CS Masters' Thesis Defense

Title:	*A New RAID Linux Flash File System*
Speaker:	Xinhua Fahy
Date:	Monday, April 2, 2012
Time:	11:00 a.m.
Location:	GMCS 405
Thesis advisor:	Dr Tao Xie

Abstract:

NAND flash memory has been widely used in the embedded market because of its desirable properties such as non-volatility, shock resistance, low power and low latency for reading and writing. However, hardware limitations like the need to erase before writing again, erase block sizes than span multiple write pages, and limited memory cell lifespan, make file system requirements for flash memory very different from those for conventional block devices. As such, conventional file system cannot be applied directly to flash memory.

To address these issues, two different approaches have been proposed: (1) add a software layer that emulates a block device using the flash device, and (2) create special-purpose file systems that are targeted to flash devices. The first approach, which is based on a Flash Translation Layer (FTL) concept, succeeds in making flash devices universally accessible, but hiding flash device characteristics from the file system inevitably leads to reduced performance compared to the second option, of creating “flash-aware” file systems.

There is currently no support for flash-based RAID file systems in standard Linux distributions. RAID support is particularly important for NAND flash devices because they have a high rate of cell failure as they age compared to disk drives. In addition, RAID support can provide performance enhancements that enable video rate recording of data, which has already been demonstrated in some high performance video cameras. As NAND flash devices continue to fall in price at a faster pace than disk drivers, price barriers will be reduced or eliminated. As such it is anticipated that RAID-capable flash file systems will eventually become a fundamental requirement for modern operating systems.

There are currently three dominant flash file systems for Linux: JFFS2, YAFFS2, and UBIFS. JFFS2 has issues scaling to large sizes and does not write the index data to flash as it goes, but instead reconstructs it every time the system mounts. This means mounting a large JFFS2 file system can take a long time, which would only be worsened in the context of RAID. YAFFS2 and UBIFS do write their indexes to flash as they go but have other issues. UBIFS is the most advanced of the three but is somewhat immature, very complex (three times the code size of JFFS2), and has high storage overhead for the index (over 10%).

This thesis considers enhancements to JFFS2 which eliminate its main deficiencies and add RAID capability. The index is off-loaded to a more expensive non-volatile storage medium such as PRAM or battery-backed SRAM. The added cost of the index memory is offset by the reduced storage overhead, faster access times, and reduced complexity compared to UBIFS. The implemented RAID-based file system, JFFSR, is compared to JFFS2, YAFFS2, and UBIFS.