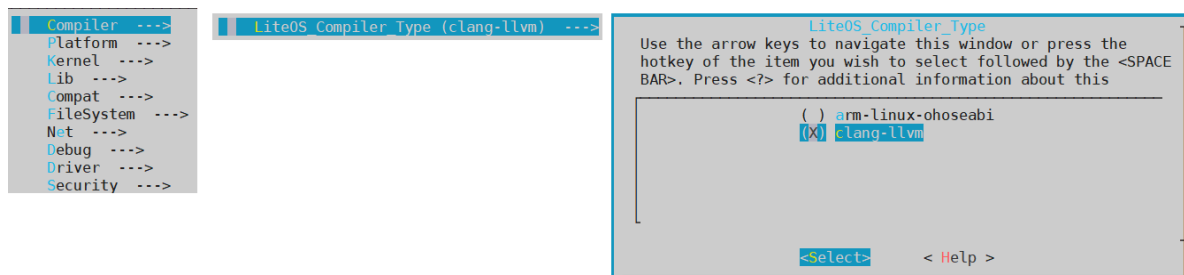
1.4.2 语法

解释如下：

- menu "xxx"表示一个菜单，菜单名是"xxx"
- menu和endmenu之间的entry都是"xxx"菜单的选项
- 在上面的例子中子菜单有2个选项："Enable Libc"、"Enable Zlib"
- 由于第二个菜单项依赖于第一个菜单项，所以第二个菜单项缩进一格

1.5 实现单选choice/endchoice

1.5.1 示例



在上述界面中，对于 LiteOS_Compiler_Type，有2个选择：arm-linux-ohoseabi、clang-llvm。在Kconfig文件中怎么描述？如下：

```

menu "Compiler"
choice
    prompt "LiteOS_Compiler_Type"
    default COMPILER_CLANG_LLVM
    help
        Enable arm-himix100 or aarch64-himix100 or compiler.

config COMPILER_HIMIX_32
    bool "arm-linux-ohoseabi"
    depends on PLATFORM_HI3518EV300 || PLATFORM_HI3516DV300 || PLATFORM_IMX6ULL
    || PLATFORM_STM32MP157

config COMPILER_CLANG_LLVM
    bool "clang-llvm"
    depends on PLATFORM_HI3518EV300 || PLATFORM_HI3516DV300 ||
PLATFORM_IMX6ULL || PLATFORM_STM32MP157

```

```
endchoice
endmenu
```

1.5.2 语法

解释如下：

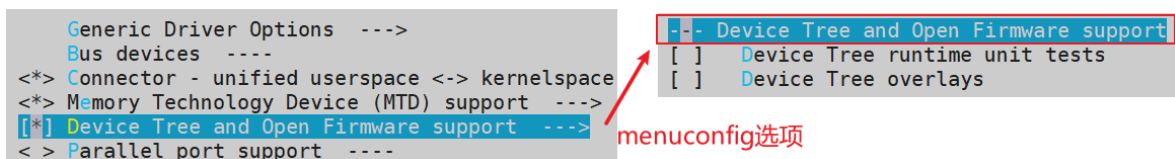
- choice表示"选择"
- choice和endchoice之间的entry是可以选择的项目
 - 它们之间，只能有一个被设置为"y"：表示编进内核
 - 它们之间，可以设置多个为"m"：表示编译为模块
 - 比如一个硬件有多个驱动程序
 - 同一时间只能有一个驱动能编进内核
 - 但是多个驱动都可以单独编译为模块

1.6 menuconfig

menuconfig xxx 和 config xxx 类似，

唯一不同的是该选项除了能设置y/m/n外，还可以实现菜单效果(能回车进入该项内部)。

1.6.1 示例



```
Generic Driver Options --->
Bus devices ----
<*> Connector - unified userspace <-> kernelspace
<*> Memory Technology Device (MTD) support --->
[*] Device Tree and Open Firmware support --->
< > Parallel port support ----
```

menuconfig选项

对于上述界面，Kconfig文件中代码如下：

```
menuconfig OF
    bool "Device Tree and Open Firmware support"
    help
        This option enables the device tree infrastructure.
        It is automatically selected by platforms that need it or can
        be enabled manually for unittests, overlays or
        compile-coverage.

if OF

config OF_UNITTEST
    bool "Device Tree runtime unit tests"
    depends on OF_IRQ
    select OF_EARLY_FLATTREE
    select OF_RESOLVE
    help
        This option builds in test cases for the device tree infrastructure
        that are executed once at boot time, and the results dumped to the
        console.

        If unsure, say N here, but this option is safe to enable.

config OF_OVERLAY
    bool "Device Tree overlays"
    select OF_DYNAMIC
```

```

select OF_RESOLVE
help
    Overlays are a method to dynamically modify part of the kernel's
    device tree with dynamically loaded data.
    While this option is selected automatically when needed, you can
    enable it manually to improve device tree unit test coverage.

endif # OF

```

1.6.2 语法

menuconfig常用格式有2种：

```

menuconfig M
if M
    config C1
    config C2
endif

```

或：

```

menuconfig M
config C1
    depends on M
config C2
    depends on M

```

第1项 menuconfig M 跟 config M 语法是一样的，不同之处在于 menuconfig M 后面可以跟着好几个依赖于M的 config C1、config C2 等子配置项。

1.7 if/endif

1.7.1 语法

在上面的menuconfig中就有 if/endif 的使用，它的语法如下：

```

"if" <expr>
<if block>
"endif"

```

1.7.2 示例

示例如下，只有定义的OF项， OF_UNITTEST 和 OF_OVERLAY 才会显示出来：

```

if OF

config OF_UNITTEST
    bool "Device Tree runtime unit tests"
    depends on OF_IRQ
    select OF_EARLY_FLATTREE
    select OF_RESOLVE
    help
        This option builds in test cases for the device tree infrastructure

```

that are executed once at boot time, and the results dumped to the console.

If unsure, say N here, but this option is safe to enable.

```
config OF_OVERLAY
    bool "Device Tree overlays"
    select OF_DYNAMIC
    select OF_RESOLVE
    help
        Overlays are a method to dynamically modify part of the kernel's
        device tree with dynamically loaded data.
        While this option is selected automatically when needed, you can
        enable it manually to improve device tree unit test coverage.

endif # OF
```

1.8 source

source 语句用于读取另一个文件中的 Kconfig 文件，如：

```
source "../../kernel/liteos_a/platform/kconfig"
```

1.9 comment

comment 语句出现在界面的第一行，用于定义一些提示信息，如：

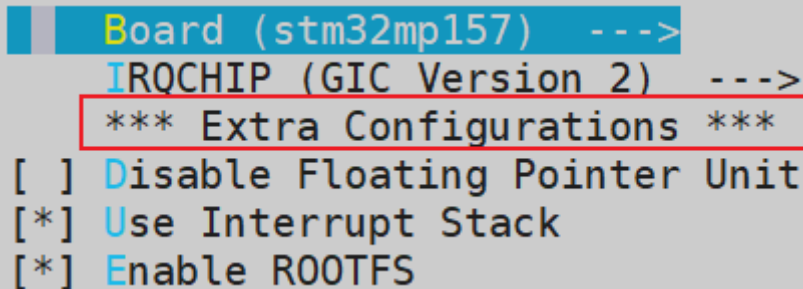
```
config ARCH_ARM
    bool

source "arch/arm/kconfig"

comment "Extra Configurations"

config ARCH_FPU_DISABLE
    bool "Disable Floating Pointer Unit"
    default n
    help
        This option will bypass floating procedure in system.
```

界面如下：



```
Board (stm32mp157) --->
IRQCHIP (GIC Version 2) --->
*** Extra Configurations ***
[ ] Disable Floating Pointer Unit
[*] Use Interrupt Stack
[*] Enable ROOTFS
```