Published on *Yaffs* (http://www.yaffs.net)

Yaffs 2 Specification

Summary

The original motivation for Yaffs 2 was to add support for the new NAND with 2kB pages instead of 512-byte pages and strictly sequential page writing order.

To achieve this, a new design is used which also realises the following benefits:

- Zero page rewrites means faster operation. (Yaffs1 uses a single rewrite in the spare area to delete a page).
- Ability to exploit simultaneous page programming on some chips.
- Improves performance relative to Yaffs1 speed(write:1.5x to 5x, delete: 4x, garbage collection: 2x)
- Lower RAM footprint (approx. 25% to 50% of Yaffs1).
- Can support Toshiba/Sandisk MLC parts.

Most of Yaffs1 and Yaffs2 code is common, therefore the code will likely be kept common with Yaffs1 vs Yaffs2 being run-time selected.

Method

The main philosophical difference between Yaffs and Yaffs2 is how discarded status is tracked. Since we can't do any re-writing, we can't use the "discarded" flags in Yaffs2.

Instead Yaffs2 uses two mechanisms to resolve data state.

- Yaffs2 chunks have more tag information, including a block sequence Id. From that we can determine the chunk sequence Id since the chunks are allocated sequentially in a block. Thus we always know the patch order for all chunks in the system.
- The above helps us track stale -vs- fresh data, but does not help determine when a file/object is deleted. Deletion is achieved by moving the object to the "unlinked" directory. We also keep track of the number of chunks (both stale and current) in the system for each object. While this number indicates that there are still chunks associated with this object we keep the deletion record. When the last trace of the object has been really erased from NAND, we can forget about the deletion record too.
- Since there is no deletion, a resize (shrinking) of a file will still have valid data chunks past the end of file on the NAND. However, we write a new ObjectHeader at the time of the resize, therefore this shows the shrunken file size.

This changes erasure slightly:

- During garbage collection we can't just look at chunk state flags, instead we must read the tags of each chunk to determine which object's chunk count we must decrement. This step must also be performed when a block is erased (as part of deletion).

This makes erasure & garbage collection more expensive (by adding reads), but remember that ion Yaffs2 we don't need to do page deletions which are much more expensive operations. Thus, all-up Yaffs2 wins.

Tag structure

Each chunk in Yaffs2 has the following information:

Field

Comment

Size for 1kb chunks

Size for 2kB chunks

blockState	Block state. non-0xFF for bad block	1 byte	1 byte
chunkId	32-bit chunk Id	4 bytes	4 bytes
objectId	32-bit object Id	4 bytes	4 bytes
nBytes	Number of data bytes in this chunk	2 bytes	2 bytes
blockSequence	sequence number for this block	4 bytes	4 bytes
tagsEcc	ECC on tags area	3 bytes	3 bytes
ecc	ECC, 3 bytes/256 bytes of data	12 bytes	24 bytes

Total		30 bytes	42 bytes

To get enough spare bytes for this tagging structure requires a chunk-size of at least 1kB.

The blockSequence increments each time a block is allocated. (ie. the first block allocated is block 1, and so on).

Scanning

The only reason we need to keep track of data status on NAND is to be able to recreate the file system state during scanning. Since we no longer have chunk deletion status flags we use a slightly different process for scanning a Yaffs2 system.

In effect, Yaffs2 recreates its state by “replaying the tape”. ie. it scans the chunks in their allocation order (block sequence Id order) rather than in their order on the media. This implies that at start up, the blocks must be read and their block sequence determined.

Performance

Times for 2kB read(units of 1uS).

Operation

Yaffs1

Yaffs2 (512b pages)

Yaffs2 (2kB pages)

Yaffs2(2kB pages, x16)

Seek	40	40	10	10
Read	220	220	220	110
Total	260	260	230	120
MB/s	7.6	7.6	8.7	16.7
Relative speed	1	1	1.1	2.2

Times for 2kB writes(units of 1uS).

Operation

Yaffs1

Yaffs2 (512b pages)

Yaffs2 (2kB pages)

Yaffs2(2kB pages, x16)

Seek	40	40	10	10
Program	800	800	200	200
Seek	40	40	10	10
Read	220	220	220	110
Total	1320	1320	660	440
MB/s	1.5	1.5	3	4.5
Relative speed	1	1	2	3

Times for 1MB delete (units of 1uS).

Operation

Yaffs1

Yaffs2 (512b pages)

Yaffs2 (2kB pages)

Yaffs2(2kB pages, x16)

Seek	20480	0	0	0
Program	409600	0	0	0
Erase	128000	128000	16000	16000
Total	558080	128000	16000	16000
MB/s	1.8	7.8	62.5	62.5
Relative speed	1	4	34	34

Times for 1MB of garbage collection at 50% dirty (units of 1uS).

Operation

Yaffs1

Yaffs2 (512b pages)

Yaffs2 (2kB pages)

Yaffs2(2kB pages, x16)

Delete 1MB	558080	128000	16000	16000
Write 0.5MB	337920	337920	168960	112640
Total	896000	465920	184960	128640
MB/s	1.1	2.1	5.4	7.7
Relative speed	1	1.9	4.9	7

Source URL: http://www.yaffs.net/yaffs-2-specification