驱动加载过程

先看几个宏：

xref/kernel-4.19/include/linux/module.h#87

[87](http://localhost:8080/source/xref/kernel-4.19/include/linux/module.h" \l "87)  #**define** **[module\_init](http://localhost:8080/source/s?refs=module_init&project=kernel-4.19)**(**[x](http://localhost:8080/source/s?refs=x&project=kernel-4.19)**) [\_\_initcall](http://localhost:8080/source/s?defs=__initcall&project=kernel-4.19)([x](http://localhost:8080/source/s?defs=x&project=kernel-4.19));

xref/kernel-4.19/include/linux/init.h#199

199 #define \_\_define\_initcall(fn, id) \_\_\_define\_initcall(fn, id, .initcall##id)

215 #define pure\_initcall(fn) \_\_define\_initcall(fn, 0)

217 #define core\_initcall(fn) \_\_define\_initcall(fn, 1)

218 #define core\_initcall\_sync(fn) \_\_define\_initcall(fn, 1s)

219 #define postcore\_initcall(fn) \_\_define\_initcall(fn, 2)

220 #define postcore\_initcall\_sync(fn) \_\_define\_initcall(fn, 2s)

221 #define arch\_initcall(fn) \_\_define\_initcall(fn, 3)

222 #define arch\_initcall\_sync(fn) \_\_define\_initcall(fn, 3s)

223 #define subsys\_initcall(fn) \_\_define\_initcall(fn, 4)

224 #define subsys\_initcall\_sync(fn) \_\_define\_initcall(fn, 4s)

225 #define fs\_initcall(fn) \_\_define\_initcall(fn, 5)

226 #define fs\_initcall\_sync(fn) \_\_define\_initcall(fn, 5s)

227 #define rootfs\_initcall(fn) \_\_define\_initcall(fn, rootfs)

228 #define device\_initcall(fn) \_\_define\_initcall(fn, 6)

229 #define device\_initcall\_sync(fn) \_\_define\_initcall(fn, 6s)

230 #define late\_initcall(fn) \_\_define\_initcall(fn, 7)

231 #define late\_initcall\_sync(fn) \_\_define\_initcall(fn, 7s)

233 #define \_\_initcall(fn) device\_initcall(fn)

从上述宏定义中可以指知道，我们通常使用的module\_init的装载优先级是6。系统根据优先级的顺序装载所有的驱动程序，数值越小，优先级越高。每个宏的具体含义参考末尾附录部分。

当bootloader装载并进入kernel后，执行的第一个方法是start\_kernel，一直到module\_init中注册的方法发生。调用的过程如如下：

xref/kernel-4.19/init/main.c#530

[530](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "530)  [asmlinkage](http://localhost:8080/source/s?defs=asmlinkage&project=kernel-4.19) [\_\_visible](http://localhost:8080/source/s?defs=__visible&project=kernel-4.19) **void** [\_\_init](http://localhost:8080/source/s?defs=__init&project=kernel-4.19) **[start\_kernel](http://localhost:8080/source/s?refs=start_kernel&project=kernel-4.19)**(**void**)

[736](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "736)   **[rest\_init](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "rest_init)**();

[396](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "396)  **static** [noinline](http://localhost:8080/source/s?defs=noinline&project=kernel-4.19) **void** [\_\_ref](http://localhost:8080/source/s?defs=__ref&project=kernel-4.19) **[rest\_init](http://localhost:8080/source/s?refs=rest_init&project=kernel-4.19)**(**void**)

[407](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "407)   **[pid](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "pid)** = [kernel\_thread](http://localhost:8080/source/s?defs=kernel_thread&project=kernel-4.19)(**[kernel\_init](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "kernel_init)**, [NULL](http://localhost:8080/source/s?defs=NULL&project=kernel-4.19), [CLONE\_FS](http://localhost:8080/source/s?defs=CLONE_FS&project=kernel-4.19));

[1058](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "1058)  **static** **int** [\_\_ref](http://localhost:8080/source/s?defs=__ref&project=kernel-4.19) **[kernel\_init](http://localhost:8080/source/s?refs=kernel_init&project=kernel-4.19)**(**void** \***[unused](http://localhost:8080/source/s?refs=unused&project=kernel-4.19)**)

[1062](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "1062)   **[kernel\_init\_freeable](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "kernel_init_freeable)**();

[1112](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "1112)  **static** [noinline](http://localhost:8080/source/s?defs=noinline&project=kernel-4.19) **void** [\_\_init](http://localhost:8080/source/s?defs=__init&project=kernel-4.19) **[kernel\_init\_freeable](http://localhost:8080/source/s?refs=kernel_init_freeable&project=kernel-4.19)**(**void**)

[1145](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "1145)   **[do\_basic\_setup](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "do_basic_setup)**();

[970](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "970)  **static** **void** [\_\_init](http://localhost:8080/source/s?defs=__init&project=kernel-4.19) **[do\_basic\_setup](http://localhost:8080/source/s?refs=do_basic_setup&project=kernel-4.19)**(**void**)

[978](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "978)   **[do\_initcalls](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "do_initcalls)**();

[955](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "955)  **static** **void** [\_\_init](http://localhost:8080/source/s?defs=__init&project=kernel-4.19) **[do\_initcalls](http://localhost:8080/source/s?refs=do_initcalls&project=kernel-4.19)**(**void**)

[960](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "960)   **[do\_initcall\_level](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "do_initcall_level)**([level](http://localhost:8080/source/s?defs=level&project=kernel-4.19));

[939](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "939)  **static** **void** [\_\_init](http://localhost:8080/source/s?defs=__init&project=kernel-4.19) **[do\_initcall\_level](http://localhost:8080/source/s?refs=do_initcall_level&project=kernel-4.19)**(**int** **[level](http://localhost:8080/source/s?refs=level&project=kernel-4.19)**) // [level](http://localhost:8080/source/s?refs=level&project=kernel-4.19) = 6

[951](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "951)  **for** ([fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19) = **[initcall\_levels](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "initcall_levels)**[[level](http://localhost:8080/source/s?defs=level&project=kernel-4.19)]; [fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19) < **[initcall\_levels](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "initcall_levels)**[[level](http://localhost:8080/source/s?defs=level&project=kernel-4.19)+1]; [fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19)++)

[952](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "952)   **[do\_one\_initcall](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "do_one_initcall)**([initcall\_from\_entry](http://localhost:8080/source/s?defs=initcall_from_entry&project=kernel-4.19)([fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19))); // 查找fn方法在内存中的地址

[874](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "874)  **int** [\_\_init\_or\_module](http://localhost:8080/source/s?defs=__init_or_module&project=kernel-4.19) **[do\_one\_initcall](http://localhost:8080/source/s?refs=do_one_initcall&project=kernel-4.19)**([initcall\_t](http://localhost:8080/source/s?defs=initcall_t&project=kernel-4.19) **[fn](http://localhost:8080/source/s?refs=fn&project=kernel-4.19)**)

[884](http://localhost:8080/source/xref/kernel-4.19/init/main.c" \l "884)   [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19)();

上述[fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19)就是module\_init([fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19))中的入参。

接下来看看[fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19)的地址是如何确定的？查看上面[initcall\_from\_entry](http://localhost:8080/source/s?defs=initcall_from_entry&project=kernel-4.19)(fn)方法的定义：

915 static initcall\_entry\_t \***initcall\_levels**[] \_\_initdata = {

916 \_\_initcall0\_start,

917 \_\_initcall1\_start,

918 \_\_initcall2\_start,

919 \_\_initcall3\_start,

920 \_\_initcall4\_start,

921 \_\_initcall5\_start,

922 \_\_initcall6\_start,

923 \_\_initcall7\_start,

924 \_\_initcall\_end,

925 };

[122](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "122)  **static** **inline** **[initcall\_t](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "initcall_t)** **[initcall\_from\_entry](http://localhost:8080/source/s?refs=initcall_from_entry&project=kernel-4.19)**([initcall\_entry\_t](http://localhost:8080/source/s?defs=initcall_entry_t&project=kernel-4.19) \***[entry](http://localhost:8080/source/s?refs=entry&project=kernel-4.19)**)

[123](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "123)  {

[124](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "124)   **return** [offset\_to\_ptr](http://localhost:8080/source/s?defs=offset_to_ptr&project=kernel-4.19)([entry](http://localhost:8080/source/s?defs=entry&project=kernel-4.19));

[125](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "125)  }

xref/kernel-4.19/include/linux/compiler.h#372

368 /\*\*

369 \* offset\_to\_ptr - convert a relative memory offset to an absolute pointer

370 \* @off: the address of the 32-bit offset value

371 \*/

372 **static inline** **void** \*offset\_to\_ptr(const int \*off)

373 {

374 **return** (void \*)((unsigned long)off + \*off);

375 }

上述过程是内核启动过程中查找fn在内存中的绝对地址的过程。

那么fn的绝对地址是如何确定的？

在执行make命令时，先将源码编译成.o，然后由ld链接器再根据链接描述文件lds，对多个.o文件进行组装。

xref/kernel-4.19/arch/x86/kernel/vmlinux.lds#82

[.init.data](http://localhost:8080/source/s?path=.init.data&project=kernel-4.19) : AT(ADDR([.init.data](http://localhost:8080/source/s?path=.init.data&project=kernel-4.19)) - 0xffffffff80000000) { KEEP(\*(SORT(\_\_\_kentry+\*))) \*([.init.data](http://localhost:8080/source/s?path=.init.data&project=kernel-4.19) [init.data.](http://localhost:8080/source/s?path=init.data.&project=kernel-4.19)\*) \*([.meminit.data](http://localhost:8080/source/s?path=.meminit.data&project=kernel-4.19)\*) \*([.init.rodata](http://localhost:8080/source/s?path=.init.rodata&project=kernel-4.19) [.init.rodata.](http://localhost:8080/source/s?path=.init.rodata.&project=kernel-4.19)\*) . = ALIGN(8); \_\_start\_ftrace\_events = .; KEEP(\*(\_ftrace\_events)) \_\_stop\_ftrace\_events = .; \_\_start\_ftrace\_eval\_maps = .; KEEP(\*(\_ftrace\_eval\_map)) \_\_stop\_ftrace\_eval\_maps = .; . = ALIGN(8); \_\_start\_kprobe\_blacklist = .; KEEP(\*(\_kprobe\_blacklist)) \_\_stop\_kprobe\_blacklist = .; . = ALIGN(32); \_\_start\_error\_injection\_whitelist = .; KEEP(\*(\_error\_injection\_whitelist)) \_\_stop\_error\_injection\_whitelist = .; \*([.meminit.rodata](http://localhost:8080/source/s?path=.meminit.rodata&project=kernel-4.19)) . = ALIGN(8); \_\_clk\_of\_table = .; KEEP(\*(\_\_clk\_of\_table)) KEEP(\*(\_\_clk\_of\_table\_end)) . = ALIGN(8); \_\_cpu\_method\_of\_table = .; KEEP(\*(\_\_cpu\_method\_of\_table)) KEEP(\*(\_\_cpu\_method\_of\_table\_end)) . = ALIGN(8); \_\_cpuidle\_method\_of\_table = .; KEEP(\*(\_\_cpuidle\_method\_of\_table)) KEEP(\*(\_\_cpuidle\_method\_of\_table\_end)) . = ALIGN(32); \_\_dtb\_start = .; KEEP(\*([.dtb.init.rodata](http://localhost:8080/source/s?path=.dtb.init.rodata&project=kernel-4.19))) \_\_dtb\_end = .; . = ALIGN(8); \_\_irqchip\_acpi\_probe\_table = .; KEEP(\*(\_\_irqchip\_acpi\_probe\_table)) \_\_irqchip\_acpi\_probe\_table\_end = .; . = ALIGN(8); \_\_timer\_acpi\_probe\_table = .; KEEP(\*(\_\_timer\_acpi\_probe\_table)) \_\_timer\_acpi\_probe\_table\_end = .; . = ALIGN(8); \_\_earlycon\_table = .; KEEP(\*(\_\_earlycon\_table)) \_\_earlycon\_table\_end = .; . = ALIGN(16); \_\_setup\_start = .; KEEP(\*([.init.setup](http://localhost:8080/source/s?path=.init.setup&project=kernel-4.19))) \_\_setup\_end = .; \_\_initcall\_start = .; KEEP(\*([.initcallearly.init](http://localhost:8080/source/s?path=.initcallearly.init&project=kernel-4.19))) \_\_initcall0\_start = .; KEEP(\*([.initcall0.init](http://localhost:8080/source/s?path=.initcall0.init&project=kernel-4.19))) KEEP(\*([.initcall0s.init](http://localhost:8080/source/s?path=.initcall0s.init&project=kernel-4.19))) \_\_initcall1\_start = .; KEEP(\*([.initcall1.init](http://localhost:8080/source/s?path=.initcall1.init&project=kernel-4.19))) KEEP(\*([.initcall1s.init](http://localhost:8080/source/s?path=.initcall1s.init&project=kernel-4.19))) \_\_initcall2\_start = .; KEEP(\*([.initcall2.init](http://localhost:8080/source/s?path=.initcall2.init&project=kernel-4.19))) KEEP(\*([.initcall2s.init](http://localhost:8080/source/s?path=.initcall2s.init&project=kernel-4.19))) \_\_initcall3\_start = .; KEEP(\*([.initcall3.init](http://localhost:8080/source/s?path=.initcall3.init&project=kernel-4.19))) KEEP(\*([.initcall3s.init](http://localhost:8080/source/s?path=.initcall3s.init&project=kernel-4.19))) \_\_initcall4\_start = .; KEEP(\*([.initcall4.init](http://localhost:8080/source/s?path=.initcall4.init&project=kernel-4.19))) KEEP(\*([.initcall4s.init](http://localhost:8080/source/s?path=.initcall4s.init&project=kernel-4.19))) \_\_initcall5\_start = .; KEEP(\*([.initcall5.init](http://localhost:8080/source/s?path=.initcall5.init&project=kernel-4.19))) KEEP(\*([.initcall5s.init](http://localhost:8080/source/s?path=.initcall5s.init&project=kernel-4.19))) \_\_initcallrootfs\_start = .; KEEP(\*([.initcallrootfs.init](http://localhost:8080/source/s?path=.initcallrootfs.init&project=kernel-4.19))) KEEP(\*([.initcallrootfss.init](http://localhost:8080/source/s?path=.initcallrootfss.init&project=kernel-4.19))) \_\_initcall6\_start = .; KEEP(\*([.initcall6.init](http://localhost:8080/source/s?path=.initcall6.init&project=kernel-4.19))) KEEP(\*([.initcall6s.init](http://localhost:8080/source/s?path=.initcall6s.init&project=kernel-4.19))) \_\_initcall7\_start = .; KEEP(\*([.initcall7.init](http://localhost:8080/source/s?path=.initcall7.init&project=kernel-4.19))) KEEP(\*([.initcall7s.init](http://localhost:8080/source/s?path=.initcall7s.init&project=kernel-4.19))) \_\_initcall\_end = .; \_\_con\_initcall\_start = .; KEEP(\*([.con\_initcall.init](http://localhost:8080/source/s?path=.con_initcall.init&project=kernel-4.19))) \_\_con\_initcall\_end = .; \_\_security\_initcall\_start = .; KEEP(\*([.security\_initcall.init](http://localhost:8080/source/s?path=.security_initcall.init&project=kernel-4.19))) \_\_security\_initcall\_end = .; . = ALIGN(4); \_\_initramfs\_start = .; KEEP(\*([.init.ramfs](http://localhost:8080/source/s?path=.init.ramfs&project=kernel-4.19))) . = ALIGN(8); KEEP(\*([.init.ramfs.info](http://localhost:8080/source/s?path=.init.ramfs.info&project=kernel-4.19))) }

编译log显示链接过程如下：

... ld -m elf\_x86\_64 -z noreloc-overflow -pie --no-dynamic-linker -T arch/x86/boot/compressed/vmlinux.lds arch/x86/boot/compressed/head\_64.o arch/x86/boot/compressed/misc.o arch/x86/boot/compressed/string.o arch/x86/boot/compressed/cmdline.o arch/x86/boot/compressed/error.o arch/x86/boot/compressed/piggy.o arch/x86/boot/compressed/cpuflags.o arch/x86/boot/compressed/early\_serial\_console.o arch/x86/boot/compressed/kaslr.o arch/x86/boot/compressed/kaslr\_64.o arch/x86/boot/compressed/mem\_encrypt.o arch/x86/boot/compressed/pgtable\_64.o arch/x86/boot/compressed/eboot.o arch/x86/boot/compressed/efi\_stub\_64.o drivers/firmware/efi/libstub/lib.a arch/x86/boot/compressed/efi\_thunk\_64.o -o arch/x86/boot/compressed/**vmlinux**

lds文件相当于ELF文件的组织框架，将如下宏展开后就将fn这个符号填充到了上面[.initcall6.init](http://localhost:8080/source/s?path=.initcall6.init&project=kernel-4.19)这个位置。

#**define** **[\_\_\_define\_initcall](http://localhost:8080/source/s?refs=___define_initcall&project=kernel-4.19)**(**[fn](http://localhost:8080/source/s?refs=fn&project=kernel-4.19)**, **[id](http://localhost:8080/source/s?refs=id&project=kernel-4.19)**, **[\_\_sec](http://localhost:8080/source/s?refs=__sec&project=kernel-4.19)**) **static** **[initcall\_t](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "initcall_t)** [\_\_initcall\_](http://localhost:8080/source/s?defs=__initcall_&project=kernel-4.19)##[fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19)##[id](http://localhost:8080/source/s?defs=id&project=kernel-4.19) [\_\_used](http://localhost:8080/source/s?defs=__used&project=kernel-4.19) \

[\_\_attribute\_\_](http://localhost:8080/source/s?defs=__attribute__&project=kernel-4.19)(([\_\_section\_\_](http://localhost:8080/source/s?defs=__section__&project=kernel-4.19)(#[\_\_sec](http://localhost:8080/source/s?defs=__sec&project=kernel-4.19) ".init"))) = [fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19);

#**define** **[\_\_define\_initcall](http://localhost:8080/source/s?refs=__define_initcall&project=kernel-4.19)**(**[fn](http://localhost:8080/source/s?refs=fn&project=kernel-4.19)**, **[id](http://localhost:8080/source/s?refs=id&project=kernel-4.19)**) [\_\_\_define\_initcall](http://localhost:8080/source/s?defs=___define_initcall&project=kernel-4.19)([fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19), [id](http://localhost:8080/source/s?defs=id&project=kernel-4.19), .[initcall](http://localhost:8080/source/s?defs=initcall&project=kernel-4.19)##[id](http://localhost:8080/source/s?defs=id&project=kernel-4.19))

#**define** **[device\_initcall](http://localhost:8080/source/s?refs=device_initcall&project=kernel-4.19)**(**[fn](http://localhost:8080/source/s?refs=fn&project=kernel-4.19)**) **[\_\_define\_initcall](http://localhost:8080/source/xref/kernel-4.19/include/linux/init.h" \l "__define_initcall)**([fn](http://localhost:8080/source/s?defs=fn&project=kernel-4.19), 6)

上面[\_\_section\_\_](http://localhost:8080/source/s?defs=__section__&project=kernel-4.19)(#[\_\_sec](http://localhost:8080/source/s?defs=__sec&project=kernel-4.19) ".init")的展开结果就是：

[\_\_section\_\_](http://localhost:8080/source/s?defs=__section__&project=kernel-4.19)([.initcall6.init](http://localhost:8080/source/s?path=.initcall6.init&project=kernel-4.19)) = fn

**附注：同一等级的优先级的驱动，加载顺序是链接过程决定的，结果是不确定的，我们无法去手动设置谁先谁后。由此也可以得出一个推论，上述的modulue\_init在动态加载方式下用postcore\_initcall等替换也是可以的，若是静态加载驱动的方式，则须确保调用到的外部方法已经在内核中初始化成功。比如：**

**xref/kernel-4.19/drivers/spi/spi.c#3621**

**[postcore\_initcall](http://localhost:8080/source/s?defs=postcore_initcall&project=kernel-4.19)**([spi\_init](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c" \l "spi_init));

**上述是module\_init的外部调用过程**。

**驱动内部加载过程**

**驱动更常见的写法如下，以spidev为例：**

**static** **struct** [spi\_driver](http://localhost:8080/source/s?defs=spi_driver&project=kernel-4.19) **[spidev\_spi\_driver](http://localhost:8080/source/s?refs=spidev_spi_driver&project=kernel-4.19)** = {

.[driver](http://localhost:8080/source/s?defs=driver&project=kernel-4.19) = {

.[name](http://localhost:8080/source/s?defs=name&project=kernel-4.19) = "spidev",

.[of\_match\_table](http://localhost:8080/source/s?defs=of_match_table&project=kernel-4.19) = [of\_match\_ptr](http://localhost:8080/source/s?defs=of_match_ptr&project=kernel-4.19)(**[spidev\_dt\_ids](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_dt_ids)**),

**},**

.[probe](http://localhost:8080/source/s?defs=probe&project=kernel-4.19) = **[spidev\_probe](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_probe)**,

.[remove](http://localhost:8080/source/s?defs=remove&project=kernel-4.19) = **[spidev\_remove](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_remove)**,

**};**

**static** **int** [\_\_init](http://localhost:8080/source/s?defs=__init&project=kernel-4.19) **[spidev\_init](http://localhost:8080/source/s?refs=spidev_init&project=kernel-4.19)**(**void**)

**{**

**...**

[status](http://localhost:8080/source/s?defs=status&project=kernel-4.19) = [spi\_register\_driver](http://localhost:8080/source/s?defs=spi_register_driver&project=kernel-4.19)(&**[spidev\_spi\_driver](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_spi_driver)**);

**...**

**return status;**

**}**

**static** **void** [\_\_exit](http://localhost:8080/source/s?defs=__exit&project=kernel-4.19) **[spidev\_exit](http://localhost:8080/source/s?refs=spidev_exit&project=kernel-4.19)**(**void**)

**{**

[spi\_unregister\_driver](http://localhost:8080/source/s?defs=spi_unregister_driver&project=kernel-4.19)(&**[spidev\_spi\_driver](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_spi_driver)**)**;**

**...**

**}**

[module\_init](http://localhost:8080/source/s?defs=module_init&project=kernel-4.19)(**[spidev\_init](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_init)**);

[module\_exit](http://localhost:8080/source/s?defs=module_exit&project=kernel-4.19)(**[spidev\_exit](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c?r=&mo=21747&fi=851" \l "spidev_exit)**);

**上面是目前基于“总线设备驱动模型”的通用写法。将设配的的差异化部分提取成配置文件dts。共通部分的逻辑抽取成了平台化代码，开发人员无需赘写。**

**Kernel在开机过程中将解析dts（dtbo）文件并组装成内存对象，kenel又提供了完备的以of\_xxx为前缀的方法来读取这些dts对象。当dts中配置的compatible字段和上面struct spi\_driver结构体中定义的字符匹配时会自动调用在.probe中被赋值的方法。**

**上述是基于“总线设备驱动模型”当下绝大多数驱动的写作模板，不同的设备在各自的xxx\_init方法中调用xxx\_register\_driver(xxx\_driver)进行驱动注册，最终进入probe方法进行各自和硬件特性相关的操作。**

**那么probe方法是如何调到的？**

**xref/kernel-4.19/drivers/spi/spidev.c#844**

**[844](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c" \l "844)** [status](http://localhost:8080/source/s?defs=status&project=kernel-4.19) = [spi\_register\_driver](http://localhost:8080/source/s?defs=spi_register_driver&project=kernel-4.19)(&**[spidev\_spi\_driver](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spidev.c" \l "spidev_spi_driver)**);

xref/kernel-4.19/include/linux/spi/spi.h#281

**[281](http://localhost:8080/source/xref/kernel-4.19/include/linux/spi/spi.h?fi=spi_register_driver" \l "281)** #**define** **[spi\_register\_driver](http://localhost:8080/source/s?refs=spi_register_driver&project=kernel-4.19)**(**[driver](http://localhost:8080/source/s?refs=driver&project=kernel-4.19)**) [\_\_spi\_register\_driver](http://localhost:8080/source/s?defs=__spi_register_driver&project=kernel-4.19)([THIS\_MODULE](http://localhost:8080/source/s?defs=THIS_MODULE&project=kernel-4.19), [driver](http://localhost:8080/source/s?defs=driver&project=kernel-4.19))

xref/kernel-4.19/drivers/spi/spi.c

**[401](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=__spi_register_driver" \l "401)**  **int** **[\_\_spi\_register\_driver](http://localhost:8080/source/s?refs=__spi_register_driver&project=kernel-4.19)**(**struct** [module](http://localhost:8080/source/s?defs=module&project=kernel-4.19) \***[owner](http://localhost:8080/source/s?refs=owner&project=kernel-4.19)**, **struct** [spi\_driver](http://localhost:8080/source/s?defs=spi_driver&project=kernel-4.19) \***[sdrv](http://localhost:8080/source/s?refs=sdrv&project=kernel-4.19)**)

**[405](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=__spi_register_driver" \l "405)** [sdrv](http://localhost:8080/source/s?defs=sdrv&project=kernel-4.19)->[driver](http://localhost:8080/source/s?defs=driver&project=kernel-4.19).[probe](http://localhost:8080/source/s?defs=probe&project=kernel-4.19) = **[spi\_drv\_probe](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=__spi_register_driver" \l "spi_drv_probe)**;

**[409](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=__spi_register_driver" \l "409)** **return** [driver\_register](http://localhost:8080/source/s?defs=driver_register&project=kernel-4.19)(&[sdrv](http://localhost:8080/source/s?defs=sdrv&project=kernel-4.19)->[driver](http://localhost:8080/source/s?defs=driver&project=kernel-4.19));

**[146](http://localhost:8080/source/xref/kernel-4.19/drivers/base/driver.c?fi=driver_register" \l "146)  int** **[driver\_register](http://localhost:8080/source/s?refs=driver_register&project=kernel-4.19)**(**struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**)

**[170](http://localhost:8080/source/xref/kernel-4.19/drivers/base/driver.c?fi=driver_register" \l "170)** **[ret](http://localhost:8080/source/xref/kernel-4.19/drivers/base/driver.c?fi=driver_register" \l "ret)** = [bus\_add\_driver](http://localhost:8080/source/s?defs=bus_add_driver&project=kernel-4.19)([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19));

xref/kernel-4.19/drivers/base/bus.c

**[639](http://localhost:8080/source/xref/kernel-4.19/drivers/base/bus.c?fi=bus_add_driver" \l "639)  int** **[bus\_add\_driver](http://localhost:8080/source/s?refs=bus_add_driver&project=kernel-4.19)**(**struct** **[device\_driver](http://localhost:8080/source/xref/kernel-4.19/drivers/base/bus.c?fi=bus_add_driver" \l "device_driver)** \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**)

**[672](http://localhost:8080/source/xref/kernel-4.19/drivers/base/bus.c?fi=bus_add_driver" \l "672)** [error](http://localhost:8080/source/s?defs=error&project=kernel-4.19) = [driver\_attach](http://localhost:8080/source/s?defs=driver_attach&project=kernel-4.19)([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19));

xref/kernel-4.19/drivers/base/dd.c

**[920](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "920)  int** **[driver\_attach](http://localhost:8080/source/s?refs=driver_attach&project=kernel-4.19)**(**struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**)

**[922](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "922)**   **return** [bus\_for\_each\_dev](http://localhost:8080/source/s?defs=bus_for_each_dev&project=kernel-4.19)([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19)->[bus](http://localhost:8080/source/s?defs=bus&project=kernel-4.19), [NULL](http://localhost:8080/source/s?defs=NULL&project=kernel-4.19), [drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19), **[\_\_driver\_attach](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "__driver_attach)**);

**[2](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "872)**   **static** **int** **[\_\_driver\_attach](http://localhost:8080/source/s?refs=__driver_attach&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**, **void** \***[data](http://localhost:8080/source/s?refs=data&project=kernel-4.19)**)

**[887](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "887)** [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [driver\_match\_device](http://localhost:8080/source/s?defs=driver_match_device&project=kernel-4.19)([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19), [dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19)); // 匹配方法： .[match](http://localhost:8080/source/s?defs=match&project=kernel-4.19) = **[spi\_match\_device](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c" \l "spi_match_device)**

**[903](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "903)** **[driver\_probe\_device](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "driver_probe_device)**([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19), [dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

**[649](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "649)  int** **[driver\_probe\_device](http://localhost:8080/source/s?refs=driver_probe_device&project=kernel-4.19)**(**struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**, **struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

**[667](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "667)** [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = **[really\_probe](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "really_probe)**([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19), [drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19));

**[448](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "448)  static** **int** **[really\_probe](http://localhost:8080/source/s?refs=really_probe&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**, **struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**)

**[510](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "510)** [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19)->[probe](http://localhost:8080/source/s?defs=probe&project=kernel-4.19)([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19)); // [probe](http://localhost:8080/source/s?defs=probe&project=kernel-4.19) = s**pi\_drv\_probe**

xref/kernel-4.19/drivers/spi/spi.c#343

**[43](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=__spi_register_driver" \l "343)**  **static** **int** **[spi\_drv\_probe](http://localhost:8080/source/s?refs=spi_drv_probe&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

**[366](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=__spi_register_driver" \l "366)** [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [sdrv](http://localhost:8080/source/s?defs=sdrv&project=kernel-4.19)->[probe](http://localhost:8080/source/s?defs=probe&project=kernel-4.19)([spi](http://localhost:8080/source/s?defs=spi&project=kernel-4.19)); // **.probe = spidev\_probe //**spi.c#**405**

至此就调用到了驱动中自定义的xxx**\_probe方法。**

**关于probe方法的调用，其过程有两个方向：上面从驱动查找设备的方向，下面是从设备查找驱动的方向：**

**[spi\_new\_device](http://localhost:8080/source/s?refs=spi_new_device&project=kernel-4.19)**

xref/kernel-4.19/drivers/spi/spi.c#596

[596](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=spi_new_device" \l "596)  **struct** [spi\_device](http://localhost:8080/source/s?defs=spi_device&project=kernel-4.19) \***[spi\_new\_device](http://localhost:8080/source/s?refs=spi_new_device&project=kernel-4.19)**(**struct** [spi\_controller](http://localhost:8080/source/s?defs=spi_controller&project=kernel-4.19) \***[ctlr](http://localhost:8080/source/s?refs=ctlr&project=kernel-4.19)**, **struct** [spi\_board\_info](http://localhost:8080/source/s?defs=spi_board_info&project=kernel-4.19) \***[chip](http://localhost:8080/source/s?refs=chip&project=kernel-4.19)**)

[634](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=spi_new_device" \l "634)   [status](http://localhost:8080/source/s?defs=status&project=kernel-4.19) = **[spi\_add\_device](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=spi_new_device" \l "spi_add_device)**(**[proxy](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=spi_new_device" \l "proxy)**);

[518](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=spi_new_device" \l "518)  **int** **[spi\_add\_device](http://localhost:8080/source/s?refs=spi_add_device&project=kernel-4.19)**(**struct** [spi\_device](http://localhost:8080/source/s?defs=spi_device&project=kernel-4.19) \***[spi](http://localhost:8080/source/s?refs=spi&project=kernel-4.19)**)

[569](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c?fi=spi_new_device" \l "569)   [status](http://localhost:8080/source/s?defs=status&project=kernel-4.19) = [device\_add](http://localhost:8080/source/s?defs=device_add&project=kernel-4.19)(&[spi](http://localhost:8080/source/s?defs=spi&project=kernel-4.19)->[dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

......

**[platform\_device\_register](http://localhost:8080/source/s?refs=platform_device_register&project=kernel-4.19)**

xref/kernel-4.19/drivers/base/platform.c#475

[475](http://localhost:8080/source/xref/kernel-4.19/drivers/base/platform.c?fi=platform_device_register" \l "475)  **int** **[platform\_device\_register](http://localhost:8080/source/s?refs=platform_device_register&project=kernel-4.19)**(**struct** [platform\_device](http://localhost:8080/source/s?defs=platform_device&project=kernel-4.19) \***[pdev](http://localhost:8080/source/s?refs=pdev&project=kernel-4.19)**)

[479](http://localhost:8080/source/xref/kernel-4.19/drivers/base/platform.c?fi=platform_device_register" \l "479)   **return** **[platform\_device\_add](http://localhost:8080/source/xref/kernel-4.19/drivers/base/platform.c?fi=platform_device_register" \l "platform_device_add)**([pdev](http://localhost:8080/source/s?defs=pdev&project=kernel-4.19));

[361](http://localhost:8080/source/xref/kernel-4.19/drivers/base/platform.c?fi=platform_device_register" \l "361)  **int** **[platform\_device\_add](http://localhost:8080/source/s?refs=platform_device_add&project=kernel-4.19)**(**struct** [platform\_device](http://localhost:8080/source/s?defs=platform_device&project=kernel-4.19) \***[pdev](http://localhost:8080/source/s?refs=pdev&project=kernel-4.19)**)

[420](http://localhost:8080/source/xref/kernel-4.19/drivers/base/platform.c?fi=platform_device_register" \l "420)   [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [device\_add](http://localhost:8080/source/s?defs=device_add&project=kernel-4.19)(&[pdev](http://localhost:8080/source/s?defs=pdev&project=kernel-4.19)->[dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

......

**[input\_register\_device](http://localhost:8080/source/s?refs=input_register_device&project=kernel-4.19)**

xref/kernel-4.19/drivers/input/input.c#2092

[2092](http://localhost:8080/source/xref/kernel-4.19/drivers/input/input.c?fi=input_register_device" \l "2092)  **int** **[input\_register\_device](http://localhost:8080/source/s?refs=input_register_device&project=kernel-4.19)**(**struct** [input\_dev](http://localhost:8080/source/s?defs=input_dev&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

[2148](http://localhost:8080/source/xref/kernel-4.19/drivers/input/input.c?fi=input_register_device" \l "2148)   [error](http://localhost:8080/source/s?defs=error&project=kernel-4.19) = [device\_add](http://localhost:8080/source/s?defs=device_add&project=kernel-4.19)(&[dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19)->[dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

......

xref/kernel-4.19/drivers/base/core.c#2068

[2068](http://localhost:8080/source/xref/kernel-4.19/drivers/base/core.c?fi=device_add" \l "2068)  **int** **[device\_add](http://localhost:8080/source/s?refs=device_add&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

[2170](http://localhost:8080/source/xref/kernel-4.19/drivers/base/core.c?fi=device_add" \l "2170)   [bus\_probe\_device](http://localhost:8080/source/s?defs=bus_probe_device&project=kernel-4.19)([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

xref/kernel-4.19/drivers/base/bus.c#515

[515](http://localhost:8080/source/xref/kernel-4.19/drivers/base/bus.c?fi=bus_probe_device" \l "515)  **void** **[bus\_probe\_device](http://localhost:8080/source/s?refs=bus_probe_device&project=kernel-4.19)**(**struct** **[device](http://localhost:8080/source/xref/kernel-4.19/drivers/base/bus.c?fi=bus_probe_device" \l "device)** \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

[524](http://localhost:8080/source/xref/kernel-4.19/drivers/base/bus.c?fi=bus_probe_device" \l "524)   [device\_initial\_probe](http://localhost:8080/source/s?defs=device_initial_probe&project=kernel-4.19)([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

xref/kernel-4.19/drivers/base/dd.c#867

[867](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "867)  **void** **[device\_initial\_probe](http://localhost:8080/source/s?refs=device_initial_probe&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

[869](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "869)   **[\_\_device\_attach](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "__device_attach)**([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19), **true**);

[793](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "793)  **static** **int** **[\_\_device\_attach](http://localhost:8080/source/s?refs=__device_attach&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**, **bool** **[allow\_async](http://localhost:8080/source/s?refs=allow_async&project=kernel-4.19)**)

[822](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "822)   [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [bus\_for\_each\_drv](http://localhost:8080/source/s?defs=bus_for_each_drv&project=kernel-4.19)([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19)->[bus](http://localhost:8080/source/s?defs=bus&project=kernel-4.19), [NULL](http://localhost:8080/source/s?defs=NULL&project=kernel-4.19), &[data](http://localhost:8080/source/s?defs=data&project=kernel-4.19), **[\_\_device\_attach\_driver](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "__device_attach_driver)**);

[727](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "727)  **static** **int** **[\_\_device\_attach\_driver](http://localhost:8080/source/s?refs=__device_attach_driver&project=kernel-4.19)**(**struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**, **void** \***[\_data](http://localhost:8080/source/s?refs=_data&project=kernel-4.19)**)

[734](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "734)   [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [driver\_match\_device](http://localhost:8080/source/s?defs=driver_match_device&project=kernel-4.19)([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19), [dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

[754](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "754)   **return** **[driver\_probe\_device](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=device_initial_probe" \l "driver_probe_device)**([drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19), [dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

**[649](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "649)  int** **[driver\_probe\_device](http://localhost:8080/source/s?refs=driver_probe_device&project=kernel-4.19)**(**struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**, **struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**)

**[667](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "667)** [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = **[really\_probe](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "really_probe)**([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19), [drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19));

**[448](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "448)  static** **int** **[really\_probe](http://localhost:8080/source/s?refs=really_probe&project=kernel-4.19)**(**struct** [device](http://localhost:8080/source/s?defs=device&project=kernel-4.19) \***[dev](http://localhost:8080/source/s?refs=dev&project=kernel-4.19)**, **struct** [device\_driver](http://localhost:8080/source/s?defs=device_driver&project=kernel-4.19) \***[drv](http://localhost:8080/source/s?refs=drv&project=kernel-4.19)**)

**[510](http://localhost:8080/source/xref/kernel-4.19/drivers/base/dd.c?fi=driver_attach" \l "510)** [ret](http://localhost:8080/source/s?defs=ret&project=kernel-4.19) = [drv](http://localhost:8080/source/s?defs=drv&project=kernel-4.19)->[probe](http://localhost:8080/source/s?defs=probe&project=kernel-4.19)([dev](http://localhost:8080/source/s?defs=dev&project=kernel-4.19));

无论是从驱动找设备还是从设备找驱动，总会调用相应总线的match函数来匹配driver和device,如.[match](http://localhost:8080/source/s?defs=match&project=kernel-4.19) = [xxx\_match\_device](http://localhost:8080/source/xref/kernel-4.19/drivers/spi/spi.c" \l "spi_match_device)。具体可参考每个总线的match函数，但大致的匹配过程是一样的，以platform\_driver为例如下：

① 先用dts中的 compatible 属性和platform\_driver中的driver.of\_match\_table 来匹配

② 再用 platform\_driver.id\_table 中的 name 和 platform\_device.name来匹配

③ 最后用platform\_device.name和 platform\_driver 中的 driver.name来匹配

大多数情况下我们会在dts中配置外接设备的 compatible 属性，并在自己模块的驱动中填写driver.of\_match\_table，所以通常第①步就会匹配成功。

接下来就是probe方法对硬件的各种操作，比如芯片的上下电等。

附录：

|  |  |  |
| --- | --- | --- |
| 初始化等级 | 对应的初始化设备或驱动的函数 | 备注 |
| early\_initcall | migration\_init和spawn\_ksoftirqd等 | 主要是register\_cpu\_notifier |
| pure\_initcall | init\_cpufreq\_transition\_notifier\_list |  |
| core\_initcall | netlink\_proto\_init、cpuidle\_init、xxx\_gpio\_init、filelock\_init、pm\_init、sock\_init、wakelocks\_init等 | 主要是一些关键部分的初始化，像gpio、通信、电源管理等部分 |
| core\_initcall\_sync |  |  |
| postcore\_initcall | backlight\_class\_init、dma\_sysclass\_init、i2c\_init、kobject\_uevent\_init、pci\_driver\_init、spi\_init、tty\_class\_init等 | 主要是一些总线的节点的创建和链表初始化等（总线驱动的加载在后面） |
| postcore\_initcall\_sync |  |  |
| arch\_initcall | xxx\_init\_device、xxx\_devices\_setup、customize\_machine、platform\_init等 | 主要是板级设备的加载（i2c、spi、usb、串口等以及一些外设的加载） |
| arch\_initcall\_sync |  |  |
| subsys\_initcall | blk\_ioc\_ini、xxx\_dma\_init、xxx\_i2c\_init\_driver、usb\_init、xxx\_spi\_init等 | 块设备驱动、以及主要的bus总线驱动 |
| subsys\_initcall\_sync |  |  |
| fs\_initcall | inet\_init、alignment\_init、chr\_dev\_init、tracer\_alloc\_buffers等 |  |
| fs\_initcall\_sync |  |  |
| rootfs\_initcall | populate\_rootfs、default\_rootfs | Rootfs先关初始化 |
| device\_initcall | 一般的外设驱动的加载函数 | module\_init = device\_initcall，外设驱动 |
| device\_initcall\_sync |  |  |
| late\_initcall | late\_resume\_init 等 |  |
| late\_initcall\_sync |  |  |