STANDARD

SERIAL COMMUNICATION

INTERFACE and PROTOCOL

DESCRIPTION

FOR

KENT DISPLAY

SIGN and DISPLAY PRODUCTS

Kent Displays

Document

25016

KENT DISPLAYS INCORPORATED
343 Portage Blvd., Kent OH USA 44240
330 673-8784, 330 673-4408 (FAX)
Website: www.kentdisplays.com
E-mail: sales@kentdisplay.com

Kent Displays Document 25016, REVISION RECORD

Revision	Date	Author	Description
A	8/5/98	CRB	Released to support the 90014 INFOSIGN Module Dev. Kit
B	10/6/98	CB/TW	Modified sections 4.4 Updated sections 5.1, 5.3 and 5.3.1.
C	11/24/98	CRB	Mod. Sec. 5.2 to reflect INFOSIGN Enhancements
D	1/7/99	CRB	Modified ¼ VGA section (5.1) to account for increased message storage & selective addressing capability. Added section 5.1.1.
E	2/11/99	CRB	Added Broadcast Message Section (4.4)
F	5/5/99	CRB	Mod. 5.1.x Sections for ¼ VGA 15126G03 Display.
G	9/28/99	CB/TE	Mod. 5.2 to handle additional font controls & "N" Cmd, added section 5.4.
H	10/7/99	CB/TE	Added J, M & U commands to section 5.2. Fixed <Binary Pixel Data> information for the 1/8 VGA Appl.
I	3/23/00	CRB	Added/Modified Section 5.5, Again
J	4/11/00	CRB	Added "Rotate ON" bullet to section 5.5.
K	5/24/00 - 10/27/00	CRB	Added App. Cmds to INFOSign Protocol to support mixed-mode operation (Added Section 5.2.1, Table 5.2.2 and 3-Line font to section 5.2.2). Added "Z" cmd. to Table 5.2.1. Removed Sample INFOSIGN App. Source Code Section.
L	2/7/01, 4/13/01	CRB	Added ";" command (table 5.2.1), "4" & "5" commands (table 5.2.3), and modified table 5.2.3, and sections 5.2.1 and 5.2.2, to support new INFOSign III product.
M	4/20/01	TE	Added (Does not display after valid message RX'ed) to Table 5.4.1 N & U command.
N	6/18-7/27/01	CB/ReA	Added >, Y & Z App. Cmds. To QUAD application, table 5.3.1.
O	8/16/01 - 10/11/01	CRB, TE	Added sections 5.6 & 5.7, Added "Swell-ON" & "Fade-ON" bullets to section 5.5.
P	11/01/01; 12/3/01	ReA	Modified Height Multiplier for Dynamic Update of INFOSign III (Section 5.2.1)
Q	1/3/02, 1/18-5/3/02	CRB	Added Dynamic Update, Partial screen & Dynamic Memory type application commands to 1/8 VGA section (section 5.4), to support the 10120 controller. Added LVGA Section (Section 5.7).
R	6/27/02	CRB	Modified ("Sectionalized") LVGA Graphic Data Commands in section 5.7
S	8/28/02, 9/4/02, 9/10/02, 11/1/02	CRB	Added sectioin 5.8 for New ¼ VGA 15297 Display Module. Added 2x Horz. Capability to Dynamic Update Parameter, for 320x80 app., Sect. 5.6. Changed "Message" terminology to "Packet" to avoid confusion when describing lower level packet components, Section 4. Added text and re-named section 4.1.1.
T	12/21/02	CRB	Removed old 1/4 VGA App. Lev. Section (4.1). Removed old app. commands from INFOSign section (5.2). Added some INFOSign App. features, in Dynamic Update parameter features (Horz. Stretch), Font Controls, Flashing, Swell & Fade.

Table of Contents

1.0 Introduction

This document describes the proper interface to control Kent Displays Sign and Display products. The document describes the interface between the display module controller and the host computer determining the sign or display product image content. This protocol description is labeled “the Kent Displays standard serial protocol” in product description documentation.

The interface described is designed to support character and graphical interface capability. The standard Kent Displays serial protocol supports both operational modes. The host computer can change the module image by outputting an ASCII character sequence, or graphic pixel data to the corresponding display module. The protocol is designed to minimize the communication packet lengths, enable error checking, support multiple address locations, and provide wake-up commands to support the unlicensed wireless communication link.

2.0 ChLCD Technology and Display/Sign Product Description

The Kent Displays sign displays are modular units containing one or more cholesteric liquid crystal display (ChLCD) cells. A ChLCD is an advanced reflective display technology. It requires no back lighting.

The bi-stable ChLCD has advantages over traditional LCD displays by containing a non-volatile memory feature. Once an image is driven onto the ChLCD, it will remain until a new image is written. The image also remains if power is removed. The non-volatile memory feature enables the display to be very energy efficient.

The ChLCD also exhibits a wide viewing angle advantage over traditional active and passive displays. The technology requires no polarizers. Its extremely high contrast characteristics are exhibited consistently at viewing angles up to 90º from perpendicular to the display. The display also exhibits excellent sunlight readability characteristics. The ChLCD technology has superior contrast characteristics than a typical active matrix display.

A typical configuration for a Kent Displays sign product conforms to the configuration outlined in Figure 2-1. For some display modules which have only one ChLCD display cell per module, the slave and master display are the same.

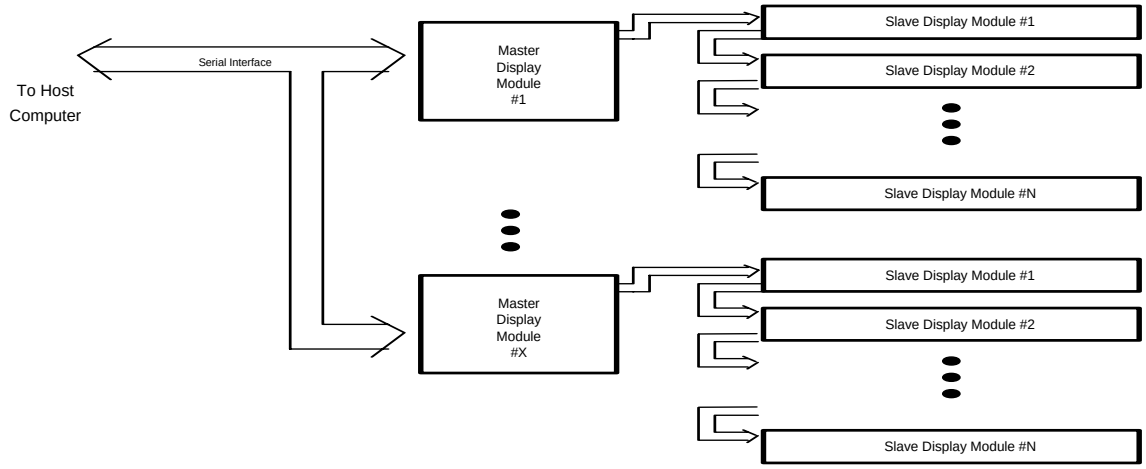

Figure 2-1: Typical Sign/Display Configuration

When Kent Display products are configured as outlined in Figure 2-1, the product contains the following features:

- Several masters can be connected to a common serial communication interface controlled by the host computer. Each master can be assigned a unique address location to support a typical "hard-wired" multi-drop or wireless RF communication hardware configuration.
- For sign display type modules, several slave display modules can be connected to a common master assembly. The image on each slave module along with each master display can be addressed and altered individually.
- The modules contain all the high voltage electronics required to create the appropriate drive signals required for the ChLCD technology. This feature enables a simple asynchronous serial communication link for interfacing and controlling each module. Communication to the master modules can also be performed using an inexpensive, RF type, wireless link, available from Kent Displays.
- The master module contains temperature compensation circuitry and firmware to enable the displays to be operated between ambient temperatures of 0º and 60º-70º C (upper temperature limit is dependent upon the display module type). This sophisticated logic enables this function to be transparent for the user. The display modules can also operate below 0º C, however the image update duration will be slower.
- The user interface dictates the image content for each display module by downloading bit-mapped image data or text character information to the master module. Due to the memory feature of the ChLCD technology, the last image written on a display module will remain indefinitely after a power interruption, unless the user requests a new image after power is restored. Each master unit can store several images or display messages for each slave display within the RAM of the master controller.

3.0 Interface or Communication Protocol Description

The Kent Displays standard serial communication protocol is designed to support typical half-duplex multi-drop, or unidirectional "simplex" configurations. The protocol can also be implemented using full duplex hardware components. The message packet formats are intended to provide all the necessary information for checking data integrity in the event data is corrupted due to a communication median failure. The protocol is designed for the host computer to act as a master, with the display modules acting as slaves.

4.0 Lower Level Communication Protocol Description

The following sub-sections outline the lower layers of the protocol, which determine the framing of the application layer data into message packets. The lower level communication protocol is common regardless of the Kent Displays sign or display product.

4.1 Host Command Format to the Master Display Modules

The messages sent by the host computer to the master display modules obey the following data packet format:

Field Description	Packet Start Byte	Master Address #	Packet #	Application Layer Command Designation	Application Layer Desig-nation # 2.	Application Layer Data Elements	Packet Check-sum	Carriage Return	Line Feed
Packet Field #	1	2	3	4	5	6	7	8	9
Byte Count / Field	1	1	1	1	0-X	0-X	1	1	1

Table 4.1.1 Host Command Message Packet Format

Where:

1. *Packet Start Indication Byte* = ASCII escape (ESC) byte; 1B (Hex). This field occupies a 1 byte width.
2. *Master Address #* = Address setting of master display module, 0-31 (0-63 possibilities for most displays) decimal, or 0-1F (0-3F) Hexadecimal. Address #0 is typically allocated for broadcast message purposes, and should not be assigned to a sign location in normal operation. This field occupies a 1 byte width.
3. *Packet #* = Packet number assigned by the host computer for the data packet, 0-255 possibilities. This field occupies a 1 byte width.
4. *Application Layer Command Designation* = Dictates the action required for the master and/or slave display module. This field occupies a 1 byte width.
5. *Application Layer Designation #2* = This field is reserved as an additional designation for the application layer message content. Depending upon the display type, it may be used to designate the slave module number, it may designate the first and last bytes for loading bit-mapped data, or it may be omitted from the message. This field a variable width field.
6. *Application Layer Data Elements* = This field may contain additional data elements required for the application layer message content. Some messages will not require this field within the message data packet. This field a variable width field.
7. *Packet Checksum* = Equals the least significant 8 bits of the summation of all elements contained in message packet fields 1-6. This field occupies a 1 byte width.
8. Carriage *return* = ASCII carriage return, or 0D in hexadecimal. This field occupies a 1 byte width.
9. *Line feed* = ASCII line Feed, or 0A in hexadecimal. This field occupies a 1 byte width.

4.1.1 Additional Host Command Communication Requirements and RF Operation

Most Kent display modules are designed to be battery operated. To enable battery operation, many Kent Display modules contain a "Wake-up and Sleep-mode" feature. To conserve battery power, these modules will power themselves down (or "Sleep") if inactivity is sensed locally on the diagnostic port(s) and if inactivity is sensed with the serial communication port. The display modules will automatically "power-up" or "Wake-up" when activity is sensed on either a local diagnostic or the serial communication port.

If a host computer starts communicating with display modules connected to a multi-drop or RF wireless communication median, all modules connected to the multi-drop or in the proximity of the host computer (if RF link) will "Wake-up" and start normal operation. To ensure all modules connected to the multi-drop or in the proximity of the host "wake-up", the following actions need to be implemented by the host computer prior to transmitting normal message packets as described previously:

1. The host should transmit a continuous stream of ASCII "U" (55 in hex) for approximately 1 second (if communicating using a “hard-wired” RS-485 or RS-232 configuration, this 1 second duration needs only to be the length of time required to transmit a few characters).

2. If the Kent Displays RF communication modules are used (Kent Display P/N's 10158 and 10146), the duration for transmitting a continuous stream of ASCII "U" (55 in hex) should be extended from 1.0 to 5.0 seconds. The 5.0-second duration is required since the RF slave modules will only check the RF band every 4.8 seconds, to conserve battery power. The 5.0-second duration is only necessary prior to the 1st packet transmission. If a significant amount of data is to be sent and with significant pauses between each packet, then the host computer should turn OFF or disable the corresponding sign (or controller) sleep mode timer.

3. The host should wait approximately 0.5 to 1 seconds after the wake-up transmission described previously to enable the sign master controllers to properly initialize.

4. If the Kent Displays RF communication modules are used, then the host computer should send each packet with the following 4 byte sequence prior to field 1, or the ESC byte of the packet (illustrated in hexadecimal):

   ```
   0x00, 0xff, 0x00, 0xff
   ```

   Also, each packet should contain the following 2-byte sequence, after field 9, or the Line Feed of the packet (illustrated in hexadecimal):

   ```
   0x55, 0x00
   ```

   Neither the 4 byte, nor the 2 byte sequences indicated are included into the checksum calculation of the packet, installed in field 7 (refer to previous table). In an RF system, the 4 and 2 byte sequences are required to assist in synchronizing the RF slave and RF host bi-directional transceivers.

The host will need to implement the 1st three items above prior to transmitting normal message packets if any of the following conditions occur:

a. The host computer is transmitting the first message packet for the given communication period.

b. The host has not transmitted any type of message packet over the serial communication median for greater than preset "sleep mode" duration programmed into the sign or controller modules (typically 20 seconds).

c. The host has physically moved to a new proximity for proper communication to other display modules (only applicable if using a wireless communication median).

4.2 Sign Response Protocol Format to a Host Command

There are two possible responses that would be transmitted by the master display module to the host computer, a positive and a negative response. Both possibilities follow the same message packet format.

If the host to master display module communication median is a Kent Displays supplied unidirectional RF communication link, the host computer will receive no responses from the display modules.

The positive response format would appear as follows:

Field Description	ACK	Packet #	Packet Checksum	Carriage Return	Line Feed
Packet Field #	1	2	3	4	5
Byte #/ Field	1	1	1	1	1

Table 4.2.1 Sign Positive Response Message Packet Format

Where:

1. *ACK* = ASCII acknowledgment character, 06 (Hex)
2. *Packet #* = Packet number of the previously transmitted host message package, 0-255 possibilities.
3. *Packet Checksum* = Equals the least significant 8 bits of the summation of all elements contained in message packet fields 1 and 2.
4. *Carriage return* = ASCII Carriage return, or 0D in hexadecimal.
5. *Line feed* = ASCII line Feed, or 0A in hexadecimal.

The negative response format would appear as follows:

Field Description	NAK	Packet #	Packet Checksum	Carriage Return	Line Feed
Packet Field #	1	2	3	4	5
Byte #/ Field	1	1	1	1	1

Table 4.2.2 Sign Negative Response Message Packet Format

Where:

1. *NAK* = ASCII acknowledgment character, 15 (Hex)
2. *Packet #* = Packet number of the previously transmitted host message package, 0-255 possibilities.
3. *Packet Checksum* = Equals the least significant 8 bits of the summation of all elements contained in message packet fields 1 and 2.
4. *Carriage return* = ASCII carriage return, or 0D in hexadecimal.
5. *Line feed* = ASCII line Feed, or 0A in hexadecimal.

4.3 Host and Sign Module Communication Action Description

A master sign/display module will respond to a host command if and only if the address within the host transmitted packet matches the hardware address setting on the respective sign module. A positive response will be sent if the host command packet is valid and the checksum received matches the checksum calculated by the master module controller. A negative response will be transmitted to the host if a checksum mismatch occurs due to a communication median problem, or if the host command was invalid.

If a negative response or no response is received within 3 seconds after a host command is issued, the host should re-transmit the message packet to the sign or controller module. If after three consecutive unsuccessful host transmissions, the host computer should assume the respective sign module is "off-line", and re-try communication at a later time.

4.4 Broadcast Messages

Broadcast messages are host transmitted messages where the host wants all signs or displays connected or in the proximity of the host (if the communication median is a wireless link) to implement the requested command within the message. Broadcast messages are typically used for less sophisticated type display systems where unique address capability is not required or for transmitting safety warning type messages, such as “Fire in Building, Please Evacuate”.

Broadcast messages are the only type of messages where the signs do NOT transmit a response packet back to the host computer. If the host packet received at the sign is not corrupted (sign calculated

checksum matches the checksum received in the packet), the sign will implement the requested command within the received packet.

Address number 0 has been allocated for broadcast message purposes. Hence**, the user should NOT select address 0** as an address location using the sign hardware components allocated for the selection.

4.5 Sample Host Lower Level Communication Protocol Code

The following code fragments illustrate how a host computer can implement the lower level data packing of application layer data and transmit the packets out of the host system. The function `Get_Target_Response` illustrates how a host system can check the packets received from a sign system. The following code fragments are written using a PC-based Visual Basic development system:

```
Sub Lower_lev_pack_data ()

'   This function packs the data previously formatted & packed by the
'   Appl_level_pack_data subroutine.
'   The Appl_level_pack_data subroutine started the checksum calculation for
'   fields 4-6, while this function will add the remaining portions of
'   the fully packed message consisting of the following:
'    - The ESC character byte (field #1), 1B in Hex
'    - The Sign Address # in Hex (field #2).
'    - A message # used for tracking messages between this host & the respective
'      target. (field #3).
'    - The application level data stream created by a separate
       Appl_level_pack_data function (fields 4-6).
'    - A message checksum equaling the least significant 8 bits of all data
'       elements contained in fields 1-6.  The checksum byte in packed into field 7..
'    - An ASCII carriage return (0D in hex) and Line feed (0A in hex) in fields 8 & 9).
       ESC = Chr$(27)
         sign_address_char = Chr$(address_index + 1)
         For Loc_Msg_num = New_msg_number_out To (New_msg_packed - 1)
             ' Message fields #1-3
           temp = ESC & sign_address_char & Chr$(Loc_Msg_num)
           For p = 1 To 3                  ' Add fields 1-3 to checksum calculation
             msg_byte = Mid(temp, p, 1)
             msg_check_sum(Loc_Msg_num) = msg_check_sum(Loc_Msg_num) + Asc(msg_byte)
           Next p
               ' Mask off all bits except the least significant 8 bits for checksum
           checksum = msg_check_sum(Loc_Msg_num) And &HFF
           Complete_sign_msg_packet(Loc_Msg_num) = temp & Appl_lev_Msg_data(Loc_Msg_num) &
                      Chr$(checksum) & Chr$(13) & Chr$(10)
         Next Loc_Msg_num
End Sub

Sub TX_new_Data_Packets ()

' This function performs the actual transmission of data to the sign modules
' after all the data has been packed by the appropriate aplication & lower level
' packing routines.  The function assumes all the appropriate flags have been
' set by the lower level packing functions to determine which data packets need
' tranmission.  The function will clear the appropriate flags after the transmission
' process has been completed.
  Label1.Visible = True
  For Loc_Msg_num = New_msg_number_out To (New_msg_packed - 1)
        ' Message fields #1-3
    Label1.Caption = ""
    loop_count = 1
    packet_length = Len(Complete_sign_msg_packet(Loc_Msg_num))
    Do Until loop_count > 3
      Label1.Caption = "TX'ing Packet # " & Str$(Loc_Msg_num) & ", " &
            Mid(Complete_sign_msg_packet(Loc_Msg_num), 1, 10) & Chr$(10) & Chr(13)
      Label1.Caption = Label1.Caption & "To Sign # " & Str$(address_index + 1) & " Attempt # " &
          Str$(loop_count) & Chr$(10) & Chr$(13)
      Comm1.InBufferCount = 0     ' Discard any in RX buffer prior to this point
' Perform the following setups prior to TX'ing data to minimize duration between TX & monitoring
```

```
‘ Serial Port
      If rf_enabled_mnu.Checked = False Then
        Timer1.Enabled = False
        Timer1.Interval = Responce_TimeAllowed
        TimeOut1 = False
        Timer1.Enabled = True  'Enable after setting interval
      End If
      Screen.MousePointer = HOURGLASS  ' Change pointer to hourglass.
      For i = 1 To packet_length                                ' TX Out Msg Packet
        Comm1.Output = Mid(Complete_sign_msg_packet(Loc_Msg_num), i, 1)
      Next i
      If rf_enabled_mnu.Checked = False Then        ' Hard wired communication configuration
        Target_Responce = Get_Target_Responce(Loc_Msg_num) ' Monitor RX Line
        If Target_Responce = True Then
          If comm_error = True Then     ' Clear communication error message
            comm_error = False
            FormTitle
          End If
          Exit Do
        End If
        loop_count = loop_count + 1
      Else
        Exit Do
      End If
      If loop_count > 3 Then    ' Display Communication Error Message
        comm_error = True
        FormTitle
      End If
    Loop
  Next Loc_Msg_num

End Sub

Function Get_Target_Responce (Msg_num)
' This function monitors the RX port & analyses the responces from the
' Target sign.  If a valid message responce is received from the target,
' (based on the TX'ed message information provided @ input parameters to
' this function) a valid indication is returned to the calling context.
'' If no responce or a NAK message is RX'ed, a Re-transmit message flag is
' sent to the calling context to signal a re-transmission.
c$ = Comm1.Input
Get_Target_Responce = False
field_pos = 1
RX_checksum = 0
type_msg_RXed = 0
inform_user = False
Do Until TimeOut1
    DoEvents
    If Not AOK Then Exit Do  ' Allow User Abort
    If c$ <> "" Then
      If Asc(Mid(c$, 1, 1)) = 6 Then         ' ACK RX'ed
        If Asc(Mid(c$, 2, 1)) = Msg_num Then
          RX_checksum = 6 + Msg_num
          If Asc(Mid(c$, 3, 1)) = RX_checksum Then Get_Target_Responce = True
          Exit Do
        End If
      End If
      If Asc(Mid(c$, 1, 1)) = 21 Then        ' NAK RX'ed
        If Asc(Mid(c$, 2, 1)) = Msg_num Then Exit Do
      End If
      Label1.Caption = Label1.Caption & "Character(s) RX'ed = " & c$ & " , # Char(s) RX'ed = " &
            Str(Len(c$)) '& Chr$(10) & Chr$(13)
    Else          ' No byte RX'ed from Serial Port
      If inform_user = False Then
        Label1.Caption = Label1.Caption & "Waiting for Sign # " & Str$(address_index + 1) & " to
             Respond to msg # " & Str$(Msg_num) & Chr$(10) & Chr$(13)
        inform_user = True
      End If
    End If
    c$ = Comm1.Input     ' Monitor RX Port Again
```

```
Loop
Timer1.Enabled = False
Screen.MousePointer = DEFAULT     ' Return to normal
End Function
```

5.0 Application Level Command Designations

The following sub-sections outline all the possible application level commands available for the various Kent Displays sign and display products. This section will describe in detail the specific application level commands located in fields 4 through 6 outlined in table 4.1.1. All host to display product messages will be packed in the lower level formats described in section 4.1.

All the responses generated by the Kent Displays modules and/or sign products will be identical to the format outlined in section 4.2.

Since the format for the application level message fields are dependent upon the type of sign or display module, each of the following subsections is assigned to a specific display product.

5.2 INFOSign II and III Host to Sign Application Level Command Descriptions (Mixed Mode)

The following table outlines the available application level command sequences for the Kent Display INFOSign II and INFOSign III display modules. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. The application level command designations are case sensitive.
- N/A designates that no bytes are designated or inserted for respective field.
- Typical Data elements are illustrated for field #6.
- First Pixel Byte address, Last Pixel Byte address are the respective address locations when the host is transmitting a binary image for a INFOSIGN display module.

Many of the following commands are only available on INFOSign III products, and INFOSign II with revision H version firmware or later.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"0" (ASCCI Zero)	<Display #>, <Message. #>, <Sense>	3	DC2 DC4 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load Text (image #1-4) into Memory (Does not display until valid "W" command received).
2	"1"	<Display #>, <Message. #>, <Sense>	3	<Binary Pixel Data> (for full screen image)	900	Load Graphic (Image #1-4) data into (Does not display until valid "W" command received).
3	"2"	<Display #>, <Message #>, <First Row # to Alter> (1-20_{10}), <Sense>	4	DC2 DC4 "*Change Disp.1 Msg X Line 3*" <ETX>	Variable	Load Partial Screen Text message in Memory to be updated in normal mode (Creates 1 line of Text on Display after corresponding "W" command is received).
4	"3"	<Display #>, <Message #>, First Row # to Alter, Number of Rows Altered (20_{10} Max), <Sense>	5	<Binary Pixel Data>	Variable (600 Bytes Max.)	Load Partial Screen graphic image in Memory to be output in Normal mode (Does not display until valid "W" command received).
*5	"4"	<Display #>, <Message #>, <First Row # to Alter> (1-20_{10}), <Sense>, ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	8	DC2 DC4 "*Change Disp.1 Msg X Line 3*" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after "W" command is received). Can not be used with 1-Line Fonts! Can only be implemented with INFOSign III type modules.
*6	"5"	<Display #>, <Message #>, First Row # to Alter, <Sense>, Number of Rows Altered (20_{10} Max), ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	9	<Binary Pixel Data>	Variable (600 Bytes Max.)	Load Partial Screen Graphic image in Memory to be displayed in Dynamic Update format (Does Not display on ChLCD until after a "W" command is received). Can only be implemented with INFOSign III type modules.
7	"6"	<Display #>, < Message #>, First Row # to Alter, Non-Invert Duration (in 0.1 increments), Invert Duration, # of Flash Cycles, <Dyn. Update Parameter 1>	7	DC3 DC2 "Flash Disp.1 Msg X Row 4" <ETX>	Variable (256 Bytes Max.)	Load Partial Screen Text message (1 line of text) to be "Flashed" in the display as indicated in the corresponding parameters in field # 5 (Does Not flash on ChLCD display until after a "W" command is received).

Table 5.2.1: INFOSIGN Host to Display Application Level Commands ("Mixed Mode")

8	"7"	<Display #>, <Message #>, <Sense>	3	N/A	0	Set/Clear Invert sense of Display image.
9	"8"	<Message #>, Duration (Hex)	2	N/A	0	Set Unique Pause Duration for respective message for Auto cycle operation (in 0.1 second increments).
10	"9"	<1st Message #>, <Last Message #>, <# of Message Cycles>, <Tog. Img. Sence Byte>	4	N/A	0	Start Automatic Image Cycling using the parameters specified.
*11	";"	N/A	0	N/A	0	Ask Sign for display (Panel) Count. Host will receive message f/sign after positive response is received.
12	"M"	<On/Off>	1	N/A	0	Turn OFF/ON Sleep Timer (ON after reset or power-up)
13	"O"	N/A	0	N/A	0	Output ChLCD Temperature from Sign. Host will receive a Temperature message f/sign after positive response is received.
14	"P"	Duration (Hex)	1	N/A	0	Set Default Pause Duration between message writes, in 0.1 second increments (25.5 sec. Duration maximum).
15	"R"	N/A	0	N/A	0	Reset Sign (Clears any data previously entered into the sign RAM).
16	"S"	N/A	0	N/A	0	Stop Automatic Message Cycling (Overrides any previous "t" command)
17	"V"	N/A	0	N/A	0	Sign Firmware ID Request. Host will receive a sign message after positive response is received.
18	"W"	<Msg. Number>	1	N/A	0	Display Message # X on Sign.
19	"X"	<1st display #>, <Last display number>	2	N/A	0	Blank Display or panel #'s indicated.
20	"Z"	N/A	0	N/A	0	Turns OFF the Addressed Controller(s)

NOTES: * - indicates command is only available in INFOSign III based products.

Table 5.2.1: INFOSIGN Host to Display Application Level Commands (Continued)

Where:

1. <Message #> = The message (or "Page" when using the INFOSoft) number, 1 thru the maximum # of messages for the sign (in hex) for the respective message packet. Refer to section 5.2.3 for determining the maximum number of the messages which can be stored within a respective INFOSign master module.
2. <Binary Pixel Data> = bit mapped data for the respective INFOSIGN panel 30x240 "landscape mode" image, starting from 1st row (top) left to 1st row (top) right, 2nd row left to 2nd row right, ...30th row left to 30th row right. In other words, bit 0 must be in bit 7 position, bit 1 must be in bit 6 position, etc…
3. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet.
4. <# of msg. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display module will assume the message cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).
5. <Sense> = <00> Off (non-inverted), and <01> On (inverted)
6. ON Update Method – Byte to indicate how the controller is to present the respective partial image or message. The following codes apply (Only available with INFOSign III based product):

<01> = "*Wipe-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to "grow" to the right as the text message is presented. Unlike the "Scroll-ON" method, characters to the left already presented in a previous frame will be stationary while new characters within the image are presented to the right.
<02> = "*Scroll-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to move to the left as the text message is presented. Unlike the "Wipe" update method, all the characters within the image will move until the final frame of the image is presented.
<03> = "*Close-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" down from the top and "sliding up from the bottom, until the final frame provides the completed image joined together.
<04> = "*Open-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" up and down from the center, until the final frame provides the completed image which has been completely "opened" from the center.
<05> = "*Rotate-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is vertically rotating from the "Up-right" position, till it reaches an "Upside Down position", then the frame sequence is reversed till the final frame of the "Up-right" position is reached.
<06> = "Swell-ON' image presentation method. This presentation method is used as an alternate method of generating a static frame of data. Unlike the previous methods where multiple frames are presented to the user, this method will only generate a single frame representative of the text or graphic data provided at the interface. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Fade-ON", this update method will erase the screen section prior (hence destroying the previous image) to generating the image in the cumulative drive method, hence creating an illusion that the new image will grow in brightness or "Swell" from a blank background during the update.
<07> = "Fade-ON" image presentation method. Like the "Swell-ON", this method is used as an alternate method of generating a static frame of data. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Swell-ON", this update method will NOT erase the screen section prior to generating the image in the cumulative drive method, hence creating an illustion that the new image section will start to appear as the previous image starts to disappear, hence the new image will appear to "Fade" over the previous.

7. OFF Update Method – Byte to indicate how the controller is to remove the respective partial image or message. The following codes apply (Only available on INFOSign III based product):
<01> = "*Wipe-OFF*" image removal method (not valid for graphic messages). This removal method is the opposite of the "Wipe-ON" presentation method, with the characters being removed from right to left. The left-most character is the last character removed from the image area.
<02> = "*Scroll-OFF*" image removal method (not valid for graphic messages). This removal method is the same ast the "Scroll-ON" presentation method, with the characters being removed from right to left. The right-most character is the last character removed from the image area.
<03> = "*Open-OFF*" image removal method. This removal method is the opposite of the "Close-ON" presentation method, with the characters being removed from center towards the top and bottom edges of the image area. The middle-most rows of the image are the last portions removed from the image area.
<04> = "*Close-OFF*" image removal method. This removal method is the opposite of the "Open-ON" presentation method, with the characters being removed from the top and bottom edges towards the center of the image area. The top and bottom rows of the image are the last portions removed from the image area.

<05> = "*No OFF*" image presentation method. This number will inform the respective controller to NOT remove the respective partial image after the respective pause duration has expired.

8. <Dyn. Update Parameter 1> - This single byte of data is bit-mapped to indicate the following features:

   - Bit 7 – Implements a "2x" horizontal image "Strech" feature if activated (set to 1). **This feature is only valid** for partial screen, "4" (for "Fades" and "Swells" only) and "6", single frame, text based commands. For instance, if implemented on a fixed distance 9 point line of text, the characters on the display will be twice as wide, and the maximum number of characters per text line (normally 30 for this font. Refer to next sub-section for explanation on font selection & control) will be half as much, or 15.
   - Bit 6 through 2 – Reserved for future use.
   - Bit 1 & 0 (L.S.B.) - Implements a vertical Height Multiplier (ie. A 10 row partial image would be output on 20 rows of the respective display if Height Multiplier is "2x"). 2 bits in this location will implement the function. Maximum multiplier is "3x".
     00=1x (Normal, un-"Stretched")
     01=2x
     10=3x

5.2.1 Special "0", "2", "4" and "6" Font Control Characters

As indicated in the referenced command descriptions, the resident sign firmware has the capability to output text messages requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The sign modules also have the capability to output selected fonts and respond to font control characters. When the following special control characters (described in hexadecimal format within the "<>") are added within field 6 of the respective application layer messages, the following actions will occur (the following control characters must be the 1st data elements entered in field #6):

a) <0B> (VT) = Turn ON Tahoma 16 Single Line Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Tahoma 16 point one line per panel font format (Font can not be used with partial screen message format, "4" command designations).

b) <09> (HT) = Turn ON Times 16 Single Line Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Times Roman 16 point one line per panel font format (Font can not be used with partial screen message formats, or "2", "4" or "6" command designations).

c) <12> (DC2) = Turn ON Dual Panel Font Control character. All characters entered in field 6 will be output in the 15 row (dual line) font format (two lines of text per INFOSIGN panel). A carriage return (0D in hexadecimal without a line feed) will need to be entered between the two lines of text. **This is a ISO/IEC 8859-1 Compliant Font.**

d) <05> (ENQ) = Turn ON 3-Line/Panel Font Control character. All characters entered in field 6 will be output in the 10 row (3 line) font format (three lines of text per INFOSIGN panel). A carriage return (0D in hexadecimal without a line feed) will need to be entered between the lines of text. **This is a ISO/IEC 8859-1 Compliant Font.**

Notes: The absence of the above control characters will command the respective sign to output the respective panel in Letter Gothic 22 point; the default sign font (Font can not be used with partial screen message format, "4" command designations). **This is a ISO/IEC 8859-1 Compliant Font**

e) <13> (DC3) = Turn ON Underline control character. All characters entered in field 6 after the reception of this control character will be output in Underline format. The absence of

this control character will command the respective sign to output the panel with the default non-underlined mode.

5.2.2 Maximum Number of INFOSign Messages or Pages

As indicated in the "<Message #>" references in table 5.2.1, the INFOSign II (with revision H or later firmware) and INFOSign III products can store several messages or pages of data for the respective sign. To maximize the amount of messages to be stored in the INFOSign, the INFOSign uses a dynamic memory allocation method. Each message type assign for each panel with a given message (or "Page") is allocated a fixed memory space, based on the message type for the given panel. The dynamic memory allocation firmware uses a "link list" to place each panel image in consecutive section of sign RAM memory.

The following memory space allocations are assigned to the message types:

#	**Mixed Mode Message Type**	**Field #4 Command Designation(s)**	**Fixed Memory Space Allocation**
1.	Full Screen Text	"0"	105 Bytes / Panel image
2.	Full Screen Graphic	"1"	910 Bytes / Panel image
3.	Partial Screen Text, Dynamic Update Text, Flashing Text	"2", "4" and "6"	43 Bytes / Panel image
4.	Partial Screen Graphic, Dyanamic Update Partial Screen Graphic	"3" and "5"	610 Bytes / Panel image

Table 5.2.1: Sign Memory Space Allocations for INFOSign Message Data

As indicated previously, the INFOSign will load panel message data into memory sequentially, into the next available memory location. Since the INFOSign has a finite amount of memory storage capacity, the interface software ("INFOSoft", or equivalent) will need to either keep track of the amount of bytes used in the INFOSign to be sent to the respective INFOSign prior to transmitting, or the INFOSign will need to be "asked" the amount of available memory located for additional panel message storage. For instance, the maximum amount of message data that can be loaded into a 15153 INFOSign II master module, or a 10060 INFOSign II controller module can not exceed 43,680 bytes. The INFOSign III controller module can store up to 46,117 bytes of message data.

The following items pertain to maximizing the number of complete messages (or "pages") stored by an INFOSign:

1. Any two INFOSign master modules of the same type will have the same panel message storage capacity. However, an INFOSign containing only 4 display panels will be able to store more complete messages (or complete "pages") than a sign containing 12 display panels. For instance, if data is loaded into an INFOSign II sign for each of its panels, the 12 panel sign will be able to store 4 full screen graphic images for each panel {12 [panels/message] x 910 [bytes/panel] x 4 [messages] = 43,680 [bytes]}, whereas the 4 panel sign will be able to store 12 full graphic images for each of its panels {4 [panels/message] x 910 [bytes/panel] x 12 [messages] = 43,680 [bytes]}

2. To maximize the number complete messages to be loaded into a sign, use text-based messages (full, partial or flashing) whenever possible. For instance, as illustrated in the previous bullet, a 12 panel sign can store 4 complete full screen graphic messages, whereas the same sign can store 34 complete full screen text based message {12 [panels/message] x 105 [bytes/panel] x 34 [messages] = 42,840 [bytes], which is less than 43,680 [bytes]}.

5.2.2.1 Controlling and Determining the Message Count

The following application command are available to the mixed mode user to enable the interface to query (or "ask") the INFOSign the amount of available RAM memory left for additional message storage, and to enable the user to clear existing memory in the event the INFOSign dynamic message storage algorithm becomes fragmented.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Designation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"Y"	N/A	0	N/A	0	Output Mixed Mode Message storage availability from Sign. Host will receive message f/sign after positive response is received.
2	">" (3e in Hex)	N/A	0	N/A	0	Clear All Previously Received Message Data (overides any previous Message trigger or Automatic cycle command received).

Table 5.2.3.1: Additional INFOSIGN Memory Allocation/Query Commands

As indicated from the table above, the interface will receive a message like the following example after the INFOSign receives a "Y" command:

```
RAM Bytes Used = 0791
RAM Bytes Available = A2AF
RAM Fragmented = 0000
```

The quantities provided on each line after the introductory text is output in ASCCI hexadecimal format. As illustrated in the example above, the given INFOSign currently has 0791 (1937_{10}) bytes occupied with user message data, has A2AF ($41{,}647_{10}$) bytes available for additional message storage, and has 0 bytes wasted due to fragmentation of the message data.

On current INFOSign designs, message data fragmentation will only occur if the user changes an existing message (for a given message and panel number) from one mixed mode message type, to another as indicated in table 5.2.3. If the amount of RAM space, which is fragmented, becomes excessive, the user (or "interface") may wish to output a "Clear All Previously Received Message Data" command (">" command), followed by transmission of all the required data using the mixed mode commands indicated in table 5.2.2.

Another good application of the "Clear All Previously Received Message Data" command (">" command) is when the interface application starts a new transmission of data to an INFOSign, which may be in continuous automatic cycle mode. After the INFOSign receives the ">" command, the INFOSign will stop generation of any messages presently being driven onto the display panels, then clear memory allocated for message storage. After transmission of the ">" command, the interface should load new message data to the respective INFOSign master or controller module.

5.3 QUAD Application Layer Comand Descriptions

The following table outlines the available application level command sequences for the Kent Display new QUAD display modules. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. The application level command designations are case sensitive.
- N/A designates that no bytes are designated or inserted for respective field.

#	(Field 4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
% 1	"A"	“Sign Row 1, Msg 1 Text” EOT “Sign Row 2, Msg 1 Text”	Variable (320 Max)	ETX	1	Load Text-based Sign message #1 into memory, and display on sign.
% 2	"B"	“Sign Row 1, Msg 2 Text” EOT “Sign Row 2, Msg 2 Text”	Variable (320 Max)	ETX	1	Load Text-based Sign message #2 into memory, and display on sign.
% 3	"C"	“Sign Row 1, Msg 3 Text” EOT “Sign Row 3, Msg 2 Text”	Variable (320 Max)	ETX	1	Load Text-based Sign message #3 into memory, and display on sign.
% 4	"D"	“Sign Row 1, Msg 4 Text”, EOT DC1 DC3 “Sign Row 2, Msg 4 Text” DC3 DC1	Variable (320 Max)	ETX	1	Load Text-based Sign message #4 into memory, and display on sign.
% 5	"E"	“Sign Row 1, Msg 5 Text” EOT DC1 “Sign Row 2, Msg 5 Text” DC1	Variable (320 Max)	ETX	1	Load Text-based Sign message #5 into memory, and display on sign.
% 6	"F"	“Sign Row 1, Msg 6 Text”, EOT DC3 “Sign Row 2, Msg 6 Text” DC3	Variable (320 Max)	ETX	1	Load Text-based Sign message #6 into memory, and display on sign.
% 7	"G"	“Sign Row 1, Msg 7 Text” EOT DC1 “Sign Row 2, Msg 7 Text” DC1	Variable (320 Max)	ETX	1	Load Text-based Sign message #7 into memory, and display on sign.
% 8	"H"	<Message #> (8-23)	1	“Sign Row 1, Msg 8-23 Text”, EOT DC3 “Sign Row 2, Msg 8-23 Text” DC3 ETX	Variable (320 Max)	Load Text-based Sign message #8, 9, .. or 23 into memory, and display on sign.
% 9	"I"	N/A	0	N/A	0	Toggle the sense of image generation.
10	"P"	Duration (Hex)	1	N/A	0	Set Pause Duration between image writes, in 0.1 second increments (25.5 sec. Duration maximum).
11	"O”	N/A	0	N/A	0	Output ChLCD Temperature from Sign. Host will receive a 4 byte message f/sign after positive response is received.
12	"R”	N/A	0	N/A	0	Reset Sign (Clears any data previously entered into the sign RAM).
13	"S"	N/A	0	N/A	0	Stop Automatic Message Cycling (Overrides any previous "t" command)
% 14	"T"	<1st Msg. #>, <Last Msg. #>	2	<Msg_Sense Cmd_Byte1 – Msg_Sense Cmd_Byte 3>, <# of msg. Cycles>	4	Start Automatic Message Cycling for messages indicated, the specified image sense, and the amount of cycles specified (all #’s in hex)

Table 5.3.1A: Available Host Computer QUAD Application Layer Commands

#	(Field 4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
15	"U"	<# of Sign Rows> <# Characters/Row>	2	N/A	0	Inform Sign its configuration.
16	"V"	N/A	0	N/A	0	Sign Firmware ID Request. Host will receive a sign message after positive response is received.
17	"W"	<Message Number>	1	N/A	0	Display Message # X already loaded on Sign.
18	"X"	N/A	0	N/A	0	Blank Sign (all sign rows)
* 19	"Y"	<Msg. #>, <Row #>, <Invert Status>, <Update Method>	4	Sign Row 1 Text!, ETX	Variable	Load Text-based Sign message #X into memory. Does not display until "W" cmd. RX'ed.
* 20	"Z"	<1st Msg. #>, <Last Msg. #>, <# of msg. Cycles>	3	N/A	0	Start Automatic Message Cycling for messages indicated, the amount of cycles specified (all #'s in hex)
* 21	">" (3e in Hex)	N/A	0	N/A	0	Clear All Previously Received Message Data (overides any previous Message trigger or Automatic cycle command received).

* Indicates Y & Z command are used together in place of A-H and T commands, and are only available on QUADs (along with ">" command) with 13028 Revision C firmware, or later.
% indicates respective command should not be used for new designs. Y, Z & > command set should be used instead.

Table 5.3.1B: Available Host Computer QUAD Application Layer Commands (Continued)

Where:

1. Text data for all rows of a sign are to be combined into a single application layer message packet, where:

   a. EOT = End of Sign Row Text Command byte (04 Hex). An EOT byte received without any prior text data will inform the respective sign that data elements for the respective sign row are not to be changed for the respective message #. The text data for the last row of the sign is not to be terminated with an EOT character.
   b. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet.

2. <Msg_Sense Cmd_Byte1 – Msg_Sense Cmd_Byte3> = Three consecutive bit mapped binary bytes indicating the sense of display messages (Normal mode = Reflective text/image with transparent background) 1 through 23. A "1" bit in the any of the following bit assignments for the 3 bytes will command the respective message into inverted mode (transparent text/image with reflective background):

   - Bit 7 (Left-most, or M.S.B.) - Message #1 (Msg_Sense Cmd_Byte1), Message #9 (Msg_Sense Cmd_Byte2) & Message #17 (Msg_Sense Cmd_Byte3) image sense assignment.
   - Bit 6 - Message #2, Message #10 & Message #18 sense assignment.
   - Bit 5 - Message #3, Message #11 & Message #19 sense assignment.
   - Bit 4 - Message #4, Message #12 & Message #20 sense assignment.

- Bit 3 - Message #5, Message #13 & Message #21 sense assignment.
- Bit 2 - Message #6, Message #14 & Message #22 sense assignment.
- Bit 1 - Message #7, Message #15 & Message #23 sense assignment.
- Bit 0 (L.S.B.) - Message #7, Message #15 & Toggle message sense for all messages after each message cycle (Msg_Sense Cmd_Byte3). A 1 bit in this location will implement the function.

3. <# of msg. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display module will assume the message cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).

4. < Invert Status> = <00> Off (non-inverted), and <01> On (inverted)

5. <Update Method> - <00> Normal, and <01> Wipe (Note: Update methods must be common between rows within the same message or page number).

5.3.1 Special "A" through "H" & “Y” Command Text Control Characters

As implied in the "A" through "H" & “Y” command descriptions, the resident sign firmware has the capability to output text messages requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The sign modules also have the capability to change fonts and to "Underline" the host requested characters. When the following special control characters (described in hexadecimal format within the "<>") are added within field 6 of the "A" through "H" application layer messages, create the following actions:

a) <11> (DC1) = Toggle Font Control character. All characters entered in field 5 after the reception of this control character will have the opposite font available font than the characters preceding the character (The QUAD modules presently have 2 fonts available, Big ((Default)) and Courier-New 9 Pt.)

b) <13> (DC3) = Underline command toggle character. All characters entered in field 6 after the reception of this control character will have the opposite Underline characteristic than the characters preceding the character.

c) The “Y” commands do not toggle the Control character. Instead, the Control character will inform the sign if the Font/Underline is on or off. If the Control character is present (prior to remaining field 6 characters), the Command is on.

When the QUAD modules start outputting text characters dictated by a host "A" through “H” commands, the sign firmware will assume the requested characters are to be output in "NORMAL" text generation mode (Big font, not underlined).

5.4 1/8 VGA Controller Host to Display Application Level Field Descriptions

The following table outlines the available application level command sequences for the Kent Display 1/8 VGA modules. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. Elements designated between < and > characters are to be output in binary. The application level command designations are case sensitive.
- N/A designates that no bytes are designated or inserted for respective field.
- Typical Data elements are illustrated for field #6.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"A"	<Display #>, <Image #> *(1-1E Hex), <Sense>	3	DC1 DC4 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load Full Screen Text *(image #1-30) message into Memory (Does not display after valid message RX'ed).
2	"B"	<Display #>, <Image #> *(1-0F Hex), <Sense>	3	<Binary Pixel Data>	4800	Load Graphic *(Image #1-15) Data into memory (Does not display after valid message RX'ed).
3	"C" (reserved for future use)	<Display #>	x	<Compressed Binary Pixel Data>	Variable	NOT IMPLEMENTED at this Time.
4	"D"	<On/Off>	1	N/A	0	Update Image from center out on/off (default is ON). *This command does not apply and is not implemented if using 10120 controller module.*
5	"E"	N/A	0	N/A	0	Turn the display OFF.
**6	"F"	<Display #>, <Image #>, <First Row # to Alter> (1-155_{10}), <Sense>, ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	8	DC1 DC3 "**Disp 1 Msg 10 Row 1**" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after "W" command is received).
**7	"G"	<Display #>, <Image #>, First Row # to Alter, <Sense>, Number of Rows Altered (16_{10} Max), ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	9	<Binary Pixel Data>	Variable (480 Max)	Load Partial Screen Graphic image in Memory to be displayed in Dynamic Update format (Does Not display on ChLCD until after a "W" command is received).
**8	"H"	<Display #>, < Message #>, First Row # to Alter, Non-Invert Duration (in 0.1 increments), Invert Duration, # of Flash Cycles, Additional Future Parameter (1 byte not used)	7	DC1 DC3 "Flash Disp.1 Msg X Row 4" <ETX>	Variable (256 Bytes Max.)	Load Partial Screen Text message (1 line of text) to be "Flashed" in the display as indicated in the corresponding parameters in field # 5 (Does Not flash on ChLCD display until after a "W" command is received).
9	"I"	<Display #>, <Img #>*(0-1E Hex), <Sense>	3	N/A	0	Invert sense of image (Img # 0 is the Global sense for all images default is off)
**10	"J"	Duration (Hex)	1	N/A	0	Set Pause Duration between Auto cycle image writes, in 0.1 second increments.
**11	"K"	<Image #>, Duration (Hex)	2	N/A	0	Set Unique Pause Duration for Message or Image # indicated (in 0.1 second increments).
12	"N"	<Display #>, <Img. #>, <First Row # to Alter> (1-160 decimal), <Sense>	4	DC3 DC4 "*Change Disp.1 Msg X Row 4*" <ETX>	Variable (27 Max if 10072, and 50 if 10120 controller).	Change Partial Screen Text message in Memory (Creates 1 line of Text in memory Does not display after valid message RX'ed).

Table 5.4.1A: 1/8 VGA Controller Host to Display Application Level Commands

* indicates no Fixed number of Images are required if using dynamic memory storage capability of 10120 controller module, hence number limitations indicated do not appply.
** indicates application level command is only available with 10120 controller module.

	(Field 4)	Designation #2 (Field #5)		Data Elements (Field #6)		
12	"O"	N/A	0	N/A	0	Output ChLCD Temperature from Sign. (Host will receive a message f/sign after positive response is received)
13	"P"	Duration (Hex)	1	N/A	0	Set Pause Duration between Auto cycle image writes, in 1 second increments *(Use "J" command in place of this command if using 10120 controller module).*
14	"Q"	<On/Off>	1	N/A	0	State of sleep timer (default is ON after power-on or reset).
15	"S"	N/A	0	N/A	0	Stop Automatic Message Cycling
16	"T"	<Display #>, <1st Img. #>, <Last Img. #>, <# of img. Cycles>, <Tog. Img. Sence Byte>	5	N/A	0	Start Automatic Image Cycling using the parameters specified.
17	"U"	<Display #>, <Img #> *(1-0F Hex), First Row # to Alter, Number of Rows Altered (88 Max.), <Sense>	5	<Binary Pixel Data>	Variable (2400 Bytes Max.)	Change Graphic-based Partial Screen image in Memory (Does not display after valid message RX'ed).
18	"V"	N/A	0	N/A	0	Sign Firmware ID Request (Host will receive a sign message after positive response is received.
19	"W"	<Display #>, <Img. #>	2	N/A	0	Write the Image *(01 – 1E in hex) to the display
20	"X"	<Display #>	1	N/A	0	Blank Display.

* indicates no Fixed number of Images are required if using dynamic memory storage capability of 10120 controller module, hence number limitations indicated do not appply.

Table 5.4.1B: 1/8 VGA Controller Host to Display Application Level Commands (Continued)
Where:

1. <image #> = The image number for the respective package. The controller can store up to 30 images. The first 15 images can be either text or graphic type messages. Images 16-30 (10 - 1E Hex) can only be text-based images. *These stated fixed number and image type limitations do NOT apply if using the 10120 controller module to drive the respective 1/8 VGA display (refer to the "Maximum Number of 1/8 VGA Images" sub-section within this section of the document for details).*

2. <Display #> = <01>, other values reserved for future use.

3. <Binary Pixel Data> = Bit mapped data arranged with the most significant bit representing the left most pixel for the given byte. The image is loaded with the display in landscape mode, starting from the byte in the 1st row (top) right to 1st row (top) left, 2nd row right to 2nd row left, ...160th row right to 160th row left.

   I*f using the 10120 controller module, the same described image loading algorithm is used, however the data is loaded from left to right. Hence, image data is loaded with the display in landscape mode, starting from the byte in the 1st row (top) left most position, to 1st row (top) right most posion (30 bytes later), 2nd row left to 2nd row right, ...160th row left to 160th row right.*

4. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet.

5. <On/Off> = <00> Off
   <01> On

6. <First Row # to Alter, Number of Rows Altered> = First Row # of respective of 160x240 display to alter (1-160 decimal) and number of rows to alter. The byte count of pixelated data in field 6 of the "U" command must equal 40 times the Number of Rows Altered.

7. < Sense> = <00>Non-inverted
   <01>Inverted

8. <# of Img. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display module will assume the Image cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).

9. <Duration Between Images>, or <Pause Duration> = binary data (1 byte each) indicating their respective durations in 1 second increments (if using 10072 controller), or 0.1 second increments (if using the 10120 controller). A value of FF hex (255 decimal) would equate to a 255 second (10072 controller), or 25.5 second (10120 controller) duration.

10. <Tog. Img. Sence Byte> = ("Toggle Image Sence Byte") indicates during an automatic cycle process, if the corresponding module is to toggle the image sense for all images when the requested messages are repeated.

11. ON Update Method – Byte to indicate how the controller is to present the respective partial image or message. The following codes apply:
    - <01> = "*Wipe-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to "grow" to the right as the text message is presented. Unlike the "Scroll-ON" method, characters to the left already presented in a previous frame will be stationary while new characters within the image are presented to the right.
    - <02> = "*Scroll-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to move to the left as the text message is presented. Unlike the "Wipe" update method, all the characters within the image will move until the final frame of the image is presented.
    - <03> = "*Close-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" down from the top and "sliding up from the bottom, until the final frame provides the completed image joined together.
    - <04> = "*Open-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" up and down from the center, until the final frame provides the completed image which has been completely "opened" from the center.
    - <05> = "*Rotate-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is vertically rotating from the "Up-right" position, till it reaches an "Upside Down position", then the frame sequence is reversed till the final frame of the "Up-right" position is reached.
    - <06> = "Swell-ON' image presentation method. This presentation method is used as an alternate method of generating a static frame of data. Unlike the previous methods where multiple frames are presented to the user, this method will only generate a single frame representative of the text or graphic data provided at the interface (similiar to the normal partial screen message types). This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Fade-ON", this update method will erase the screen section prior (hence destroying the previous image) to

generating the image in the cumulative drive method, hence creating an illusion that the new image will grow in brightness or "Swell" from a blank background during the update.

<07> = "Fade-ON" image presentation method. Like the "Swell-ON", this method is used as an alternate method of generating a static frame of data. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Swell-ON", this update method will NOT erase the screen section prior to generating the image in the cumulative drive method, hence creating an illustion that the new image section will start to appear as the previous image starts to disappear, hence the new image will appear to "Fade" over the previous.

12. OFF Update Method – Byte to indicate how the controller is to remove the respective partial image or message. The following codes apply:
    - <01> = "*Wipe-OFF*" image removal method (not valid for graphic messages). This removal method is the opposite of the "Wipe-ON" presentation method, with the characters being removed from right to left. The left-most character is the last character removed from the image area.
    - <02> = "*Scroll-OFF*" image removal method (not valid for graphic messages). This removal method is the same ast the "Scroll-ON" presentation method, with the characters being removed from right to left. The right-most character is the last character removed from the image area.
    - <03> = "*Open-OFF*" image removal method. This removal method is the opposite of the "Close-ON" presentation method, with the characters being removed from center towards the top and bottom edges of the image area. The middle-most rows of the image are the last portions removed from the image area.
    - <04> = "*Close-OFF*" image removal method. This removal method is the opposite of the "Open-ON" presentation method, with the characters being removed from the top and bottom edges towards the center of the image area. The top and bottom rows of the image are the last portions removed from the image area.
    - <05> = "*No OFF*" image presentation method. This number will inform the respective controller to NOT remove the respective partial image after the respective pause duration has expired.

13. <Dyn. Update Parameter 1> - This single byte of data is bit-mapped to indicate the following features:

    - Bit 7 through 3 – Reserved for future use.
    - Bit 0 – Bit 2 - Implement a 1x to 8x Image height feature. For instance, a 10 row partial image would be output on 80 rows (or half the display) of the respective display if all 3 bits are set high. The same 10 row image can be output on 40 rows of the display if bits 0 and 1 are set high. Clearing all bits will output the image to the normal, 1x mode where the 10 row image will be output on 10 rows of the display.

Note: Changing the contents of an image while writing the image to the screen may result in an undetermined image to be displayed.

5.4.1 "A", "N", "F" and "H" Command Text Control Characters

As implied in the referenced command descriptions, the resident controller module firmware has the capability to output text Images requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The controller modules also have the capability to change fonts and respond to font control characters. When the following special control characters (described in hexadecimal format within the "<>") are added within field 6 of the "A", "N", "F" or "H" application layer commands, the following actions are implemented (the following control characters must be the 1st data elements entered in field #6):

16x30 Pixel Characters: Each of the following described font controls will occupy a 16x30 (columns x rows pixel space/character. Up to 15 characters/text line and up to 8 text lines/display image can be output using these font controls.

a) <0B> (VT) = Turn ON Tahoma 16, 30 row Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Tahoma 16 point font format. A carriage return (0D in hexadecimal) will need to be entered between each text line if used with full screen text command.

b) <09> (HT) = Turn ON Times 16, 30 row Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Times Roman 16 point font format. A carriage return (0D in hexadecimal) will need to be entered between each text line if used with full screen text command.

10x15 Pixel Characters: The font control below will occupy a 10x15 (columns x rows) pixel space/character. Up to 24 characters/text line and up to 16 text lines/display image can be output using this font control.

c) <12> (DC2) = Turn ON Fixed Distance 11, 15 row Font Control character. All characters entered in field 6 will be output in the dual line font format. A carriage return (0D in hexadecimal) will need to be entered between each text line if used with full screen text command.

8x10 Pixel Characters: The font control below will occupy a 8x10 (columns x rows) pixel space/character. Up to 30 characters/text line and up to 16 text lines/display image can be output using this font control *(Only available with 10120 controller module)*.

d) <11> (DC2) = Turn ON Fixed Distance 9 point Font Control character. All characters entered in field 6 will be output in the Fixed distance 9 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a full screen text command.

6x8 Pixel ("5x7") Characters: The font control below will occupy a 6x8 (columns x rows) pixel space/character. Up to 40 characters/text line and up to 20 text lines/display image can be output using this font control *(Only available with 10120 controller module)*.

e) <05> (ENQ) = Turn ON 5x7 character Font Control characters. All characters entered in field 6 will be output in this format after reception. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a full screen text command.

Note: The absence of the above control characters will command the respective controller to output the respective image in Letter Gothic 22 point, 16x30 row font; the default font.

f) <13> (DC3) = Turn ON Underline control character. All characters entered in field 6 after the reception of this control character will be output in Underline format. The absence of this control character will command the respective sign to output the image with the default non-underlined mode.

g) <14> (DC4) = Turn ON *Italicized* control character. All characters entered in field 6 after the reception of this control character will be output in *Italicized* format. The absence of this control character will command the respective sign to output the image with the default non-Italicized mode. *Italicized font controls are NOT supported by 10120 controller module.*

5.4.2 1/8 Controller Response Protocol (10072 Controller Only)

The 1/8 VGA controller responds to Images the same as in section 4.2 accept the negative response contains an added byte (Error Code) to report the reason the Image received a negative response. The negative response format would appear as follows:

Field Description	NAK	Message #	Error Code	Message Checksum	Carriage Return	Line Feed
Packet Field #	1	2	3	4	5	6
Byte #/ Field	1	1	1	1	1	1

Table 5.4.2 1/8 Controller Negative Response Message Packet Format
Where:

1. *NAK* = ASCII acknowledgment character; 15 (Hex)
2. *Message #* = Message number of the previously transmitted host message package, 0-255 possibilities.
3. *Error Code* = <01> Checksum error
   <02> Buffer full
   <03> controller busy
   <04 - FF> reserved for debugging and future use.
4. *Message Checksum* = Equals the least significant 8 bits of the summation of all elements contained in message packet fields 1 and 2.
5. *Carriage return* = ASCII carriage return, or 0D in hexadecimal.
6. *Line feed* = ASCII line Feed, or 0A in hexadecimal.

5.4.3 Maximum Number of 1/8 VGA Image Storage Capacity (10120 Controller Only)

As indicated in the "<Image #>" references in table 5.4.1, the 1/8 VGA 10120 controller module can store several images, messages or pages of data for the respective display number assigned to the controller module. To maximize the amount of messages to be stored in the controller module, the controller uses a dynamic memory allocation method. Each message type for each display within a given message (or "Page") is allocated a fixed memory space, based on the message type for the given display. The dynamic memory allocation firmware uses a "link list" to place each display image in consecutive section of the controller RAM memory.

The following memory space allocations are assigned to the available message types:

#	**Message Type**	**Field #4 Command Designation(s)**	**Fixed Memory Space Allocation**
1.	Full Screen Text	"A"	833 Bytes / Display image
2.	Full Screen Graphic	"B"	4814 Bytes / Display image
3.	Partial Screen Text, Dynamic Update Text, Flashing Text	"N", "F" and "H"	55 Bytes / Display image
4.	Partial Screen Graphic	"U"	3014 Bytes / Display image
5	Dyanamic Update Partial Screen Graphic	"G"	494 Bytes / Display image

Table 5.4.3: Controller Memory Space Allocations for Different Types of Message Data

As indicated previously, the 10120, 1/8 VGA controller module will load display image data into memory sequentially, into the next available memory location. Since the controller has a finite amount of memory storage capacity, the interface software ("1/8VGASoft", or equivalent) will need to either keep track of the amount of bytes used in the respective controller address prior to transmitting, or the controller will need to be "asked" the amount of available memory for an additional display message storage. For instance, the maximum amount of message data that can be loaded into a 1/8 VGA controller can not exceed 43,817 bytes.

The following items pertain to maximizing the number of complete messages stored by a 1/8 VGA, 10120 controller module:

- To maximize the number complete messages to be loaded into a controller module, use text-based messages (full, partial, flashing or partial screen text dynamic update) whenever possible. For instance, the 1/8 VGA controller can store only 9 complete full screen graphic messages {4814 [bytes/message] x 9 [messages] = 43,326 [bytes]}, whereas the same controller can store 52 complete full screen text based message {833 [bytes/message] x 52 [messages] = 43,316 [bytes]}, which is less than the 43,817 bytes capacity. Additional message storage capacity can be obtained if partial screen text images are used whenever possible.

5.4.3.1 Controlling and Determining the Message Count (10120 Controller Only)

The following application commands are available to the user to enable the interface to query (or "ask") the 1/8 VGA controller the amount of available RAM memory left for additional message storage, and to enable the user to clear existing memory in the event the controller dynamic message storage algorithm becomes fragmented.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Designation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"M"	N/A	0	N/A	0	Output "M"essage storage availability from controller. Host will receive message f/controller after positive response is received.
2	">" (3e in Hex)	N/A	0	N/A	0	Clear All Previously Received Message Data (overides any previous Message trigger or Automatic cycle command received).

Table 5.4.4: Additional Controller Memory Allocation/Query Commands

As indicated from the table above, the interface will receive a message like the following example after the 320x80 controller receives a "M" command:

```
RAM Bytes Used = 0791
RAM Bytes Available = A398
RAM Fragmented = 0000
```

The quantities provided on each line after the introductory text is output in ASCCI hexadecimal format. As illustrated in the example above, the given 1/8 VGA controller currently has 0791 (1937_{10}) bytes occupied with user message data, has A398 ($41{,}880_{10}$) bytes available for additional message storage, and has 0 bytes wasted due to fragmentation of the message data.

On current 1/8 VGA controller designs, message data fragmentation will only occur if the user changes an existing message (for a given message and display number) from one message type, to another as indicated in table 5.4.3. If the amount of RAM space, which is fragmented, becomes excessive, the user (or the "interface") may wish to output a "Clear All Previously Received Message Data" command (">"

command), followed by transmission of all the required message data using the commands indicated in the previous table.

Another good application of the "Clear All Previously Received Message Data" command (">" command) is when the interface application starts a new transmission of data to an 1/8 VGA controller, which may be in continuous automatic cycle mode. After the controller receives the ">" command, the controller will stop generation of any messages presently being driven onto the displays, then clear memory allocated for message storage. After transmission of the ">" command, the interface should load new message data to the respective controller module.

5.5 128x32 Display Application Level Field Descriptions

The following table outlines the available application level command sequences for the Kent Displays 128x32 display modules. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. Elements designated between < and > characters are to be output in binary or Hexidecimal. The application level command designations are case sensitive (field #4).
- N/A designates that no bytes are designated or inserted for respective field.
- Typical Data elements are illustrated for field #6.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"A"	<Display #>, <Img. #> (1-48), <Sense>, <Pause Duration>	4	DC1 DC4 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load Text (image #1-48) into Memory (Does not display until valid "W" command received).
2	"B"	<Display #>, <Img #>(1-48), <Sense>, <Pause Duration>	4	<Binary Pixel Data> (for full screen image)	512	Load Graphic (Image #1-48) data into (Does not display until valid "W" command received).
3	"C"	<Display #>, <Img. #>, <First Row # to Alter> (1-25$_{10}$), <Sense>, <Pause Duration>	5	DC3 DC4 "*Change Disp.1 Msg X Row 4*" <ETX>	Variable	Load Partial Screen Text message in Memory to be updated in normal mode (Creates 1 line of Text on Display after corresponding "W" command is received).
4	"D"	<Display #>, <Img #> (1-48), First Row # to Alter, Number of Rows Altered (32$_{10}$ Max), <Sense>, <Pause Duration>	6	<Binary Pixel Data>	Variable (512 Bytes Max.)	Change Graphic Partial Screen image in Memory to be output in "Shutter" mode (Does not display until valid "W" command received).
5	"E"	<Display #>, <Img. #>, <First Row # to Alter> (1-24 decimal), <Sense>, ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	8	DC3 DC4 "Change Disp.1 Msg X Row 4" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after "W" command is received). Can NOT be used for large Fonts!
6	"F"	<Display #>, <Img #>, First Row # to Alter, <Sense>, Number of Rows Altered (16$_{10}$ Max), ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	9	<Binary Pixel Data>	Variable (256 Bytes Max.)	Load Partial Screen Graphic image in Memory in indicated Dynamic Update format (Does Not display on ChLCD until after a "W" command is received).
7	"G"	<Display #>, <Img #>, First Row # to Alter, Non-Invert Duration (in 0.1 increments), Invert Duration, # of Flash Cycles, Additional Future Parameter (1 byte not used)	7	DC3 DC4 "Flash Disp.1 Msg X Row 4" <ETX>	Variable (256 Bytes Max.)	Load Partial Screen Text message to be "Flashed" in the display as indicated in the corresponding parameters in field # 5/ (Does Not flash on ChLCD display until after a "W" command is received).
8	"H"	N/A	0	N/A	0	Turns OFF the display(s) off.
9	"I"	<Display #>, <Img #>(1-30 Hex), <Sense>	3	N/A	0	Invert sense of image (Img # 0 is the Global sense for all images. Default is non-Invert state)
10	"O"	N/A	0	N/A	0	Output ChLCD Temperature from Sign. (Host will receive a message f/sign after positive response is received)
11	"P"	Duration (Hex)	1	N/A	0	Set Global Pause Duration between Auto cycle image writes, in 0.1 second increments.
12	"Q"	<On/Off>	1	N/A	0	State of sleep timer (default is on after power-on or reset).
13	"S"	N/A	0	N/A	0	Stop Automatic Message Cycling
14	"T"	<Display #>, <1st Img. #>, <Last Img. #>, <# of img. Cycles>, <Tog. Img. Sence Byte>	5	N/A	0	Start Automatic Image Cycling using the parameters specified.
15	"U"	<Image Seq. Number>	1	N/A	0	Ouput (Default or Custom) Image Sequence stored in ROM
16	"V"	N/A	0	N/A	0	Sign Firmware ID Request (Host will receive a sign message after positive response is received.
17	"W"	<Display #>, <Img. #>	2	N/A	0	Write full, Partial, Dynamic or Flashing Image (1 - 48) to the selected display(s).
18	"X"	<Display #>	1	N/A	0	Blank Display(s).

Table 5.5.1: 128x32 Display Controller Host to Display Application Level Commands

Where:

6. <image #> = The image number for the respective package. The controller can store up to 48 text, graphic full or partial screen images. for each of the possible 2 displays/controller.
7. <Display #> = <01> for front display, <02> for back display or <03> for both.
8. <Binary Pixel Data> = Bit mapped data arranged with the most significant bit representing the left most pixel for the given byte. The image is loaded with the display in landscape mode, starting from the first byte in the 1st row (top) left to 1st row (top) right, 2nd row left to 2nd row right, ...32nd row right to 32nd row left. If interfacing to a 15240 display module, all bytes of data entered are to be mirrored (bit 0 placed in bit 7 position, bit 1 placed in bit 6 position, ect..). If interfacing to any other 128x32 display type (i.e. 15373 or 128x32 display with the "ZIF" header), the data entered is NOT mirrored.
9. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet.
10. <On/Off> = <00> Off, <01> On
11. <First Row # to Alter, Number of Rows Altered> = First Row # of respective of 128x32 display to alter (1-32 decimal) and number of rows to alter. The byte count of pixelated data in field 6 of the "D" & "F" command must equal 16 times the Number of Rows Altered.
12. < Sense> = Image Sense, <00>Non-inverted, <01>Inverted.
13. ON Update Method – Byte to indicate how the controller is to present the respective partial image or message. The following codes apply:
    <01> = "*Wipe-ON*" image presentation method (not valid for graphic "F" command messages). Characters within the image will appear to "grow" to the right as the text message is presented. Unlike the "Scroll-ON" method, characters to the left already presented in a previous frame will be stationary while new characters within the image are presented to the right.
    <02> = "*Scroll-ON*" image presentation method (not valid for graphic "F" command messages). Characters within the image will appear to move to the left as the text message is presented. Unlike the "Wipe" update method, all the characters within the image will move until the final frame of the image is presented.
    <03> = "*Close-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" down from the top and "sliding up from the bottom, until the final frame provides the completed image joined together.
    <04> = "*Open-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" up and down from the center, until the final frame provides the completed image which has been completely "opened" from the center.
    <05> = "*Rotate-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is vertically rotating from the "Up-right" position, till it reaches an "Upside Down position", then the frame sequence is reversed till the final frame of the "Up-right" position is reached.
    <06> = "Swell-ON' image presentation method (Only available on 10120 controller modules w/ revision E firmware, or later). This presentation method is used as an alternate method of generating a static frame of data. Unlike the previous methods where multiple frames are presented to the user, this method will only generate a single frame representative of the text or graphic data provided at the interface (similar to the "A" - "D" commands). This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Fade-ON", this update method will erase the screen section prior (hence destroying the previous image) to generating the image in the cumulative drive method, hence creating an illusion that the new image will grow in brightness or "Swell" from a blank background during the update.
    <07> = "Fade-ON" image presentation method (Only available on 10120 controller modules w/ revision E firmware, or later). Like the "Swell-ON", this method is used as an alternate method of generating a static frame of data. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Swell-ON", this

update method will NOT erase the screen section prior to generating the image in the cumulative drive method, hence creating an illusion that the new image section will start to appear as the previous image starts to disappear; hence the new image will appear to "Fade" over the previous.

14. OFF Update Method – Byte to indicate how the controller is to remove the respective partial image or message. The following codes apply:
    <01> = "*Wipe-OFF*" image removal method (not valid for graphic "F" command messages). This removal method is the opposite of the "Wipe-ON" presentation method, with the characters being removed from right to left. The left-most character is the last character removed from the image area.
    <02> = "*Scroll-OFF*" image removal method (not valid for graphic "F" command messages). This removal method is the same ast the "Scroll-ON" presentation method, with the characters being removed from right to left. The right-most character is the last character removed from the image area.
    <03> = "*Open-OFF*" image removal method. This removal method is the opposite of the "Close-ON" presentation method, with the characters being removed from center towards the top and bottom edges of the image area. The middle-most rows of the image are the last portions removed from the image area.
    <04> = "*Close-OFF*" image removal method. This removal method is the opposite of the "Open-ON" presentation method, with the characters being removed from the top and bottom edges towards the center of the image area. The top and bottom rows of the image are the last portions removed from the image area.
    <05> = "*No OFF*" image presentation method. This number will inform the respective controller to NOT remove the respective partial image after the respective pause duration has expired.

15. <# of Img. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display module will assume the Image cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).
16. <Duration Between Images> = (Gobal used in "P" command) binary data (1 byte each) indicating their respective durations in 0.1 second increments. A value of FF hex (255 decimal) would equate to a 25.5 second duration.
17. <Tog. Img. Sence Byte> = ("Toggle Image Sence Byte") indicates during an automatic cycle process, if the corresponding module is to toggle the image sense for all images when the requested messages are repeated. This byte follows the stated <00> Off, <01> On hex format.
18. <Pause Duration> = (used in "A" through "F" commands) indicates the number of 0.1 second increments to allow the respective message (full, partial or dynamic update) to remain on the display during autocycle or a simple dynamic update trigger operation. If a 0 (in hex) is received, the pause duration will default to the global duration dictated by the "P" command, or the default duration programmed by the respective controller during startup.
19. <Tog. Img. Sence Byte> - indicates during an automatic cycle process, if the corresponding module is to toggle the image sense for all images when the requested messages are repeated. This byte follows the stated <00> Off, <01> On (hex format).
20. <Dyn. Update Parameter 1> - This single byte of data is bit-mapped to indicate the following features:

    - Bit 7 (Left-most, or M.S.B.) - Output the dynamic image in Low "flicker" mode. A 1 in this field will command the respective controller to minimize the "Flickering" while the frames of data are displayed.
    - Bits 6 through 4 - Reserved for future use.
    - Bits 3 through 1 - Used slow down the respective "Frame Rate" for the dynamic image (ie. providing a "011" would slow down the frame rate by a factor of 3. Providing a "111"

would slow down the frame rate by a factor of 7). Providing 0's in these field will command the respective controller to output the dynamic image at the fastest rate for the respective temperature.

- Bit 0 (L.S.B.) - Implement 2x image height feature (ie. A 10 row partial image would be output on 20 rows of the respective display). A 1 bit in this location will implement the function.

Presenting a 0 hex for this byte will command the respective controller to output the dynamic image in the fastest frame rate possible, and without the 2x feature.

Note: Changing the contents of an image while writing the image to the screen could result in an undetermined image to be displayed.

5.5.1 "A", “C”, “E” and “G” Command Text Control Characters

As implied in the respective command description, the resident sign firmware has the capability to output text Images requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The sign modules also have the capability to change fonts using font control characters. When the following special control characters (described in hexadecimal format within the "<>") are added within field 6 of the application layer packets to create the following actions (the following control characters must be prior to the displayable data elements entered in field #6):

Four Line Font:

a) <14> (DC4) = Turn ON 5x7 pixel control character. All characters entered in field 6 after the reception of this control character will be output in 5 (wide) byb 7 (high) character format. A carriage return (0D in hexadecimal) will be needed to be entered between the possible 4 lines of text. 21 characters per line are allowed with this font.

Two Line Font:

b) <11> (DC1) = Turn Fixed Distance 11 Pt. Font Control character. All characters entered in field 6 will be output in the modified fixed distance 11 point format 8 (wide) by 9 (high) character format. A carriage return (0D in hexadecimal) will be needed to be entered between the 2 lines of text.

One Line Fonts (The following fonts allow a maximum of 8 character for the display):

c) <0B> (VT) = Turn ON Tahoma 16 Single Line Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Tahoma 16 point font format.
d) <09> (HT) = Turn ON Times 16 Single Line Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Times Roman 16 point font format.
e) <12> (DC2) = Turn ON Letter Gothic 22 Single Line Font Control character. All characters entered in field 6 will be output with the modified Letter Gothic 22 font format.

Note: The absence of the above control characters will command the respective sign to output the respective display in the default modified fixed distance 9 point, 3 lines/display font format. 16 characters per line are allowed with this font.

f) <13> (DC3) = Turn ON Underline control character. All characters entered in field 6 after the reception of this control character will be output in Underline format. The absence of this control character will command the respective sign to output the display with the default non-underlined mode.

5.6 320x80 Module Host to Display Application Level Field Descriptions

The following table outlines the available application level command sequences for the Kent Display 320x80 display modules. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. The application level command designations (field #4) are case sensitive.
- N/A designates that no bytes are designated or inserted for respective field.
- Typical Data elements are illustrated for field #6.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"0" (ASCCI Zero)	<Display #>, <Message. #>, <Sense>	3	DC2 DC4 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load full screen Text into Memory (Does not display until valid "T" command received).
2	"1"	<Display #>, <Message. #>, <Sense>	3	<Binary Pixel Data> (for full screen image)	3200 bytes	Load Graphic data into (Does not display until valid "T" command received).
3	"2"	<Display #>, <Message #>, <First Row # to Alter> (1-74_{10}), <Sense>	4	DC2 DC4 "*Change Disp.1 Msg X Line 3*" <ETX>	Variable	Load Partial Screen Text message in Memory to be updated in normal mode (Creates 1 line of Text on Display after corresponding "T" command is received).
4	"3"	<Display #>, <Message #>, First Row # to Alter, Number of Rows Altered (54_{10} Max), <Sense>	5	<Binary Pixel Data>	Variable (2160 Max)	Load Partial Screen graphic image in Memory to be output in Normal mode (Does not display until valid "T" command received).
5	"4"	<Display #>, <Message #>, <First Row # to Alter> (1-74_{10}), <Sense>, ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	8	DC2 DC4 "*Change Disp.1 Msg X Line 3*" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after "T" command is received).
6	"5"	<Display #>, <Message #>, First Row # to Alter, <Sense>, Number of Rows Altered (16_{10} Max), ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	9	<Binary Pixel Data>	Variable (640 Max)	Load Partial Screen Graphic image in Memory to be displayed in Dynamic Update format (Does Not display on ChLCD until after a "T" command is received).
7	"6"	<Display #>, < Message #>, First Row # to Alter, Non-Invert Duration (in 0.1 increments), Invert Duration, # of Flash Cycles, Additional Future Parameter (1 byte not used)	7	DC3 DC2 "Flash Disp.1 Msg X Row 4" <ETX>	Variable (256 Bytes Max.)	Load Partial Screen Text message (1 line of text) to be "Flashed" in the display as indicated in the corresponding parameters in field # 5 (Does Not flash on ChLCD display until after a "T" command is received).
8	"7"	<Display #>, <Message #>, <Sense>	3	N/A	0	Set/Clear Invert sense of Display image for respective Msg. #.
9	"8"	<Message #>, Duration (Hex)	2	N/A	0	Set Unique Pause Duration for respective message for Auto cycle operation (in 0.1 second increments).
10	"9"	<1st Message #>, <Last Message #>, <# of Message Cycles>, <Tog. Img. Sence Byte>	4	N/A	0	Start Automatic Image Cycling using the parameters specified.
11	":" (3B Hex)	N/A	0	N/A	0	Ask controller for display count. Host will receive message f/controller after positive response is received.
12	"A"	<On/Off>	1	N/A	0	Turn OFF/ON Sleep Timer (ON after reset or power-up)
13	"B"	<1st display #>, <Last display number>	2	N/A	0	"B"lank Display #'s indicated.
14	"C"	N/A	0	N/A	0	Output "C"hLCD Temperature from Sign. Host will receive a temperature text f/controller after positive response is received.
15	"I"	N/A	0	N/A	0	Sign Firmware "I"D Request. Host will receive a sign message after positive response is received.

Table 5.6.1: 320x80 Host to Display Application Level Commands

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
16	"O"	N/A	0	N/A	0	Turns "O"FF the Addressed Controller(s)
17	"P"	Duration (Hex)	1	N/A	0	Set Default "P"ause Duration between message writes, in 0.1 second increments (25.5 sec. Duration maximum).
18	"R"	N/A	0	N/A	0	"R"eset Controller (Clears any data previously entered into the sign RAM).
19	"S"	N/A	0	N/A	0	"S"top Automatic Message Cycling (Overrides any previous "9" command received)
20	"T"	<Msg. Number>	1	N/A	0	Generate, or "T"rigger Message # X on Display(s).

Table 5.6.1: 320x80 Host to Display Application Level Commands (Continued)

Where:

1. <Msg. #> = The message number, 1 thru the maximum # of messages for the displays (in hex) for the respective message packet. Refer to section 5.6.2 for determining the maximum number of the messages which can be stored within a respective 320x80 controller module.
2. <Binary Pixel Data> = bit mapped data for the respective 320x80 "landscape mode" image, starting from 1st row (top) left to 1st row (top) right, 2nd row left to 2nd row right, ...last row left to last row right.
3. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet.
4. <# of msg. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display module will assume the message cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).
4. <Sense> = <00> Off (non-inverted), and <01> On (inverted)
5. <On/Off> = <00> Off, <01> for On.
6. ON Update Method – Byte to indicate how the controller is to present the respective partial image or message. The following codes apply:
   <01> = "*Wipe-ON*" image presentation method (not valid for graphic "F" command messages). Characters within the image will appear to "grow" to the right as the text message is presented. Unlike the "Scroll-ON" method, characters to the left already presented in a previous frame will be stationary while new characters within the image are presented to the right.
   <02> = "*Scroll-ON*" image presentation method (not valid for graphic "F" command messages). Characters within the image will appear to move to the left as the text message is presented. Unlike the "Wipe" update method, all the characters within the image will move until the final frame of the image is presented.
   <03> = "*Close-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" down from the top and "sliding up from the bottom, until the final frame provides the completed image joined together.
   <04> = "*Open-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" up and down from the center, until the final frame provides the completed image which has been completely "opened" from the center.
   <05> = "*Rotate-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is vertically rotating from the "Up-right" position, till it

reaches an “Upside Down position”, then the frame sequence is reversed till the final frame of the “Up-right” position is reached.

<06> = "Swell-ON' image presentation method. This presentation method is used as an alternate method of generating a static frame of data. Unlike the previous methods where multiple frames are presented to the user, this method will only generate a single frame representative of the text or graphic data provided at the interface. This method will automatically use the “*No OFF*” OFF update method, outlined in the next bullet text. Unlike the "Fade-ON", this update method will erase the screen section prior (hence destroying the previous image) to generating the image in the cumulative drive method, hence creating an illusion that the new image will grow in brightness or "Swell" from a blank background during the update.

<07> = "Fade-ON" image presentation method. Like the "Swell-ON", this method is used as an alternate method of generating a static frame of data. This method will automatically use the “*No OFF*” OFF update method, outlined in the next bullet text. Unlike the "Swell-ON", this update method will NOT erase the screen section prior to generating the image in the cumulative drive method, hence creating an illustion that the new image section will start to appear as the previous image starts to disappear, hence the new image will appear to "Fade" over the previous.

7. OFF Update Method – Byte to indicate how the controller is to remove the respective partial image or message. The following codes apply:
   <01> = “*Wipe-OFF*” image removal method (not valid for graphic “F” command messages). This removal method is the opposite of the “Wipe-ON” presentation method, with the characters being removed from right to left. The left-most character is the last character removed from the image area.
   <02> = “*Scroll-OFF*” image removal method (not valid for graphic “F” command messages). This removal method is the same ast the “Scroll-ON” presentation method, with the characters being removed from right to left. The right-most character is the last character removed from the image area.
   <03> = “*Open-OFF*” image removal method. This removal method is the opposite of the “Close-ON” presentation method, with the characters being removed from center towards the top and bottom edges of the image area. The middle-most rows of the image are the last portions removed from the image area.
   <04> = “*Close-OFF*” image removal method. This removal method is the opposite of the “Open-ON” presentation method, with the characters being removed from the top and bottom edges towards the center of the image area. The top and bottom rows of the image are the last portions removed from the image area.
   <05> = “*No OFF*” image presentation method. This number will inform the respective controller to NOT remove the respective partial image after the respective pause duration has expired.
8. <Dyn. Update Parameter 1> - This single byte of data is bit-mapped to indicate the following features:

   - Bit 7 – Implements a “2x” horizontal image “Strech” feature if activated (set to 1). **This feature is only valid** for partial screen, Dynamic Update, single frame, text based “Fades” or “Swells”. For instance, if implemented on a fixed distance 9 point line of text, the characters on the display will be twice as wide, and the maximum number of characters per text line (normally 40 for this font. Refer to next sub-section for explanation on font selection & control) will be half as much, or 20.
   - Bit 6 through 3 – Reserved for future use.
   - Bit 0 – Bit 2 - Implement a “1x” to “8x” Image height or vertical “Stretch” feature. For instance, a 10 row partial image would be output on all 80 rows of the respective display if all 3 bits are set high. The same 10 row image can be output on 40 rows of the display if bits 0 and 1 are set high. Clearing all bits will output the image to the normal, 1x mode where the 10 row image will be output on 10 rows of the display.

Note: Changing the contents of an image while writing the image to the screen could result in an undetermined image to be displayed.

5.6.1 Special “0”, “2”, “4” and “6” Command Text Control Characters

As indicated in the referenced command descriptions, the resident sign firmware has the capability to output text messages requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The sign modules also have the capability to output selected fonts and respond to font control characters. When the following special control characters (described in hexadecimal format within the "<>") are added within field 6 of the respective application layer messages, the following actions will occur (the following control characters must be the 1st data elements entered in field #6):

16x30 Pixel Characters: Each of the following described font controls will occupy a 16x30 (columns x rows) pixel space/character. Up to 20 characters/text line and up to 2 text lines/display image can be output using these font controls.

a) <0B> (VT) = Turn ON Tahoma 16 Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Tahoma 16 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a “0” command.

b) <09> (HT) = Turn ON Times 16 Font Control Character. All characters entered in field 6 after the reception of this control character will be output with the modified Times Roman 16 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a “0” command.

c) <11> (DC1) = Turn ON Letter Gothic 22 Font Control character. All characters entered in field 6 after the reception of this control character will be output with the modified Letter Gothic 22 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a “0” command.

10x15 Pixel Characters: The font control below will occupy a 10x15 (columns x rows) pixel space/character. Up to 32 characters/text line and up to 5 text lines/display image can be output using this font control.

d) <12> (DC2) = Turn ON Fixed Distance 11 point Font Control character. All characters entered in field 6 will be output in the Fixed distance 11 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a “0” command.

6x8 Pixel Characters: The font control below will occupy a 6x8 (columns x rows) pixel space/character. Up to 53 characters/text line and up to 10 text lines/display image can be output using this font control.

e) <05> (ENQ) = Turn ON 5x7 character Font Control characters. All characters entered in field 6 will be output in this format after reception. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for a “0” command.

Notes: The absence of the above control characters will command the respective display controller to output the respective display in Fixed Distance 9 point; the default display font. With the

default font, each character will occupy a 8x10 pixel space. Up to 40 characters/text line and up to 8 text lines/display can be output using this default font.

f) <13> (DC3) = Turn ON Underline control character. All characters entered in field 6 after the reception of this control character will be output in Underline format. The absence of this control character will command the respective sign to output the display with the default non-underlined mode.

5.6.2 Maximum Number of 320x80 Messages

As indicated in the "<Message #>" references in table 5.6.1, the 320x80 controller module can store several messages or pages of data for the respective number of display assign to the controller module. To maximize the amount of messages to be stored in the controller module, the controller uses a dynamic memory allocation method. Each message type assign for each display with a given message (or "Page") is allocated a fixed memory space, based on the message type for the given display. The dynamic memory allocation firmware uses a "link list" to place each display image in consecutive section of the controller RAM memory.

The following memory space allocations are assigned to the available message types:

#	Message Type	Field #4 Command Designation(s)	Fixed Memory Space Allocation
1.	Full Screen Text	"0"	553 Bytes / Display image
2.	Full Screen Graphic	"1"	3214 Bytes / Display image
3.	Partial Screen Text, Dynamic Update Text, Flashing Text	"2", "4" and "6"	68 Bytes / Display image
4.	Partial Screen Graphic	"3"	2174 Bytes / Display image
5	Dyanamic Update Partial Screen Graphic	"5"	654 Bytes / Display image

Table 5.6.2: Controller Memory Space Allocations for Different Types of Message Data

As indicated previously, the 320x80 controller module will load display message data into memory sequentially, into the next available memory location. Since the controller has a finite amount of memory storage capacity, the interface software ("320x80Soft", or equivalent) will need to either keep track of the amount of bytes used in the respective controller address prior to transmitting, or the controller will need to be "asked" the amount of available memory for an additional display message storage. For instance, the maximum amount of message data that can be loaded into a 320x80 controller can not exceed 43,817 bytes.

The following items pertain to maximizing the number of complete messages (or "pages") stored by a 320x80 controller module:

- To maximize the number complete messages to be loaded into a controller module, use text-based messages (full, partial, flashing or partial screen text dynamic update) whenever possible. For instance, a 320x80 controller can store only 9 complete full screen graphic messages {3214 [bytes/message] x 13 [messages] = 41,718 [bytes]}, whereas the same controller can store 79 complete full screen text based message {553 [bytes/message] x 79 [messages] = 43,687 [bytes]}, which is less than the 43,817 bytes capacity. Additional message storage capacity can be obtained if partial screen text images are used whenever possible.

5.6.2.1 Controlling and Determining the Message Count

The following application commands are available to the user to enable the interface to query (or "ask") the 320x80 controller the amount of available RAM memory left for additional message storage, and to enable the user to clear existing memory in the event the controller dynamic message storage algorithm becomes fragmented.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Designation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"M"	N/A	0	N/A	0	Output "M"essage storage availability from controller. Host will receive message f/controller after positive response is received.
2	">" (3e in Hex)	N/A	0	N/A	0	Clear All Previously Received Message Data (overides any previous Message trigger or Automatic cycle command received).

Table 5.6.3: Additional Controller Memory Allocation/Query Commands

As indicated from the table above, the interface will receive a message like the following example after the 320x80 controller receives a "M" command:

```
RAM Bytes Used = 0791
RAM Bytes Available = A398
RAM Fragmented = 0000
```

The quantities provided on each line after the introductory text is output in ASCCI hexadecimal format. As illustrated in the example above, the given 320x80 controller currently has 0791 (1937_{10}) bytes occupied with user message data, has A398 ($41{,}880_{10}$) bytes available for additional message storage, and has 0 bytes wasted due to fragmentation of the message data.

On current 320x80 controller designs, message data fragmentation will only occur if the user changes an existing message (for a given message and display number) from one message type, to another as indicated in table 5.6.2. If the amount of RAM space, which is fragmented, becomes excessive, the user (or the "interface") may wish to output a "Clear All Previously Received Message Data" command (">" command), followed by transmission of all the required message data using the commands indicated in table 5.6.1.

Another good application of the "Clear All Previously Received Message Data" command (">" command) is when the interface application starts a new transmission of data to an 320x80 controller, which may be in continuous automatic cycle mode. After the controller receives the ">" command, the controller will stop generation of any messages presently being driven onto the displays, then clear memory allocated for message storage. After transmission of the ">" command, the interface should load new message data to the respective controller module.

5.7 Large VGA (LVGA) Module Host to Display Application Level Field Descriptions

The following table outlines the available application level command sequences for the Kent Display LVGA (640x480) display modules. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. The application level command designations (field #4) are case sensitive.
- N/A designates that no bytes are designated or inserted for respective field.
- Typical Data elements are illustrated for field #6.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"0" (ASCCI Zero)	<Display #>, <Message. #>, <Sense>, <Dyn. Update Parameter 1>	4	DC2 DC4 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load Full Screen Text data into Memory (Does not display until valid "T" command received), for Small fonts (5x7 Characters, & FD-9 Fonts).
2	"1"	<Display #>, <Message. #>, <Sense>, <Dyn. Update Parameter 1>	4	DC2 DC4 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load Full Screen Text data into Memory (Does not display until valid "T" command received), for Large fonts (FD-11 & Letter Gothic 22).
3	"2"	<Display #>, <Message. #>, <Sense>, <Starting Img. Data Location>, <Number of Image Bytes within packet>	7	<Binary Pixel Data> (See Description below)	Variable (<5k Recom-mended of 38.4k total)	Load Full Screen, VGA Format Graphic data into RAM (Does not display until valid "T" command received).
4	"3"	<Display #>, <Message. #>, <Sense>, <Starting Img. Data Location>, <Number of Image Bytes within packet>	7	<Binary Pixel Data> (See Description below)	Variable (<5k Recom-mended of 9.6k total)	Load Full Screen, ¼ VGA Format Graphic data into (Does not display until valid "T" command received).
5	"4"	<Display #>, <Message #>, <First Row # to Alter MSB>, <First Row # to Alter LSB>, <Sense>, <Dyn. Update Parameter 1>	7	DC2 DC4 "*Change Disp.1 Msg X Line 3*" <ETX>	Variable	Load Partial Screen Text message in Memory to be updated in normal mode (Creates 1 line of Text on Display after corresponding "T" command is received).
6	"5"	<Display #>, <Message #>, <First Row # to Alter MSB>, <First Row # to Alter LSB>, Number of Rows Altered (120_{10} Max), <Sense>, <Dyn. Update Parameter 1>, <Starting Img. Data Location>, <Number of Image Bytes within packet>	11	<Binary Pixel Data> (See Description below)	Variable (<5k Recom-mended of possible 9.6k total)	Load Partial Screen graphic image in Memory to be output in Normal mode (Does not display until valid "T" command received).
7	"6"	<Display #>, <Message #>, <First Row # to Alter MSB>, <First Row # to Alter LSB>, <Sense>, ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	9	DC2 DC4 "*Change Disp.1 Msg X Line 3*" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after "T" command is received. Only Small Fonts allowed)
8	"7"	<Display #>, <Message #>, <First Row # to Alter MSB>, <First Row # to Alter LSB>, <Sense>, Number of Rows Altered (16_{10} Max), ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	10	<Binary Pixel Data> (See Description below)	Variable (1280 Max)	Load Partial Screen Graphic image in Memory to be displayed in Dynamic Update format (Does Not display on ChLCD until after a "T" command is received).
9	"8"	<Display #>, < Message #>, <First Row # to Alter MSB>, <First Row # to Alter LSB>, Non-Invert Duration (in 0.1 increments), Invert Duration, # of Flash Cycles, <Dyn. Update Parameter 1>	8	DC3 DC2 "Flash Disp.1 Msg X Row 4" <ETX>	Variable (256 Bytes Max.)	Load Partial Screen Text message (1 line of text) to be "Flashed" in the display as indicated in the corresponding parameters in field # 5 (Does Not flash on ChLCD display until after a "T" command is received).
10	"9"	<Display #>, <Message #>, <Sense>	3	N/A	0	Set/Clear Invert sense of Display image for respective Msg. #.

Table 5.7.1: LVGA Host to Display Application Level Commands

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
11	":" (3A Hex)	<Message #>, Duration (Hex)	2	N/A	0	Set Unique Pause Duration for respective message for Auto cycle operation (in 0.1 second increments).
12	";" (3B Hex)	<1st Message #>, <Last Message #>, <# of Message Cycles>, <Tog. Img. Sence Byte>	4	N/A	0	Start Automatic Image Cycling using the parameters specified.
13	"<" (3C Hex)	N/A	0	N/A	0	Ask controller for display count. Host will receive message f/controller after positive response is received.
14	"I"	N/A	0	N/A	0	Sign Firmware "I"D Request. Host will receive a sign message after positive response is received.
15	"A"	<On/Off>	1	N/A	0	Turn OFF/ON Sleep Timer (ON after reset or power-up)
16	"B"	<1st display #>, <Last display number>	2	N/A	0	"B"lank Display #'s indicated.
17	"C"	N/A	0	N/A	0	Output "C"hLCD Temperature from Sign. Host will receive a temperature text f/controller after positive response is received.
18	"O"	N/A	0	N/A	0	Turns "O"FF the Addressed Controller(s)
19	"P"	Duration (Hex)	1	N/A	0	Set Default "P"ause Duration between message writes, in 0.1 second increments (25.5 sec. Duration maximum).
20	"R"	N/A	0	N/A	0	"R"eset Controller (Clears any data previously entered into the sign RAM).
21	"S"	N/A	0	N/A	0	"S"top Automatic Message Cycling (Overrides any previous ":" command received)
22	"T"	<Msg. Number>	1	N/A	0	Generate, Write, or "T"rigger Message # X on Display(s).

Table 5.7.1: LVGA Host to Display Application Level Commands (Continued)

Where:

1. <Msg. #> = The message number, 1 thru the maximum # of messages for the displays (in hex) for the respective message packet. Refer to section 5.7.2 for determining the maximum number of the messages which can be stored within a respective LVGA controller module.
2. <Binary Pixel Data> = bit mapped data for the respective LVGA "landscape mode" image, starting from 1st row (top) left to 1st row (top) right (80 bytes/row typically), 2nd row left to 2nd row right, ...last row left to last row right totalling 38,400 bytes (80 bytes/row times 480 rows for "2" command above). If sending binary pixel data for command "3" above, only 40 bytes per row and half the number of rows (240) of data are expected (totaling 9600 bytes, or 40 bytes/row x 240 rows ((1/4 VGA))).
3. <Starting Img. Data Location>, <Number of Image Bytes within packet> = 2 byte quanites each (Most significant byte 1st, followed by least significant byte) used to "sectionalize" the message command into smaller packets to optimize communication. The "Starting Img. Data Location" represents the starting location or address index for the image data in field 6, starting with 0. The "Number of Image Bytes within packet" respresents the number of field 6 bytes sent with the corresponding packet.

4. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet (used for text-based messages).
5. <# of msg. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display module will assume the message cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).
6. <Sense> = <00> Off (non-inverted), and <01> On (inverted)
7. <On/Off> = <00> Off, <01> for On.
8. ON Update Method – Byte to indicate how the controller is to present the respective partial image or message. The following codes apply:
   - <01> = "*Wipe-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to "grow" to the right as the text message is presented. Unlike the "Scroll-ON" method, characters to the left already presented in a previous frame will be stationary while new characters within the image are presented to the right.
   - <02> = "*Scroll-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to move to the left as the text message is presented. Unlike the "Wipe" update method, all the characters within the image will move until the final frame of the image is presented.
   - <03> = "*Close-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" down from the top and "sliding up from the bottom, until the final frame provides the completed image joined together.
   - <04> = "*Open-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" up and down from the center, until the final frame provides the completed image which has been completely "opened" from the center.
   - <05> = "*Rotate-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is vertically rotating from the "Up-right" position, till it reaches an "Upside Down position", then the frame sequence is reversed till the final frame of the "Up-right" position is reached.
   - <06> = "Swell-ON' image presentation method. This presentation method is used as an alternate method of generating a static frame of data. Unlike the previous methods where multiple frames are presented to the user, this method will only generate a single frame representative of the text or graphic data provided at the interface. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text section. Unlike the "Fade-ON", this update method will erase the screen section prior (hence destroying the previous image) to generating the image in the cumulative drive method, hence creating an illusion that the new image will grow in brightness or "Swell" from a blank background during the update.
   - <07> = "Fade-ON" image presentation method. Like the "Swell-ON", this method is used as an alternate method of generating a static frame of data. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Swell-ON", this update method will NOT erase the screen section prior to generating the image in the cumulative drive method, hence creating an illustion that the new image section will start to appear as the previous image starts to disappear, hence the new image will appear to "Fade" over the previous.

9. OFF Update Method – Byte to indicate how the controller is to remove the respective partial image or message. The following codes apply:
   - <01> = "*Wipe-OFF*" image removal method (not valid for graphic messages). This removal method is the opposite of the "Wipe-ON" presentation method, with the characters being removed from right to left. The left-most character is the last character removed from the image area.
   - <02> = "*Scroll-OFF*" image removal method (not valid for graphic messages). This removal method is the same ast the "Scroll-ON" presentation method, with the characters being

removed from right to left. The right-most character is the last character removed from the image area.

<03> = "*Open-OFF*" image removal method. This removal method is the opposite of the "Close-ON" presentation method, with the characters being removed from center towards the top and bottom edges of the image area. The middle-most rows of the image are the last portions removed from the image area.

<04> = "*Close-OFF*" image removal method. This removal method is the opposite of the "Open-ON" presentation method, with the characters being removed from the top and bottom edges towards the center of the image area. The top and bottom rows of the image are the last portions removed from the image area.

<05> = "*No OFF*" image presentation method. This number will inform the respective controller to NOT remove the respective partial image after the respective pause duration has expired.

10. <Dyn. Update Parameter 1> - This single byte of data is bit-mapped to indicate the following features:

    - Bit 7 through 3 – Reserved for future use.
    - Bit 0 – Bit 2 - Implement a 1x to 8x Image height feature. For instance, a 10 row partial image would be output on 80 rows of the respective display if all 3 bits are set high. The same 10 row image can be output on 40 rows of the display if bits 0 and 1 are set high. Clearing all bits will output the image to the normal, 1x mode where the 10 row image will be output on 10 rows of the display. If more than a 1x is selected for the full screen text image ("0" or "1" command), the same number of characters/line indicated in section 5.7.1 is maintained, however the number of text lines / image is 1/x (for instance, if the 2x mode is selected, the number of text lines /image is ½ the 1x values specified below in the next section.

5.7.1 Special "0", "1", "4", "6" and "8" Command Text Control Characters

As indicated in the referenced command descriptions, the resident sign firmware has the capability to output text messages requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The sign modules also have the capability to output selected fonts and respond to font control characters. When the following special control characters (described in hexadecimal format within the "<>") are added within field 6 of the respective application layer messages, the following actions will occur (the following control characters must be the 1st data elements entered in field #6):

16x30 Pixel Characters: Each of the following described font controls will occupy a 16x30 (columns x rows) pixel space/character. Up to 40 characters/text line and up to 16 (if "1x", 8 if "2x" selected) text lines/display image can be output using these font controls.

a) <11> (DC1) = Turn ON Letter Gothic 22 Font Control character. All characters entered in field 6 after the reception of this control character will be output with the modified Letter Gothic 22 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used with a full-screen text command.

10x15 Pixel Characters: The font control below will occupy a 10x15 (columns x rows) pixel space/character. Up to 64 characters/text line and up to 32 (if "1x", 16 if "2x" selected) text lines/display image can be output using this font control.

b) <12> (DC2) = Turn ON Fixed Distance 11 point Font Control character. All characters entered in field 6 will be output in the Fixed distance 11 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used for the full screen text command.

6x8 (5x7) Pixel Characters: The font control below will occupy a 6x8 (columns x rows) pixel space/character. The actual text occupies a standard 5x7 pixel space. Up to 106 characters/text line and up to 60 (if "1x", 30 if "2x") text lines/display image can be output using this font control.

c) <05> (ENQ) = Turn ON 5x7 character Font Control characters. All characters entered in field 6 will be output in this format after reception. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used with a full screen text command.

Note: The absence of the above control characters will command the respective display controller to output the respective display in Fixed Distance 9 point; the default display font. With the default font, each character will occupy a 8x10 pixel space. Up to 80 characters/text line and up to 48 (if "1x", 24 if "2x") text lines/display can be output using this default font.

e) <13> (DC3) = Turn ON Underline control character. All characters entered in field 6 after the reception of this control character will be output in Underline format. The absence of this control character will command the respective sign to output the display with the default non-underlined mode.

5.7.2 Maximum Number of LVGA Messages

As indicated in the "<Message #>" references in table 5.7.1, the LVGA controller module can store several messages or pages of data for the respective number of display(s) assigned to the controller module. To maximize the amount of messages to be stored in the controller module, the controller uses a dynamic memory allocation method. Each message type (or "Page") is allocated a fixed memory space, based on the message type for the given display. The dynamic memory allocation firmware uses a "link list" to place each display image in consecutive sections of the controller RAM memory.

The following memory space allocations are assigned to the available message types:

#	Message Type	Field #4 Command Designation(s)	Fixed Memory Space Allocation
1.	Full Screen Small Font Text	"0"	6435 Bytes / Display image
2.	Full Screen Large Font Text	"1"	2095 Bytes / Display image
3.	Full Screen Graphic (VGA Format)	"2"	38,416 Bytes / Display image
4.	Full Screen Graphic (1/4 VGA Format)	"3"	9,616 Bytes / Display image
5.	Partial Screen Text, Dynamic Update Text, Flashing Text	"4", "6" and "8"	123 Bytes / Display image
6.	Partial Screen Graphic	"5"	9,616 Bytes / Display image
7	Dyanamic Update Partial Screen Graphic	"7"	1296 Bytes / Display image

Table 5.7.2: Controller Memory Space Allocations for Different Types of Message Data

As indicated previously, the LVGA controller module will load display message data into memory sequentially, into the next available memory location. Since the controller has a finite amount of memory storage capacity, the interface software ("VGASoft", or equivalent) will need to either keep track of the amount of bytes used in the respective controller address prior to transmitting, or the controller will need to be "asked" the amount of available memory for an additional display message storage. For instance, the maximum amount of message data that can be loaded into a LVGA controller can not exceed 58,400 bytes.

The following items pertain to maximizing the number of complete messages (or "pages") stored by a LVGA controller module:

- To maximize the number complete messages to be loaded into a controller module, use text-based messages (full, partial, flashing or partial screen text dynamic update) whenever possible. For instance, a LVGA controller can store only 1 complete full screen VGA format graphic message (with room to space) {38,416 [bytes/message] x 1 [messages] = 38,416 [bytes]}, or it can only store 6 complete ¼ VGA formatted messages {9,616 [bytes/message] x 6 [messages] = 57,696 [bytes]}, whereas the same controller can store 27 complete full screen, large font text based message {2095 [bytes/message] x 27 [messages] = 56,565 [bytes]}, which are all less than the 58,400 byte capacity. Additional message storage capacity can be obtained if partial screen text images are used whenever possible.

5.7.2.1 Controlling and Determining the Message Count

The following application commands are available to the user to enable the interface to query (or "ask") the LVGA controller the amount of available RAM memory left for additional message storage, and to enable the user to clear existing memory in the event the controller dynamic message storage algorithm becomes fragmented.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Designation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"M"	N/A	0	N/A	0	Output "M"essage storage availability from controller. Host will receive message f/controller after positive response is received.
2	">" (3e in Hex)	N/A	0	N/A	0	Clear All Previously Received Message Data (overides any previous Message trigger or Automatic cycle command received).

Table 5.7.3: Additional Controller Memory Allocation/Query Commands

As indicated from the table above, the interface will receive a message like the following example after the LVGA controller receives a "M" command:

```
RAM Bytes Used = 293B
RAM Bytes Available = BAE5
RAM Fragmented = 0000
```

The quantities provided on each line after the introductory text is output in ASCCI hexadecimal format. As illustrated in the example above, the given LVGA controller currently has 293B ($10{,}555_{10}$) bytes occupied with user message data, has BAE5 ($47{,}845_{10}$) bytes available for additional message storage, and has 0 bytes wasted due to fragmentation of the message data.

On current LVGA controller designs, message data fragmentation will only occur if the user changes an existing message (for a given message and display number) from one message type, to another as indicated in table 5.7.2. If the amount of RAM space, which is fragmented, becomes excessive, the user (or the "interface") may wish to output a "Clear All Previously Received Message Data" command (">" command), followed by transmission of all the required message data using the commands indicated in table 5.7.1.

Another good application of the "Clear All Previously Received Message Data" command (">" command) is when the interface application starts a new transmission of data to an LVGA controller, which may be in continuous automatic cycle mode. After the controller receives the ">" command, the controller will stop any generation of any messages presently being driven onto the displays, then clear memory allocated for message storage. After transmission of the ">" command, the interface should load new message data to the respective controller module.

5.8 New 1/4 VGA Display Module Host to Display Application Level Field Descriptions

The following table outlines the available application level command sequences for the Kent Display 1/4 VGA (320x240) display modules, Kent Displays P/N 15297. **This section does NOT apply to the older version 1/4 VGA Display, Kent Displays P/N 15126**. The following items pertain to the table:

- Text enclosed in "", indicate the exact bytes (typically in ASCII, unless noted otherwise) to be output for the particular field. The application level command designations (field #4) are case sensitive.
- N/A designates that no bytes are designated or inserted for respective field.
- Typical Data elements are illustrated for field #6.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"0" (ASCCI Zero)	<Display #>, <Message. #>, <Sense>, <Dyn. Update Parameter 1>	4	DC2 "**Disp 1 Msg 10 Row 1**" <CR> "**Disp 1 Msg 10 Row 2**" <ETX>	Variable	Load Full Screen Text data into Memory (Does not display until valid "T" command received), for Small fonts (5x7 Characters, & FD-9 Fonts).
2	"1"	<Display #>, <Message. #>, <Sense>, <Dyn. Update Parameter 1>	4	DC2 "***Disp 1 Msg 10 Row 1***" <CR> "***Disp 1 Msg 10 Row 2***" <ETX>	Variable	Load Full Screen Text data into Memory (Does not display until valid "T" command received), for Large fonts (FD-11 & Letter Gothic 22).
3	"2"	<Display #>, <Message. #>, <Starting Img. Data Location>, <Number of Image Bytes within packet>	6	<Binary Pixel Data> (See Description below)	Variable (<5k Recom-mended of 38.4k total)	Load Full Screen, 1/4 VGA, 16 Level grey Scaled Format, Graphic data into RAM (Does not display until valid "T" command received). *(Command not implemented on initial versions of Product!).*
4	"3"	<Display #>, <Message. #>, <Sense>, <Starting Img. Data Location>, <Number of Image Bytes within packet>	7	<Binary Pixel Data> (See Description below)	Variable (<5k Recom-mended of 9.6k total)	Load Full Screen, ¼ VGA Format binary (no grey scale) Graphic data into (Does not display until valid "T" command received).
5	"4"	<Display #>, <Message #>, <First Row # to Alter>, <Sense>, <Dyn. Update Parameter 1>	5	DC3 "Change Disp.1 Msg X Line 3" <ETX>	Variable	Load Partial Screen Text message in Memory to be updated in normal mode (Creates 1 line of Text on Display after corresponding "T" command is received).
6	"5"	<Display #>, <Message #>, <First Row # to Alter>, Number of Rows Altered (100_{10} Max), <Sense>, <Dyn. Update Parameter 1>	6	<Binary Pixel Data> (See Description below)	Variable (4000 Maximum)	Load Partial Screen binary graphic image (no grey scale) in Memory to be output in Normal mode (Does not display until valid "T" command received).
7	"6"	<Display #>, <Message #>, <First Row # to Alter>, <Sense>, ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	8	DC3 "Change Disp.1 Msg X Line 3" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after "T" command is received. Only small fonts allowed!)
8	"7"	<Display #>, <Message #>, <First Row # to Alter>, <Sense>, Number of Rows Altered (16_{10} Max), ON Update Method, OFF Update Method, <Pause Duration>, <Dyn. Update Parameter 1>	9	<Binary Pixel Data> (See Description below)	Variable (640 Max)	Load Partial Screen Graphic image in Memory to be displayed in Dynamic Update format (Does Not display on ChLCD until after a "T" command is received).
9	"8"	<Display #>, < Message #>, <First Row # to Alter LSB>, Non-Invert Duration (in 0.1 increments), Invert Duration, # of Flash Cycles, <Dyn. Update Parameter 1>	7	DC3 DC2 "Flash Disp.1 Msg X Row 4" <ETX>	Variable (256 Bytes Max.)	Load Partial Screen Text message (1 line of text) to be "Flashed" in the display as indicated in the corresponding parameters in field # 5 (Does Not flash on ChLCD display until after a "T" command is received).
10	"9"	<Display #>, <Message #>, <First Row # to Alter>, <Sense>, ON Update Method, <Dyn. Update Parameter 1>	6	DC2 "Change Disp.1 Msg X Line 3" <ETX>	Variable	Load Partial Screen Text message in Memory to be Displayed in indicated Dynamic Update format (Creates 1 line of Text on Display after reception of command. No "T" command is required. Only Small fonts allowed!)
11	";" (3B Hex)	<Display #>, <Message #>, <Sense>	3	N/A	0	Set/Clear Invert sense of Display image for respective Msg. #.

Table 5.8.1: New 1/4 VGA Host to Display Application Level Commands

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Desig-nation	Content	Byte Count	Typical Content	Byte Count	Message Description
12	"<" (3C Hex)	<Message #>, Duration (Hex)	2	N/A	0	Set Unique Pause Duration for respective message for Auto cycle operation (in 0.1 second increments).
13	"=" (3D Hex)	<1st Message #>, <Last Message #>, <# of Message Cycles>, <Tog. Img. Sence Byte>	4	N/A	0	Start Automatic Image Cycling using the parameters specified.
14	"I"	N/A	0	N/A	0	Sign Firmware "I"D Request. Host will receive a sign message after positive response is received.
15	"A"	<On/Off>	1	N/A	0	Turn OFF/ON Sleep Timer (ON after reset or power-up)
16	"B"	<1st display #>, <Last display number>	2	N/A	0	"B"lank Display #'s indicated.
17	"C"	N/A	0	N/A	0	Output "C"hLCD Temperature from Sign. Host will receive a temperature text f/controller after positive response is received.
18	"O"	N/A	0	N/A	0	Turns "O"FF the Addressed Controller(s)
19	"P"	Duration (Hex)	1	N/A	0	Set Default "P"ause Duration between message writes, in 0.1 second increments (25.5 sec. Duration maximum).
20	"R"	N/A	0	N/A	0	"R"eset Controller (Clears any data previously entered into the controller RAM).
21	"S"	N/A	0	N/A	0	"S"top Automatic Message Cycling (Overrides any previous ":" command received)
22	"T"	<Msg. Number>	1	N/A	0	Generate, Write, or "T"rigger Message # X on Display(s).

Table 5.8.1: New 1/4 VGA Host to Display Application Level Commands (Continued)

Where:

1. <Msg. #> = The message number, 1 thru the maximum # of messages for the displays (in hex) for the respective message packet. Refer to section 5.8.2 for determining the maximum number of the messages which can be stored within a respective 1/4 VGA controller module.
2. <Binary Pixel Data> = bit mapped data for the respective 1/4 VGA "landscape mode" image, starting from 1st row (top) left to 1st row (top) right (40 bytes/row), 2nd row left to 2nd row right, ...last row left to last row right totalling 96,000 (non-grey scaled images). If sending binary pixel data for command "3" above, the packing method is to be determined at later time.
3. <Starting Img. Data Location>, <Number of Image Bytes within packet> = 2 byte quanites each (Most significant byte 1st, followed by least significant byte) used to "sectionalize" the message command into smaller packets to optimize communication. The "Starting Img. Data Location" represents the starting location or address index for the image data in field 6, starting with 0. The "Number of Image Bytes within packet" respresents the number of field 6 bytes sent with the corresponding packet.
4. ETX = End of Transmission Command byte (03 Hex). Indicates end of application layer message packet (used for text-based messages).
5. <# of msg. Cycles> = Indicates the number of image writes the display module is to generate before stopping. If the user provides a <00> (Hex) into this respective location, the display

module will assume the message cycling is to resume indefinitely, overriding the "sleep-mode" feature of the display module (Not recommended for long term battery operation).

6. <Sense> = <00> Off (non-inverted), and <01> On (inverted)
7. <On/Off> = <00> Off, <01> for On.
8. ON Update Method – Byte to indicate how the controller is to present the respective partial image or message. The following codes apply:
   - <01> = "*Wipe-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to "grow" to the right as the text message is presented. Unlike the "Scroll-ON" method, characters to the left already presented in a previous frame will be stationary while new characters within the image are presented to the right.
   - <02> = "*Scroll-ON*" image presentation method (not valid for graphic messages). Characters within the image will appear to move to the left as the text message is presented. Unlike the "Wipe" update method, all the characters within the image will move until the final frame of the image is presented.
   - <03> = "*Close-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" down from the top and "sliding up from the bottom, until the final frame provides the completed image joined together.
   - <04> = "*Open-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is "sliding" up and down from the center, until the final frame provides the completed image which has been completely "opened" from the center.
   - <05> = "*Rotate-ON*" image presentation method. This presentation method provides the illusion that the partial screen image is vertically rotating from the "Up-right" position, till it reaches an "Upside Down position", then the frame sequence is reversed till the final frame of the "Up-right" position is reached.
   - <06> = "Swell-ON' image presentation method. This presentation method is used as an alternate method of generating a static frame of data. Unlike the previous methods where multiple frames are presented to the user, this method will only generate a single frame representative of the text or graphic data provided at the interface. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text section. Unlike the "Fade-ON", this update method will erase the screen section prior (hence destroying the previous image) to generating the image in the cumulative drive method, hence creating an illusion that the new image will grow in brightness or "Swell" from a blank background during the update.
   - <07> = "Fade-ON" image presentation method. Like the "Swell-ON", this method is used as an alternate method of generating a static frame of data. This method will automatically use the "*No OFF*" OFF update method, outlined in the next bullet text. Unlike the "Swell-ON", this update method will NOT erase the screen section prior to generating the image in the cumulative drive method, hence creating an illustion that the new image section will start to appear as the previous image starts to disappear, hence the new image will appear to "Fade" over the previous.

11. OFF Update Method – Byte to indicate how the controller is to remove the respective partial image or message. The following codes apply:
    - <01> = "*Wipe-OFF*" image removal method (not valid for graphic messages). This removal method is the opposite of the "Wipe-ON" presentation method, with the characters being removed from right to left. The left-most character is the last character removed from the image area.
    - <02> = "*Scroll-OFF*" image removal method (not valid for graphic messages). This removal method is the same ast the "Scroll-ON" presentation method, with the characters being removed from right to left. The right-most character is the last character removed from the image area.
    - <03> = "*Open-OFF*" image removal method. This removal method is the opposite of the "Close-ON" presentation method, with the characters being removed from center towards

the top and bottom edges of the image area. The middle-most rows of the image are the last portions removed from the image area.

<04> = "*Close-OFF*" image removal method. This removal method is the opposite of the "Open-ON" presentation method, with the characters being removed from the top and bottom edges towards the center of the image area. The top and bottom rows of the image are the last portions removed from the image area.

<05> = "*No OFF*" image presentation method. This number will inform the respective controller to NOT remove the respective partial image after the respective pause duration has expired.

12. <Dyn. Update Parameter 1> - This single byte of data is bit-mapped to indicate the following features:
    - Bit 7 – Implements a "2x" horizontal image "Strech" feature if activated (set to 1). **This feature is only valid** for static or single frame text based images ("0","1", "4", "6" (Fades & swells only), "8" & "9" commands). For instance, if implemented on a fixed distance 9 point line of text, the characters on the display will be twice as wide, and the maximum number of characters per text line (normally 40 for this font. Refer to next sub-section for explanation on font selection & control) will be half as much, or 20.
    - Bit 6 through 3 – Reserved for future use.
    - Bit 0 – Bit 2 - Implement a "1x" to "8x" Image height or vertical "Stretch" feature. For instance, a 10 row partial image would be output on all 80 rows of the respective display if all 3 bits are set high. The same 10 row image can be output on 40 rows of the display if bits 0 and 1 are set high. Clearing all bits will output the image to the normal, 1x mode where the 10 row image will be output on 10 rows of the display.

5.8.1 Special "0", "1", "4", "6", "8" and "9" Command Text Control Characters

As indicated in the referenced command descriptions, the resident sign firmware has the capability to output text messages requested by the host computer. The host does not need to inform the display module which pixel to turn on or off to create a text character image. The controller module also have the capability to output selected fonts and respond to font control characters. When the following special control characters (described in hexadecimal format within the "<>") are added to the beginning of field 6 of the respective application layer messages, the following actions will occur (the following control characters must be the 1st data elements entered in field #6):

16x30 Pixel Characters: Each of the following described font controls will occupy a 16x30 (columns x rows) pixel space/character. Up to 20 characters/text line and up to 8 (if "1x", 4 if "2x" selected) text lines/display image can be output using these font controls.

a) <11> (DC1) = Turn ON Letter Gothic 22 Font Control character. All characters entered in field 6 after the reception of this control character will be output with the modified Letter Gothic 22 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used with a full-screen text command. **This is a ISO/IEC 8859-1 Compliant Font.**

10x15 Pixel Characters: The font control below will occupy a 10x15 (columns x rows) pixel space/character. Up to 32 characters/text line and up to 16 (if "1x", 8 if "2x" selected) text lines/display image can be output using this font control.

b) <12> (DC2) = Turn ON Fixed Distance 11 point Font Control character. All characters entered in field 6 will be output in the Fixed distance 11 point font format. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive

lines of text if used for the full screen text command. **This is a ISO/IEC 8859-1 Compliant Font.**

8x10 Pixel Characters: The font control below will occupy a 8x10 (columns x rows) pixel space/character. Up to 40 characters/text line and up to 24 (if "1x", 12 if "2x") text lines/display image can be output using this font control.

c) <05> (ENQ) = Turn ON Fixed Distance 9 Point characters. All characters entered in field 6 will be output in this format after reception. A carriage return (0D in hexadecimal without a line feed) will need to be entered between the consecutive lines of text if used with a full screen text command. **This is a ISO/IEC 8859-1 Compliant Font.**

Note: The absence of the above control characters will command the respective display controller to output the respective display in 5x7 pixel characters, the default display font. With the default font, each character will occupy a 6x8 pixel space. Up to 53 characters/text line and up to 30 (if "1x", 15 if "2x") text lines/display can be output using this default font.

d) <13> (DC3) = Turn ON Underline control character. All characters entered in field 6 after the reception of this control character will be output in Underline format. The absence of this control character will command the respective sign to output the display with the default non-underlined mode.

5.8.2 Maximum Number of 1/4 VGA Messages

As indicated in the "<Message #>" references in table 5.8.1, the 1/4 VGA controller module can store several messages or pages of data for the respective number of display(s) assigned to the controller module. To maximize the amount of messages to be stored in the controller module, the controller uses a dynamic memory allocation method. Each message type (or "Page") is allocated a fixed memory space, based on the message type for the given display. The dynamic memory allocation firmware uses a "link list" to place each display image in consecutive sections of the controller RAM memory.

The following memory space allocations are assigned to the available message types:

#	Message Type	Field #4 Command Designation(s)	Fixed Memory Space Allocation
1.	Full Screen Small Font Text	"0"	1635 Bytes / Display image
2.	Full Screen Large Font Text	"1"	543 Bytes / Display image
3.	Full Screen Graphic (1/4 VGA, 16 Level Grey-Scale Format)	"2"	38,416 Bytes / Display image
4.	Full Screen Graphic (1/4 VGA, Binary Format)	"3"	9,616 Bytes / Display image
5.	Partial Screen Text, Dynamic Update Text, Flashing Text	"4", "6", "8" and "9"	70 Bytes / Display image
6.	Partial Screen Binary Graphic	"5"	4,016 Bytes / Display image
7	Dyanamic Update Partial Screen Graphic	"7"	656 Bytes / Display image

Table 5.8.2: Controller Memory Space Allocations for Different Types of Message Data

As indicated previously, the 1/4 VGA controller module will load display message data into memory sequentially, into the next available memory location. Since the controller has a finite amount of memory storage capacity, the interface software ("1/4VGASoft", or equivalent) will need to either keep track of the amount of bytes used in the respective controller address prior to transmitting, or the controller will need to be "asked" the amount of available memory for an additional display message storage. For instance, the maximum amount of message data that can be loaded into a 1/4 VGA controller can not exceed 60,000 bytes.

The following items pertain to maximizing the number of complete messages (or “pages”) stored by a 1/4 VGA controller module:

- To maximize the number complete messages to be loaded into a controller module, use text-based messages (full, partial, flashing or partial screen text dynamic update) whenever possible. For instance, a ¼ VGA controller can store only 1 complete full screen, Grey-scaled format graphic message (with room to space) {38,416 [bytes/message] x 1 [messages] = 38,416 [bytes]}, or it can only store 6 complete ¼ VGA binary formatted messages {9,616 [bytes/message] x 6 [messages] = 57,696 [bytes]}, whereas the same controller can store 36 complete full screen, small font text based message {1635 [bytes/message] x 36 [messages] = 58,860 [bytes]}, which are all less than the 60,000 byte capacity. Additional message storage capacity can be obtained if partial screen text images are used whenever possible.

5.8.2.1 Controlling and Determining the Message Count

The following application commands are available to the user to enable the interface to query (or “ask”) the 1/4 VGA controller the amount of available RAM memory left for additional message storage, and to enable the user to clear existing memory in the event the controller dynamic message storage algorithm becomes fragmented.

#	(Field #4)	(Field #5)		(Field #6)		
#	Cmd. Designation	Content	Byte Count	Typical Content	Byte Count	Message Description
1	"M"	N/A	0	N/A	0	Output "M"essage storage availability from controller. Host will receive message f/controller after positive response is received.
2	“>” (3e in Hex)	N/A	0	N/A	0	Clear All Previously Received Message Data (overides any previous Message trigger or Automatic cycle command received).

Table 5.8.3: Additional Controller Memory Allocation/Query Commands

As indicated from the table above, the interface will receive a message like the following example after the ¼ VGA controller receives a “M” command:

```
RAM Bytes Used = 293B
RAM Bytes Available = C125
RAM Fragmented = 0000
```

The quantities provided on each line after the introductory text is output in ASCCI hexadecimal format. As illustrated in the example above, the given 1/4 VGA controller currently has 293B ($10{,}555_{10}$) bytes occupied with user message data, has C125 ($49{,}445_{10}$) bytes available for additional message storage, and has 0 bytes wasted due to fragmentation of the message data.

On current ¼ VGA controller designs, message data fragmentation will only occur if the user changes an existing message (for a given message and display number) from one message type, to another as indicated in table 5.8.2. If the amount of RAM space, which is fragmented, becomes excessive, the user (or the “interface”) may wish to output a “Clear All Previously Received Message Data” command (“>” command), followed by transmission of all the required message data using the commands indicated in the table above.

Another good application of the “Clear All Previously Received Message Data” command (“>” command) is when the interface application starts a new transmission of data to an 1/4 VGA controller, which may be in continuous automatic cycle mode. After the controller receives the “>” command, the controller will stop any generation of any messages presently being driven onto the displays, then clear memory

allocated for message storage. After transmission of the ">" command, the interface should load new message data to the respective controller module.

6.0 Customer Contact

For comments regarding this manual, or the operation of a display unit, contact the Kent Displays customer service department at:

Kent Displays Incorporated Customer Service, Kent Displays, Inc.
343 Portage Blvd.
Kent, OH 44240, USA
(330) 673-8784
(330) 673-4408 (FAX)
E-mail: sales@kentdisplays.com