

# LIVING COLOR FRAME SYSTEM: PC GRAPHICS TOOL FOR DATA VISUALIZATION

Long V. Truong

National Aeronautics and Space Administration  
Lewis Research Center  
Cleveland, Ohio 44135

## ABSTRACT

Living Color Frame System (LCFS) is a personal computer software tool for generating real-time graphics applications. It is highly applicable for a wide range of data visualization in virtual environment applications. Engineers often use computer graphics to enhance the interpretation of data under observation. These graphics become more complicated when "run time" animations are required, such as found in many typical modern artificial intelligence and expert systems. Living Color Frame System solves many of these real-time graphics problems.

## INTRODUCTION

Lewis Research Center is NASA's lead center in space power technology. Our technologies include the use of artificial intelligence (AI) for power management and distribution. Specific projects in AI are power scheduling, fault detection and isolation using expert systems [1,2], neural networks, and fuzzy logic.

Power AI projects typically require real-time computer graphics that involve graphics image animation, such as alternating video colors and intensities and moving objects during run time. In some of our applications these images display the system hardware diagram and its real-time status (Fig. 1). The system process control variables must be displayed along with these graphical "frames." It is also desirable to be able to zoom in and out at any level of detail of these object images (hardware components) and to have the personal computer (PC) buzzer sound when operator response is urgently requested.

Because considerable effort has been required to generate these real-time graphics applications, the need for an easy graphics generation and management tool became apparent. Although there are many sophisticated commercial graphics tools, such as Freelance (trademark of Lotus Development Corp.) and Quattro Pro (trademark of Borland International Inc.), none support or provide adequate run-time software for graphics manipulation and/or disclose their graphics file formats for user implementation of real-time applications. Because such a tool was not available for PC use, we designed a new tool to eliminate custom graphics programming and to simplify the development of our application software. It is called the Living Color Frame System (LCFS).

This paper introduces the LCFS and presents an example of real-time graphics applications in the area of monitoring and diagnostics for an electromechanical actuator system (Fig. 1).

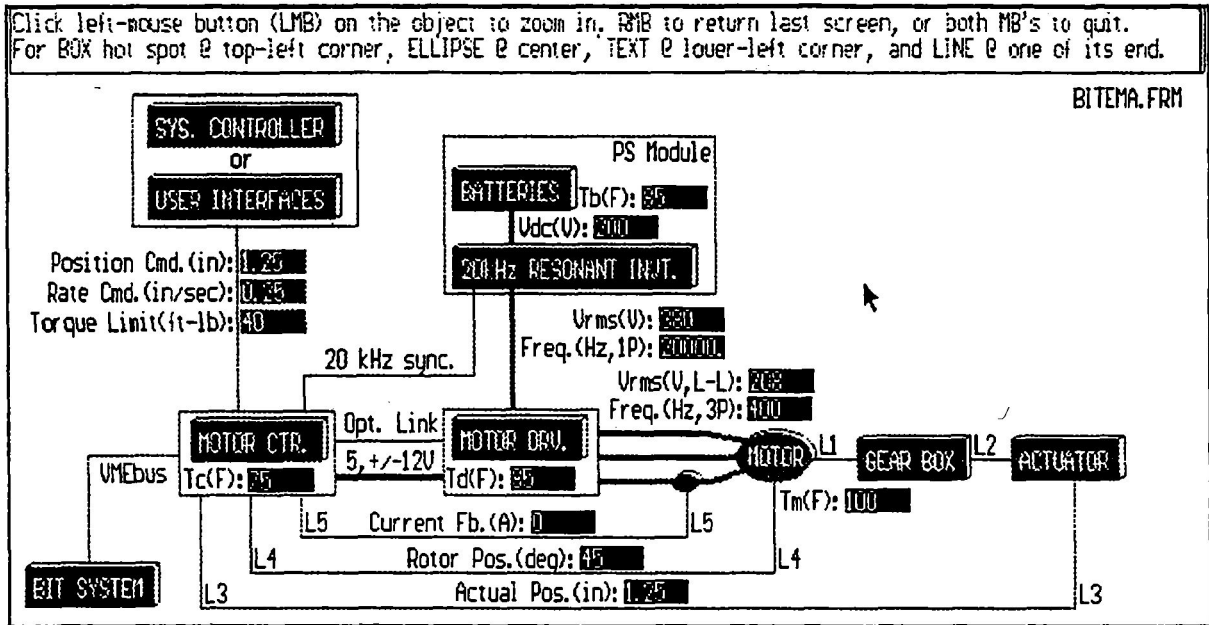


Figure 1 Example of normal display for typical electromechanical actuator system.

## LIVING COLOR FRAME SYSTEM

### I. ABOUT THE SYSTEM

Living Color Frame System, broadly speaking, is a PC software tool for real-time graphics applications. It is highly applicable for data visualization in virtual environment applications. This new graphics tool, LCFS, is user friendly, having a graphical interface, mouse-driven controls, and on-line instructions. Computer graphics screens or "frames" can be conveniently drawn, logically linked, and then dynamically recalled for display in real time by using mouse commands. Run-time software for custom video and sound effects on these frames are provided, such as display of monitored data, manipulation of graphics images, and/or sounding of a PC buzzer. In some of our applications these frames can be visual aids for managing systems. For monitoring, diagnosing, and/or controlling purposes circuit or system diagrams can be brought to *life* by using designated video colors and intensities to symbolize the status (feedback from sensors) of hardware environments. With LCFS, custom graphics programming is largely eliminated, allowing software developers to focus fully on their applications' contents. Thus, LCFS is suitable for a wide range of real-time graphics applications.

### II. ITS SOFTWARE STRUCTURE

Figure 2 shows a simplified block diagram of LCFS's software structure. It basically describes the integration of the software modules (oval shapes) and the flow of information (rectangular shapes) throughout the system. In the next three sections LCFS's software structure is discussed in terms of user application software, LCFS's built-in software, and frame data bases, respectively.

#### 1. User Application Software

As shown in Figure 2, user application software is divided into two modules: data communication and data processing. They are described here.

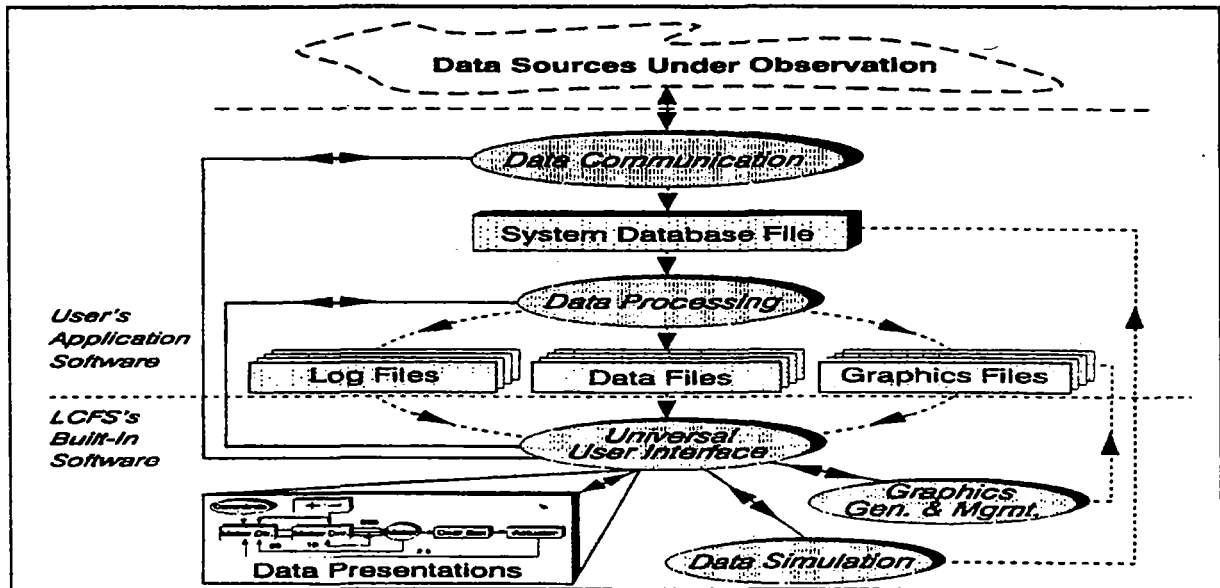


Figure 2 Simplified block diagram of LCFS's software structure.

**1.1. Data communication (DC) module.-** Because each application is unique, users are responsible for the function of the DC module (Fig. 2). Its primary task is to transfer raw monitored data from their sources (Fig. 2) to the system data base file (see section 3.1 for its format and creation). This executable module must be named "DATACOMM.EXE" in order for the system to recognize it as a "child" process during the data updating procedure. Optional routines for hardware control should also be implemented here.

**1.2. Data processing (DP) module.-** For each "static" graphics screen or file generated by the graphics generation and management module (Fig. 2), users have an option to develop a run-time interfaced program for linking and displaying additional information. Most of the software routines for this option are provided, such as superimposing numerical monitored data, manipulating graphics images to reflect the current data, and sounding the PC buzzer to get the operator's attention. These interface programs (data processing module, Fig. 2) must have the same names as their "related" graphics files for the system to acknowledge them. They can be added or deleted without interfering with LCFS's built-in software. Their typical tasks would be

- To read (routines are provided) data from the system data base file
- To analyze (user application rules) the data
- To select and prepare (routines are provided) frame information for display. Thus, run-time graphics, numerical data, and /or text messages are dynamically constructed for display in this step. See section 3 for implementations.

## 2. LCFS's Built-In Software

As shown in Figure 2, this permanent part of the system software is divided into four modules: universal user interface, data presentations, graphics generation and management, and data simulation. They are discussed here.

**2.1. Universal user interface (UI) module.-** This unique user interface module (Fig. 2) controls the entire system operations, manually or automatically. In automatic operation mode it continuously updates frame information by repeatedly executing three responsible tasks: data communication, data processing, and data presentation. On the other hand, in manual operation mode (default), it allows users random access

to many convenient built-in options. At present these options are as follows:

- Display UI's instructions (click the right-mouse button).
- Zoom in or out (click the left-mouse button on or off the frame objects).
- Set automatic operation mode (press the A key).
- Set manual operation mode (press the M key).
- Turn off the PC buzzer's sound (press the S key); see section 3.3 for activating the PC buzzer.
- List run-time messages (press the L key); see section 3.2 for implementations.
- Generate or modify graphics screens or files (press the F key); see section 2.3 for usage.
- Simulate data (press the D key); see section 2.4 for usage.
- Exit LCFS (press the Q key).

**2.2. Data presentations (DPRE) module.-** This module (Fig. 2) is responsible for refreshing frame information. Its functions are as follows:

- Spawn (executes as a child process) the data communication module to update monitored data in the system data base file (see section 3.1).
- Spawn the data processing module to prepare frame information for display.
- Display frame information.
- Return to UI module.

**2.3. Graphics generation and management (GG&M) module.-** This module is a graphics utility [3,4] for custom generation and management of computer graphics screens. Drawing options are fully displayed on a single menu and are executed by using mouse commands for fast and easy operation of the tool. Once an option is selected (click the left-mouse button on the desired menu item), users will be assisted with the on-line instructions to complete the process. Figure 3 shows some of the typical drawing elements. Its custom graphics file format (see section 3.4) is opened for user applications.

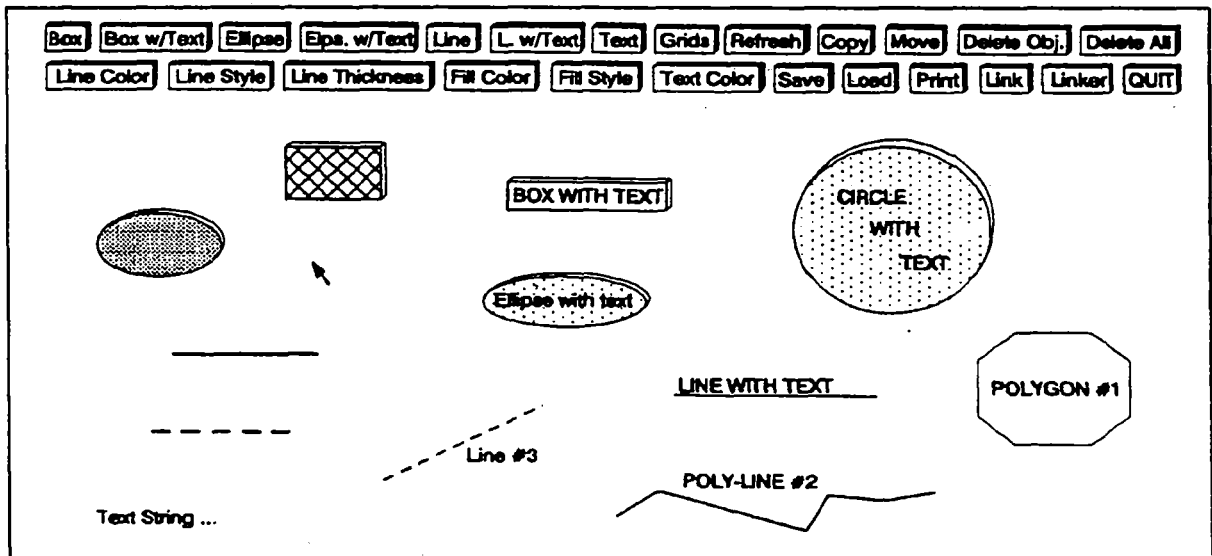


Figure 3 Samples of basic drawing elements.

**2.4. Data simulation (DS) module.-** This module (Fig. 2) allows users to modify the contents of the system data base file (see section 3.1) from the keyboard without programming. Thus, raw data can be conveniently simulated. Users can use this advantage to develop and test their application software independently from the target system.

### 3. Frame Data Bases

As mentioned in section 1.2, frame data bases or files (Fig. 2; can be printed out by any text editor) are needed for holding frame information. Four types of data files were designed for the purpose. They are discussed here.

**3.1. System data base file.-** This "master" data file (Fig. 2), named "SYSDBASE.DAT," was designed for holding any numerical data that users desire to access during the data processing procedure (see section 1.2). Routines for run-time creation of this file (mentioned in section 1.1) are provided. Figure 4 shows a typical listing of the file.

```
2.25
5.00
125
...
45.75
```

Figure 4 Typical listing of system data base file.

**3.2. Frame log files.-** By design, frame log files (Fig. 2) are created (routines are provided) during data processing procedures (see section 1.2) for holding run-time messages. These messages are optionally implemented by users to instruct operators in difficult situations, such as in an alarm situation. These files are designated with a common file extension, .LOG, and have the same names as their related frame graphics files. An example of a typical frame log file (refer to Fig. 8 for better understanding of the text) is shown in Figure 5.

```
Fault Detection:
  ACTUATOR is not at its commanded position within the given time.

Fault Isolation:
  A possible failure in one of the mechanical links between MOTOR and
  GEARBOX, or GEARBOX and ACTUATOR results in an immobilized
  actuator.

Corrective Suggestions:
  1. Shut down the power immediately.
  2. Check for visible damage between the interconnections of MOTOR,
  GEARBOX, and ACTUATOR. Repair the damage, if found. If the
  interconnections appear normal, proceed to the next step.
  3. Disconnect the couplings between MOTOR, GEARBOX, and ACTUATOR.
  Then, independently check for proper operations of MOTOR, GEARBOX,
  ACTUATOR, and the mechanical links (L1 and L2) between them.
  ...
```

Figure 5 Typical listing of frame log file.

**3.3. Frame data files.-** Being the same as frame log files, frame data files (Fig. 2) are also generated (routines are provided) during the data processing procedure (see section 1.2) and named after their related frame graphics files with a common file extension, .DAT. They are designed for holding run-time numerical data to be displayed along with color and sound instructions for highlighting purposes.

Figure 6 shows a typical listing of the file. Each record (112 bytes max.) includes seven fields and spaces as field separators.

```

021 132 15 0 4 1.250000 Position Cmd.(in.):
010 162 15 0 4 40.00000 Torque Limit(ft-lb):
...
190 263 12 1 4 55.00000 Current Fb.(Amps):

```

Figure 6 Typical listing of frame data file.

Individual fields (from left to right) are explained as follows:

- Fields 1 and 2 contain the position (x-y screen coordinate) of the text label (field 7) for the numerical data (field 6) to be displayed. Since the label is created by the GG&M module, this information (fields 1, 2, and 7) can be extracted directly from the related frame graphics file.
- Field 3 contains the background color (2 bytes max.; computer standard color codes (0-15)) of the numerical data (field 6) to be displayed. Designated background colors can be used for highlighting ranges of data.
- Field 4 contains the sound flag (1 byte, 0=disable, 1=enable). This feature is excellent in getting the operator's attention by sounding the PC buzzer. It works nicely with the color option in field 3.
- Field 5 contains the number (2 bytes max.) of digits of the numerical data (field 6) to be displayed, for sizing of the display area.
- Field 6 contains the numerical data (any real number) to be displayed.
- Field 7 contains the label (mentioned in fields 1 and 2; 80 bytes max.) of the numerical data (field 6) to be displayed.

**3.4. Frame graphics files.-** A custom frame graphics file format (designated file extension, .FRM) and run-time supported software for graphics manipulation were designed and developed for solving real-time graphics problems. Unknown graphics file format and the lack of software support in many commercial graphics tools (e.g., Lotus Freelance Plus and Quattro Pro) had made it difficult or impossible to access the graphics knowledge base from user programs in many typical real-time graphics applications.

Figure 7 shows a typical listing of a frame graphics file that allows easy access and tremendous saving of computer memory in both run time (random access memory) and storage (hard disk). Each drawing object is saved as a record (125 bytes max.), including 13 fields and 12 bytes of spaces as field separators.

```

T   noname 21 132  1  1 10 0 1  1 10 15 Position Cmd.(in.):
T   noname 10 162  0  0 10 0 1  1 10 15 Torque Limit(ft-lb):
T   noname 190 263  0  0 10 0 1  1 10 15 Current Fb.(A):
...
L   L1 429 227 451 227 10 0 1  1 10 15 L1

```

Figure 7 Typical listing of frame graphics file.

To start the tool, simply type the command LCFS at the DOS prompt [5] and hit the Enter key. After successful completion of the command, first-time users can click the right-mouse button for instructions. Thus, users are assisted with the custom on-line instructions throughout the operations. Other usage instructions (installing software, using supported software, etc.) are provided with the software package.

## REAL-TIME GRAPHICS APPLICATION EXAMPLE

As previously mentioned, the application example given pertains to the area of monitoring and diagnosis (M&D) of an electromechanical actuator (EMA) system. A simulated top-level diagram of such a system was shown in Figure 1.

Because of the nature of the electrical power system, such as shown in Figure 1, electricians normally use hardware schematics as "roadmaps" for troubleshooting problems. What could be better than having "live" schematics on the computer screen for M&D? An essentially live schematic can be achieved by using computer images with designated colors, video intensity, and sound to symbolize the component hardware status (real-time feedback from sensors), and therefore the status of the system itself. Table 1 defines these image representations or models for the example below. Note that only black-and-white shadings are shown to represent the actual colors in illustrated figures.

**Table 1** Designated Image Representations of Colors and Sound Denoting System Status.

Image Representations	Status of Hardware Components/Modules
Green	NORMAL, no failure diagnosed (requires no attention)
Yellow	WARNING, a "side effect" or minor failure, temporary or permanent, diagnosed (usually requires attention)
Red and beeping	ALARM, a "hard" or serious failure diagnosed (requires immediate attention)

For the application example an alarm situation for the EMA system was simulated and is shown in Figure 8. Note the differences in "colors" (different shadings in black and white) from the normal status display of the system in Figure 1. By looking at this color graphics presentation (Fig. 8) and having the knowledge from Table 1, operators are quickly aware of the alarm situation without losing time to analyze the data (numerical numbers), especially when minutes and seconds count:

A fault was detected and isolated. Suspected failure components are highlighted in red (solid shading) and beeping, namely the MOTOR, L1, GEARBOX, L2, and ACTUATOR.

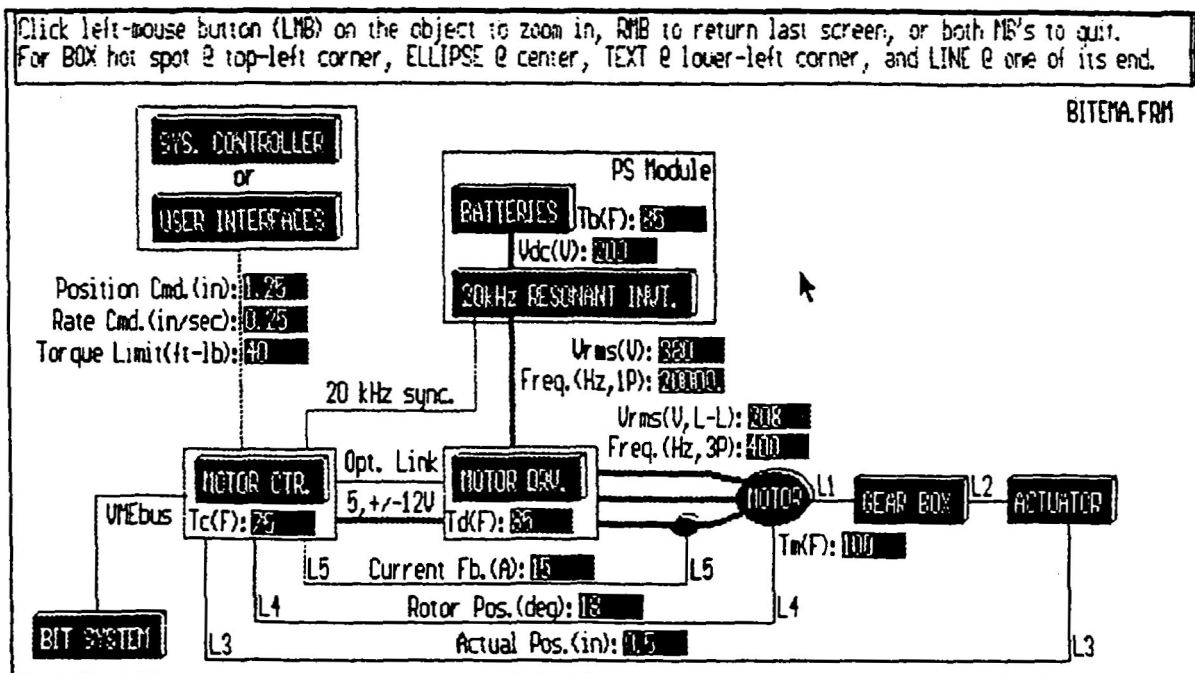


Figure 8 Example of alarm display for simulated electromechanical actuator system.

Diagnostic text messages for further explanations of the situation can also be implemented as shown earlier in Figure 5. To illustrate the zooming (in/out) capability, a second-level detail of the MOTOR DRV. module was implemented (created and linked by the GG&M module) and is shown in Figure 9 (obtained by clicking the left-mouse button on the MOTOR DRV. image (Fig. 8)).

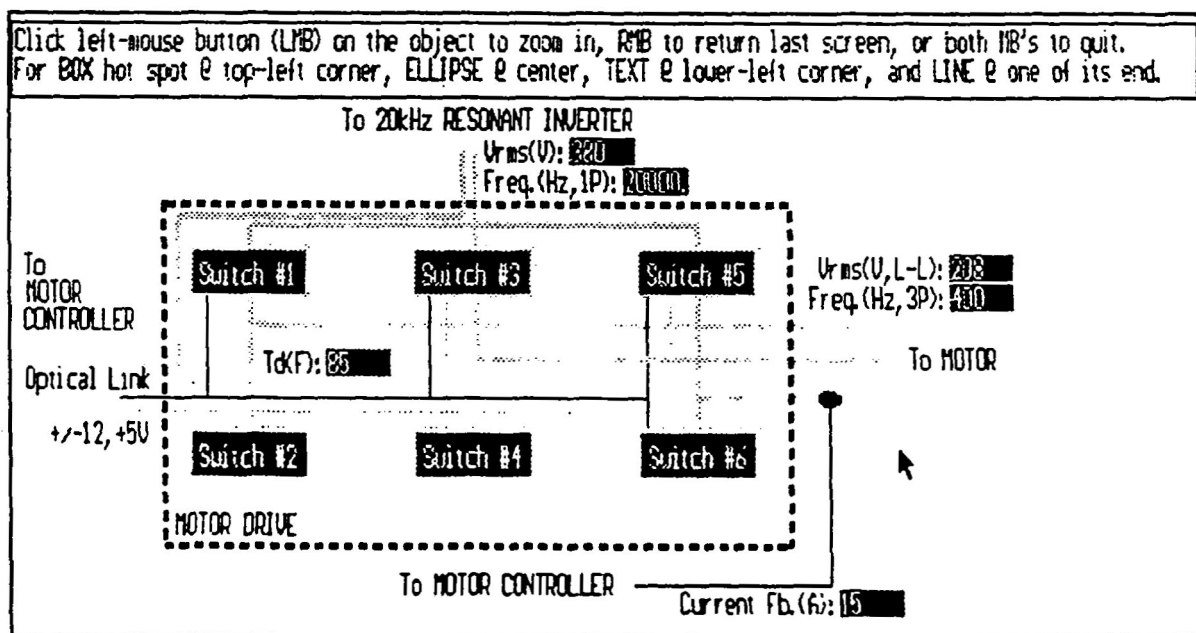


Figure 9 Example of normal display for second-level detail of MOTOR DRV. (Fig. 8).



As you can see by comparing Figures 1 and 8, *colors* are essential for adding information to the images in such dynamic applications. Thus, real-time data visualization and highlighting are key contributions of these new graphics tools.

## **SUMMARY AND CONCLUSIONS**

LCFS is an excellent tool for a wide range of real-time graphics applications, especially in data visualization for system health monitoring and management. The tool is user friendly, having a graphical interface, mouse-driven controls, and on-line instructions. A virtually unlimited (or limited only by the hard disk) number of frames can be easily created, logically wired, and then dynamically recalled for display in real time by using mouse commands. Its convenient built-in graphics tools, data simulation utility, and support of run-time software truly allow users to focus fully on their applications' contents. The system is easy to maintain and modify because of its software structure: modular and independent from the users' data bases. And finally, with the open (versus proprietary for most commercial graphics tools) format of the frame graphics file, users have the opportunity to customize their own application software. Thus, this new approach for real-time graphics applications suggests new opportunities for commercial markets.

## **REFERENCES**

1. Truong, L., et al.: Autonomous Power Expert Fault Diagnostic for Space Station *Freedom* Electrical Power Testbed. Proceedings of Third Annual Workshop on Space Operations Automation and Robotics (SOAR 1989), NASA CP-309, 1989, pp. 181-186.
2. Walters, J., et al.: Autonomous Power Expert System. The 1990 Goddard Conference on Space Applications of Artificial Intelligence, J.L. Rash, ed., NASA CP-3068, 1990, pp. 147-156.
3. Truong, L.: PC Graphics Generation and Management Tool for Real-Time Applications. NASA TM-105749, 1992.
4. Truong, L.: LCFM—Living Color Frame Maker: PC Graphics Generation and Management Tool for Real-Time Applications. NASA/COSMIC, The University of Georgia, Athens, GA 30602-4272, (706) 542-3265, 1992.
5. Microsoft MS-DOS Version 5.0, User's Guide and Reference. Microsoft Press, Redmond, WA, 1991.