AN ABSTRACT OF THE THESIS OF

Yan Xu for the degree of Master of Science in Electrical and Computer Engineering presented on October 10, 1990.

Title: Implementation of Display Control Node for a Distributed Microcontroller Network

Abstract Approved: Redacted for Privacy

James H. Herzog

As hardware becomes cheaper and cheaper, there is an increasing interest in the implementation of distributed control networks with microcontrollers. Such networks are usually inexpensive and of high performance.

In this thesis work, a low-cost display control unit for COLAN, a control-oriented network, is designed. The system is based on the 8051 single-chip microprocessor and 82716 video controller. COLAN network users can remotely display and control the screen by sending commands to the display control node through the network.

The emphasis of this thesis is in the implementation of the display control node, including hardware and software. It has been demonstrated that a basic set of display control functions has been developed, different type monitors can be supported, and on-screen instructions make the system easy to use.
IMPLEMENTATION OF DISPLAY CONTROL NODE FOR A DISTRIBUTED MICROCONTROLLER NETWORK

by

Yan Xu

A THESIS

submitted to

Oregon State University

in partial fulfillment of the requirement for the degree of

Master of Science

Completed October 10, 1990

Commencement June, 1991
APPROVED:

Redacted for Privacy

Associate Professor of Electrical and Computer Engineering in charge of major

Redacted for Privacy

Head of Department of Electrical and Computer Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented  October 10, 1990

Typed by the researcher for  Yan Xu
ACKNOWLEDGEMENTS

I would like to express my heartfelt thanks to a number of people for their help in the preparation of this thesis. Foremost among them is my major professor, Dr. J. H. Herzog, whose guidance during all my graduate studies has been invaluable.

I would also like to thank my thesis committee, Dr. R. Mohler, my minor professor, and committee members, Dr. D. Amort and Dr. G. Smith, for their active participation on my graduate committee.

Thanks are also due to the Intel Corporation for the donation of the 82716 Video Controller which made this thesis possible.

Finally, I would like to express deep appreciation to my husband, Feihong Zhu. His constant support and sacrifice made this endeavor a reality.
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Display Control Node</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Outline of Following Chapters</td>
<td>8</td>
</tr>
<tr>
<td>2. CRT DISPLAYS</td>
<td>9</td>
</tr>
<tr>
<td>2.1 CRT Fundamental</td>
<td>9</td>
</tr>
<tr>
<td>2.2 CRT Monitors</td>
<td>12</td>
</tr>
<tr>
<td>2.3 Color CRT Monitors</td>
<td>12</td>
</tr>
<tr>
<td>2.4 Display Resolution</td>
<td>14</td>
</tr>
<tr>
<td>2.5 Character Generation</td>
<td>14</td>
</tr>
<tr>
<td>2.6 CRT Controller</td>
<td>15</td>
</tr>
<tr>
<td>3. COLAN NETWORK</td>
<td>19</td>
</tr>
<tr>
<td>3.1 Computer Networks</td>
<td>19</td>
</tr>
<tr>
<td>3.2 The ISO OSI Reference Model</td>
<td>21</td>
</tr>
<tr>
<td>3.3 COLAN Network</td>
<td>22</td>
</tr>
<tr>
<td>4. DISPLAY CONTROL NODE</td>
<td>26</td>
</tr>
<tr>
<td>4.1 System Description</td>
<td>26</td>
</tr>
<tr>
<td>4.1.1 Overview</td>
<td>26</td>
</tr>
<tr>
<td>4.1.2 System Functions and Specifications</td>
<td>28</td>
</tr>
<tr>
<td>4.2 Hardware Description</td>
<td>30</td>
</tr>
<tr>
<td>4.2.1 Hardware Overview</td>
<td>30</td>
</tr>
<tr>
<td>4.2.2 8051 and 82716 Descriptions</td>
<td>31</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS cont.

Chapter                               Page

4.2.3 Interface Between 8051         33
    and Host Node-----------------------
4.2.4 Interface Between             34
    8051 and 82716----------------------
4.2.5 Memory and Mapping------------- 34
4.2.6 Video Subsystem Configuration-- 38
4.2.7 Display Object Definition------ 41

4.3 Software Description------------- 43
4.3.1 Software Overview-------------- 43
4.3.2 Video Subsystem
    Initialization Subroutine-------- 45
4.3.3 Data Reception and Display    46
    Control Subroutines---------------
4.3.4 Screen Control, Baud Rate      50
    and Color Selections--------------

5. CONCLUSION AND SUGGESTIONS FOR
   FURTHER WORK------------------------ 52

BIBLIOGRAPHY-------------------------- 55

APPENDICES

APPENDIX A TIMING CONSTANT CALCULATION-- 57

APPENDIX B DISPLAY COMMAND SUMMARY
   AND AN APPLICATION EXAMPLE---------- 64

APPENDIX C SCHEMATIC-------------------- 68
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 ISO OSI Reference Model</td>
<td>22</td>
</tr>
<tr>
<td>4.1 Intel MCS-51 Family</td>
<td>32</td>
</tr>
<tr>
<td>4.2 Register Window Organization</td>
<td>39</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Diagram of Distributed Microcontroller Network</td>
<td>3</td>
</tr>
<tr>
<td>1.2 COLAN Network with Display Control Nodes</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Cathode Ray Tube</td>
<td>10</td>
</tr>
<tr>
<td>2.2 Raster Scan</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Basic CRT Monitor Elements</td>
<td>13</td>
</tr>
<tr>
<td>2.4 Scan Sequence of a Character Row</td>
<td>16</td>
</tr>
<tr>
<td>2.5 CRT Interface</td>
<td>18</td>
</tr>
<tr>
<td>3.1 Computer Network Topologies</td>
<td>20</td>
</tr>
<tr>
<td>4.1 Structure Diagram of Display Control Node</td>
<td>27</td>
</tr>
<tr>
<td>4.2 8051 Memory Map</td>
<td>35</td>
</tr>
<tr>
<td>4.3 82716 Screen Memory Map</td>
<td>37</td>
</tr>
<tr>
<td>4.4 Program Flow Chart of Display Control Node</td>
<td>44</td>
</tr>
<tr>
<td>4.5 Flow Chart of Display and Control Subroutines</td>
<td>48</td>
</tr>
</tbody>
</table>
# LIST OF APPENDIX TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1 Task-Level Commands</td>
<td>64</td>
</tr>
<tr>
<td>B.2 Subroutine-Level Commands</td>
<td>65</td>
</tr>
</tbody>
</table>
## LIST OF APPENDIX FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1 Raster Timing</td>
<td>57</td>
</tr>
<tr>
<td>C.1 Display Board Schematic</td>
<td>68</td>
</tr>
</tbody>
</table>
IMPLEMENTATION OF DISPLAY CONTROL NODE
FOR A DISTRIBUTED MICROCONTROLLER NETWORK

CHAPTER 1
INTRODUCTION

1.1 Background

Semiconductor technology (LSI and VLSI) has made dramatic advances over the last decade. This has made computers smaller and more powerful. The progression has been of great benefit to both people and industry by taking the computer from a central computer room to the desk top of users. Since computers nowadays could be spread out around offices or buildings, it has become necessary for users to be able to communicate with each other.

Computer networks are especially invented to meet the demand of communication. There are two categories of computer networks according to their geographical distributions, Wide Area Network (WAN) and Local Area Network (LAN). WAN is distributed in a wide area, nation wide or even world wide. In contrast with WAN, LAN is distributed in a small geographical area. Because of the short distance, a LAN is usually characterized by

- low cost
- high transmission speed
- low transmission error rate.
LAN networks have wide applications. One is in the design of real-time control systems which employ several high performance microcontrollers. These microcontrollers are distributed at control points, perform specialized tasks and communicate with each other. Figure 1.1 shows the structure of a general distributed microcontroller network. In such distributed control systems, the complex control activities are broken down into several control tasks and performed by several microcontrollers referred to as nodes. Each node has two interfaces: one is the interface to the communication network; the other is to the specified application. Distributed microcontroller networks have the advantages of high performance and modularity that implies the possibility of specialized implementations of the nodes. The simplicity of maintenance and troubleshooting is also an advantage of the distributed microcontroller system.

A distributed microcontroller network model has been developed at the Department of Electrical and Computer Engineering, Oregon State University by the research group of Dr. J. H. Herzog, named the COLAN (Control-Oriented Local Area Network) network. It evolved from Task Master to COLANI, COLANII, COLANIII, COLANIV [2-3] [14-17]. The COLAN network is a low-cost network and can be easily implemented. It consists of at least one host node and several task processing nodes. Through the use of the task control structure and communication network, the complex control activities can be executed efficiently on appropriate distributed microcontrollers of COLAN networks.
Figure 1.1 Diagram of Distributed Microcontroller Network
COLAN networks can be used in many applications. For example, an education LAN, which has been developed from COLAN, includes a host node (the teacher) and several slave nodes (the students). The instructions of the teacher are sent to the students through the network and the students can "hand in" their exam or exercise to the teacher [12-13]. Several special task processing nodes of COLAN, such as step-motor control, light control, projector control and audio control, have been constructed.

The object of this thesis was to implement a simple, inexpensive COLAN node for the control of CRT display units. It results from the requirement for the remote control of display units, which is desired in many applications such as image processing, commercial advertisement and tele-education. Personal Computers (PCs) can be used in these applications, but the cost is high and the high computational capability of PCs is wasted.

1.2 Display Control Node

The display control node is a single board computer. The structure of the COLAN network with the display control nodes is shown in Fig. 1.2, where each display unit is controlled by a display control node, and the display control nodes are connected by the COLAN network with Network Interface Units. The COLAN network users can remotely control the display screen by sending the display information and commands to the display control node.
The display control node possesses the advantages of low cost, flexibility, easy implementation and extension. The display control node can be characterized by the following functions:

1. Communication function
   - Command packet receiving and routing
   - Display information receiving

2. Display and control function
   - On-line text display
   - Text re-display in page form
   - Screen clear
   - Line feed
   - Auto-scrolling screen when screen full
   - Enable display
   - Disable display

3. Mode selection
   - MDA (Monochrome Display Adapter) mode
   - CGA (Color Graphics Adapter) mode
   - EGA (Enhanced Graphics Adapter) mode

4. Color selection
   - Four colors can be selected for text display

5. "Friendly" system
   - On-screen help instruction
Figure 1.2 COLAN Network with Display Control Nodes
The hardware of the display control node consists of a single board computer (SBC) with an Intel 8051 microprocessor and the display board with an Intel 82716 video controller. The Intel 82716 is a single-chip video controller and was donated by the Intel Corporation. Unlike other CRT controllers, such as the Motorola 6845 which is used for IBM MDA and CGA type monitors, the Intel 82716 combines most video control functions, including all the dot timing logic, into one low-cost chip. It makes the hardware of the display board simpler than that required using other CRT controllers.

The display control node has two interfaces to the external world, the interface to the COLAN network and the interface to the display application. The interface to the COLAN network is implemented with the serial port of the 8051 and an asynchronous communication protocol. The interface to the display application is provided by the 82716 which produces the RGB (Red-Green-Blue) and synchronous signals to the CRT display units.

The implementations of hardware and software for display control node have been achieved. The hardware work included the implementation of the display control board and the modification of the SBC for the interfacing of the two boards. The software work included programming the display control software and integrating the display control software into the COLAN system.
1.3 Outline of Following Chapters

In Chapter 2, the fundamentals of CRT displays and CRT controllers are introduced. This establishes the necessary background for the following chapters.

Chapter 3 briefly discusses the COLAN network. The concept of computer network topology and the ISO OSI Reference Model also are presented.

Chapter 4 describes the functions of the display control node and focuses on the hardware and software implementations of the node.

Chapter 5 summarizes the work of the thesis and makes some suggestions for further work.

Appendix A contains the calculation of the horizontal and vertical synchronous constants for the 82716 video controller. Appendix B includes a summary of display control commands and an application example.
CHAPTER 2

CRT DISPLAYS

2.1 CRT Fundamental

CRT (Cathode Ray Tube) displays are one of the oldest display technologies and remain as one of the most popular. They are unmatched for high resolution displays with moderately large size and are the least expensive of all display technologies on a per-character basis.

CRT displays use the same basic technology originally developed for television. Figure 2.1 shows the key components of a typical CRT display [9]. Electrons are emitted at the base of the tube by an indirectly heated cathode. The flow of electrons from the cathode is controlled by the beam current control. These electrons are accelerated toward the faceplate by a high voltage (called the accelerating potential) applied to the anode. When the electron beam strikes the phosphor coating on the inside of the faceplate, the phosphor emits light at the point of impact.

The electron beam is aimed at a particular spot on the faceplate by the deflection coils. One coil provides horizontal deflection and the other vertical deflection. By applying the appropriate voltages to each, the electron beam can be aimed at any spot on the faceplate.

There are two methods used to generate an image on the screen. In one approach, called vector stroke, or XY, the beam is
directed to exactly trace each line that makes up the desired image. To avoid flicker, the entire image must be retraced more than 30 times per second, which places a limit on the complexity of the image that can be displayed by the vector method. The more common technique is raster-scan. Rather than having the beam trace the desired pattern, the beam always traces a standard pattern of horizontal lines. Figure 2.2 shows the trace pattern [9]. Beginning at the top left corner, the beam traces across to the right side and then returns quickly to the left edge. This retrace line is blanked (that is, not displayed) by turning off the electron beam. The trace repeats

**Figure 2.1** Cathode Ray Tube
until the entire screen has been filled with closely spaced horizontal lines. At the end of the last line, the beam is quickly returned to the top of the screen, this is called vertical retrace. The set of horizontal lines is called the raster. By turning the electron beam on and off as each line is scanned, the desired image is created on the display.
Raster-scanned displays are by far the most common type of CRT displays. They are less expensive than vector displays. The remainder of this thesis considers only raster-scanned CRT displays.

2.2 CRT Monitors

Interfacing directly to a cathode ray tube requires the generation of the high voltages for the anode and sweep signals for the horizontal and vertical deflection. These functions are common to all CRT applications, and CRT displays are generally purchased as a CRT tube with the associated circuits to perform these functions. This is called a CRT monitor.

Figure 2.3 shows the basic elements of a CRT monitor [11]. The horizontal and vertical oscillators provide the sawtooth-wave signals, which are the horizontal and vertical sweep signals respectively with the slow rise causing the scan across the display and the quick fall causing the retrace.

2.3 Color CRT Monitors

Color CRTs use the same basic scanning approach but have several additional components. Color displays use a matrix of red, green, and blue phosphor dots. Three video signals are required instead of one. Most color monitors used with computer systems are called RGB (Red-Green-Blue) monitors, since they have separate video inputs for each of the colors.
There are two different types of RGB interfaces in common use. The analog interface interprets the voltage levels on each of the R, G, and B inputs as representing the brightness of that color. This allows an infinite number of hues to be produced. On the other hand, TTL-interface monitors treat each of RGB as a digital input, turning the corresponding color on or off. Thus, only eight colors can be displayed, corresponding to the eight possible patterns of three bits. This interface is often extended to 16 colors (actually two shades of eight colors) by adding an "intensity" signal. This interface is used by
the IBM personal computer color-graphics adapter (CGA) and enhanced-graphics adapter (EGA).

2.4 Display Resolution

The resolution of a CRT display is a combination of two facts: the number of scan lines and the number of dots that can be displayed on each line. The number of scan lines is determined by the horizontal rate if the vertical scan rate is fixed which is usually chosen as 60Hz to match the power line frequency and eliminate any waviness in the display due to the deflection of the beam by stray electromagnetic fields. The number of dots that can be displayed in one line is determined by the rate at which the electron beam can be turned on and off; this is called the video bandwidth and usually equal to the dot rate. For IBM type monitors, the resolution of the CGA type is 320x200 (320 dots/line and 200 scan lines), EGA 640x350, MDA 720x350 and VGA 640x480.

2.5 Character Generation

Characters are formed on a CRT display by using a dot matrix. For minimum cost, a 5x7 matrix can be used, but the character appearance is poor. Higher-quality displays use a 7x9 basic matrix, which is extended to 7x11 to allow for the descendents in the lowercase characters g, j, p, q, and y.

The dot patterns for the characters that can be displayed are stored in a special-purpose memory called a character generator. The address of the character generator contains two parts, one
specifies the character to be displayed, the other specifies which cell line is needed. Only one cell line of each character is needed for each horizontal scan line. Figure 2.4 shows the sequence required to display a character row [20].

Character generators can be purchased as ROMs with standard character patterns. This is the least expensive approach if the standard character set is acceptable. EPROMs or custom mask-programmed ROMs can be used for nonstandard character generators. For maximum flexibility, RAM can be used for the character generator, which is initialized by the system software with the desired character patterns. This increases the system cost but allows character patterns to be dynamically redefined.

2.6 CRT Controller

The circuit that interfaces the microprocessor buses to the CRT monitor is called the CRT Controller. The key task of the CRT controller is providing the appropriate character codes and cell line addresses for the character row to be displayed to the character generator as the electron beam scans the raster pattern. The CRT controller also provides the horizontal and vertical synchronous signals for the CRT monitor.

The ASCII (American Standard Code for Information Interchange) code is typically used for the character selection input to the character generator. The ASCII code is a 7-bit code and includes many nondisplayable control characters. These codes are
**Figure 2.4** Scan Sequence of a Character Row
often used for other symbols, such as Greek letters, math symbols, or graphics characters.

Figure 2.5 shows a general block diagram of a CRT interface [11]. The data to be displayed are written to the CRT display RAM (also called the screen memory) in the form of ASCII codes by the microprocessor, and read by the CRT controller. For each character to be displayed, the CRT controller supplies the appropriate address to the screen memory. The output of the screen memory is the ASCII code for the character to be displayed, which provides the character address to the character generator. The CRT controller provides the cell line address to the character generator. The character generator outputs the dot pattern for the selected line of the selected character, which is loaded into the shift register. The shift register is clocked at the dot rate, and its output is the basic serial video data.
Figure 2.5 CRT Interface
3.1 Computer Networks

A computer network is defined as a network in which a collection of computers, terminals and other peripheral devices are connected by communication channels. The communication channels can be pairs of twisted wires, coaxial cables, optical fibers, radio communication or even satellite links. The computers and terminals are capable of exchanging information with each other through the communication media.

This combination of computer and communication technology provides each user with the ability to quickly and efficiently exchange information with other users. It enables a user in the network to access computing facilities, hardware and software at remote locations directly through his computer.

The topology of computer networks refers to the ways in which the computers are geographically connected by communication media. For local area networks, the most commonly used topologies are bus and ring structures [4]. The structure of a ring network with 8 nodes is shown in Figure 3.1a. There each node has two links and is capable of receiving data from one link and transmitting into the other. The messages are transmitted through the ring in a single direction.
Figure 3.1 Computer Network Topologies
A bus network is shown in Figure 3.1b. In bus networks, all nodes are connected to a single global bus, and all transmissions are through this bus. Routing is not needed because the bus is the only transmission medium. At any given time, there is only one node allowed to transmit messages through the bus. The transmission is in a broadcasting style. The message can reach all the nodes connected to the bus, however, only the addressed destination node will accept the message. The bus network has the following advantages and is used for the COLAN networks.

1. It is easy to delete or add nodes onto a bus.

2. Bus networks are more reliable than Ring networks. A single node failure may not have any effect on another node's operation as long as the host nodes remain functional.

3.2 The ISO OSI Reference Model

To minimize the design complexity, modern computer networks use layered architecture with the network services divided into different layers. One of the standards for such layered network is the Open System Interconnection (OSI) Reference Model which was developed by the International Standard Organization (ISO). It is often called ISO OSI Reference Model.

The ISO OSI Reference Model arranges the functions and rules of the communication among network nodes into an hierarchy of subsets, called layers. The seven layers and their functions are listed in Table 3.1. In the hierarchy, each layer is a logical entity which
accepts services from the layer immediately below it and provides services to the layer above it.

<table>
<thead>
<tr>
<th>Name</th>
<th>Layer</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Layer</td>
<td>7</td>
<td>Application services</td>
</tr>
<tr>
<td>Presentation Layer</td>
<td>6</td>
<td>Code and format conversions</td>
</tr>
<tr>
<td>Session Layer</td>
<td>5</td>
<td>Establishment and termination of logical connections between nodes</td>
</tr>
<tr>
<td>Transport Layer</td>
<td>4</td>
<td>Maintenance of data integrity between nodes</td>
</tr>
<tr>
<td>Network Layer</td>
<td>3</td>
<td>Information routing</td>
</tr>
<tr>
<td>Data Link Layer</td>
<td>2</td>
<td>Transfer of data packages</td>
</tr>
<tr>
<td>Physical Layer</td>
<td>1</td>
<td>Transfer or reception of a bit stream</td>
</tr>
</tbody>
</table>

Table 3.1 ISO OSI Reference Model

This model is highly modular and flexible. Each layer functions independently and can be designed and implemented separately.

3.3 COLAN Network

The COLAN network consists physically of at least one host node called the system scheduler, several hardware units called task processing units, and a communication network to provide physical connection. The system scheduler is usually implemented by a personal computer which provides an interface to a human operator
and an interface to the network. It acts as a "manager" in the network and communicates with the task processing units by utilizing a special command format. Each task processing unit also has two interfaces, one to the network and the other to the possible applications. These two interfaces provide resources for the communication network and for the control of applications. The communication channel is implemented by using the serial ports of host computers and the microprocessors of task processing units, which provides the transmission media for the exchange of information between the network nodes.

The COLAN networks are especially convenient for various control applications. The complex control activities are divided into tasks. Each task is programmed as a modular piece of software. The execution time and the processing unit are scheduled by the host computer sending the control command to the addressed task processing unit.

The task processing units are usually implemented with single-board computers which have the advantages of low cost and simple structure. Intel 8031, 8048, 8051, 8052 and Motorola 68C11 microprocessors have been used in the construction of task processing units. The software of the task processing units can be divided into two parts. One part is the COLAN operating system software which ensures the proper communication with the host computer and implements the physical and data link layer functions in the network [17]. The other part is the application software which provides the application functions and high layer communication
functions in the ISO OSI model. In order to achieve high performance, both the COLAN operating system software and the application task subroutines are written in assembly language.

As mentioned above, the control command packets are transmitted from the host node to the task processing nodes to activate the operation tasks. The control command packet of the COLAN network contains the address of the task processing unit, task number, execution condition and parameters. It has the following format:

\[ \{ AA \ P \ NN \ S \ DD \ DD \ DD \ DD \ DD \ DD \} \]

where

\{ \quad \text{Start Flag which indicates the start of the packet.} \}

\AA \quad \text{Hexadecimal address of task processing unit.}

\P \quad \text{Prefix. This single character informs the addressed device of the way to execute the command. Three kinds of prefix command are used.}

:\quad \text{Queued Task. The task is put at the end of Task Queue. It is activated at the conclusion of the execution of tasks preceding it on the task queue.}

! \quad \text{Immediate Task. The task is to be execute immediately.}
Synchronized Task. It works like a queued task with the exception that the task can be activated only when a synchronizing signal "$" is received.

NN Task Number. This is a two hexadecimal character code which specifies the task to be performed.

Q Postfix Control Character. It indicates how many times the task will be executed.

1. Discard upon completion

2. Requeue upon completion

3. Requeue a specified count of times

DD Data Parameter. This optional field allows 0 to 5 pairs of hexadecimal character for use in executing a task. The existence of this field and the meaning of the data in this field are command dependent.

)} End Flag which denotes the end of a control command.

For an example, the command packet

(05132.)

will cause the display control node to receive data and display it on the screen.
CHAPTER 4

DISPLAY CONTROL NODE

4.1 System Description

4.1.1 Overview

A functional, microprocessor-based CRT display control node for COLAN network was designed and constructed with the Intel 8051 microprocessor and 82716 Video Storage Display Device. Figure 4.1 shows the structure of the display control node.

The Intel 8051 is a control-oriented 8-bit microprocessor. It contains one serial I/O port, one bit-addressable parallel I/O port, 64k bytes of program memory space and 64k bytes of data memory space. The functions of the 8051 in the display control node include the communication with the host computer, decoding command packets, writing display data into the screen memory and controlling the display screen.

The Intel 82716 is a highly integrated VLSI CRT controller. With the advantages of low cost, advanced display capability and simple interface to microprocessors, the 82716 is suitable for the implementation of the display control node. The functions of the 82716 CRT controller include managing the screen memory and proving video information and synchronous signals to the CRT display unit.
The display functions are performed by the cooperation of the 8051 and the 82716. In the display control node, the 8051 is the "brain" and the video subsystem looks like a memory to the 8051. All communication with the video subsystem occurs in the screen memory which is exclusively controlled by the 82716. The 8051 writes the display data into the screen memory through the 82716.
Bus Interface Unit. The 82716 reads the display data from the screen memory and performs all necessary conversions and manipulations to convert the display data to the RGB signals.

There are two interfaces in the display control node. One is the interface between the 8051 and the COLAN network and the other is between the 8051 and the 82716. The former is implemented with the on-chip serial port of the 8051 and the asynchronous communication protocol. The COLAN network model with one host computer is considered in the implementation of the display control node. In this case, all information on the communication channel is provided either by the host computer or by a task processing unit, such as the display control node, in response to a command by the host. The display control node listens to the communication channel and receives the commands addressed to the display control node. Other commands and communication noises are discarded. Aided by the Network Interface Unit, the display control node can be connected to general COLAN networks. The interface between the 8051 and its host node and the interface between the 8051 and the 82716 will be discussed in following sections.

4.1.2 System Functions and Specifications

Display format

<table>
<thead>
<tr>
<th>MDA mode</th>
<th>EGA mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>*80 characters/row</td>
<td>*64 characters/row</td>
</tr>
<tr>
<td>*25 character rows</td>
<td>*25 character rows</td>
</tr>
</tbody>
</table>
CGA40 mode
*40 characters/row
*20 character rows

CGA80 mode
*80 characters/row
*20 character rows

**Character format**
*7x8 character dot pattern within an 8x10 frame for CGA
*7x8 character dot pattern within an 8x14 for EGA/MDA

**Character display control**
*back space
*cursor return and line feed
*page control
*auto-scrolling line up when screen full

**Screen control**
*enable display
*disable display
*clear screen

**Monitor supported**
*MDA type: resolution 640x350
*CGA40 type: resolution 320x200
*CGA80 type: resolution 640x200
*EGA type: resolution 640x350

**Display Color**
*4 colors for color monitor: white, red, green and yellow
*green for monochrome monitor
Character displayed
*96 ASCII alphanumeric characters

Character transmitted
*96 ASCII alphanumeric characters
*ASCII control characters

Data transmission rate
*1200 baud initialized
*300, 1200 and 2400 baud user selectable

Display command format
{05!T1 T2.xx}

05 is the COLAN network address of the display control node.
T1 T2 are task numbers ranging from 30H to 37H.
xx is parameter.
{, }, !, . are COLAN network command symbols (see Chapter 3).

4.2 Hardware Description

4.2.1 Hardware Overview

The hardware of the display control node consists of the SIBEC-51 and the video display board. The SIBEC-51 is an 8051 single board computer. It contains 8051 microprocessor, 64k byte memory space and bus interface circuits. The 64k byte memory space is originally divided into 32k byte program/data memory
space, 16k byte PROM programmer and 8k byte I/O space. In this application, the SIBEC-51 board has been modified on the memory space arbitration to interface with the video display board. The video display board is implemented in this work. The board includes the 82716 video controller, four 64kx4 dynamic RAMs and the circuits for the 82716 to interface with the 8051 and the dynamic RAMs. The dynamic RAMs are used as the screen memory of the 82716. In the following sections, a brief introduction for the 8051 and the 82716 is first given, then the interface between the 8051 and the host computer, the interface between the 8051 and the 82716, memory mapping and video subsystem configuration are discussed.

4.2.2 8051 and 82716 Descriptions

8051

The Intel MCS-51 family is an 8-bit single chip microprocessor family which contains the 8031, 8051, 8052, 8751 and so on. The 8051 is the original member and also the core for all MCS-51 devices. Table 4.1 gives a list on the Intel MCS-51 family.

The MCS-51 microprocessors are single chip control-oriented microcontrollers. Byte-processing and numerical operations on small data structures are facilitated by a variety of fast addressing modes for accessing the internal RAM. Extensive on-chip support is provided for one-bit variables as a separate data type, allowing direct bit manipulation and testing which is desired in control and logic systems. With the internal data and program memories, the MCS-51 microprocessors can be used in many control applications,
especially, in the embedded control applications, such as in automobiles and robots.

\begin{table}[h]
\centering
\begin{tabular}{|l|c|c|c|c|}
\hline
Names & Internal Memory & Timers/Interrupts \\
 & Program & Data & Counters \\
\hline
8051 & 4kx8 ROM & 128x8 RAM & 2x16-bit & 5 \\
8051AH & 4kx8 ROM & 128x8 RAM & 2x16-bit & 5 \\
8052AH & 8kx8 ROM & 256x8 RAM & 3x16-bit & 6 \\
8031 & none & 128x8 RAM & 2x16-bit & 5 \\
8031AH & none & 128x8 RAM & 2x16-bit & 5 \\
8032AH & none & 256x8 RAM & 3x16-bit & 6 \\
8751H & 4kx8 EROM & 128x8 RAM & 2x16-bit & 5 \\
8751H-8 & 4kx8 EROM & 128x8 RAM & 2x16-bit & 5 \\
\hline
\end{tabular}
\caption{Intel MCS-51 Family}
\end{table}

The memory space of the 8051 consists of a 128 byte internal data memory (RAM), a 64k byte program memory with the lowest 4k bytes (ROM) on chip, and a 64k byte external data memory. The 8051 has four ports providing 32 I/O lines to interface with the external world. The port 0 and port 2 are used for memory expansion, port 3 contains special control signals, and port 1 is used for I/O only. The 8051 also contains two 16-bit timer/event counters and one serial I/O port for either multiprocessor communications,
I/O expansion or full UART (Universal Asynchronous Receiving and Transmission).

**82716**

The Intel 82716 Video Storage Display Device is a low cost, highly integrated video controller. The 82716 uses RAM type character generator which resides in the screen memory. The dot timing circuits and D/A converters are contained in the chip. The 82716 has the following functional units:

* Bus Interface Unit (BIU). The 8051 accesses the screen memory through the BIU.

* Memory Interface Unit (MIU). MIU generates the control signals and row and column addresses for the screen memory.

* Timing unit. It consists of oscillators and clock generators. The video and internal clocks are generated by the timing unit.

* Synchronous generator. The synchronous generator controls the horizontal and vertical timings for raster generation.

* Pixel unit. It contains a pixel formatting unit as well as scan line buffers in which the display information is placed for each scan line. It also contains the color lookup table and D/A converters.

**4.2.3 Interface between 8051 and Host Node**

The display control node interfaces to the host computer through the 8051 on-chip serial port and RS-232 asynchronous
communication protocol. The data is transmitted in the format of 8 data bits, one stop bit and no parity checking. The baud rate of the transmission is initialized at 1200 and is user selectable at 300, 1200, or 2400. A line driver (1488) and a line receiver (1489) are used to convert signals between the TTL level and the RS-232 level.

4.2.4 Interface between 8051 and 82716

The 82716 supports both 8 and 16 bit microprocessors from all Intel compatible families. In the display control node, the 82716 accepts Read, Write, Address Latch Enable and multiplexed address and data bus from the 8051. The Byte High Enable input of the 82716 is pulled high so that it knows the processor is 8 bits. The Ready signal output by the 82716 is to extend the processor cycle when the 82716 is using the screen memory and can not service the processor request for the screen memory access immediately. Since the 8051 has no Ready input and can not be temporarily halted during a screen memory access, the Ready pin of the 82716 is programmed as a "free access indicator". The 8051 can test this signal through the software to see if the 82716 is using the screen memory and, if not, can gain the access immediately. In this application, the Ready signal of the 82716 is connected to the pin 1 of the 8051 port 1.

4.2.5 Memory and Mapping

The memory of the display control node consists of two parts, the program/data memory of the 8051 and the screen memory of the 82716. In this application, the internal 4k byte ROM of the 8051
is not used, the external program space and the data memory space are combined into one 64k byte memory space where the 32k byte space is used by the 8051 as program/data memory and the other 32k byte space is mapped into the 82716 screen memory. Figure 4.2 shows the memory map of the 8051.

![Figure 4.2 8051 Memory Map](image_url)
The screen memory is built up by four 64k×4 dynamic RAMs and the first 32k bytes are used in this application. There are two segments in the screen memory, the register segment and the data segment. The register segment contains the information on the video subsystem configuration which is initialized by the 8051 and read by the 82716 after each frame to update its on-chip registers. The data segment contains the actual display data, Object Descriptor Table, Access Table, Color Lookup Table and two character generators. The Object Descriptor Table gives the object window attributes, such as position, color and width. The Access Table contains the vertical position and priority of each object. The Color Lookup Table stores the color palette and the character generators keep the character dot patterns. Figure 4.3 shows the map of the screen memory.

The screen memory is byte accessible for the 8051 and word accessible for the 82716. The 8051 accesses the screen memory with the memory mapping. The mapping is through two logical windows, which refer to the register window and data window in the 8051 address space and register segment and data segment in the 82716 address space. The register and data windows can be relocated anywhere in the 8051 address space. In the 82716 address space, the data segment can be relocated while the register segment is fixed at 0000H to 000FH. In this application, the register window and the data window start at the same address 8000H in the 8051 memory address space with the register window occupying locations from 8000H to 801FH and the data window from 8020H to
<table>
<thead>
<tr>
<th>8051 Address (in byte)</th>
<th>82716 Address (in word)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8000H</td>
<td>0000H</td>
</tr>
<tr>
<td>8020H</td>
<td>0010H</td>
</tr>
<tr>
<td>8300H</td>
<td>0180H</td>
</tr>
<tr>
<td>8A00H</td>
<td>0500H</td>
</tr>
<tr>
<td>A000H</td>
<td>1000H</td>
</tr>
<tr>
<td>C000H</td>
<td>2000H</td>
</tr>
<tr>
<td>E000H</td>
<td>3000H</td>
</tr>
<tr>
<td>FFFFH</td>
<td>4000H</td>
</tr>
</tbody>
</table>

- **Control Registers R0 - R15**
- **Access Table**
- **Color Look Up Table**
- **Object Descriptor Table**
- **Object Data**
- **Character Generator 0**
- **Character Generator 1**

**Figure 4.3** 82716 Screen Memory Map
FFFFH. The register segment and the data segment also start at the same address 0000H in the 82716 address space with register segment taking word addresses from 0000H to 000FH and data register from 000FH to 3FFFH. The beginning addresses of the register window, data window and data segment are stored in Register 2 to Register 5 which are initialized by the 8051 writing to the pre-initialization register window 8400H to 841FH.

### 4.2.6 Video Subsystem Configuration

The information on the video subsystem configuration is kept in the register segment of the screen memory. The register segment contains 16 registers which give the screen constants, dynamic RAM organization, video timing and various base addresses. Table 4.2 shows the organization of the 16 registers in the screen memory. These registers determine the video subsystem configuration and are discussed in this section.

**R0: Video Configuration Register 0**

Dynamic RAM Configuration Bits and Slow Access Bit

The dynamic RAM configuration bits DOF, DS0 and DS1 are set to 110 so that the 82716 knows the screen memory is implemented by four 64kx4 dynamic RAMs. Since the access time of the dynamic RAM (MT4067-10) is 100 ns, the Slow Access bit is reset to 0. The 82716 will generate suitable memory control signals to the screen memory.
<table>
<thead>
<tr>
<th>Register (R)</th>
<th>Description</th>
<th>Byte Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>R15</td>
<td>Horiz. Constant 3</td>
<td>1EH</td>
</tr>
<tr>
<td>R14</td>
<td>Horiz. Constant 2</td>
<td>1CH</td>
</tr>
<tr>
<td>R13</td>
<td>Horiz. Constant 1</td>
<td>1AH</td>
</tr>
<tr>
<td>R12</td>
<td>Horiz. Constant 0</td>
<td>18H</td>
</tr>
<tr>
<td>R11</td>
<td>Access Table Base Address Counter (ATBAC)</td>
<td>16H</td>
</tr>
<tr>
<td>R10</td>
<td>Character Generator 0 &amp; 1 Base Addresses</td>
<td>14H</td>
</tr>
<tr>
<td>R9</td>
<td>Color Table Base Address (CTBA)</td>
<td>12H</td>
</tr>
<tr>
<td>R8</td>
<td>Access Table Base Address (ATBA)</td>
<td>10H</td>
</tr>
<tr>
<td>R7</td>
<td>Object Descriptor Table Base Address (ODTBA)</td>
<td>0EH</td>
</tr>
<tr>
<td>R6</td>
<td>Priority Access Quantity (PAQ)</td>
<td>0CH</td>
</tr>
<tr>
<td>R5</td>
<td>Data Segment Base Address (DSBA)</td>
<td>0AH</td>
</tr>
<tr>
<td>R4</td>
<td>Data Window/Segment Length Mask (DWSLM)</td>
<td>08H</td>
</tr>
<tr>
<td>R3</td>
<td>Data Window Base Address (DWBA)</td>
<td>06H</td>
</tr>
<tr>
<td>R2</td>
<td>Register Window Base Address (RWBA)</td>
<td>04H</td>
</tr>
<tr>
<td>R1</td>
<td>Video Configuration Register 1 (VCR1)</td>
<td>02H</td>
</tr>
<tr>
<td>R0</td>
<td>Video Configuration Register 0 (VCR0)</td>
<td>00H</td>
</tr>
</tbody>
</table>

**Table 4.2** Register Window Organization
Update Control Flag Bit

The Update Control Flag bit determines if the 82716 updates all its internal registers after each frame. In this application, the Update Control Flag bit is set to 1 so that the 82716 reads all the register segments from the screen memory into its on-chip registers at the end of each frame. This allows the display object to change in one frame time.

Display Enable Flag Bit

The Display Enable Flag is used to control the display. The 8051 can turn the display on or off by setting or resetting the Display Enable Flag bit.

R1: Video Configuration Register 1

External Video Clock Bit

The External Video Clock bit determines the 82716 CKIO (Clock Input and Output) pin to be used as an input for receiving the external dot clock or as a buffered dot clock output. In this application, the External Video Clock bit is set to 1 so that the 82716 can use the 16MHz external clock as dot clock for EGA, MDA and CGA80 modes, and 8MHz for CGA40 mode.

Free Access Enable Bit

The Free Access Enable bit enables the Ready pin of the 82716 to act as a free access indicator when the bit is set to 1. If the 82716
is using the screen memory, the Ready pin is low. The 8051 can test this pin through the software and wait until the 82716 finishes its access.

Register 2 to Register 5 hold the base addresses of the register window, data window and data segment (see the section of memory and mapping). Register 7 to Register 10 hold the base addresses for Object Descriptor Table, Access Table, Color Lookup Table and Character Generator. In this application, the base address for the Object Descriptor Table is 0500H in the 82716 address space, Access Table is 0010H, Color Lookup Table (not used) is 0180H, Character Generator 0 is 2000H and Character Generator 1 (not used) is 3000H. Register 12 to Register 15 contain the constants for horizontal and vertical timings. The calculation of these constants is included in Appendix A.

4.2.7 Display Object Definition

In this application, the display object is alphanumerical characters. Each character is 8 pixels wide and 10 scan lines high for CGA and 14 scan lines for EGA/MDA. The character object data has two formats, one is three bytes/character, called full attribute definition, and the other is one byte/character. For the full attribute definition, each character is encoded as an ASCII byte plus a 2-byte attribute word which gives the background color, foreground color, character generator selection and other display control information. For one byte definition, each character is encoded as an ASCII byte and the display control information is given in the Object Descriptor
Table. The full attribute definition has very versatile display capabilities, but requires a large amount of memory space, three times that of the one byte definition. In this application, the one byte/character format is selected because of the space limitation of the screen memory. Only four colors can be selected in this mode.
4.3 Software Description

4.3.1 Software Overview

The software of the display control node is written in the 8051 assembly language due to the time constraint during execution. The software consists of two parts, COLAN operating system and display application programs. The latter is the emphasis of this section. Figure 4.4 shows the flow chart of the software program.

In this application, the COLAN operating system is slightly modified to match the hardware configuration. Pin 1 to pin 3 of the 8051 port 1 are used for testing the Ready signal of the 82716 and selecting monitor types. The memory address space 8000H to FFFFH of the 8051 is used for the mapping of the 82716 screen memory. Any part of COLAN operating system related to these is removed to ensure that the display control node works properly.

The display application program is written as a number of subroutines. This offers the advantage of modularity. These subroutines can be classified into three parts according to their functions:

* video subsystem initialization.
* data reception and display control.
* screen control, baud rate and color selections.

To be "friendly" to users, a help instruction is shown in the host computer's screen so that users can easily determine the task number and command format for each command.
Figure 4.4 Program Flow Chart of Display Control Node
In the following sections, the functions and implementations of the display application subroutines are discussed.

4.3.2 Video Subsystem Initialization Subroutine

The initialization subroutine contains 82716 initialization, monitor selection, character generator implementation, display object definition and display window setting.

After system reset, the initialization of the 82716 is carried out by the 8051 writing to the pre-initialization Register Window, 8400H to 841FH. These 16 registers determine the configuration of the video subsystem. After the Register Window has been completely initialized, one more write cycle is directed to Register 0 to set the Update Control Flag bit to 1, which allows the 82716 to update its internal registers from the Register segment in the screen memory. Then the 8051 waits one frame time for the 82716 update and continues programming through a newly defined register window and data window starting at 8000H.

The pin 2 and pin 3 of the 8051 port 1 are used for the monitor selection which is determined by the 4-bit dip switch on the display board. The initialization program checks the pin 2 and pin 3 of the 8051 port 1 to see which monitor mode is selected (00 for CGA40, 01 for CGA80, 10 for EGA and 11 for MDA), and then sets all the corresponding constants for display control and synchronous timings.
The 82716 uses a RAM type character generator which resides in the screen memory. The character generator is built up by loading the character dot patterns of 96 displayable ASCII alphanumeric characters from the 8051's memory to the 82716 screen memory. To save the memory space, only 7x8 basic dot patterns are stored in the EPROM of the 8051's memory. The 8x10 and 8x14 frames are formed by the program adding 0s to the remaining lines.

The object definition is realized by the 8051 writing to the Object Descriptor Table starting at 8A00H. It is a four-byte long table and contains the character height, object width, color, X coordinate of starting position and the base address of object data. The object display window is a rectangle. Its width is defined by the Object Descriptor Table and length by the Access Table. Initially, the object window length is set to the character height (10 scan lines for CGA and 14 scan lines for MDA and EGA), which allows the cursor to be displayed at the first character position. The object window width is set to 40 characters for CGA40, 80 for MDA/CGA80 and 64 for EGA.

4.3.3 Data Reception and Display Control Subroutines

There are two methods for the host to send display information: one is by files and the other is by the keyboard. The former has the advantage of high efficiency, but puts serious limitations on the processing time of reception. In order to shorten this time, the display control node first receives display data and saves them in the 8051's memory starting at 6000H. Then, after all the data are received, the 8051 writes these data into screen
memory for display. This task is performed by the file transfer subroutine (Task 31).

The display information received from the host's keyboard and the display control are initiated by the character display and control subroutine (Task 32). Figure 4.5 shows the flow chart of the character display and control subroutine. First, a help instruction is sent to the host computer which tells users the control keys used for display control:

CONTROL-D End the display.
CONTROL-P Display first page.
CONTROL-Q Display last page.
CONTROL-R Page up.
CONTROL-S Page down.

The 8051 then loops to check if there is data coming. If so, it checks the received information to see if it is the control key or the display data, then calls the corresponding subroutines. The character display and control subroutines are discussed as follows.

**Character Display**

This subroutine puts the display data into the Object Data Table in the screen memory. The address is given by a pointer, called the End of Data Table which always points to the next available location for the display data. Also the number of characters currently displayed in the screen is counted. If the last character is at the end
Figure 4.5 Flow Chart of Display and Control Subroutines
of a line, the cursor return and line feed subroutine will be called to increase one character line for next character display.

**Cursor Return and Line Feed**

There are two methods to cause cursor return and line feed. One is by the Return key of the host computer, and the other is when the current character line is full.

The program first checks if the whole screen is full, if so, the Object Data Table base address is added by the maximum number of characters in one line, which causes the display to scroll up one line. If the screen not full, the object window length is increased by the height of the characters to produce a new character line. The cursor is displayed at the first position of the new character line.

**Back Space**

The backspace program is executed when the Backspace key is hit in the host computer. In contrast to the display subroutine, the Backspace program first checks if the cursor is in the first position of a character line, if so, the object window length will decrease by the character height. Otherwise, the cursor just moves back one space. The pointer of End of Data Table is also changed so that the next coming display data will be put in the correct location.
Page Functions

The combinations of the Control key with the P, Q, R and S keys are used for page control. The CTRL-P keys cause the first page of object data to be displayed; CTRL-Q for the last page; CTRL-R for the page up and CTRL-S for the page down. The page functions are implemented by changing the Object Data Table base address resided in the Object Descriptor Table.

End Display

The CTRL-D keys are used for ending display, which causes the 8051 to jump out of the loop of receiving display information and return to the COLAN operation system.

4.3.4 Screen Control, Baud Rate and Color Selections

Baud Rate Selection

The crystal used by the 8051 operates at 12MHz. The baud rate is determined by the timer 1 constant. The formula for the constant calculation is

\[
\text{timer 1 constant} = 256 - \frac{12\text{MHz}}{12 \times 32 \times \text{baud rate}}.
\]

The baud rate is initialized at 1200 and user-selectable at 300, 1200 and 2400. The command packet for the baud rate selection is

\[
[05!30.xx] \quad xx=98H \quad 300 \text{ baud}
\]

\[
xx=E6H \quad 1200 \text{ baud}
\]

\[
xx=F3H \quad 2400 \text{ baud}
\]
The baud rate is changed by the 8051 loading the parameter from the command packet into its on-chip timer 1 which will produce the corresponding baud rate for the 8051 serial port.

**Color Selection**

The command packet for color selection is

\[(05!36.0x)\]

\[x=0 \quad \text{white}\]

\[x=1 \quad \text{red}\]

\[x=2 \quad \text{green}\]

\[x=3 \quad \text{yellow}\]

The color is changed by the 8051 writing the parameter in the command packet into the corresponding bits of the Object Descriptor Table.

**Enable and Disable Display**

Enable and disable display functions are realized by setting or clearing the Display Enable bit in the Register 0 of the 82716. These functions are useful when the display needs to be synchronized with some actions. In such cases, users can first close the display, load the display data, and display it later.
CONCLUSION AND SUGGESTIONS FOR FURTHER WORK

The implementation of the display control node has been discussed in the previous chapters. The primary goal of this thesis is to design a low-cost single board computer for the control of CRT display units. This design goal has been realized by using an Intel 8051 microprocessor and 82716 video controller. The display control node has the following features:

1. Providing a basic set of functions for the display and screen control of CRT display units.

2. Supporting different type of CRT monitors.

3. Communication with host computers or other COLAN network nodes.

4. On-screen instruction that makes the system "friendly".

This work is the first implementation of the display control node for COLAN networks. The character display functions are realized based on the Intel 82716 video controller. Being only an experimental system, the display control node could be improved in many respects. Suggestions for possible improvements are presented as followings:

(1). The development of graphic display functions. The 82716 has advanced display capabilities for videotext and color graphics. It allows the management of up to 16 objects on the screen. The
graphic objects are formatted as bit maps. The screen image is constructed from various user-defined objects.

(2). The use of the full attribute definition for character objects. As discussed in Chapter 4, the full attribute definition has strong display capability. With this format, the background color, foreground color, character generator, underlining, blinking and other display controls can be selected for each character.

(3). The use of Character Generator 1, and the dynamic re-definition of Character Generator 0 and 1. The 82716 has two character generators which allow users to dynamically load the character generator with different character fonts without interrupting the display.

(4) The standard display for EGA mode. Since the working clock of the 8051AH used in the board is 12MHz, only 70 characters/line can be displayed for EGA mode. To achieve the standard EGA display (80 characters/line), the late versions of 8051 microprocessors with higher frequency clocks are needed.

(5) The support for VGA type monitors. The original version of the 82716 video controller did not support the VGA mode since the maximum working clock of the 82716 was 14MHz and the horizontal sweeping frequency of VGA type monitors is about 29KHz which results in a 34 μs line time. If there are new versions of the 82716 with higher working clocks, the display control node can be redesigned for VGA mode.
The main difficulty for (1)-(3) comes from the limitation of the 8051 memory address space. Since the graphics are stored in the screen memory in the bit-map form, a large amount of memory space is required. To solve this problem, the Intel 8086 microprocessor may be used in the display control node. The 8086 has one megabyte memory address space and therefore can fully utilize the 512 Kbyte screen memory that the 82716 can support.
BIBLIOGRAPHY


Figure A.1 shows the raster timing which is needed for the programming of vertical and horizontal constants. The constants are programmed for MDA and EGA in the resolution of 640x350 at 60Hz, non-interlaced mode; 320x200 for CGA40 and 640x200 for CGA80. The dot clock frequency is 16MHz for MDA, EGA and CGA80 and 8MHz for CGA40. The 8MHz dot clock is obtained from the 16MHz clock divided by 2. The horizontal constants are programmed in the terms of GCLK (Graphic Clock=Dot Clock Frequency/16) and the vertical constants are programmed in the terms of line time. The horizontal and vertical constants have to be "tweaked" to obtain a steady display on the screen. The following values give a flicker-free display.

Note: HC0-HC3 and VC0-VC3 values are offset by 1.

Figure A.1 Raster Timing
1. EGA

For IBM EGA Monitors, the horizontal synchronous frequency is 21.85KHz, which means the line time is 45.76 \(\mu s\). Since the GCLK is 1 \(\mu s\), the line time needs to be integer.

Let

\[
\text{Line time} = 47 \, \mu s
\]

then

\[
\text{Active Vertical zone} = 47 \times 350 = 16.45 \text{ ms}
\]
\[
\text{Dot clock period} = \frac{1}{16\text{MHz}} = 0.0625 \, \mu s
\]
\[
\text{GCLK Period} = 16 \times 0.0625 \, \mu s = 1 \, \mu s
\]
\[
\text{Active Horizontal zone} = 640 \times 0.0625 = 40 \, \mu s
\]
\[
\text{Blanking time} = \text{line time} - \text{Active Horizontal zone}
\]
\[
= 47 - 40 = 7 \, \mu s
\]

**Horizontal Constants**

Assume

\[
\text{HSYNC width} = 2 \, \mu s
\]
\[
\text{FP} = 3 \, \mu s
\]
\[
\text{BP} = 2 \, \mu s
\]

then

\[
\text{HC0} = 2 \, \mu s = 2 \text{ GCLK periods} = 000001B
\]
\[
\text{HC1} = \text{HC0} + \text{BP} = 4 \, \mu s = 4 \text{ GCLK periods} = 000011B
\]
\[
\text{HC2} = \text{Active Horizontal time} + \text{HC1}
\]
\[
= 40 + 3 = 44 \, \mu s = 44 \text{ GLCK Periods} = 101010B
\]
\[
\text{HC3} = \text{line time} = 47 \, \mu s = 47 \text{ CLCK Periods}
\]
\[
= 101110B
\]
**Vertical Constants**

Assume

- VSYNC width = 0.2 ms
- FP = 0.25 ms
- BP = 0.4 ms

then

\[ VC0 = 0.188 \text{ ms} = 4 \text{ line times} = 0000000011B \]

\[ VC1 = VC0 + BP = 0.564 \text{ ms} \]

\[ = 12 \text{ line times} = 0000001011B \]

\[ VC2 = \text{Active vertical time} + VC1 = 16.45 + 0.564 \]

\[ = 17.01 \text{ ms} = 362 \text{ line times} = 0101101001B \]

\[ VC3 = VC2 + FP = 17.25 \text{ ms} \]

\[ = 367 \text{ line times} = 0101101110B \]

**2. MDA**

For IBM MDA Monitors, the horizontal synchronous frequency is 18.43 KHz.

Let

- Line time = 55 \( \mu \)s

then

- Active Vertical zone = 55 \( \times \) 350 = 19.25 ms
- Dot clock period= \( \frac{1}{16} \text{MHz} \) = 0.0625 \( \mu \)s
- GCLK Period = 16 \times 0.0625 \( \mu \)s = 1 \( \mu \)s
- Active Horizontal zone = 640 \times 0.0625 = 40 \( \mu \)s
- Blanking time = line time - Active Horizontal zone

\[ = 55 - 40 = 15 \mu \text{s} \]
Horizontal Constants

Assume

\[ \text{HSYNC width} = 3 \mu s \]
\[ \text{FP} = 6 \mu s \]
\[ \text{BP} = 6 \mu s \]

then

\[ \text{HC0} = 3 \mu s = 3 \text{ GCLK periods} = 000010B \]
\[ \text{HC1} = 3 + 6 = 9 \mu s = 9 \text{ GCLK periods} = 001000B \]
\[ \text{HC2} = \text{Active Horizontal time} + \text{HC1} \]
\[ = 40 + 9 = 49 \mu s = 49 \text{ GLCK Periods} = 110000B \]
\[ \text{HC3} = \text{line time} = 55 \mu s = 55 \text{ CLCK Periods} \]
\[ = 110110B \]

Vertical Constants

Assume

\[ \text{VSYNC width} = 0.2 \text{ ms} \]
\[ \text{FP} = 0.6 \text{ ms} \]
\[ \text{BP} = 0.5 \text{ ms} \]

then

\[ \text{VC0} = 0.165 \text{ ms} = 3 \text{ line times} = 0000000010B \]
\[ \text{VC1} = \text{VC0} + \text{BP} = 0.715 \text{ ms} \]
\[ = 13 \text{ line times} = 0000001100B \]
\[ \text{VC2} = \text{Active vertical time} + \text{VC1} = 19.25 + 0.715 \]
\[ = 19.96 \text{ ms} = 363 \text{ line times} = 0101101010B \]
\[ \text{VC3} = \text{VC2} + \text{FP} = 20.6 \text{ ms} \]
\[ = 375 \text{ line times} = 0101110110B \]
3. CGA80

For IBM CGA Monitors, the horizontal synchronous frequency is 15.75KHz.

Let

Line time = 61µs

then

Active Vertical zone = 61 x 200 = 12.2 ms
Dot clock period= 1/16MHz = 0.0625 µs
GCLK Period = 16 x 0.0625 µs = 1 µs
Active Horizontal zone = 640 x 0.0625 = 1 µs
Blanking time = line time - Active Horizontal zone
                = 61 - 40 = 21 µs

**Horizontal Constants**

Assume

HSYNC width = 5 µs
FP = 12 µs
BP = 8 µs

then

HC0 = 5 µs = 5 GCLK periods = 000100B
HC1 = 5 + 8 = 13 µs=13 GCLK periods = 001100B
HC2 = Active Horizontal time + HC1
     = 40 + 13 = 53 µs = 53 GLCK Periods
     = 110100B
HC3 = line time = 61 µs = 60 CLCK Periods
     = 111100B
**Vertical Constants**

Assume

VSYNC width = 0.2 ms
FP = 1.8 ms
BP = 1.8 ms

then

\[ VC_0 = 0.244 \text{ ms} = 4 \text{ line times} = \text{0000000011B} \]

\[ VC_1 = VC_0 + BP = 2 \text{ ms} \]
\[ = 33 \text{ line times} = \text{0000100000B} \]

\[ VC_2 = \text{Active vertical time} + VC_1 = 12.2 + 2 \]
\[ = 14.2 \text{ ms} = 233 \text{ line times} = \text{0011101000B} \]

\[ VC_3 = VC_2 + FP = 15.98 \text{ ms} \]
\[ = 262 \text{ line times} = \text{001110001B} \]

4. **CGA40**

Let

Line time = 64 μs

then

Active Vertical zone = 64 x 200 = 12.8 ms
Dot clock period = 1/8MHz = 0.125 μs
GCLK Period = 16 x 0.125 μs = 2 μs
Active Horizontal zone = 320 x 0.125 = 40 μs
Blanking time = line time - Active Horizontal zone
\[ = 64 - 40 = 24 \mu s \]
**Horizontal Constants**

Assume

HSYNC width = 4 $\mu$s

FP = 12 $\mu$s

BP = 8 $\mu$s

then

HC0 = 4 $\mu$s = 2 GCLK periods = 000001B

HC1 = $4 + 8 = 12 $\mu$s= 6 GCLK periods = 000101B

HC2 = Active Horizontal time + HC1

\[ = 40 + 12 = 52 $\mu$s = 26 GLCK Periods \]

\[ = 011001B \]

HC3 = line time = 64 $\mu$s = 32 CLCK Periods

\[ = 011111B \]

**Vertical Constants**

Assume

VSYNC width = 0.25 ms

FP = 1.8 ms

BP = 1.8 ms

then

VC0 = 0.256 ms = 4 line times = 0000000011B

VC1 = VC0 + BP = 2.11 ms

\[ = 33 line times = 000010000B \]

VC2 = Active vertical time + VC1 = 12.8 + 2.11

\[ = 14.91 ms = 233 line times = 0011101000B \]

VC3 = VC2 + FP = 16.768 ms

\[ = 262 line times = 010000101B \]
APPENDIX B

DISPLAY COMMAND SUMMARY AND AN APPLICATION EXAMPLE

B.1 Summary of Display Commands

There are 8 task-level commands and 5 subroutine-level commands for the display applications. The task-level commands are listed in Table B.1 and the subroutine-level commands in Table B.2.

<table>
<thead>
<tr>
<th>Task No.</th>
<th>Function</th>
<th>Command Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 30</td>
<td>Baud rate selection</td>
<td>{05!30.xx}</td>
</tr>
<tr>
<td></td>
<td>xx=98H; 300</td>
<td></td>
</tr>
<tr>
<td></td>
<td>xx=E6H; 1200</td>
<td></td>
</tr>
<tr>
<td></td>
<td>xx=F3H; 2400</td>
<td></td>
</tr>
<tr>
<td>Task 31</td>
<td>File transfer</td>
<td>{05!31.}</td>
</tr>
<tr>
<td>Task 32</td>
<td>Data receiving and display</td>
<td>{05!32.}</td>
</tr>
<tr>
<td>Task 33</td>
<td>Clear screen</td>
<td>{05!33.}</td>
</tr>
<tr>
<td>Task 34</td>
<td>Disable display</td>
<td>{05!34.}</td>
</tr>
<tr>
<td>Task 35</td>
<td>Enable display</td>
<td>{05!35.}</td>
</tr>
<tr>
<td>Task 36</td>
<td>Color selection</td>
<td>{05!36.0x}</td>
</tr>
<tr>
<td></td>
<td>x=0; white</td>
<td></td>
</tr>
<tr>
<td></td>
<td>x=1; red</td>
<td></td>
</tr>
<tr>
<td></td>
<td>x=2; green</td>
<td></td>
</tr>
<tr>
<td></td>
<td>x=3; yellow</td>
<td></td>
</tr>
<tr>
<td>Task 37</td>
<td>Display demonstration</td>
<td>{05!37.}</td>
</tr>
</tbody>
</table>

Table B.1 Task-Level Commands
<table>
<thead>
<tr>
<th>Keys</th>
<th>Function</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTRL-D</td>
<td>End of Task</td>
<td>Used in Task 31, Task 32</td>
</tr>
<tr>
<td>CTRL-P</td>
<td>Display first page</td>
<td>Used in Task 32</td>
</tr>
<tr>
<td>CTRL-Q</td>
<td>Display last page</td>
<td>Used in Task 32</td>
</tr>
<tr>
<td>CTRL-R</td>
<td>Display page up</td>
<td>Used in Task 32</td>
</tr>
<tr>
<td>CTRL-S</td>
<td>Display page down</td>
<td>Used in Task 32</td>
</tr>
</tbody>
</table>

Table B.2 Subroutine-Level Commands

B.2 Display example

In this section, a short example is given to show how the data are displayed by the display control node. The purpose of the example is to make users familiar with the use of the display control node.

In this example, the host node is an IBM PC. The software for the communication is the "Procomm" program. The display unit is an IBM EGA monitor which can work in both EGA and CGA modes.

Before plugging in, the dip switch 1 in the display board should be set to 0100 (MDA), 0101 (EGA), 0110 (CGA80) or 1011 (CGA40), and the dip switch 2 should be set to 0100 (EGA) or 1000 (MDA/CGA). A cable is needed to connect the serial port of the PC with the serial port of the display control node. After setting the dip
switches and connecting the serial ports, the power can be plugged into the display control board.

At the beginning, the "Procomm" program is run at the host computer, which allows users to send the display data and commands to the display control node. In this example, a file transfer command \{05!31.\} is first sent to the display control node. Then by pushing the Pgup key on the host's keyboard, selecting the ASCII file and giving the file name, the "Procomm" program begins to send the file data to the PC's serial port bit by bit. After all the data is sent out, the CTRL-D keys are pushed to indicate the end of the file transfer. On the other side, the display control node receives the data and stores it in the data memory started at 6000H by executing the file transfer subroutine (Task 31). When the CTRL-D keys are received, the display control node ends the loop for data reception and begins to send the received data into the 82716 screen memory for display.

When the task of file transfer is finished, a task-level display command list is shown on the host's screen:

Task 30: Baud rate selection. \{05!30.xx\}
xx=98, 300; E6, 1200; F3, 2400.

Task 31: File transfer. CTRL-D to end. \{05!31.\}

Task 32: Display and control. \{05!32.\}

Task 33: Clear screen. \{05!33.\}

Task 34: Disable display. \{05!34.\}

Task 35: Enable display. \{05!35.\}
Task 36: Color selection. \(05!36.0x\)

\[ x=0 \text{ white; 1 red; 2 green; 3 yellow.} \]

Task 37: A demonstration. \(05!37.\)

Then a command packet \(05!32.\) is sent to the display control node, which causes the data reception and display subroutine to be executed (Task 32). After the command is sent out, the screen of the host computer will display a list of the subroutine-level commands:

- CTRL-D  End of display
- CTRL-P  First page
- CTRL-Q  Last page
- CTRL-R  Page up
- CTRL-S  Page down

Since the display data has been sent by the file, we just use the CTRL-P keys to display the first page of the data and the CTRL-S keys to display the following data page by page.
Figure C.1 Display Board Schematic