

```

static void noinline __init_refok rest_init(void)
{
    __releases(kernel_lock)

    int pid;

    kernel_thread(kernel_init, NULL, CLONE_FS | CLONE_SIGHAND);
    numa_default_policy();
    pid = kernel_thread(kthreadd, NULL, CLONE_FS | CLONE_FILES);
    kthreadd_task = find_task_by_pid(pid);
    unlock_kernel();

    /*
     * The boot idle thread must execute schedule()
     * at least once to get things moving:
     */
    init_idle_bootup_task(current);
    preempt_enable_no_resched();
    schedule();
    preempt_disable();

    /* Call into cpu_idle with preempt disabled */
    cpu_idle();
}

```

rest\_init() : start\_kernel() 함수는 rest\_init() 을 호출한다.

위에 함수중 cpu\_idle() 을 선언함으로서 rest\_init() 함수는 무한루프에서 대기하도록 설정되어 있다. 이렇게 대기하게 됨으로서 이 함수가 존재하던 메모리 공간을 해제하고 재활용 됨을 알 수 있다. ↓

즉 start\_kernel() 에서 rest\_init() 을 호출해서 훨씬 작은 공간을 차지하는 프로세스가 되는 것이다.

✓ idle 상태로 대기.

즉 start\_kernel()에서는 모든 init 즉 초기화를 시작하게 된다. 메모리, 인터럽트 스케줄, 인터럽트 벡터 테이블 초기화 함수를 호출하게 되는 것이다.

```

void cpu_idle(void)
{
    int cpu = smp_processor_id();

    current_thread_info()->status |= TS_POLLING;

    /* endless idle loop with no priority at all */
    while (1) {
        tick_nohz_stop_sched_tick();
        while (!need_resched()) {
            void (*idle)(void);

            check_pgt_cache();
            rmb();
            idle = pm_idle;

            if (rcu_pending(cpu))
                rcu_check_callbacks(cpu, 0);

            if (!idle)
                idle = default_idle;

            if (cpu_is_offline(cpu))
                play_dead();

            __get_cpu_var(irq_stat).idle_timestamp = jiffies;
            idle();
        }
        tick_nohz_restart_sched_tick();
        preempt_enable_no_resched();
        schedule();
        preempt_disable();
    }
}

```

그림 16 cpu\_idle 에 무한루프 함수 확인

↓

우선 CPU의 위치를 확인하기 위해 start\_kernel (그림14)의 함수를 확인해 보았다. ↓

Start\_kernel(그림14) 함수 안에 마지막 함수는 rest\_init(그림 15) 함수이다. Rest\_init(그림 15) 함수의 마지막 함수는 CPU\_idle(그림 16) 이다. ↓

CPU\_idle(그림 16) 함수는 while(1) 로 이루어져 있다. ↓

↓

Rest\_init의 함수는 cpu\_idle0 을 선언함으로서 무한루프에서 kernel 을 대기하도록 설정할 수 있다. 이렇게 대기하게 하는 이유는 kernel 이 인터럽트가 걸리는 것에 대해 대비할 수 있도록 해준다. 따라 CPU 는 while(1) 부분에서 인터럽트를 기다린다고 볼 수 있다. ↓

↑ 핵심!!