

# CRT Image Page

## *Local Station application*

Dec 28, 1989

### *Overview*

Local station user interaction occurs via the local console. Each 15 Hz cycle, the console data is polled for any changes. The raw console input data consists of 16 switch inputs from the pushbuttons, 14 lamp outputs as pushbutton indicators, an 8-bit keyboard character code, and an 8-bit knob counter. A 4800 baud serial interface is used to access these data under interrupt control. The Console Task processes the inputs for the system. Some of the pushbuttons are handled with mutually exclusive logic. An application usually keys on the light activity for actions it makes in response to pushbutton activity. Note that a station with no local console attached can still run application pages, as the display interface resides on the Crate Utility board. Only the composite video and the serial send and receive cables are attached to the local console hardware.

### *Display layout*

```
G CRT IMAGE      10/31/89 1336
SYSTEM<      >
```

The system's node# whose console is to be monitored is entered in hex. Upon interrupting, the display changes to the current local console display appearance of the system under study. Interrupt again to return to the original page.

### *Basic approach*

This particular application presumes to know a lot about the system—more than most application pages. This is in part because it was written without any special hooks provided by the system to support its implementation. Some of the features supported are fortuitous.

Console-related input data are only 4 bytes and are stored in the system global variables. Two are for the switch input data, one is for the keyboard, and one is for the knob. The switch input data is automatically processed to turn on related lights behind the switches. These two light bytes can be read also.

The basic plan is to request memory data for the display image buffer in the remote station. This buffer is 512 bytes of ascii for all the characters on the screen. (It isn't the real video buffer because that uses a 6-bit ascii encoding. The reason for having an image buffer is to support "reading" from the display.) The image buffer is requested to be returned at 15 Hz. Any characters which have been changed are then updated on the local display.

Additional data requested along with the display image buffer is the remote cursor position, the remote lights, and the remote knob change. The program logic proceeds as follows:

The local keyboard input character is checked. Whenever a new character has been typed locally, it is sent to the remote system, which will in turn respond to it just as if the character were typed on the remote console (assuming it has one). A special case is made for activating the keyboard interrupt on the remote station. If a control-Q is typed locally, then an ESC character is sent to the remote station, which is interpreted the same as a keyboard interrupt. This must be done since a keyboard interrupt used locally will produce local actions, such as returning to the local display or exiting the page. (If that is done inadvertently, merely recall the CRT Image page, and the remote screen will re-appear just as it was.)

If the location of the remote cursor changes, the local cursor is moved to keep pace with it. This allows the local user to monitor a remote user's typing and cursor motions.

If the remote lights change, the local lights are set to match them. This allows the local console lights to indicate what a user at the remote console is doing. Of course, such light changes at the remote console can arise from the actions of the application that is being run by the local user as well.

If the local lights are changed, the remote lights are set to match them. This allows the local pushbuttons to work to control the remote application.

If the local knob is adjusted, the same count of knob clicks is passed to the remote station. This allows the local knob to work.

*To summarize:*

The monitoring of the remote station's display image buffer and cursor location allows monitoring remote console activity.

Local typing, except for ESC, is sent to be acted upon remotely. Local switch changes that result in lights being lit are passed to the remote station. Local knob action is passed to the remote station. The combination of these actions allows control of a remote application.

Currently the cursor control (etch-a-sketch) buttons do not work on the remote station because the local knob counter is continuously read into the global copy.