SLAC-PUB--3638
DE85 013369

DISSECTING THE COW*

ERIC LINSTADT

Stanford Linear Accelerator Center
Stanford University, Stanford, California, 94305

INTRODUCTION

The COW, or Console On Wheels, is the primary operator interface to the SLC accelerator control system. A hardware and software description of the COW, a microcomputer based system with a color graphics display output and touchpanel and knob inputs, is given. The ease of development and expandability, due to both the modular nature of the hardware and the multitasking, interrupt driven software running in the COW, are described. Integration of the COW into the SLCNET communications network and SLC Control system is detailed.

HARDWARE DESCRIPTION

The hardware of the COW (Fig. 1) is composed of both commercially available and SLAC developed IEEE-796 (Multibus) modules. Basing the design on a standard interface allows for both the rapid development and the incremental expansion of the hardware.

The central intelligence in the COW is provided by an Intel iSBC 86/12 single board computer with 64K of RAM. Additional memory is provided by a 256 K RAM board. Firmware on board the iSBC 86/12 in 8K of EPROM is minimal, as the operating software for the COW is downloaded over the SLCNET network whenever a SLC Control Process (SCP) is initiated on the VAX[1,2].

Communications with the VAX are provided by a Computrol Multibus Megalink board, an intelligent Multibus SDLC DMA channel, through a Coherent Systems FSK modem. This channel provides a 1 Mbaud serial link, over which operator requests and messages from the VAX may be exchanged. This hardware, and its SLAC developed firmware, are identical to that used in all SLC sector micro clusters.

Graphics information is provided by a color monitor driven by a Matrox RGB-Graph 64/4 graphics controller. This board provides 512 × 512 × 4 resolution, one plane each for red, green and blue, with the fourth plane modified to allow an overlaid blink attribute. An identical board is used to drive a monochrome monitor which provides the display for a touch panel system.

Operator inputs are handled primarily by a TSD Display Products touchpanel interfaced to a SLAC produced Multibus board, which incorporates several other system functions.

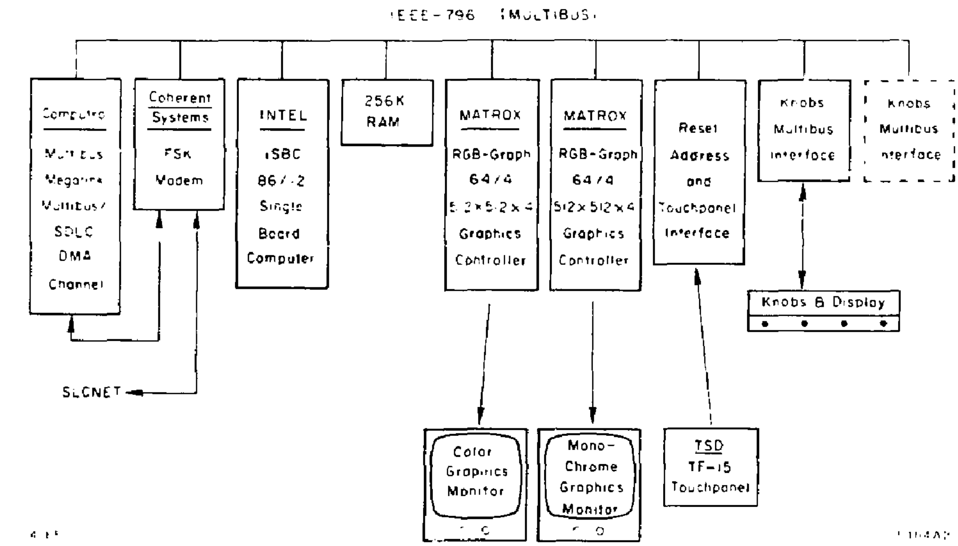

Fig. 1. Hardware block diagram of the COW.

* Work supported by the Department of Energy, contract DE-AC03-76SF00515.

Poster paper presented at the 1985 Particle Accelerator Conference, Vancouver, B.C., Canada, May 13-16, 1985

Included are Multibus reset functions and SLCNET address decoding, as well as neatly packaging system interconnection requirements.

The COW Knobs Multibus Interface, another SLAC board, provides counters, the interface to incremental shaft encoders, and their displays, for use as programmable knobs. Drivers for miscellaneous annunciators, used to acknowledge the transmission of touchpanel coordinates to the VAX, are also on this board.

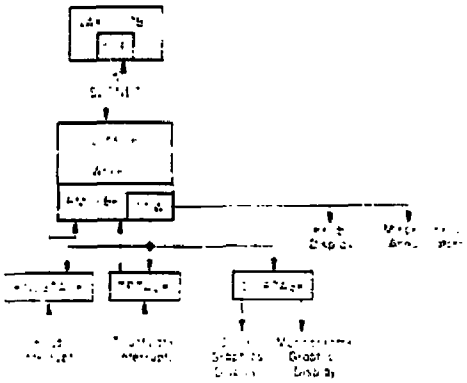

Fig. 2. COW Software and Hardware Interconnection.

SOFTWARE DESCRIPTION

The applications program in the COW runs as a job under the Intel iRMX-86 operating system. A multitasking, real time operating system, it provides system services within the COW, allocating system resources to subtasks, manages interrupt requests, and handles communication between tasks. Communications to and from the VAX over SLCNET are handled by an extension to the operating system. The presence of a sophisticated operating system allows the benefits of software modularity to be obtained, allowing the development of new features within a well defined environment, and sharing the benefit of sophisticated tools. For example, a symbolic debugger, MD86, running as a separate job in the COW over SLCNET, shortens the software debugging cycle, aiding rapid code development.

The applications code in the COW is written in PLM-86 and ASM-86 assembly language. After discovering and initializing its hardware configuration, the COW program creates two subtasks to handle touchpanel and knob interrupts.

These tasks communicate the coordinates of touchpanel pushes or the amount of knob rotation to the VAX, while the COW awaits messages from the VAX. Messages might be commands to actuate an annunciator (an LED or a beeper), or contain a character string to be written on a knob display, and these are taken care of immediately by the COW task. Messages containing information for the graphics displays are passed to a display task, DISPTASK.

The bit mapped raster graphic displays in the COW appear to the VAX as vector refresh devices. This allows the association of picture elements with a segment id, and these picture elements may be subsequently redrawn or erased by telling the COW to include or omit that particular segment. This saves both the computation time and the data communications overhead associated with the regeneration and retransmission of frequently drawn displays. DISPTASK performs all of the necessary work associated with segment management, and then processes the received graphics information and writes it to the graphics controller.

Displays are prepared on the VAX by calls to the Unified Graphics System[3]. Intermediate form, device independent graphics data are passed to a selected device driver. The only picture types available in device independent form are individual pixels, endpoints of line segments, and characters. All coordinates are in floating point format. The graphics controllers used in the COW are relatively primitive, and only allow the manipulation of individual pixels. Translation of device independent forms to graphics controller commands takes place in two steps. The device driver on the VAX converts the floating point coordinates to fixed point, and transmits them to the COW. The format of the graphics messages to the COW is shown in Table I. An assembly language subroutine called by DISPTASK then 'connects the dots', filling in the intermediate pixels of line segments and drawing the characters.

Table I. Structure of a Graphics Segment

Display Number
Segment ID
Erase Flag
Include / Omit Flag
Color and Point/Line/Character Identifier
Pointer to the Next Color and Point/ Line/Character Identifier
X Coordinate
Y Coordinate
Additional X,Y Coordinate Pairs or Number of characters in the following String
Character String
.
.
.
Next Color and Point/Line/Character Identifier
Pointer to the Next Color and Point/ Line/Character Identifier

ACKNOWLEDGEMENTS

I would like to thank the entire SLAC Instrumentation and Control group for their efforts on the COWs, and in particular acknowledge the contributions of Kevin Slattery, Phil Seward, Dave Ficklin, and Yu Chuan-Yu.

REFERENCES

1. R. E. Melen, "Design and Performance of the Stanford Linear Collider Control System", *IEEE Trans. Nucl. Sci.* NS-32, p. 230 (1985).
2. N. Phinney *et al.*, "Report on the SLC Control System", these proceedings.
3. Robert C. Beach, "The Unified Graphics System for Fortran 77", SLAC, CGTM 203, 204 and 205 (1983).

DISCLAIMER