kgdb初始化

```c
/**
 *  kgdb_arch_init - Perform any architecture specific initalization.
 *
 *  This function will handle the initalization of any architecture
 *  specific callbacks.
 */
int kgdb_arch_init(void)
{
    int ret = register_die_notifier(&kgdb_notifier);

    if (ret != 0)
        return ret;

    register_undef_hook(&kgdb_brkpt_hook);
    register_undef_hook(&kgdb_compiled_brkpt_hook);

    return 0;
}
```

即register了 2 个"undefined instruction" hooks,如下

```c
static struct undef_hook kgdb_brkpt_hook = {
    .instr_mask      = 0xffffffff,
    .instr_val       = KGDB_BREAKINST,
    .cpsr_mask       = MODE_MASK,
    .cpsr_val        = SVC_MODE,
    .fn          = kgdb_brk_fn
};

static struct undef_hook kgdb_compiled_brkpt_hook = {
    .instr_mask      = 0xffffffff,
    .instr_val       = KGDB_COMPILED_BREAK,
    .cpsr_mask       = MODE_MASK,
    .cpsr_val        = SVC_MODE,
    .fn          = kgdb_compiled_brk_fn
};
```

in arch/arm/kernel/traps.c

```c
asmlinkage void __exception do_undefinstr(struct pt_regs *regs)
{
    unsigned int instr;
    siginfo_t info;
    void __user *pc;

    pc = (void __user *)instruction_pointer(regs);

    if (processor_mode(regs) == SVC_MODE) {
#ifdef CONFIG_THUMB2_KERNEL
        if (thumb_mode(regs)) {
            instr = __mem_to_opcode_thumb16(((u16 *)pc)[0]);
            if (is_wide_instruction(instr)) {
                u16 inst2;
                inst2 = __mem_to_opcode_thumb16(((u16 *)pc)[1]);
                instr = __opcode_thumb32_compose(instr, inst2);
            }
        } else
#endif
            instr = __mem_to_opcode_arm(*(u32 *) pc);       ①
    } else if (thumb_mode(regs)) {
        if (get_user(instr, (u16 __user *)pc))
            goto die_sig;
        instr = __mem_to_opcode_thumb16(instr);
        if (is_wide_instruction(instr)) {
            unsigned int instr2;
            if (get_user(instr2, (u16 __user *)pc+1))
                goto die_sig;
            instr2 = __mem_to_opcode_thumb16(instr2);
            instr = __opcode_thumb32_compose(instr, instr2);
        }
    } else {
        if (get_user(instr, (u32 __user *)pc))
            goto die_sig;
        instr = __mem_to_opcode_arm(instr);
    }

    if (call_undef_hook(regs, instr) == 0)                  ②
        return;

die_sig:
#ifdef CONFIG_DEBUG_USER
    if (user_debug & UDBG_UNDEFINED) {
        printk(KERN_INFO "%s (%d): undefined instruction: pc=%p\n",
            current->comm, task_pid_nr(current), pc);
        __show_regs(regs);
        dump_instr(KERN_INFO, regs);
    }
#endif

    info.si_signo = SIGILL;
    info.si_errno = 0;
    info.si_code  = ILL_ILLOPC;
    info.si_addr  = pc;

    arm_notify_die("Oops - undefined instruction", regs, &info, 0, 6);
}
```

do_undefinstr()是处理“undefined instruction” trap的handler。

①
获得引起该exception的instruction

②
调用“ubdefined instruction” hook

```
static int call_undef_hook(struct pt_regs *regs, unsigned int instr)
{
    struct undef_hook *hook;
    unsigned long flags;
    int (*fn)(struct pt_regs *regs, unsigned int instr) = NULL;

    raw_spin_lock_irqsave(&undef_lock, flags);
    list_for_each_entry(hook, &undef_hook, node)
        if ((instr & hook->instr_mask) == hook->instr_val &&         ③
            (regs->ARM_cpsr & hook->cpsr_mask) == hook->cpsr_val)    ④
            fn = hook->fn;                                           ⑤
    raw_spin_unlock_irqrestore(&undef_lock, flags);

    return fn ? fn(regs, instr) : 1;
}
```

③
判断引起exception的undefined instruction是否时kgdb定义的

④
发生exception时是在SVC_MODE吗

⑤
进入kgdb breakpoint handler

```c
static int kgdb_brk_fn(struct pt_regs *regs, unsigned int instr)
{
    kgdb_handle_exception(1, SIGTRAP, 0, regs);

    return 0;
}

static int kgdb_compiled_brk_fn(struct pt_regs *regs, unsigned int instr)
{
    compiled_break = 1;
    kgdb_handle_exception(1, SIGTRAP, 0, regs);

    return 0;
}
```

kgdb_handle_exception()是kernel/debug/debug_core.c的核心函数。