Disk I/O Processing

File: DISKIO.RNO
Date: April 1978
Edition: 1

CONTENTS

1.0 GENERAL DISK I/O FLOW

This discussion assumes that the disk request has been processed to the point that an I/O list has been built and the initial sector address is known. The disk is not necessarily on the correct cylinder. The following flow describes the general processing; subsequent text will describe it more fully.

```
1-  Calculate required cylinder.
2-  If seek required then
3-       If data transfer in progress
         (non-massbuss device or massbuss
         device with active transfer on this unit)
4-            Queue request to this unit
5-            Exit
6-       Else
7-            Start seek
8-            Exit
9-  Else
10-      If data transfer in progress then
11-           Queue request for channel
12-           Exit
13-      Else
14-           Disable attention interrupts
15-           Start transfer
16-           Exit
17- End


18- On interrupt
19-      Read drive number and register # from RH
20-      Read attention summary register
21-      For each on-bit in summary register do
22-           If corresponding drive was not transferring
23-                If seek complete then queue request for channel
24-                Else process status (eg. drive coming on line)
25-      If data transfer complete then
26-           If hardware detected error then
27-                Perform error recovery
28-           Compare channel termination with predicted termination
29-           If software detected error
30-                Perform error recovery
31-      For each unit with queued requests do
32-           Select next seek and start it
33-      If channel queue (already positioned drives) is not empty then
34-           Select best transfer and start it
35-      Restore register # and drive # to RH
36- Dismiss interrupt
```

The correct cylinder is determined by dividing the sector number by the number of sectors per cylinder. To determine if a seek is needed, (2) the cylinder number is compared with the current cylinder number, which is remembered from the last transfer. (There are some limited conditions under which the drive will not be on the cylinder which is recorded in the software. In these cases, the implied seek of the drive will be used). The system can only start a seek if the drive is idle (for non-MASSBUS drives, both the drive and the controller must be idle). Therefore, if there is a data transfer in progress, the request is queued for the unit and will be started at a later time at interrupt level (4). If the drive is free, a seek will be started. If the drive is already on the correct cylinder, the seek logic is bypassed. If the drive and channel are not already busy, then th-

transfer is started; otherwise, the request is added to a queue for the required channel to be started at interrupt level at a later time. A transfer may range in size from a single word (128 words) to a whole cylinder; TOPS10 attempts to perform the longest possible transfer in order to maximize I/O throughtput. The system never attempts an implied seek in the middle of a transfer. Such a user request would be broken into two or more transfers with explicit intermediate seeks. Also, in order to simplify the code considerably, attention interrupts are disabled while doing a data transfer.

When an interrupt occurs, the system may or may not have just completed a data transfer. Since attention interrupts were disabled during the data transfer, there may be a number of outstanding attention conditions when the interrupt is actually honored. First, the system reads the attention summary register. Each drive (except the one which was completing a transfer) is checked for an attention bit on. If there is an attention bit on, and if there is a seek complete, the transfer request is added to the channel queue to be started for I/O. If there was no seek in progress, then the drive has just come on line or powered up (see later discussion for these conditions).

Once all outstanding seeks are processed, the data transfer completion is handled. If there was no error or after error correction (see error recovery later), the channel termination word is compared to the predicted channel termination word. If the check fails, then error recovery is started. After completing the processing for the interrupt, any outstanding seeks are started. For each drive, the closest (shortest) seek is the one selected for startup (a fairness count will cause the system to select the oldest transfer every 'n'th time). After seeks are started, the channel queue is checked for positioned drives and the transfer with the shortest latency is started (again unless the fairness count says otherwise). SWAPPER requests receive preference over file I/O (unless fairness count expires).

There is some special processing for interrupts on a MASSBUS device caused by the fact that the system may be attempting some operation using the device registers at UUO level at the time of the interrupt. When the interrupt occurs, the system reads RHxx and saves the drive and register number to which the RH was pointing. Before dismissing the interrupt, a DATAO is done to restore the drive number and register number. The need for reading the register from the RH at interrupt and restoring them before dismissing the interrupt is made worse by the fact that the system must wait 3 microseconds after the DATAO specifying what data is wanted before the DATAI can read the data.

There are other special considerations with the front end disk unit. In general, both the front end and TOPS10 may attempt to use the disk at the same time. The most frequent conflict occurs at system startup when the front end is using the disk at the same time that TOPS10 is running INITIA on all lines (there is a count of the times that TOPS10 tried to get the disk and found it busy; this normally rises quickly at system startup to about 40 and seldom changes thereafter. It is possible to do an assembly on the front end while timesharing continues on the -10 which might generate considerable conflict).

When the -10 attempts to get the disk and finds that it is in use by the front end, the requests is delayed (with considerable trickery to upper level code) and restarted when the drive can be gotten. Since TOPS10 may complete a seek for the front end drive and have the front end grab the disk and move it before the data transfer is started, it is possible that the drive will not be on the correct cylinder when TOPS10 tries to start the transfer. In this case, implied seek will be used since TOPS10 will not realize that the disk has been moved. This would also happen if TOPS10 got two different requests for the same cylinder and would decide that no seek is necessary when in fact, the front end had moved the heads.

2.0 DUAL PORT HANDLING

The dual port handling is very simple. It occurs only when the system attempts a data transfer on one channel and finds it busy. It then tries the other port. At no other time is the dual port facility used.

At system startup, the system reads the drive type and serial number from each drive on all channels. When the same serial number, drive type is found on two different channels, the disk is determined to be dual ported. One path (the first one found) is then the primary path and the second is the alternate path. When starting a transfer, the system will attempt to use the primary path. If that path is busy, it will then check for the alternate path; if that is available, the transfer is started. Otherwise, the request is placed in the channel queue for the primary channel.

3.0 EXCEPTION CONDITIONS

There are a number of possible error conditions that can occur while TOPS10 attempts to operate the disks. This section will attempt to list the error conditions, the circumstances under which they occur, and the action taken by the system. It will not attempt to show the 'flow diagram' of the error handling in the normal code. In general, the error processing is called as soon as possible after the error is detected.

3.1 Data Transfer Errors

These errors are detected on the completion interrupt for a data transfer (either read or write). These do not include the software detected error of the channel termination word not agreeing with the predicted channel termination word.

3.1.1 **ECC Correctable Error** - When a transfer terminates with an ECC correctable error the transfer stops after the sector in error. The software will reconstruct the data and restart the transfer at the sector following the sector in error.

3.1.2 **Non-data Error** - When a transfer completes that is not a data error (for example, a header error) the software will attempt to retry the transfer a number of times before recording the error as a hard error. The retry sequence is:

```
Retry 10 times
Recalibrate
Seek
Retry 10 times
Recalibrate
Seek
Retry 10 times
```

If after 30 tries the transfer still fails, the error is considered hard and an error is returned to the user. The data is recorded in SYSERR and a count of hard errors for this device is incremented.

If the count of hard errors reaches a system default (not 25), a message is given to the operator saying that there has been an excessive number of hard errors and the count is zeroed. The expectation is that the operator may want to set some hardware offline or call his field service rep to run a few diagnostics.

3.1.3 **Data Error** - If a data error (including header compare error) occurs which is not ECC correctable, then the system will retry the transfer and will use the offset register to vary the head position on each side of the track centerline. The retry sequence is:

```
Retry 16 times on centerline
Offset head to +200 microinches
Retry 2 times
Offset head to -200 microinches
Retry 2 times
Offset head to +400 microinches
Retry 2 times
Offset head to -400 microinches
Retry 2 times
Offset head to +600 microinches
Retry 2 times
Offset head to -600 microinches
Retry 2 times
Return to centerline
Retry disabling stop on error
Retry (every retry except the next to last is done
    with stop on error enabled, this enables the recording
    of the maximum of information in SYSERR).
```

For an RP04 the offset distances are twice the above. If the transfer is recovered at the offset position, the drive is left positioned at offset. If the next transfer on that drive is for that cylinder, it is first to be attempted at the same offset. If that fails, the head is returned to centerline and the entire above sequence is tried. If any seek is done, the heads will be on centerline (including transfers which cause the head to return to the cylinder on which an error was recovered at offset). If the device is not an RP04 or RP06 (MASSBUS drive), the error recovery is 10 retries of the sequence: read/write 10 times, recalibrate, seek.

On a hard non-recoverable error, an error is returned to the user and the system remembers the block number in error. When the file is subsequently closed, the system checks for a remembered block number. It starts reading from the bad block number+1 until it finds a good

sector (or 1000 sectors whichever is smaller). This gives the extent of the error region, which is then recorded in both the RIB of the file and the BAT block. If the program does not close the file after the error, but continues processing and hits a second error, the second error is lost.

3.2 Seek and Status Errors

In the attention summary register, on an interrupt, there may be attention interrupts for drives that were not transferring or seeking. In this case, the drive is going through some sort of status change, such as coming on line or going down.

3.2.1 **Medium-on-line = 0** - If medium-on-line is 0, the drive has just powered down. It is marked as such in the monitor tables.

3.2.2 **Drive Powered Up** - If medium-on-line (MOL)=1 and volume valid (VV) =0 then the drive has just powered up. The monitor will read the home blocks and check that the pack is the expected pack on that drive.

3.2.3 **Seek Incomplete** - On all seek errors, the error is counted and ignored. This will cause the data transfer to use the implied seek facility to perform the actual seek. If that implied seek fails, the data transfer will return an error and the appropriate retry sequence will be started(4.1.2).

3.2.4 **Hung Device** - Any time a seek or data transfer is started, the monitor starts an independent 'hung timer' that will fire in 7 seconds if the device has not responded with a completion interrupt for the operation.

If the failing request was a seek, then it is retried. If it was a data transfer, the monitor does a CONO to clear BUSY and set DONE. After this, the appropriate retry sequence for a data error is started. If 8 hung retries in a row fail, then the monitor will set the drive offline and tell the operator that it is offline (message is Inconsistent Status for Drive x).

3.2.5 **Rib Errors** - Every RIB error detected (along with every 'n' hard errors) is reported to the operator.

3.3 RAE Errors

On an RH10, Register Access Error(RAE) is ignored. The hardware will set the Selected Drive RAE at which point error recovery is started.

On the RH20, after every DATAO, a CONSZ on RAE is done, if there was an RAE, then it is cleared and the DATAO is retried. There is also a system count of RAE's per controller for the RH20's.

4.0 BAT BLOCKS

The BAT blocks provide a record of up to 63 errors on the disk. After each detected error (actually when the file is closed), the monitor will update the BAT blocks with the blocks in error and the type of error. It is possible that the BAT blocks will be filled to overflowing and there will be no room for additional entries. The system will leave bad blocks marked as allocated in the SAT table and thus avoid reallocating them. SYSERR will also complain when there are less than 5 entries remaining in the BAT block.

In general, there are some pathological cases where the total damage to a disk is unknown, but a reasonable PM of disks which includes checking SYSERR and DSKRAT and saving, refreshing (or replacing), and restoring packs with many bad spots will avoid difficulties.

5.0 DSKRAT

DSKRAT is a program which can be run to check for RIB errors and the disk space allocated as reported in the SAT table with the allocation as reported by the RIB's of the files on the pack. In general, it will find four kinds of errors:

1. RIB errors - A RIB is not consistent in format with a valid RIB. Lost blocks - These are blocks which are marked as allocated in the SAT but are not a part of any file.

2. Free blocks - These are blocks which are owned by some file on the system but are not marked as allocated in the SAT table. If one of these files is deleted, the system will get a BAZ STOPCD.

3. Multiply Defined blocks - These are blocks which are marked as owned in two or more files.

The safest procedure when a disk has significant problems in terms of free or multiply defined blocks is to BACKUP the pack, refresh it, and restore it.