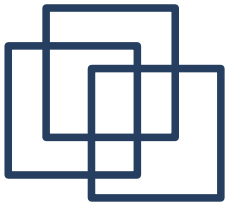
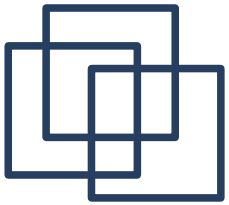


Android 2.3 での変化点

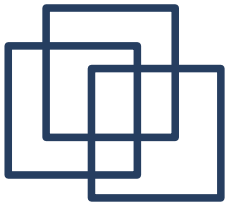
- BeagleBoard や Armadillo に Android 2.3 をのせてみた時に気が付いた点や、
  そこから脱線して調べたこと等。



init.rc の違い1

- on xxx
  - fs
  - early-fs
  - post-fs

  fs に関する項目が増えた

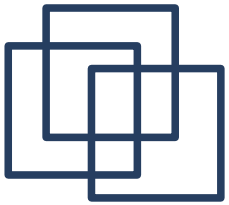


on fs

```
on fs
# mount mtd partitions
    # Mount /system rw first to give the filesystem a chance to save a checkpoint
    mount yaffs2 mtd@system /system
    mount yaffs2 mtd@system /system ro remount
    mount yaffs2 mtd@userdata /data nosuid nodev
    mount yaffs2 mtd@cache /cache nosuid nodev
```

- system、data、cache のマウントを行う

  Android 2.2 までは、on init に書いていた

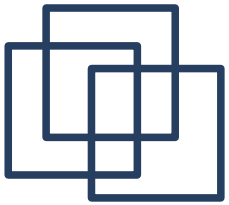


on fs

```
on fs
# mount mtd partitions
    # Mount /system rw first to give the filesystem a chance to save a checkpoint
#####    mount yaffs2 mtd@system /system
#####    mount yaffs2 mtd@system /system ro remount
#####    mount yaffs2 mtd@userdata /data nosuid nodev
#####    mount yaffs2 mtd@cache /cache nosuid nodev
```

- 上記のようにコメントアウトすると、
  イカのようなメッセージが出て落ちます

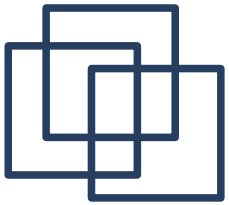
Kernel panic - not syncing: Attempted to kill init!



on fs

```
#####on fs
# mount mtd partitions
    # Mount /system rw first to give the filesystem a chance to save a checkpoint
#####    mount yaffs2 mtd@system /system
#####    mount yaffs2 mtd@system /system ro remount
#####    mount yaffs2 mtd@userdata /data nosuid nodev
#####    mount yaffs2 mtd@cache /cache nosuid nodev
```

- on fs を書いているにも関わらず、
  そのブロックに何も書いていないのが原因。
  なので、on fs ごと消す。



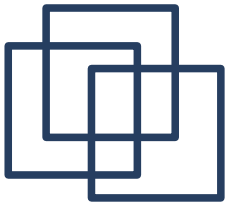
on fs

```
on fs
# mount mtd partitions
    # Mount /system rw first to give the filesystem a chance to save a checkpoint
    mount yaffs2 mtd@system /system
    mount yaffs2 mtd@system /system ro remount
    mount yaffs2 mtd@userdata /data nosuid nodev
    mount yaffs2 mtd@cache /cache nosuid nodev
```

- 何もコメントアウトしなくても、起動はする。

  マウントに失敗するので、

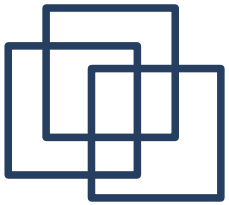
  結果的にコメントアウトしてるのと変わらない結果になる。
- on fs で system 等をマウントして起動する方法は後で。



コマンドの実行箇所

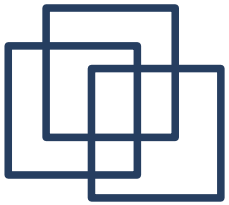
system/core/init/init.c

```
490 void execute_one_command(void)
491 {
492     int ret;
493
494     if (!cur_action || !cur_command || is_last_command(cur_action, cur_command)) {
495         cur_action = action_remove_queue_head();
496         cur_command = NULL;
497         if (!cur_action)
498             return;
499         INFO("processing action %p (%s)¥n", cur_action, cur_action->name);
500         cur_command = get_first_command(cur_action);
501     } else {
502         cur_command = get_next_command(cur_action, cur_command);
503     }
504
505     if (!cur_command)
506         return;
507
508     ret = cur_command->func(cur_command->nargs, cur_command->args);
509     INFO("command '%s' r=%d¥n", cur_command->args[0], ret);
510 }
```



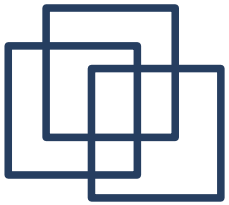
on early-fs

- on fs よりも先に実行される
- 公開されている init.rc には使用箇所無し



on post-fs

- Android 2.2 までの on init から、
  on fs と on early-fs を除いた部分が書かれてる



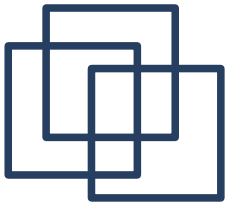
BeagleBoard の起動

- TI 等のサイトで公開されてる方法
  - kernel の起動パラメータで指定した領域をマウント

    system 、data 、cache は同じパーティションに存在する

[起動用の設定]
```
setenv bootargs 'console=ttyS2,115200n8 noinitrd root=/dev/mmcblk0p3
init=/init rootfstype=ext3 rw rootdelay=1 nohz=of omapfb.mode=1280x720MR-16@60'
```
※1 行です
```
setenv bootcmd 'mmc init; fatload mmc 0:1 0x80300000 uimage; bootm 0x80300000'
```

[mount コマンド実行の結果]
```
# mount
rootfs / rootfs rw 0 0
/dev/root / ext3 rw,relatime,errors=continue,data=writeback 0 0
tmpfs /dev tmpfs rw,relatime,mode=755 0 0
devpts /dev/pts devpts rw,relatime,mode=600 0 0
proc /proc proc rw,relatime 0 0
sysfs /sys sysfs rw,relatime 0 0
tmpfs /mnt/asec tmpfs rw,relatime,mode=755,gid=1000 0 0
tmpfs /mnt/obb tmpfs rw,relatime,mode=755,gid=1000 0 0
```

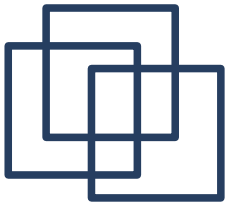


BeagleBoard の起動

- ramdisk を使用して起動する方法
  - NexusOne 等の端末と同じ起動の方法

[起動用の設定]
```
setenv bootargs 'console=ttyS2,115200n8 init=/init nohz=of omapfb.mode=1280x720MR-16@60'
setenv bootcmd 'mmc init; fatload mmc 0:1 0x80300000 uimage;
fatload mmc 0 0x81000000 ramdisk.img; bootm 0x80300000 0x81000000
```
※1 行です

[mount コマンド実行の結果]
```
rootfs / rootfs ro,relatime 0 0
tmpfs /dev tmpfs rw,relatime,mode=755 0 0
devpts /dev/pts devpts rw,relatime,mode=600 0 0
proc /proc proc rw,relatime 0 0
sysfs /sys sysfs rw,relatime 0 0
tmpfs /mnt/asec tmpfs rw,relatime,mode=755,gid=1000 0 0
tmpfs /mnt/obb tmpfs rw,relatime,mode=755,gid=1000 0 0
/dev/block/mmcblk0p3 /system ext4 ro,relatime,barrier=1,data=ordered 0 0
/dev/block/mmcblk0p5 /data ext4 rw,nosuid,nodev,relatime,barrier=1,data=ordered 0 0
/dev/block/mmcblk0p6 /cache ext4 rw,nosuid,nodev,relatime,barrier=1,data=ordered 0 0
```



ramdisk の作成

- Android のビルドで ramdisk.img は出来ている
  - out/target/product/beagleboard/ramdisk.img

    これの中身は、out/target/product/beagleboard/root

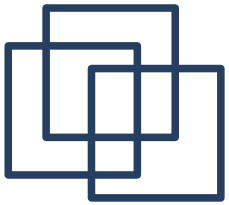
◆uboot で扱えるように作り直す

```
mkimage -A arm -O linux -T ramdisk -C none -a 0x81000000 -n ”Android Root Filesystem”
-d ./ramdisk.img ./myramdisk.img
```

※1 行です

◆init.rc の編集

```
on fs
# mount mtd partitions
    # Mount /system rw first to give the filesystem a chance to save a checkpoint
    mount ext4 /dev/block/mmcblk0p3 /system
    mount ext4 /dev/block/mmcblk0p3 /system ro remount
    mount ext4 /dev/block/mmcblk0p5 /data nosuid nodev
    mount ext4 /dev/block/mmcblk0p6 /cache nosuid nodev
```



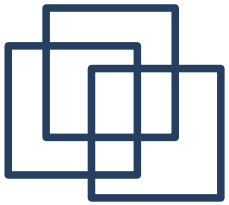
ext4 の system.img 作成

- Android 2.3 から ext4 の system.img を作成可能
  - ext4 のイメージを作成するツール

    out/host/linux-x86/bin/make_ext4fs
  - 例: system.img を作成

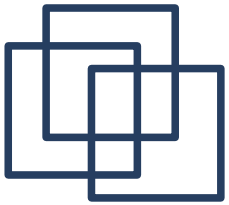
    make_ext4fs -l 128M system.img system
  - 例: system.img を SD に書き込む

    sudo dd if=system.img of=/dev/sdb3



init.rc の違い2

- lowmemorykiller への設定値
  - メモリ不足時に容赦無くプロセスを殺すやつ

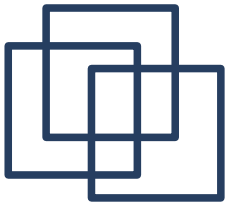


lowmemorykiller

- init.rc で設定してる値(***ADJ)に変化有り

設定項目	Android2.2	Android2.3
FOREGROUND_APP_ADJ	0	0
VISIBLE_APP_ADJ	1	1
PERCEPTIBLE_APP_ADJ		2
HEAVY_WEIGHT_APP_ADJ		3
SECONDARY_SERVER_ADJ	2	4
BACKUP_APP_ADJ	2	5
HOME_APP_ADJ	4	6
HIDDEN_APP_MIN_ADJ	7	7
CONTENT_PROVIDER_ADJ	14	
EMPTY_APP_ADJ	15	15

イカで定義されている値
frameworks/base/services/java/com/android/server/am/ActivityManagerService.java

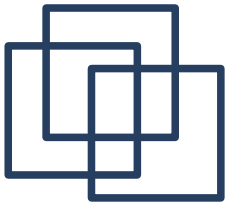


lowmemorykiller

- init.rc で設定してる値(***MEM)に変化有り

設定項目	Android2.2	Android2.3
FOREGROUND_APP_MEM	1536	2048
VISIBLE_APP_MEM	2048	3072
PERCEPTIBLE_APP_MEM		4096
HEAVY_WEIGHT_APP_MEM		4096
SECONDARY_SERVER_MEM	4096	6144
BACKUP_APP_MEM	4096	6144
HOME_APP_MEM	4096	6144
HIDDEN_APP_MIN_MEM	5120	7168
CONTENT_PROVIDER_MEM	5632	
EMPTY_APP_MEM	6144	8192

イカで定義されている値
frameworks/base/services/java/com/android/server/am/ActivityManagerService.java

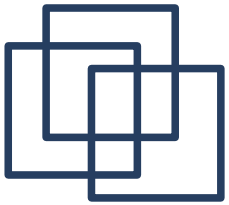


lowmemorykiller

- init.rc で lowmemorykiller に設定している値

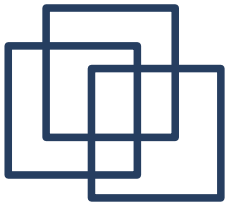
```
[Android 2.2 の設定]
write /sys/module/lowmemorykiller/parameters/adj 0,1,2,7,14,15
write /sys/module/lowmemorykiller/parameters/minfree 1536,2048,4096,5120,5632,6144
write /proc/1/oom_adj -16

[Android 2.3 の設定]
write /sys/module/lowmemorykiller/parameters/adj 0,1,2,4,7,15
write /sys/module/lowmemorykiller/parameters/minfree 2048,3072,4096,6144,7168,8192
write /proc/1/oom_adj -16
```

ADJ、MEM で設定している値の中から
選んで設定(調整)する



lowmemorykiller

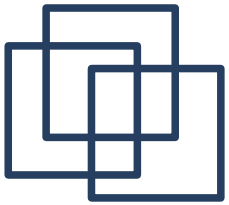
- メモリ不足時に呼ばれ、イカの処理を行う
    1. 空きページ数を調べる
    2. 空きページ数が、minfree に設定した値よりも小さくなる箇所を探す
       これを min_adj として覚えておく
    3. 全プロセスを検索し、各プロセスの oom_adj と min_adj を比較
       min_adj 以上の値を持つプロセスと、そのメモリサイズを覚えておく
    4. 殺すプロセスを選んで、シグナル（SIGKILL）発行
       殺す基準は、最もメモリを使っている& oom_adj の値が最大



lowmemorykiller

drivers/staging/android/lowmemorykiller.c

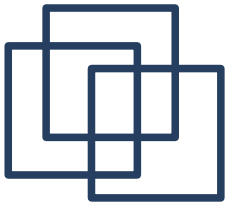
```
 82 static int lowmem_shrink(struct shrinker *s, int nr_to_scan, gfp_t gfp_mask)
 83 {
 84         struct task_struct *p;
 85         struct task_struct *selected = NULL; // 殺されるプロセスの格納先
 86         int rem = 0;
 87         int tasksize;
 88         int i;
 89         int min_adj = OOM_ADJUST_MAX + 1;    //16 (OOM_ADJUST_MAX は 15)
 90         int selected_tasksize = 0;           // 殺す候補のプロセスのサイズ
 91         int selected_oom_adj;                // 殺す候補のプロセスの oom_adj の値
 92         int array_size = ARRAY_SIZE(lowmem_adj);
            // 空きページとキャッシュされてるページの取得
 93         int other_free = global_page_state(NR_FREE_PAGES);
 94         int other_file = global_page_state(NR_FILE_PAGES) -
 95                                           global_page_state(NR_SHMEM);
```



lowmemorykiller

drivers/staging/android/lowmemorykiller.c

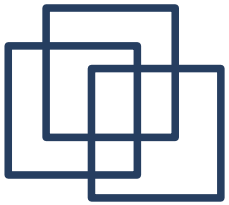
```
112         for (i = 0; i < array_size; i++) {
113                 if (other_free < lowmem_minfree[i] &&
114                     other_file < lowmem_minfree[i]) {
115                         min_adj = lowmem_adj[i];
116                         break;
117                 }
118         }
119         if (nr_to_scan > 0)
120                 lowmem_print(3, "lowmem_shrink %d, %x, ofree %d %d, ma %d¥n",
121                              nr_to_scan, gfp_mask, other_free, other_file,
122                              min_adj);
123         rem = global_page_state(NR_ACTIVE_ANON) +
124                 global_page_state(NR_ACTIVE_FILE) +
125                 global_page_state(NR_INACTIVE_ANON) +
126                 global_page_state(NR_INACTIVE_FILE);
127         if (nr_to_scan <= 0 || min_adj == OOM_ADJUST_MAX + 1) {
128                 lowmem_print(5, "lowmem_shrink %d, %x, return %d¥n",
129                              nr_to_scan, gfp_mask, rem);
130                 return rem;
131         }
132         selected_oom_adj = min_adj;
```



lowmemorykiller

drivers/staging/android/lowmemorykiller.c

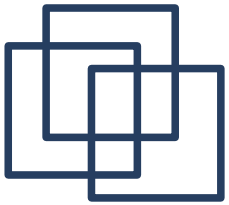
```
135         for_each_process(p) {
136                 struct mm_struct *mm;
137                 struct signal_struct *sig;
138                 int oom_adj;
140                 task_lock(p);
141                 mm = p->mm;
142                 sig = p->signal;
143                 if (!mm || !sig)
144                         { task_unlock(p); continue; }
147                 oom_adj = sig->oom_adj;
148                 if (oom_adj < min_adj)
149                         { task_unlock(p); continue; }
152                 tasksize = get_mm_rss(mm);
153                 task_unlock(p);
154                 if (tasksize <= 0) continue;
156                 if (selected) {
157                         if (oom_adj < selected_oom_adj) continue;
159                         if (oom_adj == selected_oom_adj &&
160                             tasksize <= selected_tasksize) continue;
162                 }
163                 selected = p;
164                 selected_tasksize = tasksize;
165                 selected_oom_adj = oom_adj;
168         }
```



lowmemorykiller

drivers/staging/android/lowmemorykiller.c

```
169         if (selected) {
170                 lowmem_print(1, "send sigkill to %d (%s), adj %d, size %d¥n",
171                              selected->pid, selected->comm,
172                              selected_oom_adj, selected_tasksize);
173                 lowmem_deathpending = selected;
174                 lowmem_deathpending_timeout = jiffies + HZ;
175                 force_sig(SIGKILL, selected);
176                 rem -= selected_tasksize;
177         }
```



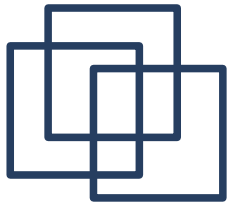
lowmemorykiller

- 動作例: min_adj 値の決定

init.rc にて以下をイカを記述することで、
write /sys/module/lowmemorykiller/parameters/adj 0,1,2,4,7,15
write /sys/module/lowmemorykiller/parameters/minfree 2048,3072,4096,6144,7168,8192
lowmem_adj[6] と lowmem_minfree[6] に値が格納される。

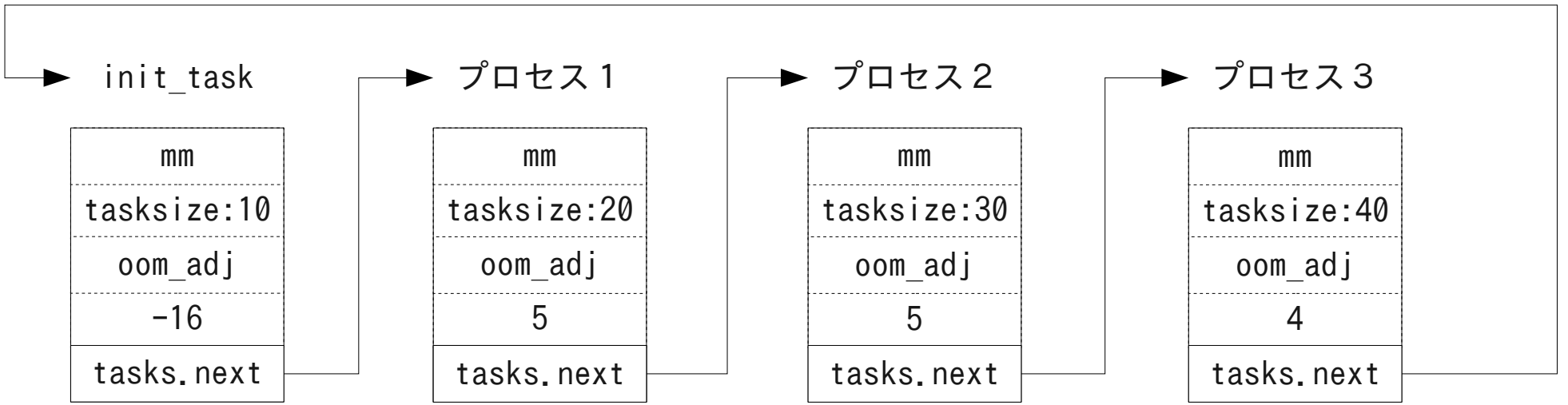
other_free = global_page_state(NR_FREE_PAGES);
other_file = global_page_state(NR_FILE_PAGES) - global_page_state(NR_SHMEM);
で得られる値がそれぞれ、
other_free = 3263;
other_file = 4444;
であるとする。

minfree[2]（4096）で、other_free（3263）の方が小さくなる。
minfree[3]（6144）で、other_file（4444）の方が小さくなる。
以上の結果から、min_adj の値は lowmem_adj[3] に格納されている 4 になる。

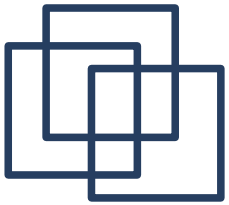


lowmemorykiller

- 動作例:殺すプロセスの検索



処理の流れ
1. init_task はmin_adj(4) より小さいので候補から外れる。
2. プロセス１はmin_adj(4) より大きいので候補になる。 min_adj を５に更新。
3. プロセス２はmin_adj(5) と同じだが、 tasksize がプロセス１より大きいので候補になる。
4. プロセス３はmin_adj(5) より小さいので候補から外れる。
5. プロセスを全て検索し終わり、殺すプロセスはプロセス２に決定。
6. プロセス２を殺す。



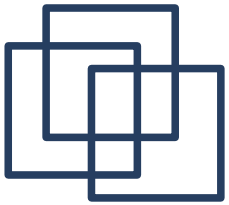
lowmemorykiller

- oom_adj の設定、確認方法
  - プロセス番号 N のプロセスの oom_adj に 8 を設定

    echo 8 > /proc/N/oom_adj
  - プロセス番号 N のプロセスの oom_adj を確認

    cat /proc/N/oom_adj
- 殺す対象から外す方法
  - oom_adj に -17 を設定すると対象外
- oom_adj の範囲
  - -16 〜 15 で、小さいほど殺され難い

    デフォルト値は 0 で、システムの状態によって変動する（はず）



lowmemorykiller

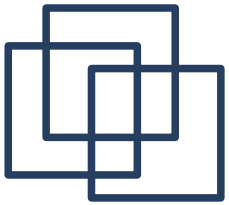
- lowmem_shrink を呼び出すには
  - shrinker_list に関数を登録
  - メモリ開放のために、

    shrink_slab は shriner_list に登録されてる関数を呼び出す

    ここで登録した lowmem_shrinker が呼び出される

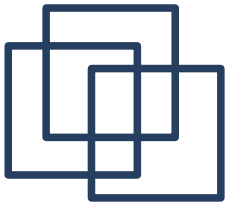
drivers/staging/android/lowmemorykiller.c

```
184 static struct shrinker lowmem_shrinker = {
185         .shrink = lowmem_shrink,
186         .seeks = DEFAULT_SEEKS * 16
187 };
189 static int __init lowmem_init(void)
190 {
191         task_free_register(&task_nb);
192         register_shrinker(&lowmem_shrinker);
193         return 0;
194 }
```



init.rc の違い3

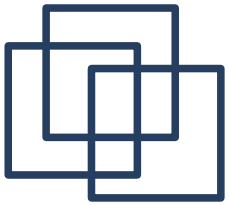
- early-init と early-boot
  - Android 2.2 までもあったけど、
    公開されてる init.rc では使われてなかった。



early-init

- Android 2.3 から追加された ueventd の起動に使用
  - ueventd は init の一部

```
# ls -l /sbin/ueventd
lrwxrwxrwx system   system            2011-01-14 14:27 ueventd -> ../init
```



early-init

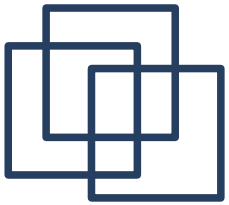
- ueventd の起動は以下のようにして判別している

system/core/init/init.c

```
652 int main(int argc, char **argv)
653 {
654     int fd_count = 0;
655     struct pollfd ufds[4];
656     char *tmpdev;
657     char* debuggable;
658     char tmp[32];
659     int property_set_fd_init = 0;
660     int signal_fd_init = 0;
661     int keychord_fd_init = 0;
662
663     if (!strcmp(basename(argv[0]), "ueventd"))
664         return ueventd_main(argc, argv);
```

system/core/init/ueventd.c

```
 34 int ueventd_main(int argc, char **argv)
 35 {
```

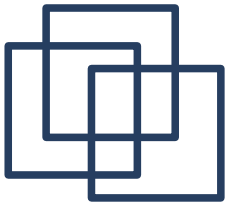


early-init

- ueventd の役割
  - Android 2.2 までは device.c で行っていた処理（デバイスファイルの作成）
    以前の PF 部の資料に説明有りのため内容は省略
  - 設定を ueventd.rc というファイルに書ける
    init.rc 同様にデバイス固有の設定を分けられる
    例: BeagleBoard だと、ueventd.omap3.rc

system/core/rootdir/ueventd.rc の抜粋

```
/dev/null                 0666   root       root
/dev/zero                 0666   root       root
/dev/full                 0666   root       root
/dev/ptmx                 0666   root       root
```



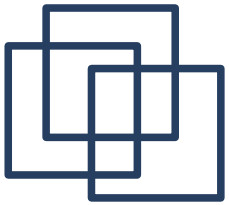
early-boot

- init.lowmem.rc
  - このファイルに記述あるけど、このファイルは読み込まれない
    将来的に使えるようになるのか、テンプレートなのか、調査不足

system/core/rootdir/init.lowmem.rc

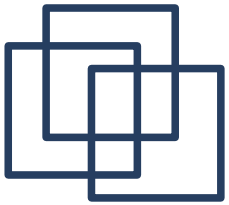
```
on early-boot
    setprop ro.FOREGROUND_APP_MEM 1536
    setprop ro.VISIBLE_APP_MEM 2048
    setprop ro.PERCEPTIBLE_APP_MEM 2048
    setprop ro.HEAVY_WEIGHT_APP_MEM 2048
    setprop ro.SECONDARY_SERVER_MEM 4096
    setprop ro.BACKUP_APP_MEM 4096
    setprop ro.HOME_APP_MEM 4096
    setprop ro.HIDDEN_APP_MEM 5120
    setprop ro.EMPTY_APP_MEM 6144

on boot
    write /sys/module/lowmemorykiller/parameters/minfree 1536,2048,3072,4096,5120,6144
```

Android 2.2 までの設定値がベースになっている



early-boot

- init.lowmem.rc が読み込まれないようなので、
  init.omap3.rc にコピペして使用することにした
  これで、ターゲット毎の設定に使える



資料作成時間無かった

- その他
  - ARMv6 向けにビルドする環境を作る
  - DalvikVM
  - Framework