Early ioremap

2016-12-13 (http://jake.dothome.co.kr/early-ioremap/) Moon Young-il (http://jake.dothome.co.kr/author/admin/) Linux Kernel (http://jake.dothome.co.kr/category/linux/)

<kernel v5.0>

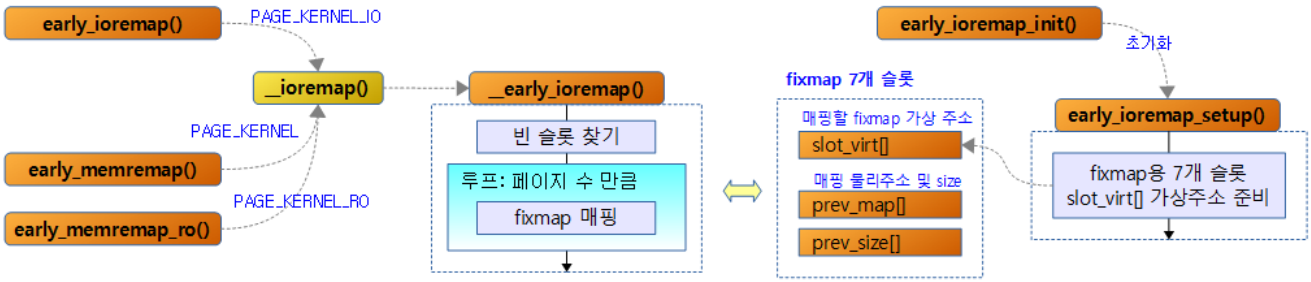
Early IO Mapping

After the paging_init( ) function is performed during kernel bootup, the memory and input/output (I/O) devices can be formally mapped and used in the virtual address space. However, we have prepared an early way to handle these actions in case they must be mapped and used before they are completed. When using such APIs, it is necessary to first access the BIOS information of a specific system and obtain the information of various memories and devices before the mapping process can be processed. The main uses are as follows:

- You need to access the ACPI table to get the configuration information of various devices.
- You need to access the EFI table to retrieve configuration information for various devices.
- If you need to read certain setting information, for example, from some devices

If io mapping is required in the early hours, it can be done using up to 256 fixmap mapping spaces of 7K increments.

The following figure shows the invocation relationship between the two functions, early_ioremap() and early_ioremap_init().



(http://jake.dothome.co.kr/wp-content/uploads/2016/12/early_ioremap-1.png)

Initialize Early IO Mapping

early_ioremap_init() – ARM32 & ARM64

Learn more about preparing for early I/O mapping at boot time before using ioremap. The following function immediately calls the early_ioremap_setup( ) function.

- ARM32 also supports early ioremap starting with kernel v4.5-rc1.
  - 참고: ARM: add support for generic early_ioremap/early_memremap (https://github.com/torvalds/linux/commit/2937367b8a4b0d46ce3312cb997e4a240b02cf15#diff-dffe014812632819972caa2be7bee00d)

arch/arm64/mm/ioremap.c

```
1 /*
2  * Must be called after early_fixmap_init
3  */
4 void __init early_ioremap_init(void)
5 {
6         early_ioremap_setup();
7 }
```

paging_init() is not complete, so you can't use the regular ioremap() function, but you need to map it temporarily in the early hours.

- fixmap, so the early_fixmap_init() function must be called first to initialize.

early_ioremap_setup()

Learn how to prepare seven 7K virtual address zones in the fixmap area for early I/O mapping.

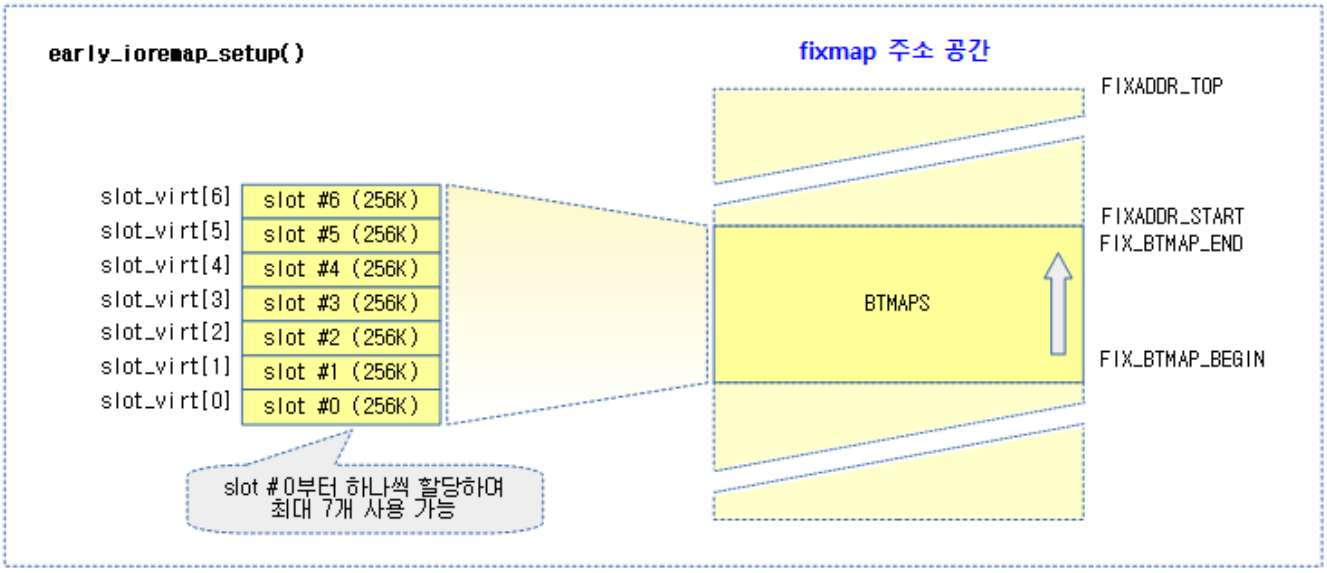
mm/early_ioremap.c

```
01 void __init early_ioremap_setup(void)
02 {
03         int i;
04
05         for (i = 0; i < FIX_BTMAPS_SLOTS; i++)
06                 if (WARN_ON(prev_map[i]))
07                         break;
08
09         for (i = 0; i < FIX_BTMAPS_SLOTS; i++)
10                 slot_virt[i] = __fix_to_virt(FIX_BTMAP_BEGIN - NR_FIX_BTMAPS*i);
11 }
```

In a slot_virt[] array of seven arrays, the virtual address assigned for the early ioremap in the fixmap is assigned.

- Loop around FIX_BTMAPS_SLOTS(5) number of lines 7~7 of the code and display a warning message if a value is set in the global prev_map[ ] array
  - Each of the seven address spaces can use up to 7K.
- In lines 9~10 of code, loop around the number of FIX_BTMAPS_SLOTS(7) and set the BTMAP virtual address of the fixmap in the global slot_virt[ ] array. There are 7 fixmap entries, each of which points to a virtual address corresponding to the fixmap's BTMAP, with an interval of 256K between entries.
  - slot_virt[0] = FIX_BTMAP_BEGIN
  - slot_virt[1] = FIX_BTMAP_BEGIN + 256K

The following figure shows that the early_ioremap_setup( ) function is called, initializing seven slots in the BTMAPS area of the fixmap used for early IO mapping.



(http://jake.dothome.co.kr/wp-content/uploads/2016/12/early_ioremap_setup-1a.png)

Key APIs

early_ioremap()

mm/early_ioremap.c

```
1  /* Remap an IO device */
2  void __init __iomem *
3  early_ioremap(resource_size_t phys_addr, unsigned long size)
4  {
5          return __early_ioremap(phys_addr, size, FIXMAP_PAGE_IO);
6  }
```

From the physical address @phys_addr, the IO devices corresponding to the @size are mapped early, and then the mapped virtual address is returned.

- Initial mapping of up to 256 io devices of 7K size.
- FIXMAP_PAGE_IO
  - ARM32
    - L_PTE_YOUNG | L_PTE_PRESENT | L_PTE_XN | L_PTE_DIRTY | L_PTE_MT_DEV_SHARED | L_PTE_SHARED have a song.
  - ARM64
    - PROT_DEVICE_nGnRE has mapping properties.

early_memremap()

mm/early_ioremap.c

```
1  /* Remap memory */
2  void __init *
3  early_memremap(resource_size_t phys_addr, unsigned long size)
4  {
5          return (__force void *)__early_ioremap(phys_addr, size,
6                                                 FIXMAP_PAGE_NORMAL);
7  }
```

From the physical address @phys_addr, the memory area corresponding to the @size is initially mapped and the mapped virtual address is returned.

- Early mapping of up to 256 memory regions of 7K size.
- FIXMAP_PAGE_NORMAL
  - ARM32
    - L_PTE_XN is a kernel memory property with an added attribute.
  - ARM64
    - It is the same as the kernel memory properties.

__early_ioremap()

mm/early_ioremap.c

```
01 static void __init __iomem *
02 __early_ioremap(resource_size_t phys_addr, unsigned long size, pgprot_t
   prot)
03 {
04         unsigned long offset;
05         resource_size_t last_addr;
06         unsigned int nrpages;
07         enum fixed_addresses idx;
08         int i, slot;
09
10         WARN_ON(system_state != SYSTEM_RUNNING);
11
12         slot = -1;
13         for (i = 0; i < FIX_BTMAPS_SLOTS; i++) {
14                 if (!prev_map[i]) {
15                         slot = i;
16                         break;
17                 }
18         }
19
20         if (WARN(slot < 0, "%s(%08llx, %08lx) not found slot\n",
21                  __func__, (u64)phys_addr, size))
22                 return NULL;
23
24         /* Don't allow wraparound or zero size */
25         last_addr = phys_addr + size - 1;
26         if (WARN_ON(!size || last_addr < phys_addr))
27                 return NULL;
28
29         prev_size[slot] = size;
30         /*
31          * Mappings have to be page-aligned
32          */
33         <span class="blob-code-inner">offset = <span class="pl-c1 x x-fi
   rst">offset_in_page</span><span class="x x-last">(</span>phys_addr<span
   class="x x-first x-last">)</span>;</span>
34         phys_addr &= PAGE_MASK;
35         size = PAGE_ALIGN(last_addr + 1) - phys_addr;
36
37         /*
38          * Mappings have to fit in the FIX_BTMAP area.
39          */
40         nrpages = size >> PAGE_SHIFT;
41         if (WARN_ON(nrpages > NR_FIX_BTMAPS))
42                 return NULL;
43
44         /*
45          * Ok, go for it..
```

```
46          */
47         idx = FIX_BTMAP_BEGIN - NR_FIX_BTMAPS*slot;
48         while (nrpages > 0) {
49                 if (after_paging_init)
50                         __late_set_fixmap(idx, phys_addr, prot);
51                 else
52                         __early_set_fixmap(idx, phys_addr, prot);
53                 phys_addr += PAGE_SIZE;
54                 --idx;
55                 --nrpages;
56         }
57         WARN(early_ioremap_debug, "%s(%08llx, %08lx) [%d] => %08lx + %08
   lx\n",
58              __func__, (u64)phys_addr, size, slot, offset, slot_virt[slo
   t]);
59
60         prev_map[slot] = (void __iomem *)(offset + slot_virt[slot]);
61         return prev_map[slot];
62 }
```

Map the requested physical address and size to the virtual address corresponding to the BTMAPS slot of fixmap with the prot attribute. If it fails, it returns null. The maximum number of io maps that can be done early on arm64 is FIX_BTMAPS_SLOTS (7).

- About BTMAPS slot management for fixmaps
  - slot_virt[]: The fixmap virtual address corresponding to each slot initialized at bootup time (in 256K units)
  - prev_map[]: Virtual address mapped to fixmap
  - prev_size[]: Mapped size

- Select the empty slot you want to use on lines 12-18 of the code. If the global prev_map[ ] value is not set after the loop of FIX_BTMAPS_SLOTS(7) and the current counter i value is set to the slot, the slot is selected.
- If an empty slot is not found while looping in line 20~22 of code, it will print a warning message and return a NULL value
- If the size is 25 or the end address is outside the system address range on lines 27~0, it will display a warning message and return a NULL value.
- In lines 29~34 of code, put the request size in prev_size[ ]. offset calculates only the remaining offset of the request physical address in pages, and the physical address is sorted down by page
- At line 35 of the code, calculate the size value sorted by page. Sort the initial requested physical start address + size value by page, and subtract the physical address that was sorted down by page (the minimum value will be in page).
- In line 40~42 of the code, count the number of pages with the size value. If the size exceeds 256K, it returns NULL
- Loop around the number of pages calculated on line 48 of code.
- If the global after_paging_init is set in line 49~50 of the code, call the __late_clear_fixmap( ) macro action.
  - e.g. arm64 and x86 just call the __set_fixmap() function without a separate late processing function.
- If the after_paging_init is not set in line 51~52, call the _ _early_set_fixmap( ) macro action.

    - e.g. in the case of arm64, it just calls the __set_fixmap() function without a separate early processing function.
    - e.g. x64 calls the __early_set_fixmap() function, which is a separate early processing function.
- On line 60 of the code, add the fixmap virtual address + offset to prev_map[slot]

__early_set_fixmap()

arm64/include/asm/fixmap.h

```
1 #define __early_set_fixmap __set_fixmap
```

If the early_ioremap() function is called before paging_init() is complete, it is mapped to a fixmap.

__late_set_fixmap()

mm/early_ioremap.c

```
1 #define __late_set_fixmap __set_fixmap
```

If the early_ioremap() function is called after paging_init() completes, it is mapped to a fixmap.

- As of kernel v4.1-rc1, early_ioremap() can be called even after booting. UNTIL THEN, BUG() WAS CALLED.
    - 참고: ARM64: allow late use of early_ioremap (https://github.com/torvalds/linux/commit/b2cedba09dd7034c144c03eadc09ee1e4d791590)

---

early_ioremap() API Provision Expiration

early_ioremap_reset()

mm/early_ioremap.c

```
1 void __init early_ioremap_reset(void)
2 {
3         early_ioremap_shutdown();
4         after_paging_init = 1;
5 }
```

Now that regular paging is ready, it's time to stop using the early_ioremap() API and use the regular ioremap().

- ACCORDING TO THE ARCHITECTURE, A BUG() WILL BE RAISED WHEN THE early_ioremap() FUNCTION IS CALLED AFTER THIS FUNCTION HAS BEEN PERFORMED.
    - In the case of x86 and arm64, the use of the early_ioremap() function is still allowed.

Other APIs

- early_memremap_ro()
  - early_memremap(), but with the addition of a read only attribute.
- early_memremap_prot()
  - early_memremap(), but you can add @prot_val attributes by requesting them.

consultation

- Ioremap (http://jake.dothome.co.kr/ioremap) | Qc
- Fixmap (http://jake.dothome.co.kr/fixmap) | Qc

---

3 thoughts to “Early ioremap”

DINGULDINGGUL
2020-11-21 13:12 (http://jake.dothome.co.kr/early-ioremap/#comment-291130)

If you look at the code of https://elixir.bootlin.com/linux/v4.11.11/source/arch/arm64/include/asm/fixmap.h#L71 (https://elixir.bootlin.com/linux/v4.11.11/source/arch/arm64/include/asm/fixmap.h#L71), you can see that

#define FIXADDR_START (FIXADDR_TOP – FIXADDR_SIZE)
enum fixed_addresses {
FIX_HOLE,
…
FIX_BTMAP_END = __end_of_permanent_fixed_addresses,
FIX_BTMAP_BEGIN = FIX_BTMAP_END + TOTAL_FIX_BTMAPS – 1,
…
}; Is.

slot_virt[0]~[6] The order of layout and FIX_BTMAP_END, FIX_BTMAP_BEGIN layout should be reversed, please check

RESPONSE (/EARLY-IOREMAP/?REPLYTOCOM=291130#RESPOND)

DINGULDINGGUL
2020-11-21 13:43 (http://jake.dothome.co.kr/early-ioremap/#comment-291131)

It is calculated by FIXADDR_TOP the enum value as an intex ((x) << PAGE_SHIFT)). The picture on the blog seems to fit.

RESPONSE (/EARLY-IOREMAP/?REPLYTOCOM=291131#RESPOND)

MOON YOUNG-IL (HTTP://JAKE.DOTHOME.CO.KR)
2020-11-21 14:05 (http://jake.dothome.co.kr/early-ioremap/#comment-291132)

Hello? You've got a quick check. We hope you have a great day. ^^

RESPONSE (/EARLY-IOREMAP/?REPLYTOCOM=291132#RESPOND)

LEAVE A COMMENT

Your email will not be published. Required fields are marked with *

Comments

name *

email *

Website

댓글 작성

‹ Ioremap (http://jake.dothome.co.kr/ioremap/)

User virtual maps (brk) › (http://jake.dothome.co.kr/user-virtual-maps-brk/)

문c 블로그 (2015 ~ 2023)