Background

Starting from a certain Conduant SDK version (uncertain which), the Conduant card firmware has lost its ability to gracefully play back modules that contain one or more corrupt/dead disks. Such modules will either freeze the Mark 5, or will play back extremely slowly with zero data.

Workaround

Disks have to be removed from the module and attached to a server so that the raw disk data can be accessed. One option is to place disks into a Mark6 module and load that into a Mark6. Alternatively, disks can be inserted into server disk cradles.

The disks of the module should appear as drives without partitions:

dhcp30222:~ # cat /proc/partitions
major minor #blocks name

```
   2        0          4 fd0
   8        0  160836480 sda
   8        1   41943040 sda1
   8        2  114861056 sda2
   8        3    4031343 sda3
  11        0    1048575 sr0
   8       16 1953514584 sdb
   8       32 1953514584 sdc
   8       48 1953514584 sdd
   8       64 1953514584 sde
   8       80 1953514584 sdf
   8       96 1953514584 sdg
```

In the above example, six disks (/dev/sdb to /dev/sdg) are still intact.

A recovery script now found in the DiFX repository (https://svn.atnf.csiro.au/difx/sites/MPIfR/mark5/mark5_scan_recovery.py) can be used to gather data from the drives, and write out recovered VDIF or Mark5B data files. A fill pattern (0x11223344) is inserted in place of missing data, identical to the formerly proper behaviour of the old Conduant card firmware.

Data from the drives can be recovered as follows:

dhcp30222:/mnt/sshfs # ~/mark5_scan_recovery.py /dev/sd{b,c,d,e,f,g}

```
Module has VSN NRAO+338/16000/2048
Module has 233 scans in the Mark5B/5C-type user directory
The original module seems to have recorded onto 8 disks
Module disk #0 --> None
Module disk #1 --> <open file '/dev/sdb', mode 'rb' at 0x7feb35d72150>
Module disk #2 --> <open file '/dev/sdc', mode 'rb' at 0x7feb35d721e0>
Module disk #3 --> <open file '/dev/sde', mode 'rb' at 0x7feb35d72300>
Module disk #4 --> None
Module disk #5 --> <open file '/dev/sdd', mode 'rb' at 0x7feb35d72270>
```

```
Module disk #6 --> <open file '/dev/sdf', mode 'rb' at 0x7feb35d72390>
Module disk #7 --> <open file '/dev/sdg', mode 'rb' at 0x7feb35d72420>
output_write() at module offset 73426876176, completed scan 0, leftover 37960 byte to next scan
output_write() at module offset 129076726760, completed scan 1, leftover 34400 byte to next scan
output_write() at module offset 208300471928, completed scan 2, leftover 11824 byte to next scan
..
```

```
dhcp30222:/mnt/sshfs # ls -al recovered | less
total 219278081
drwxr-xr-x 2 oper oper        233 Jan 23 16:31 .
drwxr-xr-x 4 oper oper          7 Jan 23 16:31 ..
-rw-r--r-- 1 oper oper 73426874368 Jan 23 16:45 GG081C_HN_No0143.vdif
-rw-r--r-- 1 oper oper 55649820672 Jan 23 16:55 GG081C_HN_No0173.vdif
-rw-r--r-- 1 oper oper 79223713792 Jan 23 17:10 GG081C_HN_No0174.vdif
-rw-r--r-- 1 oper oper 16206467072 Jan 23 17:13 GG081C_HN_No0175.vdif
...
```

Reverse Engineered Mark5 Disk Structure

A certain B.E. and H.V. at JIVE have reverse engineered the Mark5 data block structure. Mark5 recordings are scattered as 2^16 -byte sized blocks, written round-robin to disks of a module. Each block has a 8-byte header. The first 4 bytes are unused. The next 4 bytes (32 bit unsigned int) contain a sequence number in the upper 28 bits. The rest of the block contains VLBI data.

The sequences numbers of the blocks allow 1) detection of the original ordering of disks in a module, 2) detection of lost disks (lost blocks).

A certain J.W. at MPIfR spotted the User Directory storage area on the raw disks. The user directory is located at 11272704 bytes before the end of every disk. Each disk should contain an identical copy of the user directory.

Contents of a disk set can be recovered with https://svn.atnf.csiro.au/difx/sites...an_recovery.py