FASSTCOM Computers Limited

FASSTCOM COMPUTER TERMINAL SOFTWARE SPECIFICATION

For Fasstcom, Wang 2200, SuperDos, VMS and Unix computer systems

VERSION 1.0
OCTOBER 1994

FASSTCOM COMPUTERS LIMITED ,
Complexe la Roche, 1000 av. St-Charles, suite 800, Vaudreuil, Quebec, Canada, J7V 8P5

FT2201

FASSTCOM COMPUTER TERMINAL
SOFTWARE SPECIFICATIONS

Version 1.0

Introduction

This manual contains a portion only of the FASSTCOM IFT-2000 terminal. The IFT-2000 supports an enhanced Wang Terminal protocol as well as DEC's VT-220 protocol. The terminal also as a menu driven set-up utility accessible at all times using ALT-X (Pressing X while holding the ALTernate key).

This document describes the specifics concurning the Wang CS/2200 terminal emulation and the new associated features only. This manual excludes the description of the Menu driven Set-up utility and also the specifics of the VT-220 emulation. The final document will included all those information

This is a development working document only.

The terminal specifications described in this manual, are to be used for the programming of the FasstCom terminal and also for the FasstCom terminal emulation program for PCs.

These specifications will make the new terminal and the PC emulation fully compatible with all of the Wang 2x36 generations of terminals, primairly used with the Wang 2200 computers and the Wang's Basic-2 operating system.

These specifications include some enhancements compare to Wang's latest terminals, the 2536 and the 2636. Most of those enhancements are already supported by other terminal emulation programs for PCs, and are in use by other Basic-2 operating environments such as NPL and KCML under DOS, Windows, SuperDOS, VMS and Unix in one form or another.

These specifications are such that maximum compatibility is kept between all parties. When this is not possible, then a Wang style format or the actual Wang stardard is used.

Table of Contents

Table of Contents (Continued)

Compatibility

(Basic-II Only)

List of Compatible Environments

SYSTEMS

Wang 2200 Systems,
Kerrige KCML-DOS
Kerrige KCML-Windows
Kerrige KCML-Unix
Niakwa NPL-DOS
Niakwa NPL-Windows
Niakwa NPL-SuperDos
Niakwa NPL-Unix
Niakwa NPL-VMS
FasstCom FICE

TERMINALS / PC EMULATIONS

C.C. PC2200
Wang 2536 Terminal
Wang 2636 Terminal
Wang PC/Emulation
Wang 2236/2336/2436
(DW & DE Terminals)

MAC-2200
Kerrige DW-DOS
Kerrige DW-Win.
Falcon Terminal
Orchard T2200DW
Macsoft WT2

Other Terminals Under Basic-II Control

DEC VT-220
DEC VT-10/52/100
Wise 60/85

Hardware Operation

The IFT-2000 terminal has the folowing hardware interfaces :

1 Main DB25 Serial Port for Host Connection,
1 Standard DB25 Parallel Printer Port,
1 Standard DB9 Serial Port for any alternate serial device (printer, mouse, Bar Code & Mag Stripes, etc.).
1 Standard DB9 Device port (Same as serial port with 12V and 5V pins). Can also be use as ordinairy serial port by not connectting the voltage pins.
2 Standard AT Keyboard connectors, that will accept any industry standard keyboards, including specialty keyboards such as PINs ans P.O.S. keypads.

Many devices may be attached to the terminal. Control of those devices maybe from control code or from terminal configuration menu only.

PRINTER PORT

The Printer port is accessed normally, as per Wang protocol. Default printer port is the Parrallel port. Main printer port can be set to any of the ports through set-up menu. However, if any other port than the parrallel port is set as printer port, access to the parrallel port is lost. A secound printer maybe attached to the serial port or device port, as a standard RS-232 output device.

SECOND SERIAL PORT and DEVICE PORT

The Second serial port and the DEVICE ports are accessible for Input/Output operation using new I/O adresses. These ports are configurable through control sequences send by the application program. Flow control in Wang mode between the terminal and the Host, are the same for these 2 ports as for the main printer transmissions.

MOUSE (For FUTUR Implementation)
A later implementation should include mouse support using the second serial port. When this port is configure as a mouse port, access to this port by the host is no longuer available.

KEYBOARD CONTROL

A control sequence is provided for selecting from which keyboard to accept data. Programs can switch from which keyboard to accept data from during normal operation.

Simultanous usage of both keyboards is not allowed. So if keys are hit on an inactive keyboard, they will be ignored, unless the keyboard has its own buffer, in which case the behavior depends on the keyboard.

Only PC Extended AT (101 Keys) Keyboards are supported at this time.

The 12 standard function keys (F1 to F12) and also most of the other special function keys of the keyboard, Shifted and Unshifted ONLY for both cases maybe customize from the terminal set-up utility.

All other keys are mapped strickly from the table selection in the terminal set-up menu. This mapping also includes default values for the customisable keys.

MONITORS

A monitor must be added to the terminal base unit. The terminal uses VGA and SVGA interface and support any monitor with VGA or SVGA interface.

Monochrome and Color monitors are supported. Monitors maybe of any size and resolution.

If Monochrome monitor is used witt full or palette color modes, all colors are represented by various shades of grey as per the standard PC Gray scale.

Global Control

RESET ON/OFF **(TO BE CONFIRMED)**

The terminal configuration allows Activating or De-Activating the RESET Key (Ctrl R). The default setting is RESET Activated.

PRINT SCREEN **(TO BE CONFIRMED)**

Print Screen (Knowed as Local Screen Dump) is available on the Terminal by Pressing ..... . Only the TEXT of the screen is send to the printer. Underline characters are converted to no underline. This function is similar to the PRINT SCREEN on PC's and the "Hold Edit for 1 clic" on the Wang Terminal.

Flow Control :

The 2636 support 2 types of flow control. The type must be selected when configuring the terminal. It cannot be change during normal operation.

Wang Flow Control

The Wang handshaking uses two set of flow control:

Hex (FA)	=	Stop Transmitting to the screen.
Hex (FB)	=	Stop Transmitting to local printer or to the current output device different from the screen.
Hex (F8)	=	Continue Transmitting to the screen.
Hex (F9)	=	Continue Transmitting to local printer or to the current output device different from the screen.

NOTE : For the Wang flow control only, in addition to normal operation, the terminal must always transmit an Hex(F8F9) every 3 to 5 seconds whenever it is ready to receive new data during idle time. This is call the constant poll.

Standard XON/XOFF Flow control

The second type of flow control allowed is the industry standard "XON/XOFF". It is used to stop and restart transmissions, regardless of the output device in use.

When the terminal is set to this type of handshaking, HEX(F8F9) and HEX(FAFB) are no longuer used for flow control. Also, constant polling (every 3 to 5 secounds) is not required since its not industry standard protocol. The XON/XOFF is implemented 100% as of industry standard.

CONTROL SEQUENCES

Control sequences are composed of combinations of at least 2 characters. Certain rules apply such as it is possible to distinguish when those characters are to be interpreted as just characters or as control sequence characters. In this document, we include as normal characters all printable characters and all control characters such as Bell Sound or Cursor Up. Any control sequence is composed of at least 2 characters and normally will have more.

There are two main groups of control sequences, those that starts with Hex(FB) knowed as the escape sequence (no relation to the escape character on PCs) and those that starts with Hex(02).

So any character received different from Hex(FB) or Hex(02) may immediately be processed as regular characters (including control characters). This makes trapping of control sequences fairly easy.

GRAPHICS & COLOR & BLINKING

Screen management is base on 640 X 480 resolution for 80 columns operation, with screen frequency variation to allow 40 and 132 columns operation. Resolution will vary depending on the screen mode used (40, 80 or 132 columns mode). All graphic functions are done accordingly. Up to 16 colors maybe used, 8 low intensity and 8 high intensity (or blinking).

For Blinking, 2 types of operations are allowed. The terminal setting defines usage of blinking. The default setting is TRUE BLINKING. In this setting, only 8 colors total are allowed, the low intensity ones. Otherwise, when the terminal is set for NO Blinking, all 16 colors are allowed, but a blinking color must be define.

Terminal set-up allows selection between 3 different color operation; 2 color mode, palette operation and color operation.

In 2 color mode, the terminal operates like a normal black and white terminal, with all the normal attributes accessible, except the user may decide to use any foreground or background color such as white on blue instead of white on black.

Palette operation alows users to define the look of the standard caracter attribute, in case current software do not make use of the extended caracter attributes. Blinking color (in case of 16 color mode) and box line color are also control from set-up menu.

Color operation is used in conjunction with the extended set of caracter attributes which allows for extensive color manipulations.

The current active terminal color setting may also be altered by program control when in full color mode.

Cursor Control

The cursor has 2 possible attributes. Steady or Blinking. It can also be turned ON or OFF. When truned OFF it does not display. When ON, it displays according to the current attribute setting (Steady or Blinking).

HEX (05) turn the cursor ON and HEX(06) Turns it OFF. HEX(FB F4) or HEX(FB FC) or HEX(02 05 0F) turns sets the attribute to BLINKING. HEX(FB F8) sets the cursor attribute to STEADY. The default setting is STEADY.

Character Base Screen Usage

Characters should be represented such that the upper line is always empty. The top line is reserved for BOX LINES which cannot be overwritten by text.

This means that if a box line is to be displayed at the bottom of the screen, it will cause the screen to scroll, since horizontal lines always belong the the bottom caracter line which must also be on the screen, even if all blanks.

Various screen modes are available and character cell size must always be at least 1 line longuer than the maximum character size so that the upper cell line is reserved for BOX Lines.

When possible, room should be made on the lower section to fit underline without overlaping the character. Also when possible, the right column should be free such to space characters, and on larger fonts, the left column can be also free.

The following is the list of all possible screen modes. Only one screen mode at a time maybe use, and the appropriate character size is applicable on the entire screen. One line at the bottom of the screen is always reserved for special display. This is knowed as the 25th line operation on Wang terminals. This line never scrools with the rest of the screen.

Current Screen modes :

25 (24 + 1) Lines by 80 Columns (Default setting)
25 (24 + 1) Lines by 40 Columns (Graphics are enlarged)
25 (24 + 1) Lines by 132 Columns (Graphics are shrunk)

A latter implementation may support a special character attribute for over sized characters to be displayed. (Character Attribute or Line Attribute)

Special Framing Characters to the Host

Transparent Character HEX(FC)

This character preceeds characters that would otherwise be interpreted as function keys or otherwise. HEX(FC) is mainly used in conjunction with the DEAD Key, for language or underline character entry from the keyboard. Any character, no matter what the value, preceded by HEX(FC) is processed as a normal character from the character set by the Host computer.

The main usage of this character is described in the MULTI-KEYSTROKE characters segment.

EndI/Atom HEX(FD)

This character preceeds any character that is to be interpreted by the HOST as a special function character. When any function key is pressed, its value must be send preceded by HEX(FD). See Special Function Key in the table section of this document for details.

DEAD Key HEX(FE)

This caracter preceeds any character that is sent to the host, but is the first part of a 2 or more key combination character such as an underline character or language characters. Specific usage is specified in the following paragraphs "MULTI-KEYSTROKE Characters Protocol".

MULTI-KEYSTROKE Characters Protocol

Any letter or character that represents the combination of 2 bytes entry and/or multi-keystrokes, such as an Underline A or a language character using accent and is not generated from single keystroke, must use the following transmission protocol to the host.

For 2 bytes transmission (i.e. Chinese letter), The first byte is transmitted preceeded by HEX(FE) and the secound byte is transmitted preceeded by HEX(FC). Example: HEX(FE 90 FC 34). The 90 and 34 are the letter representation.

For multi-keystroke entry, the transmission uses the same format, except HEX(FC) is used to preceed only the last byte and HEX(FE) all the others. Also, the actual plain caracter (The A portion of Underline A must be keyed last. The terminal must reconize the legality of the combination of keystrokes. In case of an illegal combination, only the plain character ASCII value is transmitted. Otherwise, the actual ASCII value of the complete character must be transmitted.

Example: A is transmitted HEX(FE A0 FC C1). HEX(A0) is the underline character and HEX(C1) is Underline A.

Hex(FB) Group of control sequences

Hex(FB) is defined as the ESCAPE character.

The following set of control sequence always starts with Hex(FB). Since Hex(FB) can also be a legitimate printable character, the following rule dictates how to distinguish between to two possibilities.

When the terminal receives Hex(FB) the next character must be checked. Its value determines the use of Hex(FB).

If the following character is equal to Hex(D0) then the FB must be treated just like a regular character according to the character table and the Hex(D0) is then dropped.

Otherwise, the Hex(FB) signifies the 1st byte of a control sequence and the second character is part of that sequence.

The Hex(FB) control sequence must always be used as such (and never as a series of normal characters), regardless of the current output device selected, and even if control characters (Hex(00) to Hex(0F)) are involved.

IMPORTANT, HEX(FB) must be identified as the character (FB) or as the control sequence by checking if the following byte is HEX(D0) or not at the front end of the terminal processing, because some control sequence will contain values equal to HEX(FB). These sequences will be send out with the HEX(D0) byte which must be removed from the sequence so not to change the meaning of the control sequence send by the host.

The HEX(FB) sequence can never be transmitted as such directly from program control. The operating system generates the HEX(FB) sequence based on operations at hand, and based on Partition Selection such as SELECT PRINT 204.

To allow program control access to all HEX(FB) sequences, and also allow access to new HEX(FB) sequences that may not be directly supported by other operating systems than FICE, the control sequence HEX(02 1B ......) maybe used as the equivalent to HEX(FB......).

For example, PRINT HEX(02 1B F1); will have the same effect as SELECT PRINT 204.

The programer must be carefull using this method, since the partition device table will not be updated by the HEX(02 1B F1). So although printing will be re-routed to local printer, the device table will still show PRINTING SELECT TO 005 or whatever the provious address was.

If PRINT HEX(FB.....) is attempted within a program or even a $GIO /005(A000)HEX(FB.....), the Host will transmit HEX(FB D0) which identifies the display character HEX(FB) instead of the control sequence HEX(FB). This is valid for all versions of Basic-2, including FICE, Wang, KCML and NPL.

Case 1 : Device Output Selection HEX(FBFx)

When the terminal is powered on, by default all data received for output go to the screen, unless a different output device is selected by receiving a control code.

One output device at a time may be selected. Devices include the screen, the printer port, any additional serial ports, etc. In Wang mode, terminal flow control should be use for terminal display only, the printer flow control should be use for any other device, including of course the printer port.

The following codes send by the host computer (usually as part of a data string) switch the output device selection as follow:

Hex (FBF0)	=	Send all futur incoming data to Screen
Hex (FBF1)	=	Send all futur incoming data to terminal printer

Also, a second set of control sequence supports access to the auxiliary ports. Once routing to the current active auxiliary port is terminated, the terminal must resume original routing according to the last selection, terminal screen or terminal printer.

HEX(FB F7 35 31 68) directs all futur incoming data to the auxiliary port currently enabled.

HEX(FB F7 35 31 6C) terminates routing of data to the auxiliary port enabled.

Case 2 : Access to other auxiliary ports for input or output is done with the following protocol.

HEX(FB F3 aa bb cc dd xx) Configure Auxiliary Ports

xx = HEX(59) for auxiliary Device port
HEX(58) for auxiliary Serial port

aa = Data Speed Rate (Baud Rate)

00 = 9600	06 = 300
01 = 4800	07 = 150
02 = 2400	08 = 75
03 = 1200	09 = n/a
04 = n/a	0A = 19200
05 = 600	0B = 38400

bb = Parirty Selection
00 = No parity
01 = Odd parity
02 = Even parity

cc = Data Bites Selection
00 = 8 Data Bits
01 = 7 Data Bits

dd = Stop Bits Selection

00 = 1 Stop Bit
01 = 2 Stop Bits

HEX(FB F5 4x) Enable/Disable Auxiliary Ports

The terminal screen and main terminal printer are always active. However, only one of the Auxiliary ports may be active at a time.

In the case of the Auxiliary keyboard is enabled, the main keyboard is disabled. As soon as another auxiliary port than the keyboard is enabled or auxiliary port usage is simply disabled, the main keyboard becomes automatically enable again.

If any other auxliliary port is enable but the auxiliary keyboard, the main keyboard remains enable and active.

The folowing sequence enables or disables auxiliary ports :

HEX(FB F5 4x), where x = enable/disable function

1 = Disable Ports
2 = Enable Auxiliary Serial Port
3 = Enable Auxiliary Device Port
B = Enable Auxiliary Keyboard

The last sequence receive is the one that applies until a new one is received.

Example : HEX(FB F5 42) enable the auxiliary serial port
HEX(FB F5 43) disables the auxiliary serial port (cancels the las command) and enables the auxiliary device port.
HEX(FB F5 41) Disables the auxiliary ports.

Only zero or one auxiliairy port remains active at a time

HEX(FB F7 2F 35 3x 6y) Data Format Control To Host

Data can be received from any auxiliary port that is enable. This command sequence allows programmers to specify to the terminal how the data should be formated before it is transmitted to the host be the terminal.

When a port is enable, the data received from the port are sent to the host according to the following format :

Hex(FB F7 2F 35 33 68) - Data received from the port are directly sent to the host.

Hex(FB F7 2F 35 33 6C) - Data is embedded with a leading byte Hex(80) and is trailed by Hex(81). This allows the host to distinguish the data from terminal keyboard data.

When a port is enabled, the data received from the port are sent either as single character as it is received or in block mode with the following format :

Hex(FB F7 2F 35 34 6C) - Data is sent one character at a time as it is received.

Hex(FB F7 2F 35 34 68) - Data received are buffered and are sent as a block to the host after receiving an HEX(0D) from the port. When an HEX(2C) character is received from the port, the buffer is cleared.

HEX(FB F7 35 31 6y) Send Data to Auxiliary Ports

When this sequence is received, all futur incoming data must be send to the appropriate auxiliary port, untill a new data direction command is received. Refer to DEVICE OUTPUT SELECTION in page 15.

HEX(FB F7 35 31 68) directs all futur incoming data to the auxiliary port currently enabled.
HEX(FB F7 35 31 6C) terminates routing of data to the auxiliary port enabled.

NOTE : The setting of the auxiliary ports, including data formats, is NOT impared by the enable/disable command. These settings need be done only once on start-up.

NOTE : The auxiliary keyboard maybe enabled manually by the user by pressing Alt-K. Once RETURN (Hex(0D)) is pressed on the auxiliary keyboard, it will disable and the main keyboard will become active again.

Case 3: Data compression HEX(FBxxhh) Secound Byte <= HEX(5F)

When the same character is transmitted from the host computer, more than 3 times in a row (i.e. PRINT "AAAA"), the host computer uses a simple compression sequence so that it only needs to transmit 3 characters to do the same job.

The control sequence is Hex(FB xx hh) were hh is the hex value of the character to be printed or displayed, and xx is the number of times it must be printed or displayed in binary.

For example, if computer is programmed to do PRINT "AAAAA", the host computer will in fact transmit to the terminal HEX(FB 05 41) instead of HEX(4141414141). xx will only have a value between and including Hex(03) to Hex(5F), this means that a single character can only be repeated up to 207 times.

NOTE that repeated characters can include control characters such as the ones for cursor movement or the Bell Sound. For example, to move the cursor right 10 positions, the terminal will receive HEX(0B 0A 09). This compression applies also in Box Mode, discussed further in this manual.

VERY IMPORTANT, compression is used on all transmissions, including on HEX(02...) control sequences. Therefore, data compression support must be implemented at the front end of the terminal processing, as a filter between the host and actual terminal processing of incoming data.

Case 4 : Input Screen (RSD) HEX(FB D3 01 sr sc er ec)

When this sequence is received, the Host computer expect to receive a character base download of the screen. This is knowed as the Remote Screen Dump (RSD).

The first 2 Bytes of this sequence HEX(FB D3) tells the terminal to Start Remote Screen Dump.

The third byte has to be HEX(01). This byte confirms that the host as entered RSD mode. If this byte is any different than Hex(01) than RSD in canceled, the command ignored, the HEX(FB D3) dropped and normal operation resume.

The rest of the original sequence is to define the range of the screen dump. "sr" is the start row, "sc" is the start column, "er" is the end row and "ec" is the end column. All value are in binairy and the lowest value starts at 0.

The standard remote screen dump is of 4080 characters in size for 25 line x 80 column screen format. This size of course varies if the delimitting parameters are different than 25 lines by 80 columns. The format of the data transmitted is as follow :

Bytes 1 to 78	Contains the self identification message.
Bytes 79 - 80	Indentify the cursor position (row,colomn). Rows are numbered 0 to 24 and columns 0 to 79, in Binairy.

Then, the TEXT data is transmitted (2000 bytes in case of standard 25 lines 80 columns dump). Transmission is done ROW by ROW, from left to right.

Then the ATTRIBUTE Bytes are transmitted (2000 bytes in case of standard 25 lines 80 column dump). Transmission is done ROW by ROW, from left to right. There is 1 attribute byte for each corresponding character. The value of each attribute byte is as follow :

Bit 80 = 1 if character graphic (meaning second alt. set)
Bit 40 = 1 if reverse video
Bit 20 = 1 if blinking
Bit 10 = 1 if high intensity
Bit 08 = 1 in underline
Bit 04 = 1 if left horizontal box graphic segment
Bit 02 = 1 if vertical box graphic segment
Bit 01 = 1 if right horizontal box graphic segment

Any other attributes are not reported

RSD COMMUNICATIONS PROTOCOL

Untill the remote screen dump is completed, the following communications protocol is in effect. Once the remote screen dump completed or interrupted, normal operation resumes.

RESET : Reset (Ctrl R) is the ONLY key that if pressed will do something during RSD transmission. When pressed during the remote screen dump, HEX(12) must be transmitted to the Host.

HEX(F1): Hex(F1) is used as the escape character during RSD transmission.

HEX(12): If Hex(12) is to be transmitted as one of the valid characters in the screen dump, HEX(F1 12) must then be transmitted.

HEX(F1): If Hex(F1) is to be transmitted as one of the valid characters in the screen dump, HEX(F1 F1) must then be transmitted.

HEX(EC): If Hex(EC) is to be transmitted as one of the valid characters in the screen dump, HEX(EC EC) must then be transmitted.

Data compression can optionally be used by the terminal in the transmission, but is not madantory. If compression is used, the following format must be used :

HEX(EC) followed by the compression count (# bytes being compressed) in binairy and finally the compressed byte. Total is 3 bytes or 4 bytes for special bytes, therefore, it is only usefull to compress if the same byte is to be transmitted more than 3 times. The bytes compressed maybe any byte of the 4080 bytes transmitted, including attribute bytes.

Examples, Hex(EC 20 41) means 32 times letter 'A'
Hex(EC EC xx) is NOT LEGAL
HEX(EC 20 EC EC) means 32 times Hex(EC)

To avoid compatibility problems, compression should use a value less or equal to HEX(50) (maximum 80 characters at one time). HEX(EC) is an illegal compression count.

FLOW CONTROL FOR RSD

During transmission, the host will send a STOP or a GO signal to the terminal, whenever its receive buffer gets full. The STOP (suspend transmission) signal to the Terminal is HEX(FB D8). The GO (Resume transmission) signal to the Terminal is HEX(FB DA).

RSD SEQUENCE OF EVENTS

First, only the HOST sends to the Terminal

HOST SENDS	**TERMINAL DOES**
Hex(FB D3)	Accepts as Request for RSD
Hex(01)	RSD Confirmed
Upper Left Row	Stores 1 Byte Binairy Value
Upper Left Column	Stores 1 Byte Binairy Value
Lower Right Row	Stores 1 Byte Binairy Value
Lower Right Column	Stores 1 Byte Binairy Value
HEX(FB D8)	Start Sending Data

Then, Except for flow control (FB D8 and FB DA) of course, also except for RSD Abord, only the Terminal transmit to the Host. Cursor position starts with value 0 (HEX(00)).

TERMINAL SENDS	**HOST DOES**
HEX(F1 F2)	Accepts
78 Bytes Self ID message	Accepts
1 Byte Cursor Row Position (Binairy)	Accepts
1 Byte Cursor Col. Position (Binairy)	Accepts
DD Bytes of Screen Data	Accepts
AA Bytes of Attributes	Accepts
1 Byte LRC	Accepts
HEX(F1 F4) End of RSD	Accepts

LRC Byte is the XOR of all bytes of transmitted data except the HEX(F1) escape codes. Example, HEX(F1 12), both bytes are not included in the XOR function.

If the Host computer detects a RESET to interupt the RSD OR once the Host assumes it has received all of what it was expecting and the RSD is completed, it will transmit HEX(FB D4) to the terminal to inform the terminal that RSD as been terminated and that the terminal should resume normal operation.

HEX(FB D8), the GO flow control from the Host maybe send one last time to the terminal, after the terminate RSD HEX(FB D4) was send out. The terminal need only consider it as confirmation the the Host is ready to receive normal operation data again.

Code Functions Summary For RSD

Hex(12)	RESET (Ctrl R) (To Host)
Hex(F1 F2)	Start of RSD data bytes (To Host)
Hex(F1 12)	Data Byte of Hex(12) (To Host)
Hex(F1 F1)	Data Byte of Hex(F1) (To Host)
Hex(F1 F4)	End of RSD data bytes (To Host)
Hex(FT8)	Local Control RSD GO (**NOT USED IGNORE**)
Hex(FTA)	Local Control RSD STOP (**NOT USED IGNORE**)
Hex(FB D8)	GO RSD Transmission Flow Control From Host)
Hex(FB DA)	STOP RSD Transmission (Flow Ctrl From Host)
Hex(FB D4)	ABORD RSD Transmission (From Host)
Hex(EC EC)	Data Byte of Hex(EC) (To Host)
Hex(EC .. ..)	Data Compression (To Host)

SUMMARY OF ALL THE HEX(FB) CONTROL SEQUENCES

When receiving an Hex(FB) character, the following character must be checked. If equal to Hex(D0), then the second character is dropped and the hex(FB) is processed as a regular character.

The following is the summary list of the FB escape sequence :

CODE	DESCRIPTION
FBD0	Data byte Hex(FB)
FBxxhh	When xx is less than Hex(60), Data Compression
FBD3	Start of RSD control sequence from Host
FBD4	Abord OR Terminate RSD from Host
FBD8	RSD GO (RSD Flow Control from Host)
FBDA	RSD Stop (RSD Flow Control from Host)
FBF0	Select CRT for Output
FBF1	Select Local Printer for Output
FBF2	Reset Terminal (same as (02 0D 0C 03 0F))
FBF3	Set Auxiliary Port Parameters
FBF4	Cursor Blink (same as (02 05 0F) and (FB FC))
FBF5	Enable/Disable Auxiliary Ports
FBF6	Reset CRT ONLY, Clear terminal data ONLY from receive buffer, reset screen to default start-up stage.
FBF7	Set Data TRX Format for Auxiliary Ports
FBF8	Cursor Steady
FBFC	Cursor Blink (same as (02 05 0F) and (FB FC))
FBCx	Delay for x/6 of a second, where x = 1 to 9

The FB Cx command is a one time immediate delay command. When the terminal receives this command, the terminal must HALT (Delay) according to the delay value from 1/6 to 9/6 of a second. This delay is only valid for screen output. Screen flow control is critical in this situation to ensure data integrity.

The Hex(02) Group of Control Sequence

Contraire to the Hex(FB) control sequence, the Hex(02) sequences are never valid if the current output device is not the screen. So if Hex(02....) is received and the current output device selected is NOT the screen, then all characters must be treated as regular characters and not as part of a control sequence.

The function for each HEX(02) control sequence is determined by the character that follows the HEX(02), as per this table :

Function Code	Complete Sequence	Description
HEX (00)	HEX(02 **00** xx yy 0z) & HEX(02 **00 0B** ....)	Save & Restore Screen and also print on Line 25.
HEX (01)	HEX(02 **01** aa bb cc dd 0F)	Open Window.
HEX (02)	HEX(02 **02** xx 0F)	Select Normal or Alternate Character Sets.
HEX (03)	HEX(02 **03** xx yy 0F)	Select Active Colors.
HEX (04)	HEX(02 **04** xx yy zz)	Set Current Character Attributes.
HEX (05)	HEX(02 **05** 0F)	Cursor Blinking.
HEX (06)	HEX(02 **06** aa bb xx yy 0F)	Set Screen Mode
HEX (08)	HEX(02 **08** 09 0F)	Request Self-Identification Message.
HEX (09)	HEX(02 **09** 0x aa bb cc dd 0F)	Screen Read/Write
HEX (0B)	HEX(02 **0B** 0x)	Start Drawing Box Mode.
HEX (0C)	HEX(02 **0C** aa bbbb cccc dddd eeee ff gg hh 0F)	Graphic Function..
HEX (0D)	HEX(02 **0D** 0C 03 0F)	Initialise Terminal.
HEX (0E)	HEX(02 **0E** 0x 0F)	Select Keyboard
HEX (1B)	HEX(02 **1B** .....)	Same as HEX(FB ....)

Case 1: Save & Restore Screen HEX(02 00 06 xx 0h)
(xx MUST be Different than HEX(0B))

Hex(02 00 06 xx 0x) is used to tell the terminal to save or restore pages. This sequence is only valid when the current output device is the screen. Otherwise, these characters are treated as regular characters.

Usage of this sequence is as follow :

Hex(02 00 06 00 0E) - Save Current Screen as Page 0
Hex(02 00 06 00 0F) - Restore Page 0 to Current Screen

Hex(02 00 06 02 0E) - Save Current Screen as Page 1
Hex(02 00 06 02 0F) - Restore Page 1 to Current Screen

The last Byte is Hex(0E) to save the current screen and Hex(0F) is to restore it. The 4th byte identifies the page #. Hex(00) for Page 0, Hex(02) for Page 1, Hex(04) for Page 2, hex(06) for Page 3, Hex(08) for Page 4.

The purpose of storing the screen is so it can be restored to its current status after alterations from further operations, including Windowing. Therefore, the internal format in which the screen is stored is only important to the memory restrictions of the terminal and how fast the screen may be restored (reproduced from memory).

The PC emulators are using Bit Map image format. The screen is simply stored as bit map images, and restored the same way. This procedure is fast and all the graphics including the Box Lines are preserved.

This is not be reasonable for terminals because of memory restrictions. Instead, the screens is stored in character mode with all of the current attributes, including the Box Line Attribute. For graphics, the terminal always keep in memory the last 128 sequences that was sent to the terminal on the current screen. This information can be saved along with the screen. Upon restore, the graphics sequences would only need to be re-executed. All graphics command never exceed 16 bytes each.

Memory for Graphics commands must be cleared for the current screen when certain control sequences or control characters are received, if their purpose is to clear the screen.

For each screen save, the screen contents must be saved with all attributes, at least the last 128 graphics commands (wich are maximum 16 bytes each), the current character attribute in effect with the attribute switch status the cursor location and status and the overall screen color settings.

The total number of pages memory is 4, excuding the active screen (not considered as a page). For 132 columns operation, the number of pages is 2, excluding the current active screen.

Case 2 : Print on 25th line HEX(02 00 06 0B 0E xx)

Hex(02 00 06 0B 0E [xx]) Sets display mode to the last screen line only. This sequence is only valid when the output device is the screen, otherwise it is treated as regular characters.

This sequence allows to display text on the last screen line, the only line not scroling with the rest of the screen. This line is also out of bound for standard cursor location. Cursor Location is NOT affected by this command.

This command affects the last line of the current screen display, so the actual line # depends on the current screen mode. Current screen mode support is 24 lines only. Therefore, the special display line is line 25.

The value of xx is an OPTIONAL attribute and is used as follow:

Hex(00) - Normal display
Hex(02) - Highlight
Hex(04) - Blinking
Hex(06) - Reverse Video
Hex(08) - Reverse Highlight
Hex(0A) - Reverse Blinking
Hex(0C) - Underline

The 25th line is displayed from left to right and will scroll horizontally. Anything received from the host for the screen is displayed on the 25th line until an Hex(FF) is received. Other transmissions for other output device are allowed and do not affect the screen status. After receiving an HEX (FF) for the screen, normal screen operation resumes but what ever is on the 25th line stays. If another such sequence appears, the 25th line is overwritten by it starting from the left.

When HEX(02 00 06 0B 0F) is received, the 25th line is cleared and the screen is restored to normal state.

Case 3: Open Window HEX(02 01 aa bb cc dd 0F)

This sequence clears the area on the screen (including Box Lines and Graphics) from Line # aa to Line # cc and from column # bb to column # dd. This area of the screen becomes the only area accessible to the cursor, and position (0,0) starts at the upper left corner defined by Line # aa and Column # bb and ends at the lower right corner defined by Line # cc and Column # dd.

Inside the window, the last background and foreground color attribute used before the window command, becomes the normal background and foreground colors. All attributes are still useable inside a window.

The window starts like a new screen, all blank with cursor location at (0,0) and switch to on. Default attribute status applies, so the window starts with attribute 'Highlight' and switched OFF.

During operation, the window never exceeds its horizontal limits and scrolls inside, within the allowed number of lines. Essentially, the window operates like a normal screen, only smaller. Minimum size for a window is 2 X 2.

It is possible for the terminal to receive another Open Window Control Sequence. In this case, the same rule applies. A new window starts, erasing whatever is in the new area, and the older window is simply forgotten. This will create the effect of overlapping windows. Its up to programmers to save the screen if they which to return to the previous window.

By default, the normal screen is considered a window of 80 columns by 24 lines, starting at physical screen location of (0,0).

If the Screen Mode is set to any different size, the window maybe created anywhere within the new screen area. However, while inside as window, if a control sequence is received to change the Screen Mode to any different size than the current one, it must be ignored.

The position and size of the window must be within the current limitations of the screen. The bottom special display line remains intact from any window command and is still accessible using the PRINT TO 25TH LINE COMMAND only.

Case 4: Select Character Set HEX(02 02 xx 0F)

Hex(02 02 xx 0F) is a control sequence that selects the character table to be used for characters between Hex(80) and Hex(FF). This sequence is always of this format, and xx can only have a value of Hex(00), Hex(02) or Hex(04). This sequence is only valid for the screen. For any other output device, just output as regular characters.

Three different character sets are available, the Wang Normal Character set, the Wang Alternate Character set and the IBM standard Character set. Refer to the character set tables at the end of this manual.

The main difference between all character setsare is in the range from Hex(90) to Hex(FF) of the ASCII table. On the Wang alternate character set, characters between Hex(9C) and Hex(BF) are undefined (blank). So characters between Hex(A9) and Hex(BF) have been made to match IBM's character set exactly for those characters on the alternate set. The third character set (second alternate) is the actual IBM set.

NOTE The language sets NEVER effects the alternate tables. A single set of alternate tables exist for all languages.

Also, since under the normal character set, all characters from Hex(90) to Hex(FF) are the underlined version of characters from Hex(10) to Hex(7F), instead of olding such a table for each language, simply add an underline attribute when a character greater or equal than Hex(90) is received.

You will also note that characters with accens are between Hex(10) and Hex(1F) and those characters are never considered control characters. Control characters are limited in the range of Hex(00) to Hex(0F).

Case 5: Colour Control HEX(02 03 xx yy 0F)

This control sequence sets the new colour control parameters to the terminal. This allows programmers to alter the set up of the terminal (normally done through the terminal's own set-up menu). These colour controls must also be accessible manually through the terminal's own configuration screen.

The item for which the colour code is to change is set by xx and the yy parameter is the specific colour code ranging from 0 to 15. The codes are sent to the terminal in binary, ranging from Hex (00) to Hex(0E). The sequence is always 6 bytes and always end with HEX (0F). The possible values for xx are as follow :

HEX (00)	=	Screen Colour (Normal Background)
HEX (01)	=	Reverse Video Color (Palette)
HEX (02)	=	Normal Foreground
HEX (03)	=	Underline Color (Palette)
HEX (04)	=	Alternate Foreground 1
HEX (05)	=	Reverse & Underline (Palette)
HEX (06)	=	Alternate Foreground 2
HEX (07)	=	Highlight Color (Palette)
HEX (08)	=	Alternate Foreground 3
HEX (09)	=	Highlight & Blink (Palette)
HEX (0A)	=	Alternate Background 1
HEX (0C)	=	Alternate Background 2
HEX (0D)	=	ALternate Background 3
HEX (0E)	=	Box Line Colour
HEX (0F)	=	Blinking Colour in Case real blinking is not available.

Only Screen Colour HEX (00) takes effect immediately, and the normal screen background colour is then changed accordingly. The other colors are applicable on further display only, but do not affect whatever is already on the screen. So screen refresh is not required.

When Hex(02 03 00 xx 0F) is executed, the entire screen is cleared with the new background color. Graphics and Box line control sequences in memory must then be cleared as well.

Case 6: Display Attributes HEX(02 04 xx xx 0E)

Hex(02 04 xx yy 0E) is a control sequence for display attribute only. If such a sequence is received for any other output device than the screen, it must simply be treated as regular characters and be transmitted to the output device. This applies to all of the HEX(02) sequences. The sequence always starts with Hex(0204), it is always 5 bytes long and the last byte is always Hex (0E) or Hex (0F).

The yy parameter is the character environment attribute (mainly the background) and is always accompany the by xx parameter. The yy parameter is used as follow :

HEX (00) = Normal Display
HEX (02) = Reverse Video (Background colour is equal to the foreground determined by the xx attribute
HEX (04) = Underline Text
HEX (06) = Use Special background color 1 with xx attribute
HEX (08) = Use Special background color 2 with xx attribute
HEX (0A) = Use Special background color 3 with xx attribute
HEX (0B) = Combines HEX(02) and HEX(04) together

Note, if the text printed is already an underline character from the character set, and HEX (04) is used, the underlines simply overlaps.

The xx parameter is the actual character attribute. it is used in conjunction with the yy parameter at all times.
The Possible values for xx are as follow :

HEX (00) = Normal attribute (characters are printed normal (current foreground colour).
HEX (02) = Characters are Highlighted (or different matching colour)
HEX (04) = Blinking (Going from Foreground colour to Background and so forth OR Going from Normal Foreground to Highlight and so forth. In case blinking is not possible, use special colour code assignment
HEX (06) = Alternate Foreground Color 1.
HEX (08) = Alternate Foreground Color 2.
HEX (0A) = Alternate Foreground Color 3.
HEX (0B) = Combines HEX(02) and HEX(04) together

The attribute setting is always permanent. This means that once an HEX (02 04 02 00 0E) as been received, what ever characters are received after, they are always treated with the attribute in effect, in this case, highlight.

The HEX(0E) at the end of the sequence is used as a switch. When Hex(0E) is received, as part of the sequence or alone (as a control character), the current attribute is turned ON. When an HEX (0F) is received in the same ways, the current attribute is turned OFF.

IMPORTANT, Whenever Cariage Return "CR" is received from the Host HEX(0D), in addition to processing the "CR", the attribute switch is set to OFF, as if the terminal had received HEX(0F) after the "CR".

For example, the sequence HEX(02 04 00 02 0F) sets the current attribute to be reverse video and switched OFF. So all coming characters will be displayed normal. The current attribute is only used if an HEX(0E) turns it on. The same rule applies to the opposite. If an HEX (02 04 00 02 0E) is received, all following characters will be displayed in reverse video. Characters will be displayed normal only if an HEX(0F) or "CR" is received turning OFF the attribute. The default start-up attribute is set to Highlight and the Switch to OFF.

Case 7: Cursor Blinking HEX(02 05 0F)

Hex(02050F) is used to set the cursor in blinking mode. If the cursor is off, this command must override it, and put the cursor back on. This command is only valid for the screen display. We do not really care that the cursor blanks or not. The important thing is that for the display, this sequence is recognize as a control sequence, and that the cursor is at least turned on by it.

Case 8: Set Screen Mode HEX(02 06 aa bb 0F)

This command sequence sets the screen mode. When initialised or on power-up, the default settings for the screen mode is 25 lines by 80 columns and operates within 24 lines (Line #25 is treated in a special way).

This command allows to change this the following way: aa is the number of lines. At the momentit can only be set at 25. bb is the number of columns and can only be 40, 80 or 132. character size must change accordingly.

When this command is received, the entire screen is reseted and cleared, just as if it was being power-up, except with the new screen size parameters.

IMPORTANT, screen size parameters are given in BINAIRY. So 25 is actually Hex(19), 40 is Hex(28), 80 is Hex(50) and 132 is Hex(84). For example, the sequence to set the screen to 132 columns and 25 Lines would be HEX(02 06 19 84 0F).

Case 9: Request Self-identification Message HEX(02 08 09 0F)

Hex(02 08 09 0F), when this sequence is received for the display screen only, the terminal must transmit the following message to the host : "*FT01DW Rxxxx SSSSSB 8+O (xxx....)" and Hex(0D). If any other device than the screen is selected, than this sequence must be process as regular output characters.

Rxxxx is the firmware revision number and must have 4 digits, right justify with trailing zeros (i.e. 0001).

SSSSSB is the current speed set-up of the terminal and must have 5 digits right justified with trailing blanks (i.e. 19200 or _9600). The letter B is mandatory.

8+O is the current bit and parity setting of the terminal, (i.e. 8 Bits plus Odd). Other examples would be 7+N, 7+E, 8+E, 8+N.

(xxx...) is the current country setting of the terminal and must be exactly the names showed in the following list of countries supported, in column "Country Name". Those names may have up to 16 characters excluding parantisis. The parentisis around the name is mandatory.

The positions and spaces in this message is crucial. However, the size of the message may vary depending on the country name text only. The transmission must end with an Hex(0D).

LIST OF COUNTRY NAMES SUPPORTED
(Note that "*" means this country will have a dedicated version of the terminal)

	Language Set	**Country Name**
	French/Flemish	(AZERTY FRENCH)
	Canadian	(CANADIAN)
	Cyrillic/Latin	(CYRILLIC)
	Danish/Norwegian	(DANISH/NORWEGIAN)
	Finnish/Swedish	(FINNISH/SWEDISH)
	German	(GERMAN)
	Icelandic	(ICELANDIC)
*	Katakana	(KATAKANA)
*	Conventional Chinese	(CHINESE)
	Netherlands	(NETHERLAND)
	Spanish/Spanish Latin	(SPANISH)
	Swiss French/German	(SWISS)
	United Kingdom	(UNITED KINGDOM)
	United States	(USA)

Case 10: Screen Read/Write HEX(02 09 0x aa bb cc dd 0F)

NOT FOR IMMEDIATE IMPLEMENTATION

This command sends or receives to/from the Host computer, the Bit Map representation of the screen from position Line # aa and Column # bb and of the size of cc Lines and dd Column. Note that cc and dd are the actual size of the box to be capture in lines and columns. Internally those must be translated into graphic x,y coordinates.

Comminucations protocol will be establish at a futur date. Same protocal as RSD may be used, benifiting the established structured and applicability.

Case 11: Box Line Mode HEX(02 0B 0x ..... 0F)

This control sequence signals the start of Box Line Mode. The Wang and all other vendors in this platform, can print Box Lines using the PRINT BOX(l,c) statement, were l and c are the dimensions of the box in lines and columns.

When HEX(02 0B 02..... 0F) is received, the Box Line Mode is to draw a Box. When HEX(02 0B 0B.... 0F) is received, than the sequence is to erase a box, or a portion of a box.

Instead of simply sending a BOX like control sequence and let the terminal or PCs handle the actual drawing of the Box, its the host that actually draws the box by sending a series of cursor movement commands to the terminal or PC. The drawing of the Box always starts from the current cursor location. The box is drawn clockwise, and when finished, the cursor naturally comes back to its original location.

The HEX(020802) is the start of the sequence, and the sequence will end with HEX(0F). The length of the sequence and the possible characters received vary depending on the Box size to print. However, the smallest possible sequence is 6 Bytes long HEX(02 08 02 0B 0B 0F) and the maximum size is 16.

The is a maximum size because even if the box is large and drawn by cursor movement, the Host will use the data compression control sequence HEX(0B xx yy) within this sequence to send repetitive cursor movement characters such as cursor down Hex(0A). So there is a limit on how big this sequence can get which is 16 bytes.

The following is a list of the possible control characters that maybe received within the sequence to Draw a Box:

HEX (02 0B 02..... 0F) Draw Box Sequence

Hex(0B) This signals a change in directions, which means we are about to make a 90 degree turn. Also, when this is received, a vertical line must be drawn from top to bottom, at the center of the current character location.

Hex(0A) Move Cursor Down by one line, and print a vertical line, top to bottom, at the center of the new cursor location.

Hex(0C) Move Cursor Up by one line and print a vertical line, top to bottom, at the center of the new cursor location.

Hex(08) Move Cursor Left by one position. If at least one Hex(0B) as been received, then print a line at the top of the character box of the line below the current line, from the center of the previous cursor position to the center of the current cursor position. **This must cause scrolling if the current line is the 24th line.**

Hex(09) Move Cursor Right by one position and print a line at the top of the character box of the current line, from the center of the previous cursor position to the center of the current cursor position.

Hex(FB) Treat as compressed Data, See Compressed Data Explaination. If you receive this while in Box Line Mode, it will be accompanied by the number of repetetion and then by HEX(08), HEX(09), HEX(0A) or HEX(0C). Simply just execute the command as described under Line Box Mode. Example, if you receive Hex(FB060A) it means to move the cursor down 6 times and print a vertical line in the center of each character position as you go down since we are in Box Line Mode.

Hex(0F) This Ends Box Line Mode. Hex(0F) is the end of any Box command sequence.

HEX(02 0B 0B..... 0F) Erase Box Sequence

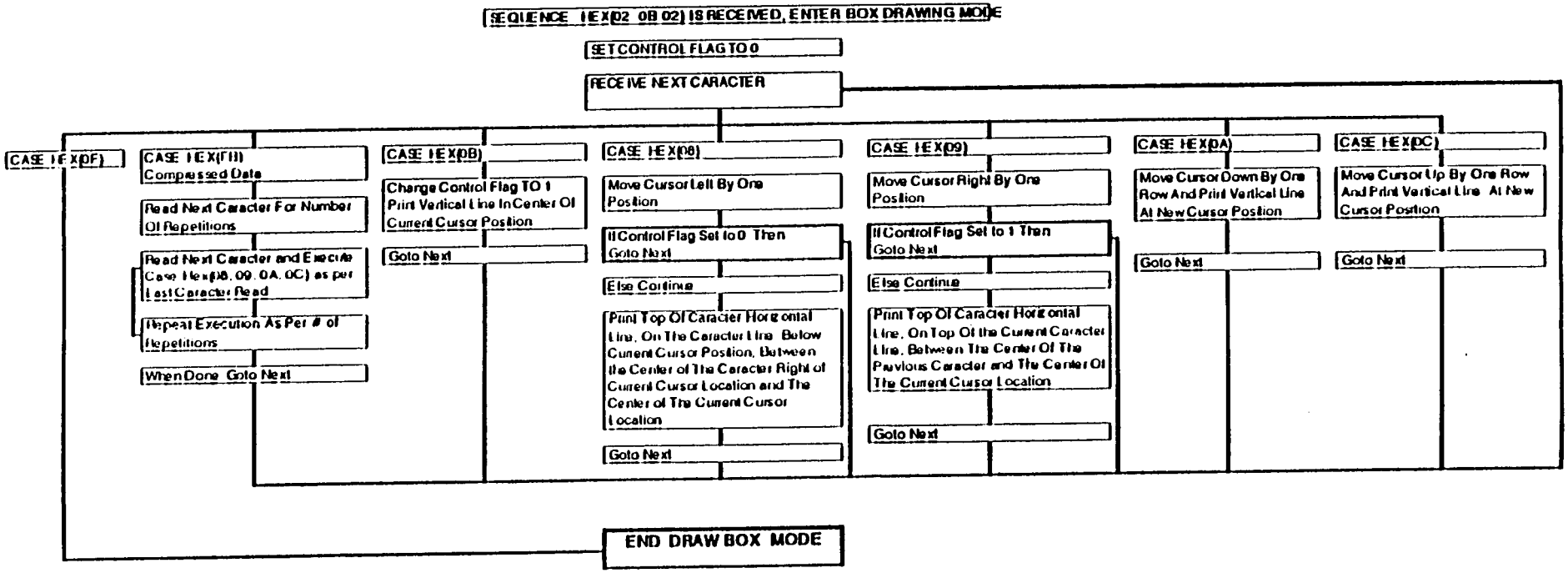
For Erasing part or an entire box, the control sequence is the same. All the control characters must behave the same way as for drwing a box, except, make sure to draw lines with the same color as the bacground color, in effect, erasing any line that maybe at the corresponding location.

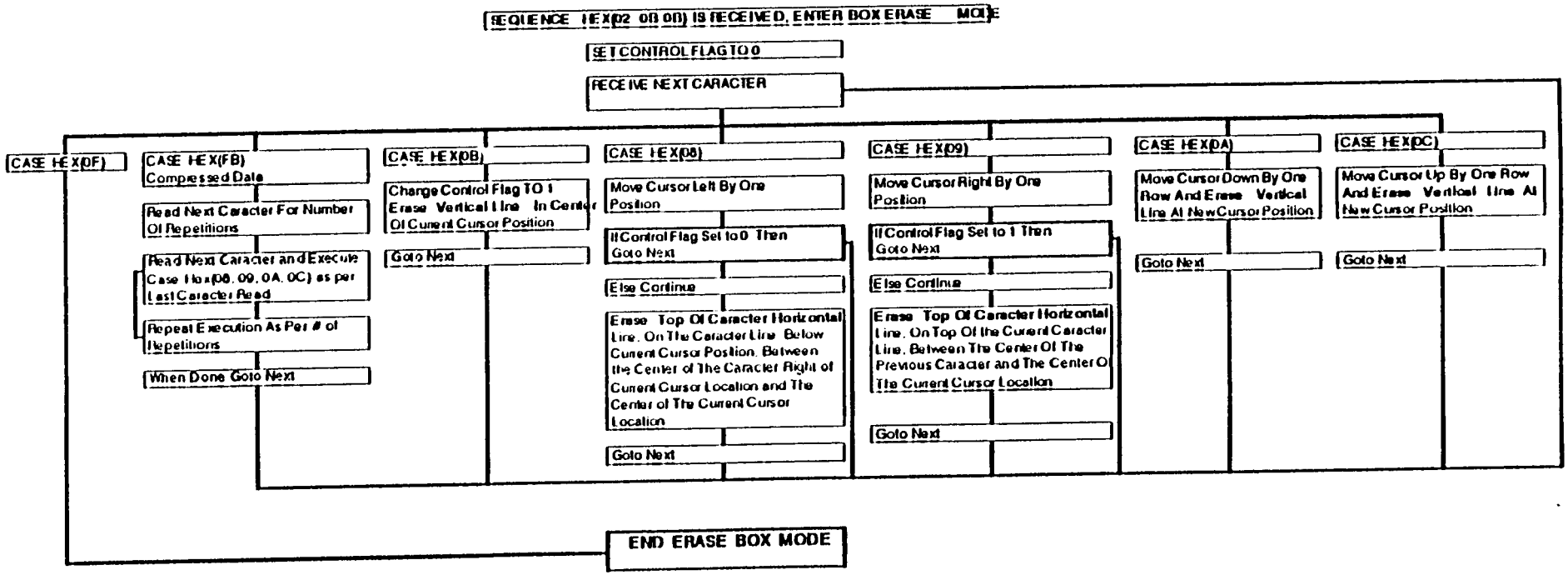
Erasing (or drawing with background color), may also cause scroling if the operation is initiated from the 24th line, just as drawing a line.

Erasing a line is legal even if there was no lines there to begin with. This means, that there is no need for keeping track of line attributes.

The following chart describes the procedure for both Line or Box drawing and for Line or Box erasing.

WANG TERMINAL BOX DRAWING ALGORITHM





Case 12: Graphics HEX(02 0C aa bbbb cccc dddd eeee ff gg hh 0F)

Some limited graphics are permitted are available through the use of this control sequence. The sequence is always 15 bytes long and is used as follow :

Param.	**USAGE**
aa	Type of Graphic in Binary Hex(00) = Pixel Display Hex(01) = Draw Line Hex(02) = Draw Box (Graphic Box) Hex(03) = Draw Circle or Arc. An Arc is drawn and becomes a circle when the start angle reaches the end angle. Hex(04) = Draw Pie Shape
bbbb	Two (2) Bytes Binary Value 'Y' coordinate. For Types 01 and 02, its the 'Y' coordinate of the first point. For Types 03 and 04 its 'Y' coordinate for the center of.
cccc	Two (2) Bytes Binary Value 'X' coordinate. For Types 01 and 02, its the 'X' coordinate of the first point. For Types 03 and 04 its 'X' coordinate for the center of.
dddd	Two (2) Bytes Binary Value, second 'Y' coordinate or Start Angle. For Type 00, this value is equal to Hex(0000) and ignored. For Types 01 and 02, its the 'Y' coordinate of the last point. For Types 03 and 04, its the start angle value.
eeee	Two (2) Bytes Binary Value, second 'X' coordinate or Start Angle. For Type 00, this value is equal to Hex(0000) and ignored. For Types 01 and 02, its the 'X' coordinate of the last point. For Types 03 and 04, its the end angle value.
ff	One (1) Byte Binary value, the value of the radius for types 03 and 04. Value is equal to Hex(00) and ignored for all other types.
gg	One (1) Byte Binary value, for line colour. Used by all types, as the line colour only and not as a fill colour. Value may range between hex(00) and Hex(0F).
hh	One (1) Byte Binary value, for filling this form. This do not apply to Types 00 and 01. For those types, the value is always Hex(FF). For Types 02,03 and 04, its defines what if anything is put inside the form. The values are Hex(00) to Hex(0F) for one of 16 colour codes (8 low intensity and 8 high intensity). Hex(FF) means no fill.

Case 13: Terminal Initialization HEX(02 0D 0C 03 0F)

Hex(02 0D 0C 03 0F), when received, instruct the terminal to re-initialize to power-on state. This sequence, like the others, is only valid when the current device is the screen. Otherwise, treat as regular characters.

FUNCTION KEYS
Standard PC Keyboard Set-Up KEYBOARD OPTION 1

Name	Output Value	Standard PC Keyboard	Input Value PC Keyboard
SF' 0	Hex(FD)	Alt + F10	Hex(F1)
SF' 1	Hex(FD01)	F1	Hex(BB)
SF' 2	Hex(FD02)	F2	Hex(BC)
SF' 3	Hex(FD03)	F3	Hex(BD)
SF' 4	Hex(FD04)	F4	Hex(BE)
SF' 5	Hex(FD05)	F5	Hex(BF)
SF' 6	Hex(FD06)	F6	Hex(C0)
SF' 7	Hex(FD07)	F7	Hex(C1)
SF' 8	Hex(FD08)	F8	Hex(C2)
SF' 9	Hex(FD09)	F9	Hex(C3)
SF'10	Hex(FD0A)	F10	Hex(C4)
SF'11	Hex(FD0B)	Shift + F1	Hex(D4)
		or F11	Hex( )
SF'12	Hex(FD0C)	Shift + F2	Hex(D5)
		or F12	Hex( )
SF'13	Hex(FD0D)	Shift + F3	Hex(D6)
SF'14	Hex(FD0E)	Shift + F4	Hex(D7)
SF'15	Hex(FD0F)	Shift + F5	Hex(D8)
SF'16	Hex(FD10)	Shift + F6	Hex(D9)
SF'17	Hex(FD11)	Shift + F7	Hex(DA)
SF'18	Hex(FD12)	Shift + F8	Hex(DB)
SF'19	Hex(FD13)	Shift + F9	Hex(DC)
SF'20	Hex(FD14)	Shift + F10	Hex(DD)
SF'21	Hex(FD15)	CTRL + F1	Hex(DE)
		Shift + F11	Hex( )
SF'22	Hex(FD16)	CTRL + F2	Hex(DF)
		Shift + F12	Hex( )
SF'23	Hex(FD17)	CTRL + F3	Hex(E0)
SF'24	Hex(FD18)	CTRL + F4	Hex(E1)
SF'25	Hex(FD19)	CTRL + F5	Hex(E2)
SF'26	Hex(FD1A)	CTRL + F6	Hex(E3)
SF'27	Hex(FD1B)	CTRL + F7	Hex(E4)
SF'28	Hex(FD1C)	CTRL + F8	Hex(E5)
SF'29	Hex(FD1D)	CTRL + F9	Hex(E6)
SF'30	Hex(FD1E)	CTRL + F10	Hex(E7)
SF'31	Hex(FD1F)	ALT + F1	Hex(E8)
		CTRL + F11	Hex( )

FUNCTION KEYS

Standard PC Keyboard, Niakwa Layout Set-Up OPTION 2

Name	Output Value	Standard PC Keyboard	Input Value PC Keyboard
SF' 0	Hex(FD01)	F1	Hex(BB)
SF' 1	Hex(FD02)	F2	Hex(BC)
SF' 2	Hex(FD03)	F3	Hex(BD)
SF' 3	Hex(FD04)	F4	Hex(BE)
SF' 4	Hex(FD05)	F5	Hex(BF)
SF' 5	Hex(FD06)	F6	Hex(C0)
SF' 6	Hex(FD07)	F7	Hex(C1)
SF' 7	Hex(FD08)	F8	Hex(C2)
SF' 8	Hex(FD09)	F9	Hex(C3)
SF' 9	Hex(FD0A)	F10	Hex(C4)
SF'10	Hex(FD0B)	Shift + F1	Hex(D4)
SF'11	Hex(FD0C)	Shift + F2	Hex(D5)
SF'12	Hex(FD0D)	Shift + F3	Hex(D6)
SF'13	Hex(FD0E)	Shift + F4	Hex(D7)
SF'14	Hex(FD0F)	Shift + F5	Hex(D8)
SF'15	Hex(FD10)	Shift + F6	Hex(D9)
SF'16	Hex(FD11)	Alt + F1	Hex(E8)
SF'17	Hex(FD12)	Alt + F2	Hex(E9)
SF'18	Hex(FD13)	Alt + F3	Hex(EA)
SF'19	Hex(FD13)	Alt + F4	Hex(EB)
SF'20	Hex(FD14)	Alt + F5	Hex(EC)
SF'21	Hex(FD15)	Alt + F6	Hex(ED)
SF'22	Hex(FD16)	Alt + F7	Hex(EE)
SF'23	Hex(FD17)	Alt + F8	Hex(EF)
SF'24	Hex(FD18)	Alt + F9	Hex(F0)
SF'25	Hex(FD19)	Alt + F10	Hex(F1)
SF'26	Hex(FD1A)	Ctrl + F1	Hex(DE)
SF'27	Hex(FD1B)	Ctrl + F2	Hex(DF)
SF'28	Hex(FD1C)	Ctrl + F3	Hex(E0)
SF'29	Hex(FD1D)	Ctrl + F4	Hex(E1)
SF'30	Hex(FD1E)	Ctrl + F5	Hex(E2)
SF'31	Hex(FD1F)	Ctrl + F6	Hex(E3)

FUNCTION KEYS

Standard PC Keyboard, Matching Banks Layout 1, Set-Up OPTION 3

Name	Output Value	Standard PC Keyboard	Input Value PC Keyboard
SF' 0	Hex(FD01)	F1	Hex(BB)
SF' 1	Hex(FD02)	F2	Hex(BC)
SF' 2	Hex(FD03)	F3	Hex(BD)
SF' 3	Hex(FD04)	F4	Hex(BE)
SF' 4	Hex(FD05)	Ctrl + F1	Hex(DE)
SF' 5	Hex(FD06)	Ctrl + F2	Hex(DF)
SF' 6	Hex(FD07)	Ctrl + F3	Hex(E0)
SF' 7	Hex(FD08)	Ctrl + F4	Hex(E1)
SF' 8	Hex(FD09)	Ctrl + F5	Hex(E2)
SF' 9	Hex(FD0A)	Ctrl + F6	Hex(E3)
SF'10	Hex(FD0B)	Ctrl + F7	Hex(E4)
SF'11	Hex(FD0C)	Ctrl + F8	Hex(E5)
SF'12	Hex(FD0D)	F5	Hex(BF)
SF'13	Hex(FD0E)	F6	Hex(C0)
SF'14	Hex(FD0F)	F7	Hex(C1)
SF'15	Hex(FD10)	F8	Hex(C2)
SF'16	Hex(FD11)	Shift + F1	Hex(D4)
SF'17	Hex(FD12)	Shift + F2	Hex(D5)
SF'18	Hex(FD13)	Shift + F3	Hex(D6)
SF'19	Hex(FD13)	Shift + F4	Hex(D7)
SF'20	Hex(FD14)	Alt + F1	Hex(E8)
SF'21	Hex(FD15)	Alt + F2	Hex(E9)
SF'22	Hex(FD16)	Alt + F3	Hex(EA)
SF'23	Hex(FD17)	Alt + F4	Hex(EB)
SF'24	Hex(FD18)	Alt + F5	Hex(EC)
SF'25	Hex(FD19)	Alt + F6	Hex(ED)
SF'26	Hex(FD1A)	Alt + F7	Hex(EE)
SF'27	Hex(FD1B)	Alt + F8	Hex(EF)
SF'28	Hex(FD1C)	Shift + F5	Hex(D8)
SF'29	Hex(FD1D)	Shift + F6	Hex(D9)
SF'30	Hex(FD1E)	Shift + F7	Hex(DA)
SF'31	Hex(FD1F)	Shift + F8	Hex(DB)

FUNCTION KEYS

Standard PC Keyboard, Matching Banks Layout 2, Set-Up OPTION 3

Name	Output Value	Standard PC Keyboard	Input Value PC Keyboard
SF' 0	Hex(FD01)	F1	Hex(BB)
SF' 1	Hex(FD02)	F2	Hex(BC)
SF' 2	Hex(FD03)	F3	Hex(BD)
SF' 3	Hex(FD04)	F4	Hex(BE)
SF' 4	Hex(FD05)	F5	Hex(BF)
SF' 5	Hex(FD06)	F6	Hex(C0)
SF' 6	Hex(FD07)	F7	Hex(C1)
SF' 7	Hex(FD08)	F8	Hex(C2)
SF' 8	Hex(FD09)	Ctrl + F1	Hex(DE)
SF' 9	Hex(FD0A)	Ctrl + F2	Hex(DF)
SF'10	Hex(FD0B)	Ctrl + F3	Hex(E0)
SF'11	Hex(FD0C)	Ctrl + F4	Hex(E1)
SF'12	Hex(FD0D)	Ctrl + F5	Hex(E2)
SF'13	Hex(FD0E)	Ctrl + F6	Hex(E3)
SF'14	Hex(FD0F)	Ctrl + F7	Hex(E4)
SF'15	Hex(FD10)	Ctrl + F8	Hex(E5)
SF'16	Hex(FD11)	Shift + F1	Hex(D4)
SF'17	Hex(FD12)	Shift + F2	Hex(D5)
SF'18	Hex(FD13)	Shift + F3	Hex(D6)
SF'19	Hex(FD13)	Shift + F4	Hex(D7)
SF'20	Hex(FD14)	Shift + F5	Hex(D8)
SF'21	Hex(FD15)	Shift + F6	Hex(D9)
SF'22	Hex(FD16)	Shift + F7	Hex(DA)
SF'23	Hex(FD17)	Shift + F8	Hex(DB)
SF'24	Hex(FD18)	Alt + F1	Hex(E8)
SF'25	Hex(FD19)	Alt + F2	Hex(E9)
SF'26	Hex(FD1A)	Alt + F3	Hex(EA)
SF'27	Hex(FD1B)	Alt + F4	Hex(EB)
SF'28	Hex(FD1C)	Alt + F5	Hex(EC)
SF'29	Hex(FD1D)	Alt + F6	Hex(ED)
SF'30	Hex(FD1E)	Alt + F7	Hex(EE)
SF'31	Hex(FD1F)	Alt + F8	Hex(EF)

SPECIAL FUNCTION KEYS

Name	Output	Standard PC Keyb.	FasstCom PC Keyboard	Input Value
Back TAB	Hex(FD7F)	Shift+TAB	BTAB	Hex(8F)
Clear	Hex(FD81)	ALT + C	Alt + C	Hex(AE)
			Clear (End)	Hex(E6)
Continue	Hex(FD84)	CTRL + C	CTRL + C	Hex(03)
			Ctnue	Hex(EF)
Delete	Hex(FD49)	DEL	DEL	Hex(D3)
Down	Hex(FD45)	Down Arrow	Down Arrow	Hex(D0)
Edit/Can.	Hex(FD E1)	End	Escape	Hex(1B)
Erase	Hex(FD48)	Ctrl + E	Ctrl + E	Hex(05)
			Erase Key	Hex(F0)
Global	Hex(FD7C)	CTRL + G	CTRL + G	Hex(07)
Halt/Step	**Hex(xx)***	**CTRL + S**	**CTRL + S**	**Hex(13)**
			Halt Key	**Hex(E5)**

*** When Using Wang Flow Control, Send (xx)=(13)**
When using XON/XOFF Flow Control Send (xx)=(07)

Home	Hex(FD41)	Ctrl + H	Ctrl + H	Hex(08)
			Home Key	Hex(C7)
Insert	Hex(FD4A)	INST	INST	Hex(D2)
Left	Hex(FD4D)	Left Key	Left Key	Hex(CB)
Load	Hex(FDA1)	CTRL +Home	CTRL +Home	Hex(F7)
			Shift+Run	Hex(E4)
Page Down	Hex(FD43)	Page Down	Page Down	Hex(D1)
Page Up	Hex(FD42)	Page Up	Page Up	Hex(C9)
Reset	Hex(12)	CTRL + R	CTRL + R	Hex(12)
			Reset Key	Hex(EE)
Return	Hex(0D)	Return	Return	Hex(0D)
Right	Hex(FD4C)	Right Key	Right Key	Hex(CD)
Run	Hex(FD82)	Home Key		Hex(C7)
			Run Key	Hex(CF)
SDelete	Hex(FD59)	Shift +Del	Shift +Del	Hex( )
SDown	Hex(FD55)	Ctrl+Down	Ctrl+Down	Hex( )
Set-Up	***Internal***	**Alt + X**	**Alt + X**	
SGlobal	Hex(FD7D)	Alt + G	Alt + G	Hex(A2)
Shift Can	Hex(FD50)	Ctrl + End	Ctrl + End	Hex(F5)
			Shift +Run	Hex(E4)
Shift Up	Hex(FD56)	Ctrl + Up	Ctrl + Up	Hex( )
SInsert	Hex(FD5A)	Shift+Inst	Shift+Inst	Hex( )
SLeft	Hex(FD5D)	Ctrl+Left	Ctrl+Left	Hex(F3)
SP Down	Hex(FD53)	Ctrl + PD	Ctrl + PD	Hex(F6)
Spage Up	Hex(FD52)	Ctrl + PU	Ctrl + PU	Hex( )
SRight	Hex(FD5C)	Ctrl+Right	Ctrl+Right	Hex(F4)
TAB	Hex(FD7E)	TAB	TAB	Hex(09)
Up	Hex(FD46)	Up Arrow	Up Arrow	Hex(C8)

MOUSE SUPPORT

CRT CONTROL Character TABLE

The following charts show the control codes, the character set, and the alternate character set for the 24 x 80 CRT of the Model 2336DW console. In the normal character set, the codes HEX(90) to HEX(FF) are underlined versions of characters from HEX(10) to HEX(7F); thus, adding HEX 80 to a non-underlined character's HEX value yields the HEX code of its underlined counterpart.

Table The CRT Control Codes

HEX	Action
00	Null
01	Moves cursor to the home position (top left corner of the CRT)
02	Start of multibyte control sequence
03	Clears the screen and homes the cursor
04	Reserved
05	Cursor on
06	Cursor off
07	Audio alarm
08	Cursor left 1 space (nondestructive backspace)
09	Cursor right 1 space (nondestructive)
0A	Cursor down 1 line (line feed)
0B	Reserved
0C	Cursor up 1 line
0D	Carriage return
0E	Activates attribute
0F	Attribute off; restores normal intensity

WANG NORMAL Character SET TABLE

High-order HEX Digit

Low-order HEX Digit

	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	ā	Space	0	@	P	`	p	Space	ą	_	0	@	P	`	p
1	ē	!	1	A	Q	a	q	♦	ę	!	1	A	Q	a	q
2	ī	"	2	B	R	b	r	▶	į	"	2	B	R	b	r
3	ō	#	3	C	S	c	s	◀	ǫ	#	3	C	S	c	s
4	ū	$	4	D	T	d	t	—	ų	$	4	D	T	d	t
5	ā	%	5	E	U	e	u	⌙	ą	%	5	E	U	e	u
6	ē	&	6	F	V	f	v	|	ę	&	6	F	V	f	v
7	ī	'	7	G	W	g	w	··	į	'	7	G	W	g	w
8	ō	(	8	H	X	h	x	'	ǫ	(	8	H	X	h	x
9	ū	)	9	I	Y	i	y	'	ų	)	9	I	Y	i	y
A	ą	*	:	J	Z	j	z	^	ą	*	:	J	Z	j	z
B	ę	+	;	K	[	k	§	■	ę	+	;	K	[	k	§
C	ų	,	<	L	\	l	£	!!	ų	,	<	L	\	l	£
D	Ā	-	=	M	]	m	ė	↕	Ā	-	=	M	]	m	ė
E	Ō	.	>	N	↑	n	ç	ß	Ǫ	.	>	N	↑	n	ç
F	Ū	/	?	O	—	o	¢	¶	Ū	/	?	O	—	o	¢

WANG ALTERNATE Character SET TABLE

Dec	Hex	Octal	Binary	Name	Character
128	80	200	1000 0000	C cedilla	Ç
129	81	201	1000 0001	u umlaut	ü
130	82	202	1000 0010	e acute	é
131	83	203	1000 0011	a circumflex	â
132	84	204	1000 0100	a umlaut	ä
133	85	205	1000 0101	a grave	à
134	86	206	1000 0110	a ring	å
135	87	207	1000 0111	c cedilla	ç
136	88	210	1000 1000	e circumflex	ê
137	89	211	1000 1001	e umlaut	ë
138	8A	212	1000 1010	e grave	è
139	8B	213	1000 1011	i umlaut	ï
140	8C	214	1000 1100	i circumflex	î
141	8D	215	1000 1101	i grave	ì
142	8E	216	1000 1110	A umlaut	Ä
143	8F	217	1000 1111	A ring	Å
144	90	220	1001 0000	É acute	É
145	91	221	1001 0001	ae ligature	æ
146	92	222	1001 0010	AE ligature	Æ
147	93	223	1001 0011	o circumflex	ô
148	94	224	1001 0100	o umlaut	ö
149	95	225	1001 0101	o grave	ò
150	96	226	1001 0110	u circumflex	û
151	97	227	1001 0111	u grave	ù
152	98	230	1001 1000	y umlaut	ÿ
153	99	231	1001 1001	O umlaut	Ö
154	9A	232	1001 1010	U umlaut	Ü
155	9B	233	1001 1011	cent sign	¢
156	9C	234	1001 1100	pound sign	£
157	9D	235	1001 1101	yen sign	¥
158	9E	236	1001 1110	Pt	₧
159	9F	237	1001 1111	function	ƒ
160	A0	240	1010 0000	a acute	á
161	A1	241	1010 0001	i acute	í
162	A2	242	1010 0010	o acute	ó
163	A3	243	1010 0011	u acute	ú
164	A4	244	1010 0100	n tilde	ñ
165	A5	245	1010 0101	N tilde	Ñ
166	A6	246	1010 0110	a macron	ª
167	A7	247	1010 0111	o macron	º
168	A8	250	1010 1000	opening question mark	¿
169	A9	251	1010 1001	upper left box	⌐
170	AA	252	1010 1010	upper right box	¬
171	AB	253	1010 1011	1/2	½
172	AC	254	1010 1100	1/4	¼
173	AD	255	1010 1101	opening exclamation	¡
174	AE	256	1010 1110	opening guillemets	«
175	AF	257	1010 1111	closing guillemets	»
176	B0	260	1011 0000	light block	▒
177	B1	261	1011 0001	medium block	▒
178	B2	262	1011 0010	dark block	▓
179	B3	263	1011 0011	single vertical	│
180	B4	264	1011 0100	single right junction	┤
181	B5	265	1011 0101	2 to 1 right junction	╡
182	B6	266	1011 0110	1 to 2 right junction	╢
183	B7	267	1011 0111	1 to 2 upper right	╖
184	B8	270	1011 1000	2 to 1 upper right	╕
185	B9	271	1011 1001	double right junction	╣
186	BA	272	1011 1010	double vertical	║
187	BB	273	1011 1011	double upper right	╗
188	BC	274	1011 1100	double lower right	╝
189	BD	275	1011 1101	1 to 2 lower right	╜
190	BE	276	1011 1110	2 to 1 lower right	╛
191	BF	277	1011 1111	single upper right	┐

Dec	Hex	Octal	Binary	Name	Character
192	C0	300	1100 0000	single lower left	└
193	C1	301	1100 0001	single lower junction	┴
194	C2	302	1100 0010	single upper junction	┬
195	C3	303	1100 0011	single left junction	├
196	C4	304	1100 0100	single horizontal	─
197	C5	305	1100 0101	single intersection	┼
198	C6	306	1100 0110	2 to 1 left junction	╞
199	C7	307	1100 0111	1 to 2 left junction	╟
200	C8	310	1100 1000	double lower left	╚
201	C9	311	1100 1001	double upper left	╔
202	CA	312	1100 1010	double lower junction	╩
203	CB	313	1100 1011	double upper junction	╦
204	CC	314	1100 1100	double left junction	╠
205	CD	315	1100 1101	double horizontal	═
206	CE	316	1100 1110	double intersection	╬
207	CF	317	1100 1111	1 to 2 lower junction	╧
208	D0	320	1101 0000	2 to 1 lower junction	╨
209	D1	321	1101 0001	1 to 2 upper junction	╤
210	D2	322	1101 0010	2 to 1 upper junction	╥
211	D3	323	1101 0011	1 to 2 lower left	╙
212	D4	324	1101 0100	2 to 1 lower left	╘
213	D5	325	1101 0101	2 to 1 upper left	╒
214	D6	326	1101 0110	1 to 2 upper left	╓
215	D7	327	1101 0111	2 to 1 intersection	╫
216	D8	330	1101 1000	1 to 2 intersection	╪
217	D9	331	1101 1001	single lower right	┘
218	DA	332	1101 1010	single upper left	┌
219	DB	333	1101 1011	inverse space	▉
220	DC	334	1101 1100	lower inverse	▄
221	DD	335	1101 1101	left inverse	▌
222	DE	336	1101 1110	right inverse	▐
223	DF	337	1101 1111	upper inverse	▀
224	E0	340	1110 0000	alpha	α
225	E1	341	1110 0001	beta	β
226	E2	342	1110 0010	Gamma	Γ
227	E3	343	1110 0011	pi	π
228	E4	344	1110 0100	Sigma	Σ
229	E5	345	1110 0101	sigma	σ
230	E6	346	1110 0110	mu	µ
231	E7	347	1110 0111	tau	τ
232	E8	350	1110 1000	Phi	Φ
233	E9	351	1110 1001	theta	θ
234	EA	352	1110 1010	Omega	Ω
235	EB	353	1110 1011	delta	δ
236	EC	354	1110 1100	infinity	∞
237	ED	355	1110 1101	phi	∅
238	EE	356	1110 1110	epsilon	ε
239	EF	357	1110 1111	intersection of sets	∩
240	F0	360	1111 0000	is identical to	≡
241	F1	361	1111 0001	plus/minus sign	±
242	F2	362	1111 0010	greater/equal sign	≥
243	F3	363	1111 0011	less/equal sign	≤
244	F4	364	1111 0100	top half integral	⌠
245	F5	365	1111 0101	lower half integral	⌡
246	F6	366	1111 0110	divide by sign	÷
247	F7	367	1111 0111	approximately	≈
248	F8	370	1111 1000	degree	°
249	F9	371	1111 1001	filled in degree	∙
250	FA	372	1111 1010	small bullet	·
251	FB	373	1111 1011	square root	√
252	FC	374	1111 1100	superscript n	ⁿ
253	FD	375	1111 1101	superscript 2	²
254	FE	376	1111 1110	box	■
255	FF	377	1111 1111	phantom space	

IBM CARACTER SET TABLE FOR USE AS ALT-2

Dec	Hex	Octal	Binary	Name	Character
128	80	200	1000 0000	C cedilla	Ç
129	81	201	1000 0001	u umlaut	ü
130	82	202	1000 0010	e acute	é
131	83	203	1000 0011	a circumflex	â
132	84	204	1000 0100	a umlaut	ä
133	85	205	1000 0101	a grave	à
134	86	206	1000 0110	a ring	å
135	87	207	1000 0111	c cedilla	ç
136	88	210	1000 1000	e circumflex	ê
137	89	211	1000 1001	e umlaut	ë
138	8A	212	1000 1010	e grave	è
139	8B	213	1000 1011	i umlaut	ï
140	8C	214	1000 1100	i circumflex	î
141	8D	215	1000 1101	i grave	ì
142	8E	216	1000 1110	A umlaut	Ä
143	8F	217	1000 1111	A ring	Å
144	90	220	1001 0000	E acute	É
145	91	221	1001 0001	ae ligature	æ
146	92	222	1001 0010	AE ligature	Æ
147	93	223	1001 0011	o circumflex	ô
148	94	224	1001 0100	o umlaut	ö
149	95	225	1001 0101	o grave	ò
150	96	226	1001 0110	u circumflex	û
151	97	227	1001 0111	u grave	ù
152	98	230	1001 1000	y umlaut	ÿ
153	99	231	1001 1001	O umlaut	Ö
154	9A	232	1001 1010	U umlaut	Ü
155	9B	233	1001 1011	cent sign	¢
156	9C	234	1001 1100	pound sign	£
157	9D	235	1001 1101	yen sign	¥
158	9E	235	1001 1110	Pt	₧
159	9F	237	1001 1111	function	ƒ
160	A0	240	1010 0000	a acute	á
161	A1	241	1010 0001	i acute	í
162	A2	242	1010 0010	o acute	ó
163	A3	243	1010 0011	u acute	ú
164	A4	244	1010 0100	n tilde	ñ
165	A5	245	1010 0101	N tilde	Ñ
166	A6	246	1010 0110	a macron	ª
167	A7	247	1010 0111	o macron	º
168	A8	250	1010 1000	opening question mark	¿
169	A9	251	1010 1001	upper left box	⌐
170	AA	252	1010 1010	upper right box	¬
171	AB	253	1010 1011	1/2	½
172	AC	254	1010 1100	1/4	¼
173	AD	255	1010 1101	opening exclamation	¡
174	AE	256	1010 1110	opening guillemets	«
175	AF	257	1010 1111	closing guillemets	»
176	B0	260	1011 0000	light block	▒
177	B1	261	1011 0001	medium block	▒
178	B2	262	1011 0010	dark block	▓
179	B3	263	1011 0011	single vertical	│
180	B4	264	1011 0100	single right junction	┤
181	B5	265	1011 0101	2 to 1 right junction	╡
182	B6	266	1011 0110	1 to 2 right junction	╢
183	B7	267	1011 0111	1 to 2 upper right	╖
184	B8	270	1011 1000	2 to 1 upper right	╕
185	B9	271	1011 1001	double right junction	╣
186	BA	272	1011 1010	double vertical	║
187	BB	273	1011 1011	double upper right	╗
188	BC	274	1011 1100	double lower right	╝
189	BD	275	1011 1101	1 to 2 lower right	╜
190	BE	276	1011 1110	2 to 1 lower right	╛
191	BF	277	1011 1111	single upper right	┐

Dec	Hex	Octal	Binary	Name	Character
192	C0	300	1100 0000	single lower left	└
193	C1	301	1100 0001	single lower junction	┴
194	C2	302	1100 0010	single upper junction	┬
195	C3	303	1100 0011	single left junction	├
196	C4	304	1100 0100	single horizontal	─
197	C5	305	1100 0101	single intersection	┼
198	C6	306	1100 0110	2 to 1 left junction	╞
199	C7	307	1100 0111	1 to 2 left junction	╟
200	C8	310	1100 1000	double lower left	╚
201	C9	311	1100 1001	double upper left	╔
202	CA	312	1100 1010	double lower junction	╩
203	CB	313	1100 1011	double upper junction	╦
204	CC	314	1100 1100	double left junction	╠
205	CD	315	1100 1101	double horizontal	═
206	CE	316	1100 1110	double intersection	╬
207	CF	317	1100 1111	1 to 2 lower junction	╧
208	D0	320	1101 0000	2 to 1 lower junction	╨
209	D1	321	1101 0001	1 to 2 upper junction	╤
210	D2	322	1101 0010	2 to 1 upper junction	╥
211	D3	323	1101 0011	1 to 2 lower left	╙
212	D4	324	1101 0100	2 to 1 lower left	╘
213	D5	325	1101 0101	2 to 1 upper left	╒
214	D6	326	1101 0110	1 to 2 upper left	╓
215	D7	327	1101 0111	2 to 1 intersection	╫
216	D8	330	1101 1000	1 to 2 intersection	╪
217	D9	331	1101 1001	single lower right	┘
218	DA	332	1101 1010	single upper left	┌
219	DB	333	1101 1011	inverse space	▓
220	DC	334	1101 1100	lower inverse	▄
221	DD	335	1101 1101	left inverse	▌
222	DE	336	1101 1110	right inverse	▐
223	DF	337	1101 1111	upper inverse	▀
224	E0	340	1110 0000	alpha	α
225	E1	341	1110 0001	beta	ß
226	E2	342	1110 0010	Gamma	Γ
227	E3	343	1110 0011	pi	π
228	E4	344	1110 0100	Sigma	Σ
229	E5	345	1110 0101	sigma	σ
230	E6	346	1110 0110	mu	µ
231	E7	347	1110 0111	tau	τ
232	E8	350	1110 1000	Phi	Φ
233	E9	351	1110 1001	theta	θ
234	EA	352	1110 1010	Omega	Ω
235	EB	353	1110 1011	delta	δ
236	EC	354	1110 1100	infinity	∞
237	ED	355	1110 1101	phi	∅
238	EE	356	1110 1110	epsilon	ε
239	EF	357	1110 1111	intersection of sets	∩
240	F0	360	1111 0000	is identical to	≡
241	F1	361	1111 0001	plus/minus sign	±
242	F2	362	1111 0010	greater/equal sign	≥
243	F3	363	1111 0011	less/equal sign	≤
244	F4	364	1111 0100	top half integral	⌠
245	F5	365	1111 0101	lower half integral	⌡
246	F6	366	1111 0110	divide by sign	÷
247	F7	367	1111 0111	approximately	≈
248	F8	370	1111 1000	degree	°
249	F9	371	1111 1001	filled in degree	∙
250	FA	372	1111 1010	small bullet	·
251	FB	373	1111 1011	square root	√
252	FC	374	1111 1100	superscript n	ⁿ
253	FD	375	1111 1101	superscript 2	²
254	FE	376	1111 1110	box	■
255	FF	377	1111 1111	phantom space	

IBM Character SET TABLE FOR USE AS ALT-2

Finnish
Swedish

High Order HEX Digit

Low Order HEX Digit

	1	2	3	4	5	6	7
0	@		0	É	P	é	p
1	¤	!	1	A	Q	a	q
2		"	2	B	R	b	r
3	o	#	3	C	S	c	s
4	\	$	4	D	T	d	t
5	§	%	5	E	U	e	u
6	£	&	6	F	V	f	v
7	á	'	7	G	W	g	w
8	à	(	8	H	X	h	x
9	è	)	9	I	Y	i	y
A	ù	*	:	J	Z	j	z
B	[	+	;	K	Ä	k	ä
C	°	,	<	L	ö	l	ö
D	]	-	=	M	Å	m	å
E	↑	.	>	N	Ü	n	ü
F	Æ	/	?	O	ı	o	

Cyrillic/Latin

High Order HEX Digit

Low Order HEX Digit

	1	2	3	4	5	6	7
0	â		0	@	P	Ю	П
1	ê	!	1	A	Q	А	Я
2	î	"	2	B	R	Б	Р
3	ô	#	3	C	S	Ц	С
4	û	$	4	D	T	Д	Т
5	ä	%	5	E	U	Е	У
6	ë	&	6	F	V	Ф	Ж
7	ï	'	7	G	W	Г	В
8	Ä	(	8	H	X	Х	Ь
9	ö	)	9	I	Y	И	Ы
A	ü	*	:	J	Z	Й	З
B	[	+	;	K	Ё	К	Ш
C	§	,	<	L	\	Л	Э
D	]	-	=	M	Ъ	М	Щ
E		.	>	N	↑	Н	Ч
F		/	?	O		О	▒

INTERNATIONAL Character SETS

United States

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	`	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	#	3	C	S	c	s
4	û	$	4	D	T	d	t
5	ä	%	5	E	U	e	u
6	ë	&	6	F	V	f	v
7	ï	'	7	G	W	g	w
8	ö	(	8	H	X	h	x
9	ü	)	9	I	Y	i	y
A	à	*	:	J	Z	j	z
B	è	+	;	K	[	k	§
C	ù	,	<	L	\	l	£
D	Ä	-	=	M	]	m	é
E	ö	.	>	N	↑	n	ç
F	Ü	/	?	O	←	o	ç

Azerty French
Flemish

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	`	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	#	3	C	S	c	s
4	û	$	4	D	T	d	t
5	ä	%	5	E	U	e	u
6	ë	&	6	F	V	f	v
7	ï	'	7	G	W	g	w
8	ö	(	8	H	X	h	x
9	ü	)	9	I	Y	i	y
A	à	*	:	J	Z	j	z
B	è	+	;	K	[	k	§
C	ù	,	<	L	\	l	£
D	Ä	-	=	M	]	m	é
E	ö	.	>	N	↑	n	ç
F	Ü	/	?	O		o	ç

INTERNATIONAL CARACTER SETS

Katakana

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	°		0	@	P	ｦ	ｰ
1	ｱ	!	1	A	Q	ｧ	ｱ
2	ｲ	"	2	B	R	ｨ	ｲ
3	ｳ	#	3	C	S	ｩ	ｳ
4	ｴ	$	4	D	T	ｪ	ｴ
5	ｵ	%	5	E	U	ｫ	ｵ
6	ｶ	&	6	F	V	ｬ	ｶ
7	ｷ	'	7	G	W	ｭ	ｷ
8	ｸ	(	8	H	X	ｮ	ｸ
9	ｹ	)	9	I	Y	ｯ	ｹ
A	ｺ	*	:	J	Z	ｻ	ｺ
B	ｻ	+	;	K	[	ｼ	ｻ
C	ｼ	,	<	L	¥	ｽ	ｽ
D	ｽ	-	=	M	]	ｾ	ｾ
E	ｾ	.	>	N	↑	ｿ	
F	ｿ	/	?	O	←	ﾀ	ﾟ

Netherlands

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	°	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	#	3	C	S	c	s
4	û	$	4	D	T	d	t
5	ä	%	5	E	U	e	u
6	ë	&	6	F	V	f	v
7	ï	'	7	G	W	g	w
8	ö	(	8	H	X	h	x
9	ü	)	9	I	Y	i	y
A	à	*	:	J	Z	j	z
B	è	+	;	K	[	k	§
C	ù	,	<	L	\	l	£
D	Ä	-	=	M	]	m	é
E	ö	.	>	N	↑	n	ç
F	Ü	/	?	O	←	o	¢

INTERNATIONAL Character SETS

Canadian

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	°	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	#	3	C	S	c	s
4	û	$	4	D	T	d	t
5	Â	%	5	E	U	e	u
6	Ê	&	6	F	V	f	v
7	Î	'	7	G	W	g	w
8	Ô	(	8	H	X	h	x
9	Û	)	9	I	Y	i	y
A	à	*	:	J	Z	j	z
B	è	+	;	K	[	k	ç
C	ù	,	<	L	\	l	é
D	À	-	=	M	]	m	é
E	È	.	>	N	↑	n	ç
F	Ù	/	?	O		o	ç

Danish Norwegian

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	\	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	#	3	C	S	c	s
4	û	$	4	D	T	d	t
5	ä	%	5	E	U	e	u
6	ë	&	6	F	V	f	v
7	ï	'	7	G	W	g	w
8	ö	(	8	H	X	h	x
9	ü	)	9	I	Y	i	y
A	à	*	:	J	Z	j	z
B	[	+	;	K	Æ	k	æ
C	§	,	<	L	Ø	l	ø
D	]	-	=	M	Å	m	å
E	é	.	>	N	↑	n	£
F	Ü	/	?	O		o	ç

INTERNATIONAL Character SETS

German

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	°	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	‼	3	C	S	c	s
4	û	$	4	D	T	d	t
5	←	⅋	5	E	U	e	u
6	ë	&	6	F	V	f	v
7	ï	'	7	G	W	g	w
8	à	(	8	H	X	h	x
9	è	)	9	I	Y	i	y
A	ù	*	:	J	Z	j	z
B	[	+	;	K	Ä	k	ä
C	\	,	<	L	Ö	l	ö
D	]	-	=	M	Ü	m	ü
E	é	.	>	N	↑	n	§
F	ß	/	?	O	!	o	ç

Swiss French
Swiss German

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	â		0	@	P	`	p
1	ê	!	1	A	Q	a	q
2	î	"	2	B	R	b	r
3	ô	#	3	C	S	c	s
4	û	$	4	D	T	d	t
5	ä	⅋	5	E	U	e	u
6	ë	&	6	F	V	f	v
7	ï	'	7	G	W	g	w
8	ö	(	8	H	X	h	x
9	ü	)	9	I	Y	i	y
A	à	*	:	J	Z	j	z
B	è	+	;	K	[	k	↑
C	ù	,	<	L	\	l	↓
D	ä	-	=	M	]	m	↕
E	é	.	>	N	↑	n	ç
F	ü	/	?	O		o	

INTERNATIONAL Character SETS

Spanish
Spanish/Latin

High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	á		0	@	P	°	p
1	é	!	1	A	Q	a	q
2	í	"	2	B	R	b	r
3	ó	‼	3	C	S	c	s
4	ú	$	4	D	T	d	t
5	à	⅋	5	E	U	e	u
6	è	&	6	F	V	f	v
7	ò	’	7	G	W	g	w
8	·	(	8	H	X	h	x
9	ï	)	9	I	Y	i	y
A	ü	*	:	J	Z	j	–
B	Ü	+	;	K	[	k	ı
C	ç	,	<	L	\	l	£
D	Ç	–	=	M	]	m	¿
E	ñ	.	>	N	↑	n	¡
F	Ñ	/	?	O		o	ç

Icelandic

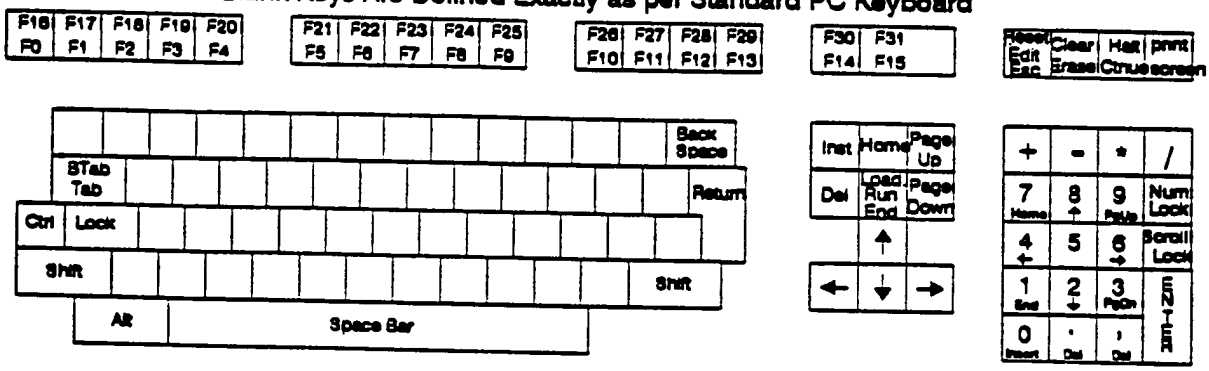
High Order HEX Digit

Low Order HEX Digit	1	2	3	4	5	6	7
0	Á		0	Đ	P	ð	p
1	É	!	1	A	Q	a	q
2	í	"	2	B	R	b	r
3	ó	#	3	C	S	c	s
4	ú	$	4	D	T	d	t
5	ý	⅋	5	E	U	e	u
6	á	&	6	F	V	f	v
7	é	’	7	G	W	g	w
8	[	(	8	H	X	h	x
9	]	)	9	I	Y	i	y
A		*	:	J	Z	j	z
B		+	;	K	Đ	k	þ
C	í	,	<	L	\	l	↑
D	ó	–	=	M	Æ	m	æ
E	ú	.	>	N	Ø	n	ö
F	ý	/	?	O		o	

FASSTCOM EXTENDED PC KEYBOARD LAYOUT

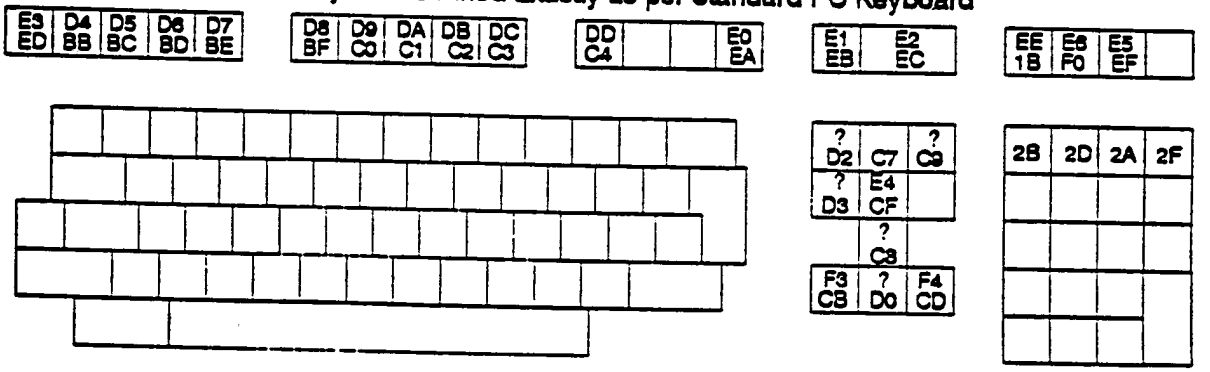
DESCRIPTION OF THE KEYS

Blank Keys Are Defined Exactly as per Standard PC Keyboard



KEYBOARD ASCII HEX CODE GENERATED

Blank Keys Are Defined Exactly as per Standard PC Keyboard



KEYBOARD SCAN CODES

Blank Keys Are Defined Exactly as per Standard PC Keyboard