A LaTeX Report Template

The report subtitle

Garza Matteo
Matr. 755295, (matteo.garza@mail.polimi.it)

Report for the master course of Real Time Operative System (RTOS)
Reviser: PhD. Patrick Bellasi (bellasi@elet.polimi.it)

Received: April, 01 2011

Abstract

Android Operative System is the most diffuse OS in lots of devices (expecially smartphones and tablets). In this paper we will analyze how Android manages memory on device. We discuss about application memory and some of the most used low level memory management systems used by Android OS.

1 Introduction

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce non eros nec dui sagittis placerat. Sed vulputate accumsan massa? Quisque dapibus consequat magna! Mauris ante erat, ullamcorper eu, iaculis a, posuere at, massa. Donec diam orci, luctus non, tristique at, volutpat vulputate, dui. Suspendisse potenti. Vivamus lacinia justo id enim! Maecenas ut odio. Donec nisi urna, gravida eu, sollicitudin eget, gravida quis, dolor. Nulla quam erat, convallis id, lobortis vel, rutrum tempor, felis. Nam id ante. Sed hendrerit ultricies sapien. Morbi interdum convallis elit. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Suspendisse potenti. Praesent egestas.

2 Low level memory management

In this part, we discuss about how the memory has managed in Android devices. For most of the releases in Android, it was used PMEM and ASHMEM. These kind of libraries was too simple, and was patched with some SoC patches, such as NVMAP for nVidia Tegra devices and CMEM for TI OMAP ones. The most important patch was CMA (Contiguous Memory Access), expecially with DMABUF patch. With the release of Android 4.0 (Ice Cream Sandwich) a brand new library has released, ION. We discuss about differences between ION and CMA approach, and, in the state-of-art, we discuss of a future integration between them.

2.1 PMEM and ASHMEM

PMEM (Process MEMory) is the first memory driver implemented on Android devices (since G1). It is used to manage shared memory regions sufficiently large (from 1 to 16MB).

This regions must be phisically contiguous between user space and kernel drivers (such as GPU, or DSP). It was written specifically to be used in a very limited hardware platform, and it could be disabled on x86 architectures.

ASHMEM (Android SHared MEMory) is a shared memory allocator subsystem, similar to POSIX, but with a different behaviour. It also gives to the developer an easier and file-based API. It used named memory, freeable by the kernel. Apparently, ASHMEME supports low memory devices better than PMEM, because it could free shared memory units when it is needed.

Figure 1: Some single-column figure caption.

2.2 CMA and DMABUF

CMA (Contiguous Memory Allocator) let the device to alloc big chunk of memory after the system has booted. Differently from similar framework, it let regions of system-reserved memory to be reused in a trasparent way, letting memory not to be wasted. When an alloc is istantiated, this framework migrates all the system page. Thus to build a big chunk of phisically contiguous memory.
Why do an OS have to use chunks of memory? Because virtual memory tends to fragment pages. An intensive use

of memory let the system not able to find contiguous memory in a very short time after boot. Recently, the requirement of huge pages in applications raises, expecially for trasparent huge pages. Another question is devices (such as cameras) that needs DMA over areas phisically contiguous.
CMA reserve an huge area of memory at boot time, only for huge request of memory. For every region, block of pages can be flaggable as three type.

- movable : typically, cache pages or anonymous pages, accessed by page table or page cache radix tree
- kernel recallable : they can be given back to the kernel by request.
- inamovable : these are typically pointer referred pages (such as pages invoked by a kmalloc())

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec et ligula. Nullam in libero. Donec dictum pede in justo. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam congue. Aliquam egestas. Nunc eu est ac nibh mattis vestibulum. Curabitur aliquet bibendum odio. Etiam hendrerit. Nunc a velit quis dui molestie consequat. Sed et turpis et mi feugiat tincidunt. Sed sollicitudin. Ut risus? Duis eget orci eu turpis consectetur fringilla? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam tellus ligula, placerat vitae, tempor vitae, varius id; est! Nullam et ipsum eget tellus eleifend sollicitudin? Fusce urna massa, imperdiet vitae, convallis in; lacinia sed, tortor.
And this is the reference to a single column figure (see **Figure 2**). Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam consectetur neque at orci. Curabitur non metus. Praesent congue porta nisl. Suspendisse ultricies, sem ac ultrices aliquam, erat nisi fermentum est; a rhoncus mauris arcu eget nibh. Aliquam sollicitudin velit non erat. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nulla diam, facilisis vel, accumsan sed; molestie egestas, massa. Fusce malesuada, ipsum et pulvinar aliquet, est dolor laoreet enim, quis porttitor erat mauris eget sapien. Integer vitae urna. Duis lectus.

2.3 ION

In december 2011, PMEM is marked as deprecated, and then replaced by ION memory allocator. ION is a memory manager that Google has developed from the 4.0 release of Android (Ice Cream Sandwich), mainly to resolve the interface issue between different memory management between different Android device. In fact, some SoC developer implemented different memory manager. We can cite two of them:

NVMAP, implemented on nVidia Tegra

CMEM, implemented on TI OMAP

All this vendor will pass to ION soon.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec et ligula. Nullam in libero. Donec dictum pede in justo. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam congue. Aliquam egestas. Nunc eu est ac nibh mattis vestibulum. Curabitur aliquet bibendum odio. Etiam hendrerit. Nunc a velit quis dui molestie consequat. Sed et turpis et mi feugiat tincidunt. Sed sollicitudin. Ut risus? Duis eget orci eu turpis consectetur fringilla? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam tellus ligula, placerat vitae, tempor vitae, varius id; est! Nullam et ipsum eget tellus eleifend sollicitudin? Fusce urna massa, imperdiet vitae, convallis in; lacinia sed, tortor.

3 The Second Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean magna. Nunc non ante eget nibh condimentum tempor. Nullam ullamcorper lectus eget mauris. Nam neque orci; rhoncus at, pulvinar quis, elementum sit amet, turpis. Mauris posuere nisi ut justo. Morbi non lorem vitae mauris interdum faucibus. Vestibulum ut sapien in augue faucibus fringilla. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Etiam vestibulum fringilla libero. Curabitur libero diam, hendrerit sit amet, ornare eget, imperdiet vel, purus!

3.1 The first subsection of the second Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam consectetur ante at eros. Vestibulum mi nisi, venenatis sollicitudin, tempus sed, auctor id, tortor. Fusce orci. Duis tellus arcu, euismod sed, consequat sit amet, elementum vel, mauris. Curabitur leo diam; dapibus quis, condimentum vitae, dignissim ut, diam. Nulla et nulla eget elit volutpat sagittis.

3.2 The second subsection of the second Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris eget mauris. Nulla facilisi. Ut condimentum tempor eros? Integer metus mauris, consectetur sit amet, tempor a, facilisis eu, nisl. Vestibulum at turpis. Ut vitae tortor pretium nisl vestibulum blandit. Nulla nibh urna, semper et, elementum at, mattis ut, nisi! Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Morbi vel ligula eget lacus convallis venenatis. Aliquam lacinia tincidunt felis. Ut dui.

References

Figure 2: Some wide-figure caption.