A New Type of Embedded File System Based on SPM

Tianzhou Chen, Feng Sha, Wei Hu, and Qingsong Shi

College of Computer Science, Zhejiang University, Hangzhou, Zhejiang, China, 310027
{tzchen,zhaoyi,ehu,zjsqs}@zju.edu.cn

Abstract. Commonly, embedded file systems reside in main memory to manage the external memory such as flash memory in embedded systems. With the progress of hardware of embedded systems, the gap of speed between main memory and CPU is becoming larger and larger. The traditional embedded file systems can not be able to support real-time response enough for their main memory management policy. A new type of embedded file system is presented to provide more real-time performance. This new embedded file system, SPM file system, is based on the internal SRAM on chip and is able to reduce response time.

Keywords: Embedded System, File System, Scratched-Pad Memory.

1 Introduction

Scratch-pad Memory (SPM) has played an important role in embedded systems by collaborating with cache. SPM has many advantages for embedded systems such as lower energy consumption. Though the size of SPM is much smaller compared with the DRAM, it has been used in many high-end embedded microprocessors. The Intel PXA 27x family is the processor covered this new feature. The performance of the whole embedded systems can be improved by using SPM when developing systems. SPM has more potential than DRAM. SPM is at least 10 times faster than the normal DRAM. And the speed of SPM will be increased faster than DRAM: the speed of SPM is increasing by 60% a year versus only 7% a year for DRAM [1].

Embedded systems have to provide real-time, fast speed and power-aware features to the users. But the memory references in the embedded system are always bottleneck of the whole systems. Though Cache has been popular in embedded systems, the only use of cache is not sufficient. Programmers can not utilize Cache unrestricted to improve the performance because of the security of the systems. The utilization of SPM can provide performance improvement to the embedded systems. A detailed recent study compares the tradeoffs of a cache as compared to a SPM. Their results show that a SPM has 34% smaller area and 40% lower power consumption than a cache memory of the same capacity [2].

In tradition, the embedded file systems, which are mostly based on flash memory, do not care the SPM on chip. Commonly the main target of embedded file systems is to manage a large capacity flash memory. eNVy [2] proposed a large non-volatile main memory storage system built primarily with Flash memory. eNVy uses a copy-on-write scheme, page remapping, a small amount of battery backed SRAM, and high

© Springer-Verlag Berlin Heidelberg 2007

bandwidth parallel data transfers for Flash memory. A hybrid cleaning algorithm (combining FIFO and greedy algorithms) for large Flash arrays is used to reclaim space. The algorithm is designed to evenly wear the array, thereby extending its life time. And more other excellent works have been done on Flash memory [4-7]. These are all static table-driven schemes and operate on Flash arrays and it is not efficient to manage large-scale flash memory.

As we know from above, there are no embedded file systems concerning the use of SPM to improve the performance of response time. SOC [9] is also the direction of embedded systems in the future. Thus the software for embedded systems should be more efficient. In this paper, we present a new type of embedded file system named SPMFS (SPM File System) to take advantage of the SPM on chip.

The rest of the paper is organized as follows. Section 2 describes the architecture of SPMFS. Section 3 covers the experimental results. Section 4 concludes.

2 Architecture of SPM File System

Commonly the embedded file systems are stored in the flash or some other external storage according to the storage hierarchy. These embedded file systems have many functions to manage the flash, provide access response and so on. Thus these file systems do not need to take their size into accounts. The SPMFS is designed for SOC chip to improve the performance of the file system.

The traditional embedded file systems integrate many functions together. The SPMFS is not same to these file systems. It has to be fit for the size of the SPM on chip. The SPM can not hold so many data and instructions simultaneously for its small size. Thus we design a MFSC (Micro File System Component) for our new file system architecture.

MFSC is main component of the SPMFS. It will run in SPM from the start of the system. There are many other functions, which are reduced from MFSC and should be provided by file system. They will reside in main memory. In our design, SPMFS consists of three parts: MFSC, BFSM (Basic File System Management) and ComM (Communication Management). The architecture is shown in Figure 1.

MFSC: this part is stored in internal Flash memory on chip. To get short response time, MFSC will be not altered to avoiding writing to flash memory. If and only if the MFSC has been a new version, the MFSC in internal Flash memory will be updated. When the system starts, the MFSC will execute with the whole file system. When the MFSC executes, it will maintain an open file table for files used continually and the data area for these files. The data area can be considered as a buffer of the data in main memory.

SPMFS must ensure that this MFSC can run in SPM. Because of the size limitation of SPM, the MFSC has to be designed as small as possible. Thus some parts in traditional embedded file system must be taken from the MFSC and the remainders have to be cut down or modified in order to make all necessary functionalities of SPMFS can be able to be contained in SPM. The most design principle is minimum and optimal. According to the foregoing design principle, we only provide necessary functions in MFSC to manage SPM.