u-boot运行在0x0800,0000开始的physical adress，但链接u-boot.elf的.lds却如下

in u-boot.lds

```
OUTPUT_FORMAT("elf32-littlearm", "elf32-littlearm", "elf32-littlearm")
OUTPUT_ARCH(arm)
ENTRY(_start)
SECTIONS
{
 . = 0x00000000;
 . = ALIGN(4);
 .text :
 {
  *(.__image_copy_start)
  arch/arm/cpu/armv7/start.o (.text*)
  *(.text*)
 }
 . = ALIGN(4);
 .rodata : { *(SORT_BY_ALIGNMENT(SORT_BY_NAME(.rodata*))) }
 . = ALIGN(4);
 .data : {
  *(.data*)
 }
 . = ALIGN(4);
 . = .;
 . = ALIGN(4);
 .u_boot_list : {
  KEEP(*(SORT(.u_boot_list*)));
 }
 . = ALIGN(4);
 .image_copy_end :
 {
  *(.__image_copy_end)
 }
 .rel_dyn_start :
 {
  *(.__rel_dyn_start)
 }
 .rel.dyn : {
  *(.rel*)
 }
 .rel_dyn_end :
 {
  *(.__rel_dyn_end)
 }
 _end = .;
 . = ALIGN(4096);
 .mmutable : {
  *(.mmutable)
 }
 .bss_start __rel_dyn_start (OVERLAY) : {
  KEEP(*(.__bss_start));
  __bss_base = .;
 }
 .bss __bss_base (OVERLAY) : {
  *(.bss*)
   . = ALIGN(4);
```

```
      __bss_limit = .;
    }
    .bss_end __bss_limit (OVERLAY) : {
     KEEP(*(.__bss_end));
    }
    .dynsym _end : { *(.dynsym) }
    .dynbss : { *(.dynbss) }
    .dynstr : { *(.dynstr*) }
    .dynamic : { *(.dynamic*) }
    .plt : { *(.plt*) }
    .interp : { *(.interp*) }
    .gnu : { *(.gnu*) }
    .ARM.exidx : { *(.ARM.exidx*) }
    .gnu.linkonce.armexidx : { *(.gnu.linkonce.armexidx.*) }
  }
```

link script指示text是从0x00000000开始的。

但在u-boot.elf的System.map中

08000000 T __image_copy_start

08000000 T _start

08000020 t _undefined_instruction

08000024 t _software_interrupt

08000028 t _prefetch_abort

0800002c t _data_abort

08000030 t _not_used

08000034 t _irq

08000038 t _fiq

0800003c t _pad

......

Why ?

---

Analyse u-boot's Makefile

in u-boot/Makefile

生成u-boot target

```
$(obj)u-boot:	depend \
		$(SUBDIR_TOOLS) $(OBJS) $(LIBS) $(obj)u-boot.lds
		$(GEN_UBOOT)
```

```
GEN_UBOOT = \
		cd $(LNDIR) && $(LD) $(LDFLAGS) $(LDFLAGS_$(@F)) \
			$(__OBJS) \
			--start-group $(__LIBS) --end-group $(PLATFORM_LIBS) \
			-Map u-boot.map -o u-boot
```

u-boot的依赖

1. $(SUBDIR_TOOLS)

```
SUBDIR_TOOLS = tools
```

u-boot/tools目录下生成的是host下的utility

2. $(OBJS)

```
OBJS := $(addprefix $(obj),$(head-y))
```

$(head-y)

```
head-y := $(CPUDIR)/start.o
head-$(CONFIG_4xx) += arch/powerpc/cpu/ppc4xx/resetvec.o
head-$(CONFIG_MPC85xx) += arch/powerpc/cpu/mpc85xx/resetvec.o
```

in u-boot/config.mk

```
CPUDIR=arch/$(ARCH)/cpu/$(CPU)
```

对G2而言，CPUDIR=arch/arm/cpu/armv7

$(head-y) = arch/arm/cpu/armv7/start.o

3. $(LIBS)

```
LIBS := $(addprefix $(obj),$(sort $(LIBS-y)))
```

$(LIBS)由$(LIBS-y)组成

```
LIBS-y += lib/

LIBS-$(HAVE_VENDOR_COMMON_LIB) += board/$(VENDOR)/common/          (not include)

LIBS-y += $(CPUDIR)/                                          (arch/arm/cpu/armv7)

ifdef SOC

LIBS-y += $(CPUDIR)/$(SOC)/
                                             (arch/arm/cpu/armv7/pegmatite)

endif

LIBS-$(CONFIG_IXP4XX_NPE) += drivers/net/npe/                 (not include)

LIBS-$(CONFIG_OF_EMBED) += dts/                                   (not include)

LIBS-y += arch/$(ARCH)/lib/                                   (arch/arm/lib)

LIBS-y += fs/

LIBS-y += net/

LIBS-y += disk/

LIBS-y += drivers/

LIBS-y += drivers/dma/

LIBS-y += drivers/gpio/

LIBS-y += drivers/i2c/

LIBS-y += drivers/input/

LIBS-y += drivers/mmc/

LIBS-y += drivers/mtd/

LIBS-$(CONFIG_CMD_NAND) += drivers/mtd/nand/                  (not include)

LIBS-y += drivers/mtd/onenand/

LIBS-$(CONFIG_CMD_UBI) += drivers/mtd/ubi/                    (not include)

LIBS-y += drivers/mtd/spi/
```

```
LIBS-y += drivers/net/

LIBS-y += drivers/net/phy/

LIBS-y += drivers/pci/

LIBS-y += drivers/power/ \

        drivers/power/fuel_gauge/ \

        drivers/power/mfd/ \

        drivers/power/pmic/ \

        drivers/power/battery/

LIBS-y += drivers/spi/

LIBS-$(CONFIG_FMAN_ENET) += drivers/net/fm/                    (not include)

LIBS-$(CONFIG_SYS_FSL_DDR) += drivers/ddr/fsl/             (not include)

LIBS-y += drivers/serial/

LIBS-y += drivers/usb/eth/

LIBS-y += drivers/usb/gadget/

LIBS-y += drivers/usb/host/

LIBS-y += drivers/usb/musb/

LIBS-y += drivers/usb/musb-new/

LIBS-y += drivers/usb/phy/

LIBS-y += drivers/usb/ulpi/

LIBS-y += common/

LIBS-y += lib/libfdt/

LIBS-$(CONFIG_API) += api/                                 (not include)

LIBS-$(CONFIG_HAS_POST) += post/                           (not include)

LIBS-y += test/
```

```
ifneq (,$(filter $(SOC), mx25 mx27 mx5 mx6 mx31 mx35 mxs vf610))        (not include)

LIBS-y += arch/$(ARCH)/imx-common/

endif


LIBS-$(CONFIG_ARM) += arch/arm/cpu/

LIBS-$(CONFIG_PPC) += arch/powerpc/cpu/                                   (not include)


LIBS-y += board/$(BOARDDIR)/                                   (board/pegmatite)
```

==> (The u-boot for Granite2 and Gemstone2 include the following source code)

```
LIBS-y += lib/

LIBS-y += arch/arm/cpu/armv7/

LIBS-y += arch/arm/cpu/armv7/pegmatite

LIBS-y += arch/arm/lib/

LIBS-y += fs/

LIBS-y += net/

LIBS-y += disk/

LIBS-y += drivers/

LIBS-y += drivers/dma/

LIBS-y += drivers/gpio/

LIBS-y += drivers/i2c/

LIBS-y += drivers/input/

LIBS-y += drivers/mmc/

LIBS-y += drivers/mtd/
```

```
LIBS-y += drivers/mtd/onenand/

LIBS-y += drivers/mtd/spi/

LIBS-y += drivers/net/

LIBS-y += drivers/net/phy/

LIBS-y += drivers/pci/

LIBS-y += drivers/power/ \

        drivers/power/fuel_gauge/ \

        drivers/power/mfd/ \

        drivers/power/pmic/ \

        drivers/power/battery/

LIBS-y += drivers/spi/

LIBS-y += drivers/serial/

LIBS-y += drivers/usb/eth/

LIBS-y += drivers/usb/gadget/

LIBS-y += drivers/usb/host/

LIBS-y += drivers/usb/musb/

LIBS-y += drivers/usb/musb-new/

LIBS-y += drivers/usb/phy/

LIBS-y += drivers/usb/ulpi/

LIBS-y += common/

LIBS-y += lib/libfdt/

LIBS-y += test/

LIBS-y += arch/arm/cpu/

LIBS-y += board/pegmatite/
```

4. $(obj)u-boot.lds

u-boot.lds由下面命令生成，u-boot/u-boot.lds

```
$(obj)u-boot.lds: $(LDSCRIPT) depend
            $(CPP) $(CPPFLAGS) $(LDPPFLAGS) -ansi -D__ASSEMBLY__ -P - <$< >$@
```

$(LDSCRIPT) = ???

从u-boot/Makefile中下面的script来确定某个.lds看,④被选中，即$(LDSCRIPT) = arch/arm/cpu/armv7/u-boot.lds

```
ifndef LDSCRIPT
      #LDSCRIPT := $(TOPDIR)/board/$(BOARDDIR)/u-boot.lds.debug
      ifdef CONFIG_SYS_LDSCRIPT
            # need to strip off double quotes
            LDSCRIPT := $(CONFIG_SYS_LDSCRIPT:"%"=%)
                                                ①
      endif
endif

# If there is no specified link script, we look in a number of places for it
ifndef LDSCRIPT
      ifeq ($(CONFIG_NAND_U_BOOT),y)
                                                                  ②
            LDSCRIPT := $(TOPDIR)/board/$(BOARDDIR)/u-boot-nand.lds
```

```
        ifeq ($(wildcard $(LDSCRIPT)),)
            LDSCRIPT := $(TOPDIR)/$(CPUDIR)/u-boot-nand.lds
        endif
    endif
    ifeq ($(wildcard $(LDSCRIPT)),)
        LDSCRIPT := $(TOPDIR)/board/$(BOARDDIR)/u-boot.lds
                                    ③
    endif
    ifeq ($(wildcard $(LDSCRIPT)),)
        LDSCRIPT := $(TOPDIR)/$(CPUDIR)/u-boot.lds
                                                    ④
    endif
    ifeq ($(wildcard $(LDSCRIPT)),)
        LDSCRIPT := $(TOPDIR)/arch/$(ARCH)/cpu/u-boot.lds
                                    ⑤
        # We don't expect a Makefile here
        LDSCRIPT_MAKEFILE_DIR =
    endif
    ifeq ($(wildcard $(LDSCRIPT)),)
$(error could not find linker script)
    endif
endif
```

①

CONFIG_SYS_LDSCRIPT is not defined

②

CONFIG_NAND_U_BOOT is not defined

③

board/pegmatite/u-boot.lds ? not exist

④

arch/arm/cpu/armv7/u-boot.lds, exist，所以选中

⑤

$(LDSCRIPT) is not null now.

怎么把obj链接成elf文件?

```
$(LD) $(LDFLAGS) $(LDFLAGS_$(@F)) $(__OBJS) --start-group $(__LIBS) --end-group
$(PLATFORM_LIBS) -Map u-boot.map -o u-boot
```

$(__OBJS) = arch/arm/cpu/armv7/start.o

$(__LIBS) 就是上面依赖3总所有编译生成的obj file。

这里start.o比较特殊，ld必须把它放在整个elf memory layout的头上，所以单独列出来。

```
$ arm-linux-gnueabi-objdump -D u-boot
```

```
-boot：     文件格式 elf32-littlearm
```

```
Disassembly of section .text:


08000000 <__image_copy_start>:
 8000000:       ea000013        b       8000054 <reset>
 8000004:       e59ff014        ldr     pc, [pc, #20]   ; 8000020 <_undefined_instruction>
 8000008:       e59ff014        ldr     pc, [pc, #20]   ; 8000024 <_software_interrupt>
 800000c:       e59ff014        ldr     pc, [pc, #20]   ; 8000028 <_prefetch_abort>
 8000010:       e59ff014        ldr     pc, [pc, #20]   ; 800002c <_data_abort>
 8000014:       e59ff014        ldr     pc, [pc, #20]   ; 8000030 <_not_used>
 8000018:       e59ff014        ldr     pc, [pc, #20]   ; 8000034 <_irq>
 800001c:       e59ff014        ldr     pc, [pc, #20]   ; 8000038 <_fiq>


08000020 <_undefined_instruction>:
 8000020:       08000100        stmdaeq r0, {r8}


08000024 <_software_interrupt>:
 8000024:       08000160        stmdaeq r0, {r5, r6, r8}


08000028 <_prefetch_abort>:
 8000028:       080001c0        stmdaeq r0, {r6, r7, r8}


0800002c <_data_abort>:
 800002c:       08000220        stmdaeq r0, {r5, r9}
```

08000030 <_not_used>:

 8000030:       08000280        stmdaeq r0, {r7, r9}

08000034 <_irq>:

 8000034:       080002e0        stmdaeq r0, {r5, r6, r7, r9}

08000038 <_fiq>:

 8000038:       08000340        stmdaeq r0, {r6, r8, r9}

......

正好对应arch/arm/cpu/armv7/start.S

```
.globl _start
_start: b       reset
        ldr     pc, _undefined_instruction
        ldr     pc, _software_interrupt
        ldr     pc, _prefetch_abort
        ldr     pc, _data_abort
        ldr     pc, _not_used
        ldr     pc, _irq
        ldr     pc, _fiq
#ifdef CONFIG_SPL_BUILD
_undefined_instruction: .word _undefined_instruction
_software_interrupt:    .word _software_interrupt
_prefetch_abort:        .word _prefetch_abort
_data_abort:            .word _data_abort
_not_used:              .word _not_used

......
```

$(PLATFORM_LIBS)是arm toolchain的libgcc

该命令被展开后如下：

cd /home/walterzh/work/gerrit/build-bundle/poky/build/tmp/work/granite2-poky-linux-gnueabi/u-boot-marvell/v2014.01+gitAUTOINC+446d3f8ae8-r0/git && arm-poky-linux-gnueabi-ld --sysroot=/home/walterzh/work/gerrit/build-bundle/poky/build/tmp/sysroots/granite2 -pie -T u-boot.lds --gc-sections -Bstatic -Ttext 0x08000000 arch/arm/cpu/armv7/start.o --start-group arch/arm/cpu/armv7/built-in.o arch/arm/cpu/armv7/pegmatite/built-in.o arch/arm/cpu/built-in.o arch/arm/lib/built-in.o board/pegmatite/built-in.o common/built-in.o disk/built-in.o drivers/built-in.o drivers/dma/built-in.o drivers/gpio/built-in.o drivers/i2c/built-in.o drivers/input/built-in.o drivers/mmc/built-in.o drivers/mtd/built-in.o drivers/mtd/onenand/built-in.o drivers/mtd/spi/built-in.o drivers/net/built-in.o drivers/net/phy/built-in.o drivers/pci/built-in.o drivers/power/battery/built-in.o drivers/power/built-in.o drivers/power/fuel_gauge/built-in.o drivers/power/mfd/built-in.o drivers/power/pmic/built-in.o drivers/serial/built-in.o drivers/spi/built-in.o drivers/usb/eth/built-in.o drivers/usb/gadget/built-in.o drivers/usb/host/built-in.o drivers/usb/musb-new/built-in.o drivers/usb/musb/built-in.o drivers/usb/phy/built-in.o drivers/usb/ulpi/built-in.o fs/built-in.o lib/built-in.o lib/libfdt/built-in.o net/built-in.o test/built-in.o --end-group /home/walterzh/work/gerrit/build-bundle/poky/build/tmp/work/granite2-poky-linux-gnueabi/u-boot-marvell/v2014.01+gitAUTOINC+446d3f8ae8-r0/git/arch/arm/lib/eabi_compat.o -L /home/walterzh/work/gerrit/build-bundle/poky/build/tmp/sysroots/granite2/usr/lib/arm-poky-linux-gnueabi/4.8.1 -lgcc -Map u-boot.map -o u-boot

这里ld option为

-pie -T u-boot.lds --gc-sections -Bstatic -Ttext 0x08000000

-pie

Create a position independent executable,也就是生成relocatable code(因为没有绝对地址寻址，只有相对地址寻址)

-Bstatic

不要链接dynamic library,这是显而易见的。

-Ttext 0x08000000

把code定位在0x08000000,也就是128M地址处。

这也就是虽然u-boot.lds中是

 . = 0x00000000;

但最终链接生成的code是位于0x0800,0000。

另外，-pie也是关键，只有relocatable code才可以从0搬移到0x08000000而依然可以运行。

有个疑问，为什么不直接在u-boot.lds写

. = 0x08000000;

这样-pie option可以没有，不是更直白更简单吗？

现在这种写法的好处是u-boot的载入地址可以在include/configs/pegmatite.h中由下面的macro决定

#define CONFIG_SYS_TEXT_BASE 0x08000000