Block Retirement in ETFS#

Overview#

The ETFS filesystem is most commonly paired with NAND flash. One characteristic of NAND flash is that blocks may wear out over time. This new feature is intended to allow ETFS to "retire" blocks from service before they fail completely. Since the ETFS filesystem, and the flash driver that it's paired to, has a direct view into the health and activity of the NAND flash, it's uniquely suited to identify blocks as they wear out, and shuffle information around in order to discontinue use of the soon-to-be bad block.

Requirements#

- Create a method for the ETFS filesystem to be notified that a block is wearing out, or is otherwise unhealthy
- ETFS shall have the ability to salvage data from the failing block, and save it.
- ETFS shall mark the block as bad
- ETFS shall prevent further use of the affected block
- A supporting function in the hardware driver "devio" layer shall be created to allow the ETFS library to mark the media as retired.

Design#

Because ETFS is hardware-agnostic, it is up to the "devio" layer to tell ETFS when a block is failing. How the devio layer determines that a block is failing is out of the scope of this change.

Once ETFS receives notice from the devio layer that a block is failing, it will copy data from the failing block to other locations within the partition. There are some conditions which may prevent the saving of all data:

1. The filesystem is full, and there is no extra space to save away the data from the unhealthy block
2. The block has suffered a sudden, and complete failure, such that ECC is insufficient to recover the data read from it.

In the above cases, filesystem damage is unavoidable.

After data has been successfully saved from the unhealthy block, the ETFS filesystem will:

1. Mark the block as BAD in the internal ETFS data structures
2. Attempt to erase the block
3. Attempt to mark the block as BAD on media (requires support from the new devio-layer function).

In the current design, the ETFS library is notified, and retires a block, synchronous to a media operation. The media operation will be blocked while the "block retirement" operation is in progress. If the media operation was part of a client I/O request, the client request will also be blocked during this time.

If an unhealthy block is detected during the startup scan, the retirement procedure is deferred until the startup scan is complete. Processing of any blocks needing "retiring" detected in this way, will be done before the partition is mounted.

Implementation#

The ETFS driver is broken up into three layers: filesystem, devio, chipio; only the devio and filesystem are relevent to this feature.

The devio layer is responsible for such low level operations as: read a transaction, read a cluster, write a cluster, erase a block. The etfs filesystem layer is hardware agnostic, and is responsible for tracking the location of data on media, ensuring even wear, and enforcing data and filesystem integrity.

Currently, all calls from the ETFS file system layer to the devio layer are done through one interface: etfs_devcall(). This function then becomes a convenient place to check for the ETFS_TRANS_RETIRE flag being returned from the low-level driver function. To allow this, a new argument had to be added to etfs_devcall(): the block number.

```
-int etfs_devcall(struct devctrl *dcp, int status)
+int etfs_devcall(struct devctrl *dcp, int status, unsigned blknum)
```

Most places that etfs_devcall() is invoked, the block number was already being calculated. That calculation was simply repeated in the new "blknum" argument.

The "status" argument to etfs_devcall() is the return value from one of the devio_* calls. etfs_devcall() needs to check this argument for the presence of the ETFS_TRANS_RETIRE flag. If set, etfs_devcall() needs to clear it (so as not to confuse later error handling), and then act on the request.

```
+  /* Take action if this is a block retirement request */
+   if (status & ETFS_TRANS_RETIRE){
+       /* Clear the RETIRE bit, so as not to confuse
+        * the check below
+        */
+       status &= ~ETFS_TRANS_RETIRE;
+       etfs_log(_SLOG_WARNING, "ETFS Block retirement request, on blk %d", blknum);
```