由各个目录下的built-in.o,最终链接而成的vmlinux(位于kernel source根目录下)是最原始的ELF格式的kernel.

438e08f1-r0/linux-granite2-standard-build$ ll arch/arm/boot/

总用量 12416

drwxr-xr-x 4 walterzh walterzh   4096 11月 5 21:30 ./

drwxr-xr-x 14 walterzh walterzh   4096 10月 20 12:03 ../

drwxr-xr-x 2 walterzh walterzh   4096 11月 5 21:30 compressed/

drwxr-xr-x 2 walterzh walterzh   4096 10月 20 12:04 dts/

-rwxrwxr-x 1 walterzh walterzh 6297792 11月 5 21:30 Image*

-rw-r--r-- 1 walterzh walterzh   112 11月 5 21:30 .Image.cmd

-rw-r--r-- 1 walterzh walterzh 3193608 11月 5 21:30 uImage

-rw-r--r-- 1 walterzh walterzh   347 11月 5 21:30 .uImage.cmd

lrwxrwxrwx 1 walterzh walterzh   16 10月 20 12:03 vmlinux -> ../../../vmlinux*

-rwxrwxr-x 1 walterzh walterzh 3193544 11月 5 21:30 zImage*

-rw-r--r-- 1 walterzh walterzh   139 11月 5 21:30 .zImage.cmd

arch/arm/boot/vmlinux link to the ELF kernel

而Image则是该ELF kernel的binary

438e08f1-r0/linux-granite2-standard-build$ cat arch/arm/boot/.Image.cmd

cmd_arch/arm/boot/Image := arm-poky-linux-gnueabi-objcopy -O binary -R .comment -S vmlinux arch/arm/boot/Image

438e08f1-r0/linux-granite2-standard-build$ cat arch/arm/boot/.zImage.cmd

cmd_arch/arm/boot/zImage := arm-poky-linux-gnueabi-objcopy -O binary -R .comment -S
 arch/arm/boot/compressed/vmlinux arch/arm/boot/zImage

zImage是arch/arm/boot/compressed/vmlinux的bianary

438e08f1-r0/linux-granite2-standard-build$ cat arch/arm/boot/.uImage.cmd

cmd_arch/arm/boot/uImage := /bin/bash /home/walterzh/work/gerrit/build-bundle/poky/build/tmp/work/granite2-poky-linux-gnueabi/linux-granite-upstream/3.18.7+gitAUTOINC+e2438e08f1-r0/linux/scripts/mkuboot.sh -A arm -O linux -C none -T kernel -a 0x00008000 -e 0x00008000 -n 'Linux-3.18.7-yocto-standard' -d arch/arm/boot/zImage arch/arm/boot/uImage

而uImage则是依据arch/arm/boot/zImage，由linux/scripts/mkuboot.sh create。

walterzh@walterzh-ThinkPad-T440p:~/work/gerrit/build-bundle/poky/build/tmp/work/granite2-poky-linux-gnueabi/linux-granite-upstream/3.18.7+gitAUTOINC+e2438e08f1-r0/linux-granite2-standard-build$ cat arch/arm/boot/compressed/.vmlinux.cmd

cmd_arch/arm/boot/compressed/vmlinux := arm-poky-linux-gnueabi-ld.bfd   --defsym _kernel_bss_size=459700 -p --no-undefined -X -T arch/arm/boot/compressed/vmlinux.lds arch/arm/boot/compressed/head.o arch/arm/boot/compressed/piggy.gzip.o arch/arm/boot/compressed/misc.o arch/arm/boot/compressed/decompress.o arch/arm/boot/compressed/debug.o arch/arm/boot/compressed/string.o arch/arm/boot/compressed/hyp-stub.o arch/arm/boot/compressed/lib1funcs.o arch/arm/boot/compressed/ashldi3.o arch/arm/boot/compressed/bswapsdi2.o -o arch/arm/boot/compressed/vmlinux

arch/arm/boot/compressed/vmlinux由如下obj文件链接而成：

arch/arm/boot/compressed/head.o

arch/arm/boot/compressed/piggy.gzip.o

arch/arm/boot/compressed/misc.o

arch/arm/boot/compressed/decompress.o

arch/arm/boot/compressed/debug.o

arch/arm/boot/compressed/string.o

arch/arm/boot/compressed/hyp-stub.o

arch/arm/boot/compressed/lib1funcs.o

arch/arm/boot/compressed/ashldi3.o

arch/arm/boot/compressed/bswapsdi2.o

arch/arm/boot/compressed/piggy.gzip.o的生成

walterzh@walterzh-ThinkPad-T440p:~/work/gerrit/build-bundle/poky/build/tmp/work/granite2-poky-linux-gnueabi/linux-granite-upstream/3.18.7+gitAUTOINC+e2438e08f1-r0/linux-granite2-standard-build$ cat arch/arm/boot/compressed/.piggy.gzip.cmd

cmd_arch/arm/boot/compressed/piggy.gzip := (cat arch/arm/boot/compressed/../Image | gzip -n -f -9 > arch/arm/boot/compressed/piggy.gzip) || (rm -f arch/arm/boot/compressed/piggy.gzip ; false)

arch/arm/boot/Image被gzip压缩生成arch/arm/boot/compressed/piggy.gzip

通过arch/arm/boot/compressed/piggy.gzip.S,把piggy.gzip打包成obj文件。

$ cat piggy.gzip.S

```
        .section .piggydata,#alloc
        .globl  input_data
input_data:
```

```
        .incbin     "arch/arm/boot/compressed/piggy.gzip"
        .globl      input_data_end
input_data_end:
```

生成的被压缩的vmlinux ELF可执行文件的layout如下：

```
$ cat arch/arm/boot/compressed/vmlinux.lds

/*
 *
 * Automatically generated file; DO NOT EDIT.
 * Linux/arm 3.18.7 Kernel Configuration
 *
 */
/*
 * Helper macros to use CONFIG_ options in C/CPP expressions. Note that
 * these only work with boolean and tristate options.
 */
/*
 * Getting something that works in C and CPP for an arg that may or may
 * not be defined is tricky.  Here, if we have "#define CONFIG_BOOGER 1"
 * we match on the placeholder define, insert the "0," for arg1 and generate
 * the triplet (0, 1, 0).  Then the last step cherry picks the 2nd arg (a one).
 * When CONFIG_BOOGER is not defined, we generate a (... 1, 0) pair, and when
 * the last step cherry picks the 2nd arg, we get a zero.
```

```
 */

/*

 * IS_ENABLED(CONFIG_FOO) evaluates to 1 if CONFIG_FOO is set to 'y' or 'm',

 * 0 otherwise.

 *

 */

/*

 * IS_BUILTIN(CONFIG_FOO) evaluates to 1 if CONFIG_FOO is set to 'y', 0

 * otherwise. For boolean options, this is equivalent to

 * IS_ENABLED(CONFIG_FOO).

 */

/*

 * IS_MODULE(CONFIG_FOO) evaluates to 1 if CONFIG_FOO is set to 'm', 0

 * otherwise.

 */

/*

 *  Copyright (C) 2000 Russell King

 *

 * This program is free software; you can redistribute it and/or modify

 * it under the terms of the GNU General Public License version 2 as

 * published by the Free Software Foundation.

 */

OUTPUT_ARCH(arm)

ENTRY(_start)

SECTIONS
```

```
{
  /DISCARD/ : {
    *(.ARM.exidx*)
    *(.ARM.extab*)
    /*
     * Discard any r/w data - this produces a link error if we have any,
     * which is required for PIC decompression.  Local data generates
     * GOTOFF relocations, which prevents it being relocated independently
     * of the text/got segments.
     */
    *(.data)
  }
  . = 0;
  _text = .;
  .text : {
    _start = .;
    *(.start)
    *(.text)
    *(.text.*)
    *(.fixup)
    *(.gnu.warning)
    *(.glue_7t)
    *(.glue_7)
  }
  .rodata : {
```

```
  *(.rodata)
  *(.rodata.*)
}
.piggydata : {
  *(.piggydata)
}
. = ALIGN(4);
_etext = .;
.got.plt : { *(.got.plt) }
_got_start = .;
.got : { *(.got) }
_got_end = .;
/* ensure the zImage file size is always a multiple of 64 bits */
/* (without a dummy byte, ld just ignores the empty section) */
.pad : { BYTE(0); . = ALIGN(8); }
_edata = .;
_magic_sig = (0x016f2818);
_magic_start = (_start);
_magic_end = (_edata);
. = ALIGN(8);
__bss_start = .;
.bss : { *(.bss) }
_end = .;
. = ALIGN(8); /* the stack must be 64-bit aligned */
.stack : { *(.stack) }
```

```
  .stab 0 : { *(.stab) }
  .stabstr 0 : { *(.stabstr) }
  .stab.excl 0 : { *(.stab.excl) }
  .stab.exclstr 0 : { *(.stab.exclstr) }
  .stab.index 0 : { *(.stab.index) }
  .stab.indexstr 0 : { *(.stab.indexstr) }
  .comment 0 : { *(.comment) }
}
```

可看出被gzip压缩的vmlinx的binary (Image)位于整个_text segment的尾部，而decompress的code在前面。

```
$ cat arch/arm/boot/compressed/nm vmlinux | grep input_data
00004718 R input_data
0030ba8b R input_data_end
```

compressed Image相对text segment的偏移为从[0x4718, 0x30ba8b)

```
$ cat arch/arm/boot/compressed/nm vmlinux | grep text
0030ba8c D _etext
00000000 T _text
```

从上可看出从0到0x4718的code就是为了解压compressed kernel的。

zImage则是compressed vmlinux的bianry

$ cat arch/arm/boot/.zImage.cmd

cmd_arch/arm/boot/zImage := arm-poky-linux-gnueabi-objcopy -O binary -R .comment -S
 arch/arm/boot/compressed/vmlinux arch/arm/boot/zImage

最后一部由zImage生成uImage

$ arch/arm/boot/.uImage.cmd

cmd_arch/arm/boot/uImage := /bin/bash /home/walterzh/work/gerrit/build-bundle/poky/build/tmp/work/granite2-poky-linux-gnueabi/linux-granite-upstream/3.18.7+gitAUTOINC+e2438e08f1-r0/linux/scripts/mkuboot.sh -A arm -O linux -C none -T kernel -a 0x00008000 -e 0x00008000 -n 'Linux-3.18.7-yocto-standard' -d arch/arm/boot/zImage arch/arm/boot/uImage

scripts/mkuboot.sh只是mkimage的wrapper.

walterzh@walterzh-ThinkPad-T440p:~/work/gerrit/build-bundle/poky/build/tmp/sysroots/x86_64-linux/usr/bin$ ./mkimage

Usage: ./mkimage -l image

-l ==> list image header information

./mkimage [-x] -A arch -O os -T type -C comp -a addr -e ep -n name -d data_file[:data_file...] image

-A ==> set architecture to 'arch'

-O ==> set operating system to 'os'

-T ==> set image type to 'type'

-C ==> set compression type 'comp'

-a ==> set load address to 'addr' (hex)

-e ==> set entry point to 'ep' (hex)

-n ==> set image name to 'name'

-d ==> use image data from 'datafile'

-x ==> set XIP (execute in place)

./mkimage [-D dtc_options] [-f fit-image.its|-F] fit-image

-D => set options for device tree compiler

-f => input filename for FIT source

Signing / verified boot not supported (CONFIG_FIT_SIGNATURE undefined)

./mkimage -V ==> print version information and exit

for example:

mkimage -A arm

-O linux

-T kernel

-C none

-a 0x00008000

-e 0x00008000

-n 'Linux-3.18.7-Yocto'

-d arch/arm/boot/zImage

arch/arm/boot/uImage (生成)

walterzh@walterzh-ThinkPad-T440p:~/work/gerrit/build-bundle/poky/build/tmp/sysroots/x86_64-linux/usr/bin$ ./mkimage -l ~/tmp/uImage-3.18.7-yocto-standard

Image Name: Linux-3.18.7-yocto-standard

Created: Tue Oct 20 12:03:19 2015

Image Type: ARM Linux Kernel Image (uncompressed)

Data Size: 3196512 Bytes = 3121.59 kB = 3.05 MB

Load Address: 00008000

Entry Point: 00008000

总结一下：

① Linux kernel source building creates vmlinux

② vmlinux ==> Image by objcopy (binary file)

③ compress Image and get piggy.gzip by gzip

④ create piggy.gzip.o by piggy.gzip.S

⑤ create compressed vmlinux by merging decompress utility code and piggy.gzip.o

⑥ compressed vmlinux ==> zImage by objcopy (binary file)

⑦ zImage ==> uImage create by mkimage utility