The system console was the computer operator's main interface with SCOPE/Hustler. The console was controlled by the PP program DSD, which displayed operator-selectable information on the two physical displays. Commands were entered via the attached keyboard.

There were about 24 different logical screens (displays) that could be shown, each denoted with a letter. Any of the screens could be shown on either of the two CRTs that constituted the console display. DSD accepted commands from the keyboard. Commands consisting of two letters were commands to control which displays were active on the two cathode ray tubes. For instance, XB meant “Show the X display on the left tube, and the B display on the right tube”. (In fact, the XB combination was the one used most frequently.) One-letter commands meant to show the named display on the left tube, retaining the current contents of the right tube.

Characters typed in at the keyboard were echoed on the bottom of one of the displays—the left one, I believe. DSD implemented command completion; that is, it would automatically fill in the rest of a command once that command could be uniquely determined from the characters already typed at the keyboard. When a command was determined to be completed, it was displayed in an rippling manner that is difficult to describe. When a complete command was entered, DSD would update portions of the echoed command line more frequently than others. The characters being displayed more frequently would be displayed more brightly. Different characters were brightened successively at different times, causing the rippling effect. Pressing the carriage return key would cause the command to execute.

DSD allowed you to type only characters that might result in a syntactically correct command. Any other character would cause “ILLEGAL ENTRY” to be flashed on the screen. ILLEGAL ENTRY occurred fairly often when inexperienced systems programmers tried to type characters that had already been filled-ahead by DSD’s command completion mechanism.

In practice, most use of the console related to mounting magnetic tapes. The system didn’t require a lot of active intervention on the part of the operators. Systems programmers probably made more extensive use of the console when they had the system for “Systems time”.

### Console Displays

The following displays were available to console users. Most were used infrequently, even by systems programmers.

| A System dayfile, including accounting messages and hardware diagnostic messages. Usually lines scrolled by very quickly. This display was useful to systems programmers but rarely to operators. |
### Control point display
This was nearly always up on the right screen. It listed control points with their status: Clear, Next, or program running. For occupied control points, a lot of information was crammed into a few lines. Included was user id, current job cost, CPU status (RCL=periodic recall, AUTO=automatic recall, WAIT=wants the CPU, CPU A=current has the CPU), latest dayfile message, number of PPs currently at that control point, and a host of others. CPU programs could put up special flashing B-display messages which caused the job to be paused until the operator took action. This capability was normally used only by certain programs designed to run as control point jobs. (Control point jobs were those that were explicitly run at a given control point and did not leave the control point until they terminated. The system backup program SELDUMP is an example.) The command B=n where n<>0 meant to display only control point jobs and Hustler jobs that had been at a control point for over 5 seconds. (Or is that n seconds?)

### Central memory display
This was the display showing the 32 words from central memory, using absolute addressing. Each word was displayed as 5 groups of 4 octal digits, plus display code. C had a base address associated with it that was remembered even while the display was not active. The C display was organized into 4 groups of 8 words. The command

- \( \text{C}x, \text{addr} \) (0 \( \leq x \leq 3 \)) set the base of the xth group to the octal address addr.
- \( \text{C}4, \text{addr} \) set the base of the entire display.

The + and - keys, when entered on an otherwise blank line, allowed system programmer to change the base of the display and hence page through memory.

### Equipment status display
This display contained one line for each device on the system. Each included the following information: Device number Type, a two-character identifier: FD for front end computer, AY or single density 844 disk drives, etc. ON/OFF status Channels attached to a device Destination/route (for printers, etc.) Visual pack id (for disks)

### File Name Table display
Each file in the FNT was displayed, with a designation of L for non-permanent local file, C for common file, and P for permanent file. Each TTYxxx file listed at MANAGER’s control point was the TTYTTY file for user xxx.

### Central memory display
This was a very similar to the C display but with words shown as 4 groups of 5 digits, plus display code. This was more conducive for viewing the words as instructions.

### Central memory-resident FNTs for input and output queue files
This display became obsolete upon implementation of permanent file I/O queues.

### Status of ARGUS devices
For instance, it contained the sequence number of the job currently printing on a model 512 line printer. Contained HELP ME if the printer was out of paper or had other trouble. When there was something useful on this display, ARGUS would put up a B display message SEEÂ -J-Â DISPLAY. The J display was named “Mickus Mouse Devices” for obscure historical reasons.

### Control statements
At a given control point. This showed the current PRU (disk sector) of control statements for a job as long as it was at a control point. Unless the job was already executing the last control statement that happened to be in that PRU, this allowed you to look ahead a few control statements to see what the job was going to do next. Sometimes the K display was used by operators to allow them to prefetch reels of tape for a tape-hungry job, as the Visual Reel Numbers were usually present in control statements. However, generally the K display wasn't used much.

### User programmable display
A CPU program running as a control point job could request to write to the L display. For instance, there was a CPU program named PAGE that displayed coded files on the console using the L display. The Operations staff maintained a user library named L*CG (for Chuck Gillengerten) that had some programs that displayed status information on the L display; also, one of these programs displayed a picture of Mickey Mouse. L stood for Left; the L display, when used, was generally shown on the left screen. There was a similar very rarely used R display that was intended to be shown on the Right screen if you wanted to take over both screens from a CPU program.
Peripheral Processor display. M had one line for each of the physical PPs in the system, with a display of the program running (if any) and the PP's input and output “registers”. (These were not physical registers, but small blocks of memory used by software to communicate with the PPs.) The M display was fascinating to watch, but generally used only during testing or problem resolution.

CPTask display. This was the central processor task analogue to the M display. As mentioned elsewhere, Michigan State had rewritten some of the CDC PP routines to run in the CPU for better performance. These were then dubbed CP tasks.

Listing of canned Operator Responses available for the MDROP command. This command had the syntax (po)MDROP,msg#,operator comment. where po was the two-character pool ordinal of a job to drop. MDROP would kill a job and place in its dayfile a message plus an operator comment. The O display listed the English messages that corresponded to the message numbers entered by the operator. The O display was rarely used because the MDROP command was rarely used, and the operators probably had the commonly-used message codes memorized anyway. The O display was one of the few completely static displays. The letter O would probably have been the first one to be recycled if a need had arisen to implement new displays.

Tape status display. This contained information for each tape drive, including the Visual Reel Number (VRN) of the tape mounted, whether a write-enable ring was present, the pool ordinal of the job that owned the drive, other information. The P display was used rather often by operators.

CPTask debugger display. This was used by systems programmers to “expand” a CPTask to show its registers during debugging.

User programmable display, intended to be shown on the Right screen. See the L display.

PP Interpreter display. This had a list of each of the physical PPs in the system, like the M display. When a PP program was being interpreted, its P and A registers were displayed here. Unfortunately, I can't remember anything about the PP Interpreter.

PP Interpreter display. Contained a list of PPs subject to interpretation. Normally, of course, this list was empty.

Next Sequence Number display. For each E/I 200 site, and for interactive and DISPOSEd jobs, there was a line containing the sequence number to be assigned to the next job coming from that source. Very infrequently used.

System status display. This was a catch-all display that normally ran on the left screen during production. The top part of the screen was dedicated to displaying available resources and the load on the system. This included: free disk space the number of free FNT entries RBT FL the number of jobs on the input and output queues Below that were listed all the pool jobs, one per line. The + and - keys scrolled this part of the display. Batch jobs were listed first, because they might ask for tapes and tape requests came up on this display. Also user jobs paused by user request were indicated here. The command X=n where n<>0 caused only jobs waiting for operator action to be displayed. Since tape mounting was the most common task of an operator, the X display was the one display to watch as an operator.

DSD command directory. The syntax (but alas, not the meaning) of each console command was listed here in alphabetical order. Typing a few characters at the console while the Y display was active caused the Y display to move to the part of its display that started with the characters typed. DSD's command fill-ahead made this display largely unnecessary. It would have been more useful if a description of each command had been included. However, it was somewhat useful because sometimes you knew the command that you wanted, but could not recall the parameters or their order (control point, or CM address, or pool id, etc.)

Index to displays. Z contained a one-line description of each lettered display.

Here is a very abbreviated list of DSD commands.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGE</td>
<td>Used to clear a condition about which DSD is complaining. The bottom of the B display will contain the error text. This was supposed to be used with caution, as the warning condition being cleared might well be a serious system problem.</td>
</tr>
<tr>
<td>ATTENDED</td>
<td>Set the attended flag, which stated that the system is running with operators on duty. This affected dayfile operation. The attended flag was queried by LOGIN, MENU, and a few other CPU programs.</td>
</tr>
<tr>
<td>UNATTENDED</td>
<td>Cleared the attended flag.</td>
</tr>
<tr>
<td>AUTO</td>
<td>Brought up MANAGER at control point 1 (if control point 1 was clear) and if this was the main system in a multiple machine environment, brought up ARGUS at control point 2. (After the acquisition of the Cyber 750, the 750 and the older 6500 ran for some years in a two-machine cluster.) AUTO also set control points 3-15 to NEXT, meaning that Hustler could schedule jobs to run at those control points.</td>
</tr>
<tr>
<td>BLITZ</td>
<td>Cleared all control points, the opposite of AUTO. Useful for harmlessly freezing the system if something nasty was happening, such as disk space going into a tailspin.</td>
</tr>
<tr>
<td>DIRECT,seqnum, {I</td>
<td>O</td>
</tr>
<tr>
<td>EVICT,seqnum, {I</td>
<td>O</td>
</tr>
<tr>
<td>UNLOCK BY xxx TO comment.</td>
<td>“Unlocked” DSD to allow certain especially dangerous commands. xxx was your initials; they were recorded in the system dayfile. This provided some accountability. comment was an English description of why you were using dangerous commands.</td>
</tr>
<tr>
<td>LOCK</td>
<td>Undid an UNLOCK command.</td>
</tr>
<tr>
<td>addr,n,value.</td>
<td>Set the n-th byte of central memory in at word absolute octal address addr to the octal value value. A byte was 4 octal digits, and there were 5 bytes per word, so n could be from 0 to 4. Sample usage: 5727,3,7700. This is an example of a command that required UNLOCK.</td>
</tr>
</tbody>
</table>