Snapshot Boot - fast power up/down with PM function

Machida, Hiroyuki
Sony Corp.

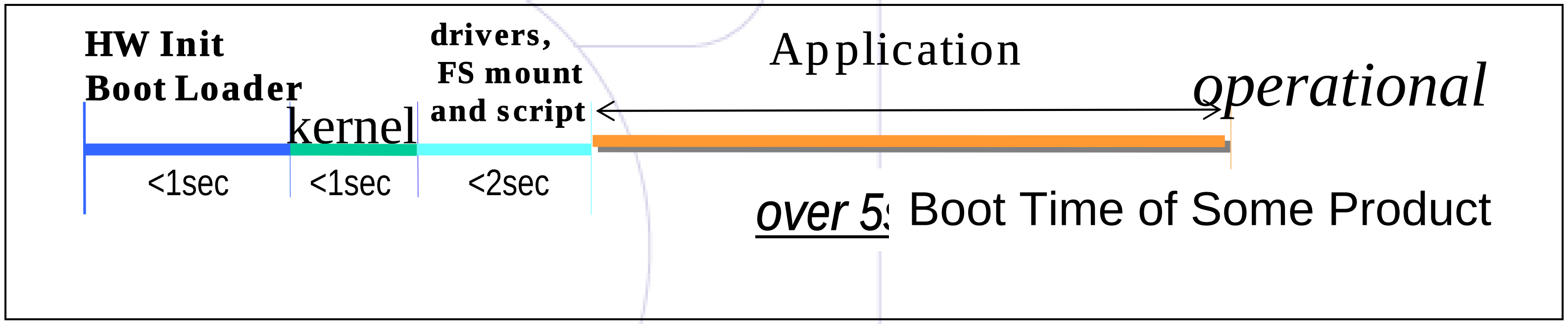
machida – AT – sm.sony.co.jp

Introduction

- Kernel 2.6 has now PM framework, which can be used to boost reduction of boot up time.
- In this presentation, we'll show our preliminary work for fast power up/down methods with PM function.
- And also we'll show how to enable hibernation on your platform to invite CE developers to this area.

CE Linux Forum

Why Snapshot boot ?



- Application init time includes;
  - Before reaching main()
    - Loading Application Image to RAM (mmap & page fault)
    - Dynamic Linking
    - Global constructors
  - IPCs between applications to activates entire system
- Need to address Application init time
- We'll also take care about followings, after Application init time optimization done
  - HW should be designed to achieve quick init.
  - 100msec-order optimizations are needed for non-application area

PM functions for Fast Power Up and Down

- Utilize Kernel 2.6 power management function for fast power up/down
  - Suspend/Resume
  - Hibernation

- Suspend/Resume can reduce;
  - HW Init
  - Boot Loader init/loading kernel
  - Most part of kernel init
  - Driver Loading
  - Application init time, before reaching main()
  - (Can't reduce I/O init)

- Hibernation can reduce;
  - Some part of kernel init
  - Application init time, before reaching main()
  - (Can't reduce I/O init nor loading memory)

Suspend/Resume for Power Down/Up

- Utilize Suspend/Resume for Power Down/Up
- Additional features to conventional suspend/resume
  - DEFERRED_RESUME
    - Support deferred I/O init on resume
  - SAFE_SUSPEND
    - RO remount on suspend and restore to RW on resume

- Pros
  - Most loading and init can be skipped
- Cons
  - Power consumption on suspend state

Snapshot Boot - 1

- Utilize un-hibernation for Power Up
  - For Power down, don't save system image, just use conventional real power down, instead of hibernate.
- Additional features to conventional hibernation
  - PRESERVE_SWSUSP_IMAGE
    - Preserve system image on un-hibernation
  - HIBERNATE_ON_FLASH
    - Use flash rom to store system image
  - DEFERRED_RESUME
    - Support deferred I/O init on resume (un-hibernation)
  - FAST_CLEAN_SHUTDOWN
    - Fast safe shutdown for Quick Power Down

Snapshot Boot - 2

- Kernel 2.6 hibernation can choice following three methods

  #1 by BIOS (boot loader) - PM_DISK_FIRMWARE

  #2 by Kernel - PM_DISK_SHUTDOWN/REBOOT

  #3 by BIOS and kernel - PM_DISK_PLATFORM

- Pros
  - Low power consumption
- Cons
  - The #1 method is difficult to work with deferred I/O resume
  - The #2 method can't skip kernel init

Current Status - 1

- Suspend/Resume
  - Implemented/patch is available
    - Platform Supporting patch (PPC440 EBONY, ARM9 OSK)
    - DEFERRED_RESUME
    - SAFE_SUSPEND
  - Need to be done/investigate for
    - Where suspend initiation code to be stored
      - DRAM in self refresh mode cannot be accessed
    - Need to unmount/mount removable media
    - Need more drivers to support suspend/resume methods

Current Status - 2

- SnapShot boot
  - Patch is available
    - FAST_CLEAN_SHUTDOWN
  - Implemented
    - Platform Support with NOR flash (PPC440 EBONY, ARM9 OSK)
    - PRESERVE_SWSUSP_IMAGE
  - Now working on
    - DEFERRED_RESUME on un-hibernation

Current Status - 3

- Issues we met…
  - SWSUSP is not fast, because it takes following steps
    - kernel initializes almost all I/O
    - Freeze processes
    - Load hibernation image
    - Shutdown almost I/O
    - Copy hibernation Image
    - Resume I/O and processes
  - mtdblock doesn’t provide good performance
  - most platform doesn’t enable PM support for device drivers including MTD

- Need to be done/investigate for SnapShot boot
  - integrate #3 and DEFERRED_RESUME
  - investigate mtdblock problem
  - investigate to reduce hibernation image
  - investigate SwSusp2 features, like compression of hibernation image
  - DMA usage on un-hibernation (2.6.11 has bug?)
  - need rw mount on every snapshot boot

Encourage PM function development

- 2.6 kernel already have PM framework, you can easily to support on your platform!
- Device Driver support
  - Mr. Nigel has shown how to support suspend/resume method on your device driver at SanJose CELF Tech. meeting Jan/2005.
- Adding Suspend/Resume platform support
- Adding Hibernation platform support
  - http://tree.celinuxforum.org/CelfPubWiki/SwSuspendPortingNotes

How to enable suspend/resume - 1

```
+static int ebony_pm_prepare(suspend_state_t state)
+{
+
+       if (state != PM_SUSPEND_MEM)
+               return -EINVAL;
+       return 0;
+}
+
+static int ebony_pm_finish(suspend_state_t state)
+{
+       return 0;
+}
+
+static struct pm_ops ebony_pm_ops = {
+       .pm_disk_mode   = PM_DISK_FIRMWARE,
+       .prepare        = ebony_pm_prepare,
+       .enter          = ebony_pm_enter,
+       .finish         = ebony_pm_finish,
+};
+
+static int __init ebony_pm_init(void)
+{
+       pm_set_ops(&ebony_pm_ops);
+       return 0;
+}
```

- Prepare 3 methods for PM_OPS
  - prepare() and finish() are almost empty.
  - enter() is a main function to be prepared

- Register PM_OPS using init function

How to enable suspend/resume - 2

```
+static int ebony_pm_enter(suspend_state_t state)
+{
+               :
+       if (state != PM_SUSPEND_MEM)
+               return -EINVAL;
+
+       /*  Save MSR and Stop all interrupts */
+       save_msr = mfmsr();
+       _nmask_and_or_msr((MSR_CE|MSR_EE), 0);
+       /* save current CPM */
+       cpm_save_er = mfdcr(DCRN_CPC0_ER);
+       /* save UIC0 enable registers */
+       uic_save_er = mfdcr(DCRN_UIC_ER(UIC0));
                :
+       /* mask UIC0 interrupts, except External Intr #5 */
+       mtdcr(DCRN_UIC_ER(UIC0), UIC0_EIR5_BIT);
                               :
+       /* set up CPM */
+       cpm_er = IBM_CPM_ALL & ~IBM_CPM_UIC0;
+       /* we need this to work with printk on serial console */
+       serial8250_suspend_port_busy(0);
+
+       /* Enable interrupts and Enter SLEEP mode */
+       ibm440gp_sleep(cpm_er, (MSR_EE|MSR_WE));
+       /*  Stop all interrupts, again */
+       _nmask_and_or_msr((MSR_CE|MSR_EE), 0);
+       /* Restore CPM, before resume serials for printk() */
+       mtdcr(DCRN_CPC0_ER, cpm_save_er);
+       /* we need this to work with printk on serial console */
+       serial8250_resume_port_busy(0);
++
+       /* Restore UIC0 enable registers */
+       mtdcr(DCRN_UIC_ER(UIC0), uic_save_er);
+
+       /* Restore MSR */
+       mtmsr(save_msr);
+
+       return 0;
+}
```

- enter() body of suspend/resume
  - save some registers
  - mask interrupts
  - goto sleep mode, with waiting resume event.
  - (update jiffies if you can do)
  - restore stuff

How to enable hibernation

- http://tree.celinuxforum.org/CelfPubWiki/SwSuspendPortingNotes
  - You need to write just two new files to enable hibernation
  - Please try and enjoy it!