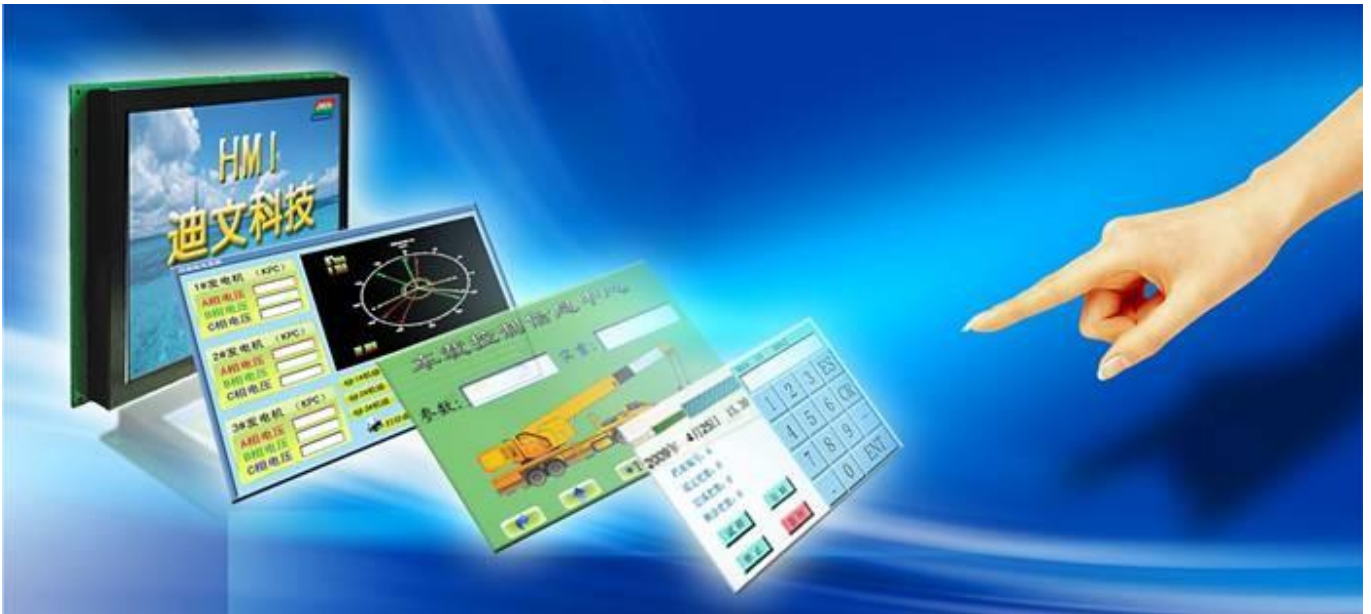




DWIN_HMI COMMAND SET

Beijing DWIN Technology Co., Ltd.



Updated English Version 2.1 at 2011.07.15

Index

1 SERIAL PORT DESCRIPTION	1
1.1 SERIAL PORT WORK MODE	1
1.2 DATA FRAME STRUCTURE	1
1.3 COMMUNICATION FRAME BUFFER (FIFO)	1
1.4 BYTE TRANSMISSION SEQUENCE	1
1.5 TRANSMITTING DIRECTION	2
2 COMMAND TABLE	2
3 COMMAND DESCRIPTION	4
3.1 HANDSHAKING(0x00)	4
3.2 SET THE CURRENT COLOR PALETTE(0x40)	4
3.3 SET CHARACTER INTERVAL (0x41)	5
3.4 ACCESS SPECIFIED POSITION'S COLOR(0x42, 0x43)	5
3.5 CURSOR DISPLAY (0x44)	5
3.6 TEXT DISPLAY(0x53, 0x54, 0x55, 0x6E, 0x6F, 0x98)	6
3.6.1 Standard character library display (0x53, 0x54, 0x55, 0x6E, 0x6F)	6
3.6.2 Character Library Display (0x98)	6
3.6.3 Set or Cancel Text Box (0x45)	8
3.7 POINT DISPLAY(0x50, 0x51, 0x74, 0x72)	8
3.7.1 Set point (0x50, 0x51)	8
3.7.2 Dynamic Curve Display (0x74)	9
3.7.3 Direct Display Operation (0x72)	9
3.8 LINE DISPLAY(0x56, 0x5D, 0x75, 0x76)	10
3.8.1 Connecting specified points (0x56, 0x5D)	10
3.8.2 Spectrum Display (0x75)	10
3.8.3 Broken Line Display (0x76)	11
3.8.4 Connect Line According to Offset (0x78)	11
3.9 ARC CURVE DISPLAY (0x57)	11
3.9.1 Arc and Circular domain display (0x57)	11
3.9.2 Arc segment display (0x5704)	12
3.10 AREA DISPLAY	12
3.10.1 Rectangle box or rectangle domain display(0x59, 0x69, 0x5A, 0x5B, 0x5C)	12
3.10.2 Polygon Display (0x64)	13
3.10.3 Double Color Bitmap Fill (0x73)	13
3.11 CLEAR SCREEN (0x52)	14
3.12 AREA SELECTED MOVEMENT(0x60, 0x61, 0x62, 0x63)	14
3.13 IMAGE OR ICON DISPLAY (0x70, 0x71, 0x99, 0xE2)	14
3.13.1 Image Display (0x70)	14
3.13.2 Image Display & Calculate accumulated sum (0x7B)	15
3.13.3 Cut Icon(0x71、0x9C、0x9D)	15
3.13.4 User-defined Icon Display (0x99)	15
3.13.5 Save current screen image in HMI (0xE2)	15
3.13.6 Save current screen image Area to Temporary Buffer (0xE9)	16
3.13.7 Restore screen image Area Saved in Temporary Buffer (0x7F)	16
3.14 BACKLIGHT BRIGHTNESS CONTROL (0x5E, 0x5F)	16

3.14.1 Backlight close (0x5E)	16
3.14.2 Set touchpad(keyed) backlight mode (0x5E)	16
3.14.3 Turn the backlight to maxim brightness (0x5F)	16
3.14.4 Adjust backlight brightness (0x5F)	16
3.15 TOUCH PANEL OPERATIONS (0x72, 0x73, 0x78, 0x79, 0xE4)	16
3.15.1 Touched position automatically upload(0x72, 0x73)	16
3.15.2 Automatically upload key code(0x78, 0x79)	17
3.15.3 Touch panel calibration mode (0xE4)	17
3.16 WORK MODE CONFIGURATION (0xE0)	17
3.17 ANIMATE CONTROL (0x9A)	18
3.17.1 Open timed-loop executive instruction function	18
3.17.2 Close timed-loop executive instruction function	19
3.18 TEMPORARY BUFFER OPERATION (0xC0, 0xC1, 0xC2)	19
3.18.1 Write temporary buffer (0xC0)	19
3.18.2 Read temporary buffer (0xC2)	19
3.18.3 Pixel setting in temporary buffer (0xC101)	19
3.18.4 Connecting lines using data in temporary buffer (0xC102)	19
3.18.5 Display broken line graph using data in temporary buffer (0xC103)	20
3.18.6 Fast display broken line graph using data in temporary buffer (0xC104)	20
3.18.7 Display broken line graph in scale using data in temporary buffer (0xC105)	20
3.18.8 Display window limited B-direction curve in scale using data in temporary buffer (0xC106)	21
3.18.9 Using temporary buffer as set-point buffer (0xC107)	21
3.18.10 Using temporary buffer to display multi-parameters (0xC108)	22
3.18.11 Using temporary buffer to buffer instruction, achieve synchronous display (0xC110)	23
3.19 KEYBOARD OPERATION(0x71, 0xE5)	24
3.19.1 Key code upload (0x71)	24
3.19.2 Key code setting(0xE5)	24
3.20 READ AND WRITE IN USER'S MEMORY (0x90, 0x91)	24
3.20.1 Write in random data memory (0x90 64KB)	24
3.20.2 Write in sequential data memory (0x90 30MB)	24
3.20.3 Read data memory (0x91)	25
3.21 DOWNLOAD CHARACTER LIBRARY AND CONFIGURATION FILES (0xF2)	25
3.22 SIMPLE ALGORITHM SUPPORT (0xB0)	25
3.22.1 Pinyin input method (0xB001、0xB004)	25
3.22.2 MAC calculate (0xB002)	25
3.22.3 Array sort (0xB003)	26
3.23 BUZZER CONTROL (0x79)	26
3.24 CLOCK(RTC) DISPLAY AND READ OFF (0x9B, 0xE7)	26
3.24.1 Close clock display	26
3.24.2 Open clock display	26
3.24.3 Clock setting	26
3.24.4 Read off current time (the Gregorian calendar)	27
3.24.5 Read off current time (the Luna calendar)	27
3.25 PLAY MUSIC(0x30, 0x32, 0x33)	27
3.25.1 Play specified address music (0x30)	27
3.25.2 Volume adjustment (0x32)	27
3.25.3 Stop playing (0x33)	27

3. 26 CONFIGURATION FILE (TOUCHPAD INTERFACE, KEYED INTERFACE, ANIMATE, ICON LIBRARY).....	28
3.26.1 Touchpad interface switch (0x1E, 0x1A configuration file)	28
3.26.2 Keyed interface switch (configuration file 0x1B)	30
3.26.3 Animate auto-play(configuration file 0x1C)	31
3.26.4 Icon display(configuration file 0x1D).....	32
3. 27 HMI AND VIDEO SWITCHING (0x7A)	33
3. 28 FORCE A FULL SCREEN REFRESH ONE TIME (0XD0)	33
4 UPDATE METHOD OF HMI SOFTWARE.....	34
5 REVISION HISTORY	35

1 Serial Port Description

1.1 Serial Port Work Mode

DWIN's all HMI products use asynchronous, UART serial port with serial port mode--8n1;i.e. Using 10 bits per dat byte: one initial bit, 8 data bits(LSB first) and one halt bit.

- When powered on, if the terminal I/O 0 pin is high or floating, serial port band rate is set by users beforehand, ranging from 1200 to 115200 bps.(More details can be found in command 0xE0.)
- When powered on, if the terminal I/O 0 pin is low, serial port band rate is fixed at 921600 bps.

Note: there are three setting modes in DWIN HMI I/O pin:

Settled by wire jumper pad, I/O0 is lower level in short circuit; high level in open circuit

Settled by contact pin, I/O0 is lower level in short circuit; high level in open circuit

Directed by status of USB port, I/O0 is lower level with USB connection and high level without USB connection.

1.2 Data Frame Structure

DWIN HMI serial port's data frame contains 4 data blocks, as shown in the table below:

Data Block	1	2	3	4
Examples	0xAA	0x70	0x01	0xCC 0x33 0xC3 0x3C
Description	header, fixed as 0xAA	command	Data, maximum 249 bytes	frame end (trailer)

1.3 Communication Frame Buffer (FIFO)

DWIN HMI has 24 Rx frame FIFO buffers, which is push-up storage; therefore, users can continuously send data to HMI before the Rx frame FIFO buffers overflow.

DWIN HMI has a hardware pin (named /Busy in user interface) to denote the FIFO status. Normally, /Busy pin keeps high (RS232 is negative); when there is only one Rx frame could be buffered, /Busy pin alters to low level. (RS232 is positive)

Since fast processing, there is no need for users concerning the status of /Busy pin. (DWIN's picture distributed software, for example, does not care about /Busy pin's status)

However, for applications of sending multiple date frame in a short time, such as rapidly refresh hundreds of monitor parameters at the same time, users are advised to use /Busy signal controlling serial port; and when /Busy signal is low, stop sending data to HMI.

If "drop-frame" phenomenon(some data do not display) occurs in the process of using HMI, overflowing of buffers may be the reason. Oscilloscope can help to inspect whether Busy signal alters. If so, deciding the transmitting speed or add hardware inspection to deal with Busy signal.

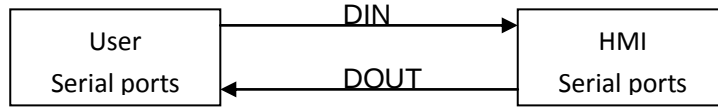
1.4 Byte Transmission Sequence

DWIN HMI commands and data are Hex format. For word data (2 byte), it is transmitted by MSB First Mode. E.g., X coordinate is 100 whose Hex format is 0x0065; then when x is transmitted to HMI, the proper sequence is 0x00 0x64.

1.5 Transmitting Direction

On DWIN HMI, Transmitting direction is defined as follows:

- Downstream (TX) when user sends data to HMI, data is received with DIN pin in HMI;
- Upstream (Rx) when HMI sends data to user, data is transmitted from DOUT pin in HMI.



2 Command Table

Categories	Instruc tions	Command parameter	Illustration
Hand shake	0x00	None	Check the configuration and version
Parameter configuration	0x40	Fcolor+Bcolor	Palette setting
	0x41	D_X (0x00-0x7F) +D_Y (0x00-0x7F)	Character space setting
	0x42	X+Y	Move the appointed color to background color palette
	0x43	X+Y	Move the appointed color to foreground color palette
	0x44	Mode + X + Y +Wide (0x01-0x1F) +Height (0x01-0x1F)	Cursor display mode setting
Text display	0x53	X + Y+ String	8X8 lattice ASCII character
	0x54		16×16lattice GBK
	0x55		32×32 GB2312
	0x6E		12×12 GBK
	0x6F		24×24 GB2312
	0x98	X+Y+Lib_ID+C_mode+C_dot+Fcolor+Bcolor +String	Display any lattice, any encoded string.
Points Setting	0x50	(x,y) ₀ + (x,y) ₁ ++ (x,y) _n	More points setting in the background color.(delete point)
	0x51		More points in the foreground color.
	0x74	X+Y _s +Y _e +Bcolor+ (y, Fcolor) ₁ ++ (y, Fcolor) _n	Dynamic curve display.
	0x72	Address(H:M:L)+Data_word ₀ ++ Data_word _n	Operation to the buffer of video card.
Lines & polygon	0x56	(x,y) ₀ +(x,y) ₁ ++(x,y) _n	Polygon display: Line the points with foreground colored segment.
	0x6D		Polygon delete: Line the points with background colored segment
	0x75	X+Y+Height_max+Height ₀ + Height ₁ ++ Height _n	Spectrum display: display a continuous vertical line with the same end in a fast rhythm.
	0x76	X+X_dis(0x00-0xFF)+Y ₀ + Y ₁ ++ Y _n	Line chat display (Xi=X+i*X_dis,Yi=Yi)
Arcs	0x57	(Type,x,y,r) ₀ +(Type,x,y,r) ₁ +.....+ (Type,x,y,r) _n	Arcs display
Rectangles	0x59	(X _s ,Y _z ,X _e ,Y _e) ₀ +(X _s ,Y _z ,X _e ,Y _e) ₁ ++ (X _s ,Y _z ,X _e ,Y _e) _n	Show rectangles: display rectangles by foreground color)
	0x69		Delete rectangles: display rectangles by background color
Areas Operation	0x64	X+Y+Color	Fill in the appointed area
	0x52	无	Clear screen
	0x5A	(X _s ,Y _z ,X _e ,Y _e) ₀ +(X _s ,Y _z ,X _e ,Y _e) ₁ ++ (X _s ,Y _z ,X _e ,Y _e) _n	Areas deleting
	0x5B		Fill in more than one appointed areas.
	0x5C		Areas color changing
	0x60	(X _s ,Y _z ,X _e ,Y _e ,n) ₀ +(X _s ,Y _z ,X _e ,Y _e ,n) ₁ +.....+ (X _s ,Y _z ,X _e ,Y _e ,n) _n	Appointed areas ring-shifting to the left
	0x61		Appointed areas ring-shifting to the right
	0x62		Appointed areas shifting to the left
	0x63		Appointed areas shifting to the right

Pictures & Icons	0x70	Picture_ID	Picture showing
	0x71	Picture_ID+X _s +Y _s +X _e +Y _e +X+Y	Display part of a picture in the memory (background display)
	0x9C	Picture_ID+X _s +Y _s +X _e +Y _e +X+Y	Display part of a picture in the memory (background does not display)
	0xE2	Picture_ID	Picture saving
	0x99	(x,y,Icon_ID) ₀ +(x,y,Icon_ID) ₁ +.....+(x,y,Icon_ID) _n /无	User-defined icons display
Animation	0x9A	0xFF/Pack_ID	Turn off/on the automatic implementation of the user's pre-setting instruction set
Temporary Buffer Operation	0xC0	Address(H:L)+ Data_word ₀ ++ Data_word _n	Writing data to the temporary buffer
	0xC1	0x01+Address+Pixel_Number(H:L)	Display the pre-set date points in the temporary buffer
		0x02+Address+Line_Number(H:L)	Display the pre-set date lines in the temporary buffer
		0x03+Address+X+Y+Line_Number+D_x+Dis_x+K_y+Color	dynamic curve scaling: connecting the data points in the temporary buffer zone
		0x04+Addr1+X+Y+Line_Number+0x01+Dis_x+Color1+Addr0+ Color0	Oscilloscope : connecting the data points in temporary buffer in a flicker-free high-speed
	0xC1	0x05+Address+X+Y+Line_Number+D_x+Dis_x+M_y+D_y+Color	Using the data in the temporary buffer to display line charts.
		0x06+Address+X+Y+Line_Number+D_x+Dis_x+M_y+D_y+Color+Ymin+Ymax	Using the data in the temporary buffer zoom to display a window-constrained bi-directional line chart
		0x10+Address+Frame_Number	Using the command in the temporary buffer to perform a synchronize display
Database Operation	0xF2	0xF2+0xF2+0x5A+0xA5+Lib_ID	Font modification
	0x90	0x55+0xAA+0x5A+0xA5+Address (H:MH:ML:L) +Data	Write data to the user's database (32MB)
	0x91	Address+Read_Length(H:L)	Read data from the database (32MB)
Key board Operation	0x71	K_code	Key code uploading
	0xE5	0x55+0xAA+0x5A+0xA5+K_Code ₀ ++K_Code _n	Key code port modification
Touch pad Operation	0x72	Touch_X+Touch_Y	Uploading the last data after the touch-screen is released, (which can turn off by 0xE0 instruction)
	0x73		Uploading data when pressing the touch panel(uploading once only by setting the command of 0xE0)
	0xE4	0x55+0xAA+0x5A+0xA5	Touch panel adjusting
	0x78	Touch_Code	Uploading the defaulted key code when switching the touch interface.
Buzzer Operation	0x79	BZ_time	Buzzing once only (10×Bz_time mS)
Backlight Control	0x5E	Non or 0x55+0xAA+0x5A+0xA + V_ON+V_OFF+ON_TIME	Turn off the backlight or control the backlight mode by touching or keying.
	0x5F	Non or PWM_T(0x00-0x3F)	Turn the backlight on or adjusting the brightness by PWM.
Clock Operation	0x9B	0x5A 0x5B(read)/0x00(off)/0xFF+M+TM+Color+X+Y(ON)	Clock on/off ; read the clock
	0xE7	0x55+0xAA+0x5A+0xA5 + YY:MM:DD:HH:MM:SS	Clock adjusting
Parameter Configuration	0xE0	Panel_Set+Bode_Set+Para1	Configuring the user's serial port speed and the touch-screen data uploading.
Algorithm	0xB0	Downlode:0x01+PY_Code answer: 0x01+HZ_num+String	

		Download :0x02+A+B+C+D answer: 0x02+E+F	Calculating(A × B + C) / D, E is 4 bytes quotient, F is 2 bytes remainder
		Downlode :0x03+Data_Pack0 answer: 0x03+ Data_Pack1	Array listing of unsigned integers(2 bytes)
Volume Operation	0x30	Start_Seg+Play_number+Play_time	Play the music in the appointed zoom
	0x32	Volume_L+Volume_R+0x00	Volume adjusting
	0x33	0x55+0xAA+0x5A	Stop playing
	0x3F	'OK'	Sound-op response
Video Operation	0x7A	Work_Mode+Video_mode+Video_CH	HMI and Video mode switching(support CVBS / S-VIDEO signal input and NTSC / PAL format).
Configuration file Operation		Pic_Now+(x _s ,y _z ,x _e ,y _e)+P_next+P_cut+Touch_Code	Touch interface automatically switching (0x1E font files)
		Pic_Now+0x00:K_Code+Pnext+P_cut+Touch_Code	Touch interface automatically switching (0x1B font files)
		Delay+Length+ Command	Play auto-instructions(0x1C font files)
		Pic_ID+(x _s ,y _z ,x _e ,y _e)	Icon Character Definition (0x1D font files)
		Command_Length+Command+String	Uploading the instructions pre-settled by users(0x1A font file)
		***	Configuration files of user's commands (0x1F font files)
Upgrading		DWIN_M600_BOOT!	Upgrading the core soft wares on line through Serial

3 Command Description

3.1 Handshaking(0x00)

Tx: AA 00 CC 33 C3 3C

Rx: AA 00 'OK_V*.*' P1 P2 P3 Pic_ID CC 33 C3 3C

- OK_V*.*, *.* is the current version of HMI
- P1 is the panel setting mode using in current HMI
- P2 is the serial baud rate set by current users
- P3 is Touch Panel setting mode.(undefined in T or K serial terminal)

DWIN HMI takes 0.5~2 seconds for initialization (varies with different power supplies) and response nothing before initialization finished. So users can detect whether HMI is ready by sending handshaking command.

3.2 Set the current color palette(0x40)

TX: AA 40 <FC> <BC> CC 33 C3 3C

Rx: None

- <FC>: foreground color palette, 2 bytes(16 bit, 65k color), default reset value is 0xFFFF(white)
- <BC>: background color palette, 2 bytes(16 bit, 65k color), default reset value is 0x001F(blue)
- color palette of 16 bit is defined to 5R6G5B mode as table below:

16 bit definitions of Color Palette																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Define	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
	Red 0xF800					Green 0x07E0						Blue 0x001F				



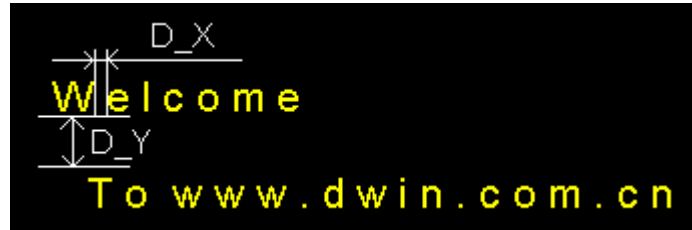
Once set, it will maintain that way unless it resets; it will restore the default values once HMI hardware resets.

3.3 Set character interval (0x41)

TX: AA 41 <D_X> <D_Y> CC 33 C3 3C

Rx: none

- <D_X>: Character interval in X direction (column interval), ranging from 0x00 to 0x7F. Default reset value is 0x00.
- <D_Y>: Character interval in Y direction (line interval), ranging from 0x00 to 0x7F. Default reset value is 0x00.



Once set, it will maintain that way unless it resets; it will restore the default values once HMI hardware resets.

3.4 Access specified position's color(0x42, 0x43)

Tx: AA <CMD> <X> <Y> CC 33 C3 3C

Rx: None

- <CMD>: command 0x42 accesses specified position's color to background color palette; command 0x43 accesses specified position's color to foreground color palette.
- <X> <Y>: specified position's coordinate.(in DWIN HMI, coordinate represent with 2 bytes)
e.g. : AA 42 00 10 01 00 CC 33 C3 3C
to access the color of position x=16 (0x0010) y=256 (0x0100) to background color palette.

3.5 Cursor display (0x44)

Tx: AA 44 <Cursor_EN> <X> <Y> <Cursor_Width> <Cursor_Height> CC 33 C3 3C

Rx: None

- <Cursor_EN>
0x01 Cursor display on; and will display on (x,y)
0x00 Cursor display closed
- <X>, <Y> is the coordinate's top left corner position of cursor
- <Cursor_Width> display the width of cursor, ranging from 0x01 to 0x1F
- <Cursor_Height> display the height of cursor, ranging from 0x01 to 0x1F



If cursor display is closed, other parameters in this command are of no use.

e.g. : AA 44 01 00 80 00 60 10 03 CC 33 C3 3C

to enable a cursor display in (128,96) with 16 pixels width and 3pixels height

3.6 Text display(0x53, 0x54, 0x55, 0x6E, 0x6F, 0x98)

3.6.1 Standard character library display (0x53, 0x54, 0x55, 0x6E, 0x6F)

Tx: AA <CMD> <X> <Y> <String> CC 33 C3 3C

Rx: None

➤ <CMD>

0x53: display 8*8 dot matrix ASCII character

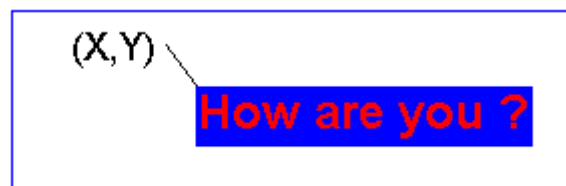
0x54: display 16*16 dot matrix extended code string (ASCII character displayed as half-width 8*16 dot matrix)

0x55: display 32*32 dot matrix internal code string (ASCII character displayed as half-width 16*32 dot matrix)

0x6E: display 12*12 dot matrix extended code string (ASCII character displayed as half-width 6*12 dot matrix)

0x6F: display 24*24 dot matrix internal code string (ASCII character displayed as half-width 12*24 dot matrix)

- <X> <Y> display the initial position of character string(the coordinate's top left corner of first character)
- <String> string to be shown. Chinese Character is encoded by GB23212 (0x55, 0x6F, internal code) or GBK (0x54, 0x6E, extended internal code); colors are set by command 0x40; character interval is set by command 0x41; and auto-wrap to next line at the end of line. Command 0x0D and 0x0A represent "back to line start" and "wrap to next line"



e.g.

AA 55 00 80 00 30 48 6F 77 20 61 72 65 20 79 6F 75 20 3F CC 33 C3 3C

to display string "How are you?" from position (128,48)

3.6.2 Character Library Display (0x98)

Tx: AA <X> <Y> <Lib_ID> <C_Mode> <C_dots> <Fcolor> <Bcolor> <String> CC 33 C3 3C

Rx: None

- <X> <Y> display 1st character's top left coordinate of the string.
- <Lib_ID> to select Character Library, ranging from 0x00 to 0x3B, corresponding to the command 0xF2

There are sixty font libraries stored in 32MB FLASH and following is Lib_ID definition:

Lib_ID	Vol.	Instruction	Default Value
0x00-0x1F	128KB	32 packets of small Font Libraries for special icons and different scripts ASCII display	0x00= ASCII fo library(please do n revise) 0x02-0x1F: empty
0x20-0x3B	1MB	28 packets of big 1MB Font Libraries ✧ Single font library can store within-16 dots GBK. (12x12 or 16x16) within-32 dots GB2312 font libraries (12x12\16x16\24x24\32x32) ✧ Assembly application is allowed to be used with biggest 28MB dot font library. In this occasion, Lib_ID in 0x98 and 0xF2 command the address of head font library.	0x20=12 dots GBK 0x21=16 dots GBK 0x22=24 dots GB231 0x23=32 dots GB231 0x24-0x3B: empty

- <C_Mode> to select Text Display Mode and Encoding Mode as follows:

Bit	.7-.4	.3-.0
Definition	Display Mode	Character Encoding Method
Description	.7=1 text display, foreground color =0 no text display .6=1 text display, foreground color =0 no text display .5,.4 undefined, write0.	0x00 8bit Character library contains most 256 bytes 0x01 GB2312 Chinese internal code 0x02 GBK Chinese extended internal code or Korea HANGUL code 0x03 BIG5 Traditional Chinese Code 0x04 SJIS Japanese Code 0x05 UNICODE UNICODE Code (UTF16) 0x06-0x0F: Undefined



Samples of Display Mode Setting:

Foreground color	ON	ON	OFF
Background color	ON	OFF	ON
<C_mode>high value 2bit	0xC*	0x8*	0x4*
character display effect	B area	A area	C area

e.g., if C_Mode=0x*1,internal code encoding, then C_dots=0x07 will display 64*64 Chinese character and 32*64 ASCII string.

- <Fcolor> foreground color of character
- <Bcolor> background color of character
- <String> string data, display interval is set by command 0x41 and automatically wrap at the end of a line.

e.g.:

AA 98 00 80 00 30 00 C0 04 F8 00 00 1F 48 6F 77 20 61 72 65 20 79 6F 75 20 3F CC 33 C3 3C

- <C_dots> character size setting

C_Dots	Character Library Type (C_Mode set the lower 4bit)		
	0x00 or 0x05	0x01-0x04	
		ASCII Character	Non-ASCII Character
0x00	8*8	6*12	12*12
0x01	6*12	8*16	16*16

0x02	8*16	12*24	24*24
0x03	12*24	16*32	32*32
0x04	16*32	20*40	40*40
0x05	20*40	24*48	48*48
0x06	24*48	28*56	56*56
0x07	28*58	32*64	64*64
0x08	32*64	—	40*80
0x09	12*12	—	48*96
0x0A	16*16	—	56*112
0x0B	24*24	—	64*128
0x0C	32*32	—	80*80
0x0D	40*40	—	96*96
0x0E	48*48	—	112*112
0x0F	56*56	—	128*128
0x10	64*64	—	—
0x11	40*80	—	—
0x12	48*96	—	—
0x13	56*112	—	—
0x14	64*128	—	—
0x15	80*80	—	—
0x16	96*96	—	—
0x17	112*112	—	—
0x18	128*128	—	—
0x19	6*8	—	—
0x1A	8*10	-	-
0x1B	8*12	-	-
0x1C	100*200	-	-
0x1D	200*200		
0x1E	64*48		

3.6.3 Set or Cancel Text Box (0x45)

Set Text Box

Tx: AA 45 <Xs> <Ys> <Xe> <Ye> CC 33 C3 3C

Rx: None

➤ <Xs> <Ys> <Xe> <Ye> text box position.

After set the text box, the text display will be auto linefeed within the text box.

Cancel Text Box

Tx: AA 45 00 CC 33 C3 3C

Rx: None

After cancel the text box, the text display will be auto linefeed within the full screen.

3.7 Point display(0x50, 0x51, 0x74, 0x72)

3.7.1 Set point (0x50, 0x51)

Tx: AA <CMD> <(x0,y0) (x1,y1)(xi, yi)> CC 33 C3 3C

Rx: None

➤ <CMD>

0x50: display point using background color(delete point)

0x51: display point using foreground color(set point)

➤ <(x0,y0) (x1,y1)(xi,yi)> coordinates to be set; and the maxim point number in one frame is 62.

e.g.: AA 51 00 00 00 00 00 03 00 06 00 05 00 20 CC 33 C3 3C

to display 3 points using foreground color at (0,0),(3,6),(5,32) as follows:



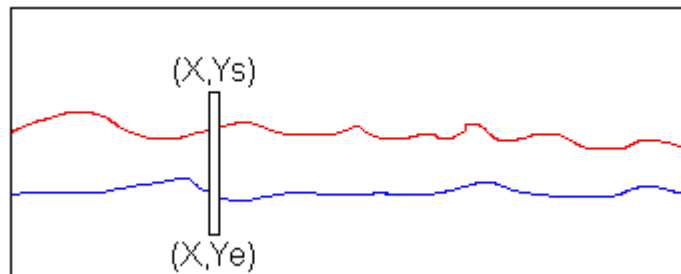
3.7.2 Dynamic Curve Display (0x74)

Tx: AA 74 <X> <Ys> <Ye> <Bcolor> <(Y0,Fcolor0), (Y1,Fcolor1).....(Yi, Fcolori)> CC 33 C3 3C

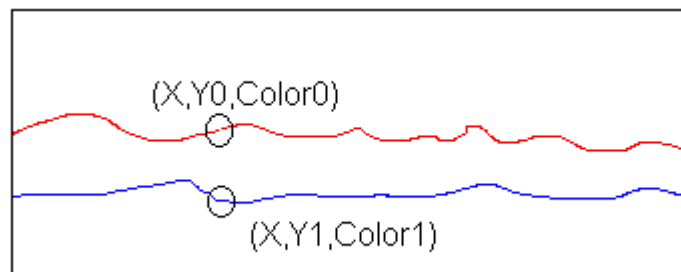
Rx: none

This command is mainly used for implementation of multi-dynamic curves on single screen. Terminals process commands by following steps:

Step 1: Erase the vertical line from (X,Ys) to (X,Ye), using specified color<Bcolor>. Empty the original content.



Step 2: Set point at (X,Yi), using specified color<Fcolori>.



This command will not alter Palette's default settings.

3.7.3 Direct Display Operation (0x72)

Tx: AA 72 <Address_H:M:L> <Pixel_data0.....Pixel_datan> CC 33 C3 3C

Rx: None

This command is mainly used to download figures to HMI, which users generally do not use.

3.8 Line Display(0x56, 0x5D, 0x75, 0x76)

3.8.1 Connecting specified points (0x56, 0x5D)

Tx: AA <CMD> <(x0,y0) (x1,y1)(xi, yi)> CC 33 C3 3C

Rx: None

➤ <CMD>

0x56: Connect specified points by lines, using foreground color(set by command 0x40).

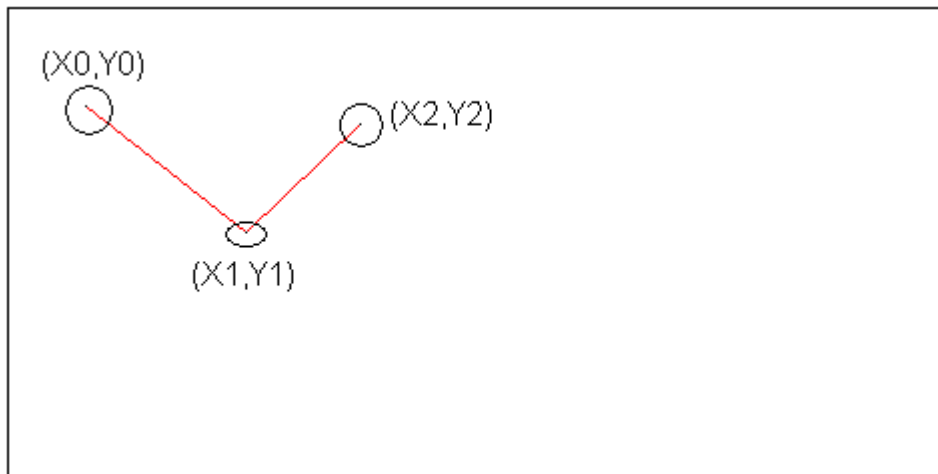
0x5D: Connect specified points by lines, using background color(set by command 0x40).

➤ <(x0,y0) (x1,y1)(xi,yi)>: coordinates of specified points.

e.g.:

AA 56 00 28 00 32 00 78 00 70 00 B1 00 3A CC 33 C3 3C

Connect 3 points, (40,50),(120,112),(177,58), to lines, using foreground color, as follows:



3.8.2 Spectrum Display (0x75)

Tx: AA 75 <(x, y)>, <H_max>, <H0.....Hi> CC 33 C3 3C

Rx: None

➤ <(x,y)>: x is initial coordinate on spectrum's X-axis. x=x+1 after displaying a spectrum line; y is horizontal position on spectrum and one spectrum line's starting and end coordinates are y and (y-Hi) respectively.

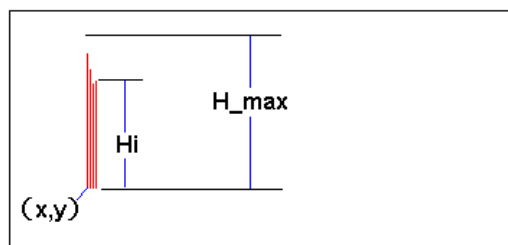
➤ <H_max> Maxim height of a spectrum line.

if H_max=0x01-0xFF, spectrum line's height Hi is a variable of 1 byte.

if H_max=0x00, the following 2 bytes represent H_max and Hi is a variable of 2 bytes.

➤ <H0.....Hi>: height of one spectrum line, one or two bytes.

Spectrum line's color is set by Palette Command 0x40 .When display, spectrum line, height Hi, will display by foreground color, other spectrum lines, height H_max-Hi, will display (be erased) by background color.



3.8.3 Broken Line Display (0x76)

Tx: AA 76 <x>, <x_dis>, <Y0.....Yi> CC 33 C3 3C

Rx: None

- <x>: the initial coordinate on broken line's X-axis. $x = x + x_dis$ after connecting a point by line.
- <x_dis>: the increment of coordinate on X-axis.
- <Y0.....Yi>: the top points' coordinates of broken lines, using foreground color to display.

This command's function is similar to command 0x56 except that coordinate X here is auto-calculated by HMI for fast connecting.

3.8.4 Connect Line According to Offset (0x78)

Tx: AA 78 <x, y>, <dx0,dy0>, <dx1,dy1>,, <dxn, dyn> CC 33 C3 3C

Rx: None

- <x, y>: is initial coordinate of the line.
- <dxn, dyn>: 1 byte offset of x and y, MSB (.7) is sign bit, "1" indicates negative.

3.9 Arc Curve display (0x57)

3.9.1 Arc and Circular domain display (0x57)

Tx: AA 57 (<Type_0> <X_0> <Y_0> <R_0>)(<Type_i> <X_i> <Y_i> <R_i>) CC 33 C3 3C

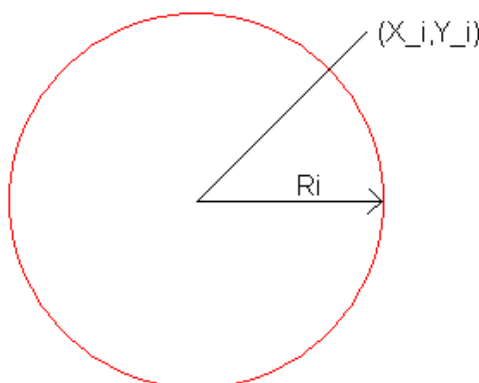
Rx: None

- <Type_i> to select type
 - 0x00 display specified arc, using invert color
 - 0x01 display specified arc, using foreground color (set by command 0x40)
 - 0x02 display specified circular domain, using invert color
 - 0x03 fill specified circular domain, using foreground color(set by command 0x40)
- <X_i> <Y_i> coordinates of the center of arc or circular domain
- <R_i> radius of the arc or circular domain, ranging from 0x01 to 0xFF

e.g.:

AA 57 01 00 60 00 60 40 CC 33 C3 3C

to display an arc, using foreground color, with the center (96,96) and radius 64 as follows:



3.9.2 Arc segment display (0x5704)

Tx: AA 57 04 <X> <Y> <R> <A_S> <A_E> CC 33 C3 3C

Rx: None

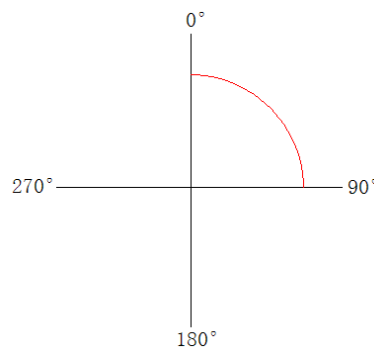
- <X> <Y>: coordinate of arc
- <R>: radius of arc, 0x0001-0x03FF
- <A_S>: start angle of arc, 0x00-0x02D0 (0-720), the unit is 0.5° .
- <A_E>: end angle of arc, 0x00-0x02D0 (0-720), the unit is 0.5° .

Display color is foreground color set by command 0x40.

e.g.:

AA 57 04 00 60 00 60 00 40 00 00 00 B4 CC 33 C3 3C

to display an arc, using foreground color, with the center (96,96) and radius 64 , start angle is 0° , end angle is 90° , as follows:



3.10 Area Display

3.10.1 Rectangle box or rectangle domain display(0x59, 0x69, 0x5A, 0x5B, 0x5C)

Tx: AA <CMD> (<Xs_0> <Ys_0> <Xe_0><Ye_0>)(<Xs_i> <Ys_i> <Xe_i><Ye_i>) CC 33 C3 3C

Rx: None

- <CMD >
 - 0x59: to display rectangle box, using foreground color(set by command 0x40), with one pixel line width
 - 0x69: to display rectangle box, using background color(set by command 0x40), with one pixel line width
 - 0x5A: to fill rectangle domain, using background color(set by command 0x40)
 - 0x5B: to fill rectangle domain, using foreground color(set by command 0x40)
 - 0x5C: to display specified rectangle domain, using invert color(set by command XOR 0xFF). It has reverted when using invert color again.
- <Xs_i> <Ys_i> <Xe_i> <Ye_i>: (Xs_i,Ys_i) is top left coordinate of rectangle box or domain; and (Xe_i,Ye_i) is bottom right coordinate.

e.g.:

AA 5C 00 40 00 40 00 80 00 80 CC 33 C3 3C

to invert color of rectangle domain specified by top left coordinates (64,64) and bottom right coordinate (128,128) as follows:



3.10.2 Polygon Display (0x64)

Tx: AA 64 <(x, y) > <Color> CC 33 C3 3C

Rx: None

- <(x,y)>: seed point position of area filling
- Color: fill color

Note:

The initial filled areas color should be the same as the seed point color, otherwise will be only fill the areas whose color is the same as seed point (other color area will be treated as boundary).

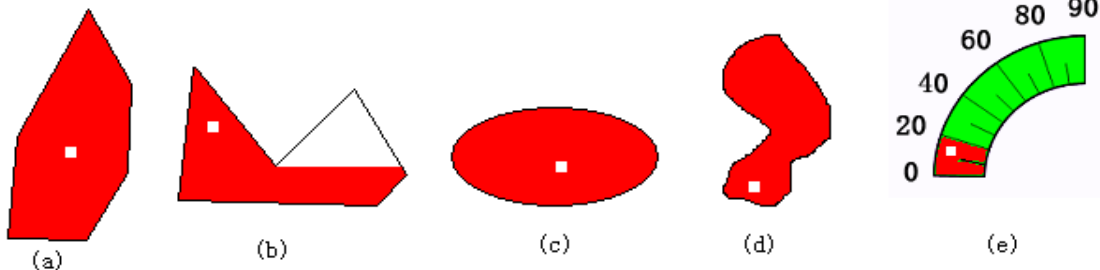
Only applies to "convex polygons", "concave polygon area" will have some area can not be filled (as figure b), by setting different seeds point position to achieve "sunken polygon area" completely fills.

Will not change the palette properties.

e.g.:

AA 64 00 64 00 64 F8 00 CC 33 C3 3C

White point is seed point, be filled actually.



3.10.3 Double Color Bitmap Fill (0x73)

Tx: AA 73 Color0 Color1 (Xs, Ys) (Xe, Ye) (X, Y) Data CC 33 C3 3C

Rx: None

- Color0: 0 bit represents filled color
- Color1: 1 bit represents filled color
- (Xs, Ys): the upper left corner coordinate of the bitmap display area boundary.
- (Xe, Ye): the bottom right corner coordinate of the bitmap display area boundary.
- (X, Y): starting coordinate of bitmap filling.
- Data: Double color bitmap information.

3.11 Clear Screen (0x52)

Tx: AA 52 CC 33 C3 3C

Rx: None

to fill full screen with background color(set by command 0x40)(clear screen)

3.12 Area Selected Movement(0x60, 0x61, 0x62, 0x63)

Tx: AA <CMD> (<Xs_0> <Ys_0> <Xe_0> <Ye_0> <N_0>)(<Xs_i> <Ys_i> <Xe_i> <Ye_i> <N_i>) CC 33 C3 3C

Rx: None

➤ <CMD >

0x60: to right-circular shift selected area,i.e.,to move the far-left area shift to far-right.

0x61: to left-circular shift selected area,i.e.,to move the far-right area shift to far-left.

0x62: to move selected area right,i.e.,to move from right to left,far-left area lost, using background color to fill far-right area(set by command 0x40)

0x63: to move selected area left,i.e.,to move from left to right,far-right area lost, using background color to fill far-left area(set by command 0x40)

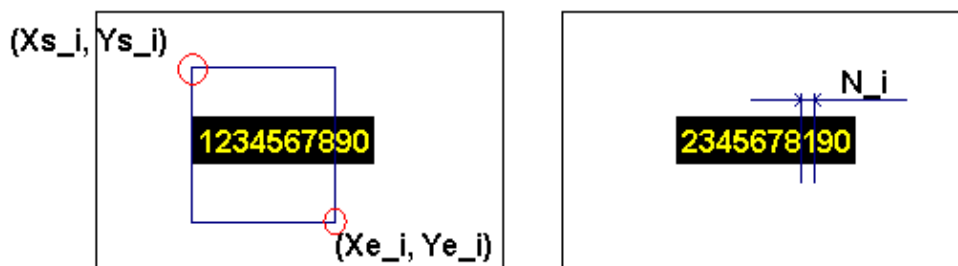
➤ <Xs_i> <Ys_i> <Xe_i> <Ye_i>: (Xs_i,Ys_i): top left and bottom right coordinates of selected area.

➤ <N_i>: numbers of dot matrix in area movement, ranging from 0x01 to 0x0F.

e.g.:

AA 60 00 40 00 40 00 80 00 80 08 CC 33 C3 3C

to move the area, selected by top left coordinate(64,64) and bottom right coordinate(128,128), right 8 dot matrix from left as follows:



3.13 Image or Icon Display (0x70, 0x71, 0x99, 0xE2)

3.13.1 Image Display (0x70)

Tx: AA 70 <Pic_ID> CC 33 C3 3C

Rx: None

<Pic_ID>: index ID(set by command 0xE2) of the image saved in HMI Flash Memory

e.g.

AA 70 00 CC 33 C3 3C to display the zeroth image saved in HMI

AA 70 01 02 CC 33 C3 3C to display the 258th image saved in HMI

If the number of saved images exceeds 256, image ID can overtake 255 only when represented directly by 2 bytes.

3.13.2 Image Display & Calculate accumulated sum (0x7B)

Tx: AA 7B <Pic_ID> CC 33 C3 3C

Rx: AA 7B <Checksum_H:L> CC 33 C3 3C

Rx: None

- <Pic_ID>: index ID(set by command 0xE2) of the image saved in HMI Flash Memory
- <Checksum_H:L>: 2 Byte accumulated sum of current image.

This command is used to check the image which download to the flash, to ensure the download is correct.

3.13.3 Cut Icon(0x71、0x9C、0x9D)

Tx: AA 71 <Pic_ID> <Xs> <Ys> <Xe> <Ye> <X> <Y> CC 33 C3 3C

or: AA 9C <Pic_ID> <Xs> <Ys> <Xe> <Ye> <X> <Y> CC 33 C3 3C

or: AA 9D <Pic_ID> <Xs> <Ys> <Xe> <Ye> <X> <Y> CC 33 C3 3C

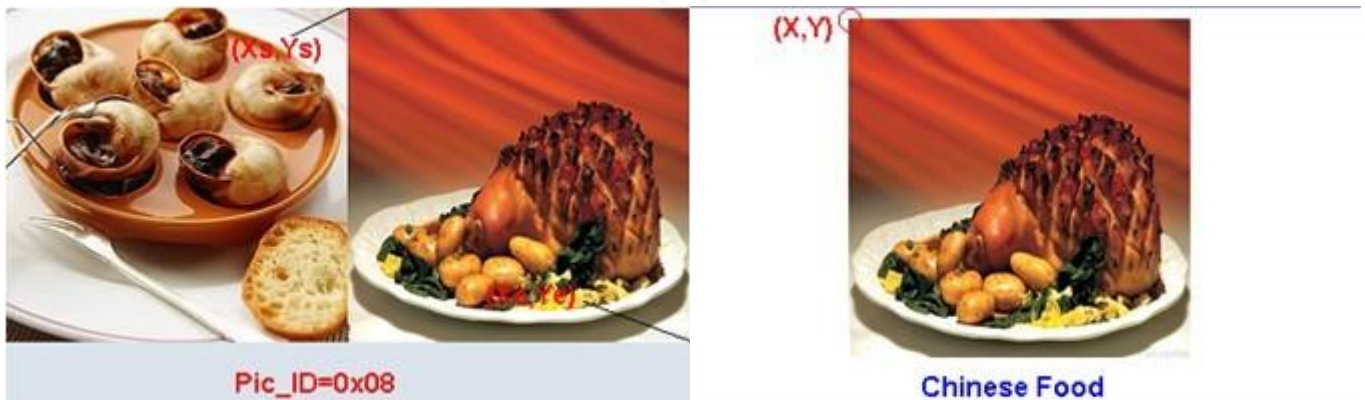
Rx: None

- <Pic_ID>: index ID(set by command 0xE2) of icons saved in HMI Flash Memory
- <Xs> <Ys> <Xe> <Ye>: (Xs, Ys): the top left coordinate of cutting area in original icon; (Xe,Ye): the bottom right coordinate of cutting area.
- <X> <Y>: (X,Y): the top left coordinate of cut icon in current screen

e.g.:

AA 71 08 01 90 00 00 03 1F 01 90 00 C8 00 14 CC 33 C3 3C

to cut 8th icon's area selected by (400,0) and (799,400) and display at(200,200) in current screen as follows:



3.13.4 User-defined Icon Display (0x99)

Tx: AA 99 <X0,Y0,Icon_ID0>.....<Xn,Yn,Icon_IDn> CC 33 C3 3C

Rx: None

- <Xn,Yn>: the display position of user icon
- <Icon_IDn>: index ID of user icon, 2 bytes

User icon need pre-define in 0x1D configuration file. Details in 3.26.4 section

3.13.5 Save current screen image in HMI (0xE2)

Tx: AA E2 <Pic_ID> CC 33 C3 3C

Rx: None

- <Pic_ID>: image index ID to be saved

3.13.6 Save current screen image Area to Temporary Buffer (0xE9)

Tx: AA E9 Xs Ys Xe Ye CC 33 C3 3C

Rx: None

- Xs Ys Xe Ye: coordinate of the upper left corner and the lower right corner.

3.13.7 Restore screen image Area Saved in Temporary Buffer (0x7F)

Tx: AA 7F Xs Ys Xe Ye CC 33 C3 3C

Rx: None

- Xs Ys Xe Ye: coordinate of the upper left corner and the lower right corner of the restored display area.

3.14 Backlight Brightness Control (0x5E, 0x5F)

3.14.1 Backlight close (0x5E)

Tx: AA 5E CC 33 C3 3C

Rx: None

3.14.2 Set touchpad(keyed) backlight mode (0x5E)

Tx: AA 5E 55 AA 5A A5 <V_ON> <V_OFF> <ON_TIME> CC 33 C3 3C

Rx: None

- <V_ON>: backlight turns on when hit the touchpad(or keyboard)
- <V_OFF>: backlight turns off when no hitting last for a while
- <ON_TIME>: the time for backlight to hold on; unit is 0.5 second.

The 1st hitting only turns on the backlight when backlight closed.

The touchpad(keyed) backlight function need command 0xE0 to initial.

3.14.3 Turn the backlight to maxim brightness (0x5F)

Tx: AA 5F CC 33 C3 3C

Rx: None

3.14.4 Adjust backlight brightness (0x5F)

Tx: AA 5F <PWM_T> CC 33 C3 3C

Rx: None

- <PWM_T>: backlight brightness value controlled by PWM, ranging from 0x00 to 0x3F; 0x00 means closed while 0x3F equals to maxim brightness.

For backlight mode CCFI in HMI, backlight brightness cannot be controlled by PWN; it is open/off way.

3.15 Touch Panel Operations (0x72, 0x73, 0x78, 0x79, 0xE4)

3.15.1 Touched position automatically upload(0x72, 0x73)

When hit the touch panel, HMI can automatically upload coordinate of touched position as following format

Tx: None

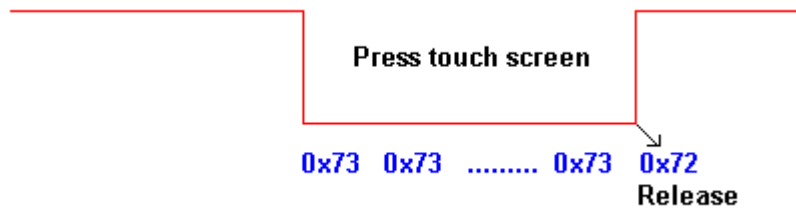
Rx: AA 73 <X> <Y> CC 33 C3 3C

to upload when hit the touch panel for one or more times(set by command 0xE0)

AA 72 <X> <Y> CC 33 C3 3C

to upload when hit the touch panel for only once(set by command 0xE0)

- <X> <Y>: ordinates of touched position, corresponding to screen resolution



If user use the touchpad function(set by command 0xE0) and enable the function that upload nothing when hit invalid area, then hitting the touch panel will not upload command 0x72 and 0x73.

3.15.2 Automatically upload key code(0x78, 0x79)

if user enable touchpad or keyed function(set by command 0xE0) and key code back transmission, then when hit the valid area or keys, HMI will upload user pre-defined key code of 2 bytes(defined in 0x1E,0x1B configuration file.)

Tx: None

Rx: AA 78 <Touch_Code> CC 33 C3 3C corresponding;

Note: 0x79 is corresponding to 0x73(pressed the touch panel)

0x78 is corresponding to 0x72(raised the touch panel)

3.15.3 Touch panel calibration mode (0xE4)

Tx: AA E4 55 AA 5A A5 CC 33 C3 3C

after sending commands, as instructions on screen, hit the top left, top right and bottom left of screen respectively.

HMI will upload following commands when calibration finished

Rx: AA E4 4F 4B CC 33 C3 3C

DWIN's unique drift compensation technique can guarantee that our product only need calibrate once after equipped in product life circle unless users re-equipped the touch panel

3.16 Work Mode Configuration (0xE0)

Tx: AA E0 55 AA 5A A5 <TFT_ID> <Bode_Set> <Para1> <Para2> CC 33 C3 3C

Rx: AA E0 4F 4B <TFT_ID> <Bode_Set> <Para1> <Para2> CC 33 C3 3C

- <TFT_ID>: to set TFT panel configurations (**V5.3 or later is not open to user, write 0x00**):

- <Bode_Set>: <Bode_Set>: to set the serial communication baud rate:

Bode_set	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07
Baud rate	1200	2400	4800	9600	19200	38400	57600	115200
As the following baud rate is not supported by the PC port, please set carefully								
Bode_set	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
Baud rate	28800	76800	62500	125000	250000	230400	345600	691200

- <Para1>: to set touch panel and keyboard work mode, described as table blow

Para1	Bit description
.7	0= to upload command 0x72 when release touch panel 1= not to upload command 0x72 when release touch panel

.6	Upload coordinate: 0= to upload command 0x73 every 100ms until release touch panel 1= to upload command 0x73 only once when press touch panel Touch control mode, and Para1.0=1: 0= to upload command 0x79 every 100ms until release touch panel 1= to upload command 0x79 only once when press touch panel
.5	0= HMI will not switch to touchpad interface after pressing touch panel. 1= HMI will switch to touchpad interface as configured file 0x1E described after pressing touch panel.
.4	0= backlight will not be controlled by touch panel or keyboard 1= backlight will be controlled by touch panel or keyboard; meanwhile, users can control it through command 0x5E/0x5F.
.3	0=Buzzer switched on for Touch screen and keyboard 1=Buzzer switched off for Touch screen and keyboard
.2	0= display in 0° 1=display in 90°
.1	0= In touch-control mode, buzzer is switched on all along. 1= In touch-control mode, buzzer only ring once when the active position is touched. When Para1. 1=1, Para1.3=1 should be set accordingly in the same time.
.0	In the touch-control mode: 0= No 0x79 command is being uploaded; 1=Upload 0x79.

- <Para2>: to set display mode, described as table blow (Formerly V6.0 Version is not available to customers, so this byte should to be omitted.)

Para1	Bit description
.7	0=refresh mode of previous Version V5.0 (automatic high-speed refresh, 0x71+0x98 result flicker.) 1= automatic refresh display every 200ms, auto delay refresh time after operate command, ensure continuous instructions can be displayed synchronized.
.6	Upload coordinate: 0= upload coordinates change with Para1.2 (deflection 90°) 1= upload coordinates will not change with Para1.2
.5	0= display deflection 180°(reverse angle display) 1= normal angle display.
.4	0= display text(0x53, 0x54, 0x55, 0x6E, 0x6F, 0x98), auto recovery picture background, ignore text background color. 1= not auto recovery picture background when display text.
.3	1, reservation
.2	1, reservation
.1	1, reservation
.0	1, reservation

Configuration parameters are not saved when power off. Re-power will recover the parameters pre-set by user through DWIN debugging tools.

3.17 Animate Control (0x9A)

It is used to convenient for users to save regular instructions (e.g. use 0x71 to display animation) in HMI, without user intervention when using, and save codes.

3.17.1 Open timed-loop executive instruction function

Tx: AA 9A Pack_ID CC 33 C3 3C

Rx: None

- <Pack_ID>: instructions ID, 0x00-0x0F, 8KB per sets, including up to 64 HMI instructions, 128KB per instruction.

HMI can only operate 1 group instruction, which is defined in 0x1C configuration file, including up to 16 instruction group.

3.17.2 Close timed-loop executive instruction function

Tx: AA 9A FF CC 33 C3 3C

Rx: None

3.18 Temporary Buffer Operation (0xC0, 0xC1, 0xC2)

M100 can't support this command. The temporary buffer content can be changed by 0x52.

3.18.1 Write temporary buffer (0xC0)

Tx: AA C0 <Address> <Word0.....Wordn> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer--RAM; totally 40k word, ranging from 0x0000 to 0x9FFF.
- <Word0.....Wordn>: data to be written.

3.18.2 Read temporary buffer (0xC2)

Tx: AA C2 <Address> <Data_Length> CC 33 C3 3C

Rx: AA C2 <Data_Pack> CC 33 C3 3C

- <Address>: the starting address, word address, in temporary buffer--RAM; totally 40k word, ranging from 0x0000 to 0x9FFF.
- <Data_Length>: word length, 2 bytes.
- <Data_Pack>: data to be read.

3.18.3 Pixel setting in temporary buffer (0xC101)

Tx: AA C1 01 <Address> <Pixel_Number> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer--RAM; totally 40k word, ranging from 0x0000 to 0x9FFF.
 - <Pixel_Number>: numbers of pixels with maximum 13653. Every pixel takes up 3 word unit.
- pixel data format defined in temporary buffer: (X,Y,Color).

3.18.4 Connecting lines using data in temporary buffer (0xC102)

Tx: AA C1 02 <Address> <Line_Number> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer--RAM; totally 40k word, ranging from 0x0000 to 0x9FFF.
- <Line_Number>: the number of lines with maximum 8191. Every line takes up 5 word unit.

line data format defined in temporary buffer: (Xs, Ys, Xe, Ye, Color).

3.18.5 Display broken line graph using data in temporary buffer (0xC103)

Tx: AA C1 03 <Address> <x> <y> <Line_Number> <D_x> <Dis_x> <K_y> <Color> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer(RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <x>: the starting coordinate in X-axis
- <y>: the zero position(the lowest point) in Y-axis; actual position equals to y-Ly
- <Line_Number>: the number of lines with maximum 40960, ranging from 0x000 to 0x9FFF. Every line takes up 1 word unit.
- <D_X>: pixels' distance in read buffer, ranging from 0x01 to 0xFF, i.e. Address=Address+D_x
- <Dis_X>: the increment in X-axis, ranging from 0x01 to 0x0F, i.e. x=x+Dis_x
- <K_y>: amplification factor of Y-axis, ranging from 0x00 to 0xFF; 1/16 is the unit, i.e. K_y=32 means amplify Y-axis twice.
- <Color>: the color of lines without any variation in system palette.

the line data format in temporary buffer is defined as Ly, 2 bytes, which is the height of pixel.

3.18.6 Fast display broken line graph using data in temporary buffer (0xC104)

Tx: AA C1 04 <Adr1> <x> <y> <Line_Number> <D_x> <Dis_x> <Color1> <Addr0> <Color0> CC 33 C3 3C

Rx: None

- <x>: the starting coordinate in X-axis
- <y>: the zero position(the lowest point) in Y-axis; actual pixel position equals to y-Ly.
- <Line_Number>: number of lines with maximum 40960. Every line takes up 1 word unit.
- <D_x> fixed write 0x01
- <Dis_X>: the increment in X-axis, ranging from 0x01 to 0x0F; i.e. x=x+Dis_x after connecting.
- <Addr0><Addr1>: the starting address(word address) in temporary buffer(RAM), totally 40k word, ranging from 0x0000 to 0x9FFF.
- <Color0> <Color1>: the color of lines without any variation in system palette.

the line data format in temporary buffer is defined as Ly, 2 bytes, which is the height of pixel.

this command is similar to command 0xC103 except:

A.the pixels' distance in read buffer is fixed to 1.

b.before connecting pixel one, erase former pixel zero first, which realizes **flicker-free** display.

When using baud rate 115200bps, this command enable broken line graph display speed up to maximum 5500 pixel per second

3.18.7 Display broken line graph in scale using data in temporary buffer (0xC105)

Tx: AA C1 05 <Address> <x> <y> <Line_Number> <D_x> <Dis_x> <M_y> <D_y> <Color> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer(RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.

- <x>: the starting coordinate in X-axis.
- <y>: the zero position(the lowest point) in Y-axis; actual pixel position equals to y-Ly.
- <Line_Number>: number of lines with maximum 40960, ranging from 0x0000 to 0x9FFF. Every line takes up 1 word unit.
- <D_x>: pixels' distance in read buffer, ranging from 0x01 to 0xFF;i.e.Address=Address+D_x after connecting.
- <Dis_x>: the increment in X-axis, ranging from 0x01 to 0x0F;i.e.x=x+Dis_x after connecting.
- <M_y> <D_y>: amplification factor of Y-axis, ranging from 0x00 to 0xFF; and height equals to $Y \times M_y / D_y$, e.g. $M_y=4$, $D_y=2$ means amplify Y-axis twice.
- <Color>: color of lines without any variation in system palette.

The line data format in temporary buffer is defined as Ly, 2 bytes, which is the height of pixel.

3.18.8 Display window limited B-direction curve in scale using data in temporary buffer (0xC106)

Tx: AA C1 06 <Address> <x> <y> <Line_Number> <D_x> <Dis_x> <M_y> <D_y> <Color> <Ymin> <Ymax> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer(RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <x>: the starting coordinate in X-axis.
- <y>: the zero position (the lowest point) in Y-axis; actual pixel position equals to y-Ly or y+Ly.
- <Line_Number>: number of lines with maximum 40960, ranging from 0x0000 to 0x9FFF. Every line takes up 1 word unit.
- <D_x>: pixels' distance in read buffer, ranging from 0x01 to 0xFF;i.e.Address=Address+D_x after connecting.
- <Dis_x>: the increment in X-axis, ranging from 0x01 to 0x0F;i.e.x=x+Dis_x after connecting.
- <M_y> <D_y>: amplification factor of Y-axis, ranging from 0x00 to 0xFF; and height equals to $Y \times M_y / D_y$, e.g. $M_y=4$, $D_y=2$ means amplify Y-axis twice.
- <Color>: color of lines without any variation in system palette.
- <Ymin> : Y-axis coordinate of the lower limit
- <Ymax>: Y-axis coordinate of the upper limit

The line data format in temporary buffer is defined as Ly, 2 bytes:

Ly.15 is the connecting direction, 0=positive direction, 1=negative direction. Ly=0x8010 means negative direction height is 16.

Ly.14- Ly.0 is connecting height.

3.18.9 Using temporary buffer as set-point buffer (0xC107)

Clear set-point buffer:

Tx: AA C1 07 00 <Address> <X_Len> <Y_len> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer(RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <X_Len>: X-direction width of set-point buffer screen window (pixel number), 0x0000-0xFFFFF.

- <Y_len>: Y-direction width of set-point buffer screen window (pixel number), 0x0000-0x0FFF.

Note:

Set-point buffer RAM=(X*Y)/16, e.g. build a 64*64 set-point buffer from temporary buffer address 0x0000, so, take the temporary buffer address space is 0x0000-0x00FF.

Build point in set-point buffer:

Tx: AA C1 07 01 <Address> <X_Len> <Y_len> <Xs,Ys> <Color> <Mode> <(X0,Y0)···(Xi,Yi)> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer (RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <X_Len>: X-direction width of set-point buffer screen window (pixel number), 0x0000-0x0FFFF.
- <Y_len>: Y-direction width of set-point buffer screen window (pixel number), 0x0000-0x0FFF.
- <Xs,Ys>: coordinate position of screen window
- <Color>: pixel color, not affect 0x40.
- <(X0,Y0)···(Xi,Yi)>: position of set point, no cross-border points. Display position X=Xs+Xi Y=Ys+Yi
- <Mode>: mode of set point.

0x00=delete point in set-point buffer, 0x10=delete point in set-point buffer and current page.

0x01=set point in set-point buffer, 0x11=set point in set-point buffer and current page.

Recover set-point buffer to current page:

Tx: AA C1 07 02 <Address> <X_Len> <Y_len> <Xs,Ys> <Color> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer (RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <X_Len>: X-direction width of set-point buffer screen window (pixel number), 0x0000-0x0FFFF.
- <Y_len>: Y-direction width of set-point buffer screen window (pixel number), 0x0000-0x0FFF.
- <Xs,Ys>: coordinate position of screen window
- <Color>: pixel color, the same as current page displayed.

Only set point in set-point buffer (mode 01), must clear set-point buffer before recover to current page.

This instruction is used for solid icon operation flexibility, as move cross cursor.

3.18.10 Using temporary buffer to display multi-parameters (0xC108)

Tx: AA C1 08 00 <Address> <Parameter_N> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer(RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <Parameter_N>: number of multi-parameters, 0x01-0xFF

Note: Display as right-aligned.

Temporary buffer data format defined as follows:

Temporary buffer relative address	definition	description
0x00	mode	Mode.15-Mode.12 display the maximum number of integer digits Mode.11-Mode.8 display the number of decimal places, 0=no decimal places Mode.7 =0 invalid integer bit zero is not displayed =1 invalid integer bit zero is displayed e.g.:Mode.15-Mode.12=3, Mode.7=1, 0.3 will be displayed as 000.3. Mode.6-Mode.4 no definition, write 0 Mode.3-Mode.0 size of fonts (based on Lib_ID=0 , default ASCII font) =0 8*8; =1 8*12; =2 6*12; =3 8*16; =4 12*24; =5 16*32; =6 20*40; =7 24*48; =8 28*56; =9 32*64; =0x0A-0x0F no definition
0x01	X	X coordinate of starting display position
0x02	Y	Y coordinate of starting display position
0x03	F_Color	Fore-ground color of display parameter Will not change the color palette.
0x04	B_Color	Back-ground color of display parameter Will not change the color palette.
0x05	Parameter	Parameter data, 4 bytes signed integer, Parameter.31 is the sign bit.

e.g. parameter set:

Displayed parameter	parameter	Mode(16*32 font, mode 6.3, invalid integer bit zero is not displayed)
12345	0x00 00 30 39	0x6005
123.450	0x00 01 E2 3A	0x6305
-123.450	0xFF FE 1D C6	0x6305

AA C0 00 00 63 05 00 0A 00 0A F8 00 00 1F 00 01 E2 3A CC 33 C3 3C

AA C1 08 00 00 01 CC 33 C3 3C

Display 123.450 at (10, 10)

3.18.11 Using temporary buffer to buffer instruction, achieve synchronous display (0xC110)

Tx: AA C1 10 <Address> <Frame_Number> CC 33 C3 3C

Rx: None

- <Address>: the starting address, word address, in temporary buffer(RAM); totally 40k word, ranging from 0x0000 to 0x9FFF.
- <Frame_Number>: number of continuous display instruction frames, 0x01-0xFF

Definition of temporary buffer instruction frames:

First address	definition	Frame structure
<Address>	The first instruction frame	frame length + instruction + data display picture 0: 03 70 00 00 the number of bytes must be even number.
<Address>+0x80	The second instruction frame	
.....		
<Address>+0x80×(K-1)	The Kth instruction frame	

e.g. write: AA C0 00 00 09 55 00 00 00 00 30 31 32 33 CC 33 C3 3C

AA C0 00 80 09 53 00 64 00 64 34 35 36 37 CC 33 C3 3C

Synchronous display: AA C1 10 00 00 02 CC 33 C3 3C

Note:

Since temporary buffer store data as word, one instruction frame address is 0x80, but the actual storage space length is 256 bytes.

This instruction is mainly used to solve the problem that many parameters need to refresh for display, in order to avoid delays caused by traffic (particularly low baud rate) result in refresh asynchronous.

3.19 Keyboard Operation(0x71, 0xE5)

3.19.1 Key code upload (0x71)

Tx: None

Rx: AA 71 <K_code> CC 33 C3 3C

- <K_code>: user pre-defined key codes, uploaded with speed--5 keys per second, when HMI's keyboard interface detects the pressing action

3.19.2 Key code setting(0xE5)

Tx: AA E5 55 AA 5A A5 <K_Code0.....K_Code63> CC 33 C3 3C

Rx: None

- <K_Code0.....K_Code63>: key codes to be set. The number is fixed at 64. Only 16 key codes are valid for 4x4 keyboard interface.

3.20 Read and Write in User's Memory (0x90, 0x91)

The physical media of database is NAND Flash with 100 thousand erasable times and 10 years life circle.

3.20.1 Write in random data memory (0x90 64KB)

Tx: AA 90 55 AA 5A A5 01 DE <Address> <Data0.....Data1> CC 33 C3 3C

Rx: AA 90 4F 4B CC 33 C3 3C

- <Address>: the starting address to write in data memory, 2 bytes, ranging from 0x0000 to 0xFFFF.
- <Data0.....Data1>: data series to be written.

3.20.2 Write in sequential data memory (0x90 30MB)

Tx: AA 90 55 AA 5A A5 <Address> <Data0.....Data1> CC 33 C3 3C

Rx: AA 90 4F 4B CC 33 C3 3C

- <Address>: the starting address to write in data memory, 4 bytes, ranging from 0x00000000 to 0x01DDFFFF.
- <Data0.....Data*i*>: data series to be written.

Different from random data memory, sequential data memory can only write sequentially, i.e. It cannot write randomly.

The whole sequential memory is divided into 239 128KB data page. When meet with the head of page (address=**02 00 00), it will erase current page with no data backup before erasing and no effect on other pages. It is recommended to do some continuous data of large amount storage such as paperless recording and audio recording.***

3.20.3 Read data memory (0x91)

Tx: AA 91 <Address> <Length> CC 33 C3 3C

Rx: AA 91 <Address> <Length> <Data0.....Data*i*> CC 33 C3 3C

- <Address>: the starting address to write in data memory, 4 bytes, ranging from 0x00000000 to 0x01DDFFFF.
- <Length>: the length of data memory to read, 2 bytes. Most read 64KB once.
- <Data0.....Data*i*>: data series to read out.

3.21 Download Character Library and Configuration Files (0xF2)

Tx: AA F2 F2 F2 5A A5 <Lib_ID> CC 33 C3 3C

Rx: Please Tx Text_Lib !

then users can download corresponding character library

when character library saving finished, HMI will response:

*****One Text_Lib Saved OK !*****

- <Lib_ID>: the storage position of character library. Totally 60 character libraries exist, in which 0x00-0x1F are 32 small character libraries(configured files) and 0x30-0x3B are 28 1MB large libraries.

Please do not modify character library located at Lib_ID=0x00, 0x20, 0x21, 0x22 and 0x23 unless users need to design their own Chinese character library; otherwise, command 0x53, 0x54, 0x55, 0x6E and 0x6F will wrongly display.

3.22 Simple Algorithm Support (0xB0)

3.22.1 Pinyin input method (0xB001、0xB004)

Tx: AA B0 01 <PY_Code> CC 33 C3 3C or AA B0 04 <PY_Code> CC 33 C3 3C

Rx: AA B0 01 <HZ_Num> <HZ_String> CC 33 C3 3C or AA B0 04 <HZ_Num> <HZ_String> CC 33 C3 3C

- <PY_Code>: Chinese pinyin, capital display with maximum 6 bytes
- <HZ_Num>: the number of Chinese character of this pinyin while 0 interprets wrong pinyin. B001 instruction is 1 byte, B004 instruction is 2 bytes.
- <HZ_String> : all Chinese characters of this pinyin using internal code
0xB001 is in accordance with GB2312-80 first class fonts. 0xB004 is in accordance with GBK expansion font.

3.22.2 MAC calculate (0xB002)

Tx: AA B0 02 <A,B,C,D> CC 33 C3 3C

Rx: AA B0 02 <E,F> CC 33 C3 3C

- <A,B,C,D>: to calculate $(A \times B + C) / D$ while A,B,C and D are all unsigned integral of 2 bytes
- <E,F>: results while E is 4 bytes quotient and F is 2 bytes remainder

3.22.3 Array sort (0xB003)

Tx: AA B0 03 <Pack0> CC 33 C3 3C

Rx: AA B0 03 <Pack1> CC 33 C3 3C

- <Pack0>: array need to be sorting, 2 bytes
- <Pack1>: array after sorting, 2 bytes, ascending sort.

3.23 Buzzer Control (0x79)

Tx: AA 79 CC 33 C3 3C

Rx: None

buzzer beepers 100ms.

3.24 Clock(RTC) Display and Read off (0x9B, 0xE7)

3.24.1 Close clock display

Tx: AA 9B 00 CC 33 C3 3C

Rx: None

3.24.2 Open clock display

Tx: AA 9B FF <RTC_Mode> <Text_Mode> <Color> <X> <Y> CC 33 C3 3C

Rx: None

- <RTC_Mode>: clock display mode setting
 - 0x00: HH:MM:SS
 - 0x01: 20YY-MM-DD HH:MM:SS
- <Text_Mode>: clock display character setting
 - 0x00: 8*8
 - 0x01: 6*12
 - 0x02: 8*16
 - 0x03: 12*24
 - 0x04: 16*32
 - 0x05: 20*40
 - 0x06: 24*48
 - 0x07: 28*56
- <Color>: character's color
- <X> <Y> : clock's position

3.24.3 Clock setting

Tx: AA E7 55 AA 5A A5 <YY:MM:DD:HH:MM:SS> CC 33 C3 3C

Rx: None

- <YY:MM:DD:HH:MM:SS>: time format to be set as year:month:day:hour:minute:second, represented by BCD code.

e.g.

AA E7 55 AA 5A A5 08 11 28 12 57 00 CC 33 C3 3C

to set current time to 12:57:00, Nov. 28th, 2008.

3.24.4 Read off current time (the Gregorian calendar)

Tx: AA 9B 5A CC 33 C3 3C

Rx: AA 9B 5A <YY:MM:DD:WW:HH:MM:SS> CC 33 C3 3C

- <YY:MM:DD:WW:HH:MM:SS>: current clock data

08 12 25 04 09 58 00, for example, represents 09:58:00, Thursday, Dec. 25th, 2008.

if HMI do not support clock function, it will return unknown results.

3.24.5 Read off current time (the Luna calendar)

Tx: AA 9B 5B CC 33 C3 3C

Rx: AA 9B 5B <YY:MM:DD:Chinese Zodiac:Heavenly stem:earthly branch> CC 33 C3 3C

- <YY:MM:DD:Chinese Zodiac:Heavenly stem:earthly branch>: current Luna calendar, including Chinese Zodiac, Heavenly stem, earthly branch, represented by internal code.

09 02 03 "niu yi chou", for example, represents current date Feb. 3rd, 2009, in which 2009 is "niu" year and chronogram is "yichou"

3.25 Play Music(0x30, 0x32, 0x33)

This instruction needs corresponding hardware (DWIN DMA5601 stereo recording/playing module) support.

3.25.1 Play specified address music (0x30)

Tx: AA 30 <Start_SEG> <SEG_Number> <Play_Time> CC 33 C3 3C

Rx: AA 3F 4F 4B start to play

AA 3F 4F 4B end playing

- <Start_SEG>: starting address of music segments to be played, ranging from 0x00 to 0xFF;
- <SEG_Number>: numbers of music segments to be played, ranging from 0x00 to 0xFF;
- <Play_Time>: repeated times, ranging from 0x00 to 0xFF.

3.25.2 Volume adjustment (0x32)

Tx: AA 32 <Volume_L> <Volume_R> 00 CC 33 C3 3C

Rx: AA 3F 4F 4B

- <Volume_L>: volume of left channel, ranging from 0x00 to 0x3F
- <Volume_R>: volume of right channel

3.25.3 Stop playing (0x33)

Tx: AA 33 55 AA 5A CC 33 C3 3C

Rx: AA 3F 4F 4B

3. 26 Configuration File (Touchpad interface, Keyed interface, Animate, Icon library)

DWIN HMI can realize simplified Operating System function through following configuration files, which can reduce considerable codes.

3.26.1 Touchpad interface switch (0x1E, 0x1A configuration file)

DWIN HMI, touchpad included, can download configuration file beforehand, which can reduce the code scale; and set touchpad interface to auto-switch mode as to realize user intervention-free function.

Developing process:

Step One: design a user interface of identical physical resolution of HMI and download it to HMI terminal.

DMT64480S057_11WT, for example, means set interface's resolution as 640x640.

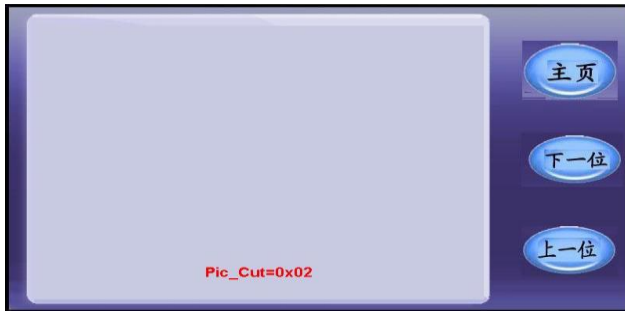
Step Two: generate configuration file

Configuration file is a binary file with maximum 8192 touchpad commands. Every command takes up 16 bytes, defined as follows:

Starting address	Length (Byte)	Definition	Description
0x00	2	Pic_Now	serial number of the picture display on current screen if high byte of Pic_Now is 0xFF, it means touchpad command end.
0x02	4	X _s ,Y _s	left top coordinate of effective touchpad area
0x06	4	X _e ,Y _e	right bottom coordinate of effective touchpad area
0x0A	2	Pic_Next	switch to next picture serial number when press the effective area. if high byte of Pic_Next is 0xFF, it means no switch allowed.
0x0C	2	Pic_Cut	serial number of touchpad animate picture if high byte of Pic_Cut is 0xFF, it means no more touchpad animate pictures.
0x0E	2	Touch_Code	upload touchpad code when press the effective area(act as a message to trigger users' soft) if high byte of Touch_Code is 0xFF, it means no more uploading. if high byte of Touch_Code is 0xFE, it means the uploading data serial indexes to 0x1A configuration file where low byte of Touch_Code means the index ID. In the 0x1A configuration file, every index length is fixed to 256 bytes with the first one is the effective length.

Table 3-26-1 definitions of touchpad interface configuration file

The process of generating configuration files is the process in which users arrange the interface-switch procedures, which graphic designer usually do. Configuration files can be compiled with Ultra Edit or some kinds of compile system, such as C51 and ASM51. The example showed in figure 3-26-1 represents how to compile a configuration file with the classic ASM51 compiler in DOS system, which is also the most familiar with engineers.



;DMT80480S070_02WT Example of HMI touchpad interface switch

;when press "the next" button in 19th picture, HMI will switch to 20th picture and upload key code(0x0031)

;to utilize the effect of button in 2nd picture after pressing ;character in blue is the coordinates of effective area

ORG 0000H

DW 19,654,195,792,292,20,2,1 ;one touchpad command

DW OFFFHH ;all commands definition finished.

END

Example above only compiles one touchpad key and one interface switch. More interface switches need users' more commands.

compile finished configuration file--*.ASM to HEX file, using compiler ASM51; then transform HEX file to BIN file which is the right configuration file.

Step 3: download the configuration file to HMI terminal.

using download command in 0xF2 library, download the generated binary configuration file to 0x1E character library in HMI terminal.

Step 4: set HMI to touchpad interface auto-switch mode

set the 5th bit of Para1--Para1.5,0x20 to 1, using command 0xE0. When users press the screen, HMI terminal will automatically switch touchpad interface and upload user pre-defined key code instead of uploading coordinates.

Step 5: for testing whether interface switch works well, step 2 and 3 may need repeat times.

Definition of configuration file 0x1A

configuration file 0x1A consists of maximum 256 commands with a maximum of 255 bytes per command. It occupies 256 bytes in memory. Commands definitions are represented as follows:

Starting address	Data length (Byte)	Definition	Description
0x00	1	Length_Command	The length of this command. If Length_Command=0x00, it represents that this command is a combination of commands--batch command.
0x01	indefinite	command	the command to be send first if Length_command is not 0x00 or 0x01.

		Command pointer (Cmd_EN,Cmd_ID, Tx_Delay)	if Length_Command = 0x00, what behind is the pointer of combination command. Each pointer takes up 3 bytes with a maximum 85 command-pointers, combination command. Pointer definitions, as follows: Cmd_EN: 0x00 represents the end of combination command sending; and others represent command sent. Cmd_ID: ID of commands to be sent, ranging from 0x00 to 0xFF, correspond to the non-combination command numbers in 0x1A; and it do not support nested of combination command. Tx_Delay: time intervals of each sending commands; unit is 0.1 second. when sending combination commands, HMI do not respond to any users and peripherals.
--	--	---	--

Designing touchpad interface with configuration files can not only largely reduce code amount, it can reduce the difficulty of development ;more important, we hope this can change our regular minds about products designing and market development, as follows:

- **Completely separate products' algorithm and interface designing;** algorithm designing is the core competitiveness of a company, thus the precious research resources should not waste on large amount copious interface designing.
- **Products research can process in parallel;** not only the interface designing and algorithm can process in parallel, interface designing itself can as well by several graphic designer; because touchpad key code has a function of "trigger message", engineers can design and debug algorithm in parallel.
- **Improve products' reliability;** all users' programmers lie in the same parallel level in principal, thus function modules are independent and easy for testing.
- **Easy upgrade;** when product is stable, product's upgrade is basically the upgrade of interface while algorithm needs little.
- Using the configuration file method, it is easy to realize **interface customization**--more skins, because users only need to insert different interface in configuration file and the key code uploaded is the same.
- Largely reduce the developing time of a new product, which enhances market competitiveness. Following is an example that DWIN develops a product for customers:

a.Using 1 to 3 working days to confirm basic demands with customers.

b.Using 1 to 2 working days, our graphic designer can provide the product's interface based on standard HMI; meanwhile, our structure engineers are ready to provide product's three-dimensional rendering for customers to confirm.

c. After confirmation, our hardware engineers start to design PCB, structure engineers hand in the design to factories in Shenzhen for fast molding and graphic designers begin to modify interface as to submit software engineer the process documents.

d.One week later, we can provide customers the full-fledged prototype for inspection.

3.26.2 Keyed interface switch (configuration file 0x1B)

Similar to touchpad interface, keyed interface is also triggered through buttons. The configuration files are saving in file 0x1B. Configured file 0x1B consists of maximum 5957 keyed commands and 22 bytes of each keyed command. See definitions as table 3-26-2:

Starting address	Length (Byte)	Definition	Description
0x00	2	Pic_Now	The number of picture display on current screen. If high byte of Pic_Now is 0xFF, it means the end of keyed command.
0x02	2	0x00:Key_Code	Key code
0x04	2	Pic_Next	Switch to the number of picture of the next interface. If high byte of Pic_Next is 0xFF, it means that no switch allowed. If high byte of Pic_Next is 0xFE, it means that process the picture switch.
0x06	14	Pic_Cut,Xs,Ys,Xe,Ye,X,Y	High byte of Pic_next=0xFE: definitions of picture area and cut area. High byte of Pic_next=others: no definitions.
0x14	2	Touch_Code	Code to be upload after pressing, as the message to trigger user software; if high byte of Touch_Code is 0xFF, it means that no code upload. If high byte of Touch_Code is 0xFE, it means that data uploaded has indexed configured file 0x1A and low byte of Touch_Code is the index ID. In configured file 0x1A,each length of index is fixed to 256 bytes and the first byte represents the effective length.

Table 3-26-2 Definitions in configuration file of keyed interface

3.26.3 Animate auto-play(configuration file 0x1C)

Composed of 16 groups instruction set, each group is 8KB, including up to 64 HMI instruction, each instruction occupies 128 bytes storage space.

HMI instruction definition is as following:

First address	Definition	Description
0x00	Command_Delay	Delay time, unit is 8ms, 0x00 means no delay
0x01	Command_Length	Length of current instruction 0x00: this command is invalid Other value: instruction length (calculate from 0x02)
0x02-0x7F	Instruction	HMI instruction without frame head and frame end.

e.g. three pictures switching, the configuration file is as follows:

```

ORG 0000H                ; first group, PADIO=00H
DB 125,2,70H,00H        ; display picture 0
ORG 0080H
DB 125,2,70H,01H        ; display picture 1
ORG 0100H
DB 125,2,70H,02H        ; display picture 2
ORG 2000H                ; second group, PADIO=01H
DB 125,2,70H,03H        ; display picture 3
ORG 2080H
DB 125,2,70H,04H        ; display picture 4

```

Use instruction 9A: AA 9A 00 CC 33 C3 3C ;picture 0,1,2 switching
AA 9A 01 CC 33 C3 3C ;picture 3,4 switching

The first address of each instruction is : 0000H、0080H、0100H...

The first address of each instruction set is : 0000H、2000H、4000H...

3.26.4 Icon display (configuration file 0x1D)

DWIN HMI has one picture cut command 0x71, which enables users cut a area of picture saved in HMI and paste it to the specified location in current interface.(details can be found in section 3.13.2) However, this is not convenient for customers. Thus we add command 0x99 and Icon definition library--file 0x1D for customers to call icon display through text method.

Format of command 0x99, described in table 3-26-3:

command	Data	Description
0x99	(x,y,Icon_ID) ₀ +.....+(x,y,Icon_ID) _n	(x,y) is the top left coordinate of icon; Icon_ID is the index ID in Icon library.

Table 3-26-3 format of command 0x99

Icon library configuration file consists of maximum 13107 binary files(correspond to Icon_ID=0x0000-0x3332). Each Icon definition takes up 10 bytes as follows:

Starting address	length (Byte)	Definition	Description
0x00	2	Pic_ID	Picture ID of the saved Icon
0x02	4	X _s ,Y _s	The top-left coordinate of icon area
0x06	4	X _e ,Y _e	The bottom-right coordinate of icon area

Table 3-26-4 definition of icon

e.g.

ORG 0000H

```
DW 10,0,0,100,100 ; Icon_ID is 0000H
DW 11,0,0,100,100 ; Icon_ID is 0001H
DW 12,0,0,100,100 ; Icon_ID is 0002H
```

Use instruction 99: AA 99 00 00 00 00 00 00 64 00 64 00 01 00 00 00 64 00 02 CC 33 C3 3C

Will display icon at position (0, 0)、(100, 100)、(0, 100)

For DWIN HMI, when received command 0x99, it will follow these steps:

- According to Icon_ID, it will take out Pic_ID, (X_s,Y_s), (X_e,Y_e) from position Icon_ID×10 in configured file 0x1D
- The same as command 0x71, it will cut picture: 0x71 Pic_ID, (X_s,Y_s), (X_e,Y_e) (X,Y)
- For the next icon display process.

The way of generating configuration file can reference section 3.26.1

To match up configuration file, using command 0x99 can easily solve these problems:

- For analog dial, we can make different scales into icons and **directly** call it according to the variables instead of looking-up table again.
- For **special characters**, even those UNICODE does not contain, we can call it **in the way of icon**, which avoids the complex character library generating.
- It is convenient to use **Windows elements** such as **cursor** and **mouse pointer**.

3. 27 HMI and Video Switching (0x7A)

For analog video:

Tx: AA 7A <Work_Mode> <Video_Mode> <Video_CH> CC 33 C3 3C

Rx: None

- <Work_Mode>: 0x00=HMI 0x01=Video
- <Video_Mode>: 0x00=PAL 0x01=NTSC
- <Video_CH>: 0x00=CVBS interface input video signal, 0x01=S terminal input video signal.

For digital video player based on SD card:

Tx: AA 7A <Work_Mode> <KEY_VALUE> CC 33 C3 3C

Rx: None

- <Work_Mode>: 0x00=HMI 0x01=Video
- <KEY_VALUE>: video play mode, definition of function key is as follows:

KEY_VALUE	Definition of function key
0x00	lift key
0x01	play/pause key press
0x02	volume on
0x03	volume off
0x04	menu operation: OK press
0x05	menu operation: BACK press
0x06	menu operation: FORWARD press
0x07	menu operation: EXIT press
0x08	menu operation: VOLUME UP press
0x09	menu operation: VOLUME DOWN press
0x0A	menu operation: CHINESE/ENGLISH SELECT press

3. 28 Force a full screen refresh one time (0XD0)

Tx: AA D0 CC 33 C3 3C

Rx: None

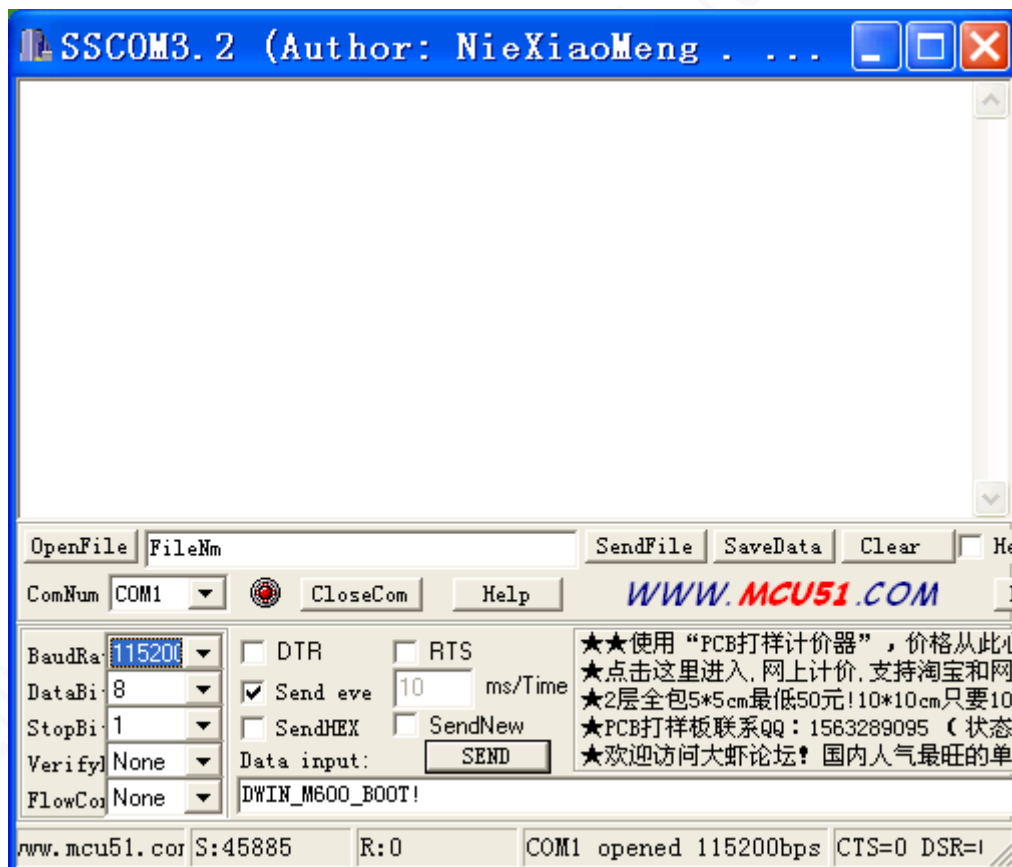
4 Update Method of HMI Software

Tools:

- One DC regulated power supply. Adjust the output voltage to suitable value;
- One serial line;
- One computer which has hardware serial port and serial debugging assistant--software SSCOM3.2.

Update Steps:

- a. Turn HMI power off and connect serial with computer's COM1.
- b. Open software SSCOM3.2 and click "Open File" to select HMI's program, such as M600_V40.BIN.
- c. Write "**DWIN_M600_BOOT!**" on send column and set **auto-send** interval to 10
- d. Hook on "**send new line**" and "**auto-send**", then turn HMI power on.
- e. The serial port will receive "Erase". If it does not receive it, check connection and whether HMI damaged.
- f. About 1 second later, it will receive "Please Tx File!", then hook off "auto-send" and click "send file".
- g. Wait another 3 to 10 seconds, when serial port receive "*****END*****", it means the end of downloading.
- h. Turn HMI **power off** and software updating finished.



Serial port interface before HMI power on

5 Revision History

Date	Revision History	Kernel Version	Document version
2008.12.01	First Publish.	— —	Ver1.0
2008.12.09	Revise the use method of configuration file and add instructions about keyed interface.	— —	Ver1.1
2008.12.11	Add several special character libraries in command 0x98	— —	Ver1.1
2008.12.15	Add command 0xC104 Revise command 0xC103 and add the function of amplifying Y-axis in scale.	— —	Ver1.1
2008.12.25	Unified instruction set of T and S series. Add command 0x9B, which has function of reading out clock.	— —	Ver1.2
2009.01.19	Add command 0x5E's function of touchpad or keyed controlling to backlight. Add command 0xE0's definition of PARA1.4, open backlight control.	— —	Ver1.3
2009.02.02	Add command 0x9B's function of Luna Calendar support.	— —	Ver1.3
2009.02.12	Add 0x1A configuration file which supports instructions to combination commands in touchpad operation.	— —	Ver1.3
2009.03.13	Added the instruction of 0x64(area filling).	— —	Ver1.4
2009.04.07	Added the instruction of 0x5704.	— —	Ver1.4
2009.06.29	Added the instruction of 0x9C, Added the instruction of 0xC106.	— —	Ver1.5
2009.07.16	Revised the instruction of 0x9A and the illustration of the configuration file of 0x1C. M600 achieve auto timing cycle implementation instruction, convenient for user design animation.	— —	Ver1.5
2009.07.23	Revised the definitions of the configuration file of 0x1A and 0x1E, so the user instruction extended to 1024.	— —	Ver1.5
2009.08.07	Added the instruction of 0xC110. Convenient for low baud rate synchronization display.	— —	Ver1.5
2009.08.08	Added instruction of 0x79、0xE0, added the function of controlling the buzzer by touching or software.	— —	Ver1.6
2009.09.03	Added the instruction of 0x7A to switch the HMI model and the Video model.	— —	Ver1.6
2009.09.17	Added the instruction of 0xC2 to read the data in the temporary buffer, support the function of RMA.	— —	Ver1.6
2009.12.27	Added the instruction of 0x7B to picture verification for downloading success	— —	Ver1.6
2010.01.04	Because of kernel modification, revised the <TFT_ID> product list in instruction of 00xE0 and revised 3 calibrating spots in touching calibration instruction (0xE4)	Ver4.5	Ver1.7
2010.02.21	Added the definition of C_Mode.5 in 0x98 instruction for supporting auto line warp in lengthways. For 128MB version HIM kernel, software needed to be re-download word stocks and pictures after above Ver5.0 updated.	Ver4.5	Ver1.7

For version 128MB HMI kernel, upgrade to V5.0 or later, need to re-download fonts			
2010.03.06	Added ID of uploading current picture in 0x00 instructions. Added Para1.2 definition in 0xE0 instruction for 0° and 90° deflection Frame timeout is not allowed in interface while only supporting frame terminator	Ver5.0	Ver1.8
2010.06.08	Added the C_Mode.4 definition to support space adjustment of ASCII character in 0x98 instruction	Ver5.1	Ver1.8
2010.07.01	Added the instruction of 0x9D, which is the same as 0x9C without auto-recovery of current background. Added the instruction of 0xD0 which aimed for forcing refreshing. Added the instruction of 0xB004 which is expand to GBK font-library	Ver5.2	Ver1.9
2010.07.15	Added the touching uploading 0x79 instruction with increasing Para1.1 and Para1.0 definition for touching-control pattern and buzzer sound only when click the effective touch area.	Ver5.3	Ver1.9
2010.08.02	Call off the TFT_ID modification function of 0xE0 instruction for avoiding abnormal display due to setting up mistake. Added 0xC107 dot buffer-zone in picture operation.	Ver5.3	Ver1.9
2010.08.09	Added the instruction of 0xC108 to multi-parameter display	Ver5.3	Ver1.9
2010.08.23	Added 0x78 for fast incremental connection. Added 64*48 font for 0x98.	Ver5.3	Ver1.9
2010.10.20	Added extra textbox restriction and cancellation command 0x45 to display the text paragraph in specific area; Touch panel adjusting Point is modified to cross mark; Two fonts 24x48, 28x56 are added to clock command 0x9B.	Ver5.3	Ver2.0
2011.01.27	Added 0x7A instruction supports video player HMI based on SD-card.	Ver5.5	Ver2.0
2011.05.11	Added 0x73 instruction supports two-color bitmap fill. 0x98 instruction added function of supporting large dot matrix characters.	Ver5.5	Ver2.0
2011.07.01	Modify 0xE0: TFT_ID can't be modified, parameters are not saved after power-off, added background automatically restored; Added high-speed (6.225 or 12Mbps) download function for pictures and fonts.	Ver6.0	Ver2.1

If you have any questions about this document or DWIN HMI, please feel free to send mails to our HMI product support with the email address: dwinhmi@263.net

Or you can call:

(86) 10-62628965 or 62655113

Welcome to visit our web site which contains DWIN HMI's latest documents:

www.dwin.com.cn

Thank you for your support to DWIN, which motives us all the time.