AGU SERIES DISPLAY USER MANUAL

Orient Display

Version	Date	Note
V01	2018.3.20	The first release

1. AGU Display Features:

1. The control board supports a maximum resolution of 320*480, making it suitable for displays with sizes up to 3.5 inches and smaller.
2. User interface uses the standard 8N1 UART 232 protocol with 3.3V TTL voltage levels.
3. Onboard storage for UI images using SPI FLASH, controllable backlight circuit, and power management circuit.
4. It's an instruction set serial display, allowing UI development by sending relevant commands and control information through the UART interface.
5. Operating voltage options: 3.3V or 5.0V.
6. It includes touchscreen control commands and is optional with RTP (Resistive Touch Panel) or CTP (Capacitive Touch Panel).

2. Differences from Traditional Serial Screens:

1. It offers a set of 23 commands, supporting command chaining (up to 40 commands in a chain).
2. Supports creating circular progress bars using commands.
3. Allows for the display of digital art characters.
4. Offers five selectable overlay modes for efficient use of flash resources.
5. Supports arbitrary rotation of the screen in all four viewing angles using commands.
6. Onboard storage of UI images in FLASH, which is 100% open for user utilization. Fonts, images, or user parameters can be stored freely at any location in FLASH, with configuration of corresponding addresses in commands.
7. Features high-speed serial updating of the onboard SPI FLASH. It supports a maximum baud rate of 1,500,000, and a 32Mbit flash can be completed within 45 seconds including formatting time.

3. System Command Format:

Command Format	RW	Bit Width	Function Description
BL(brt);\r\n	W	brt:0 or 255	Backlight Brightness Control [0: Off 255: On]
PS(x,y,color);\r\n	W	x,y,color:0~65535	Draw a pixel [x,y as coordinate; color: color value]
PL(x1,y1,x2,y2,color);\r\n	W	x1,y1,x2,y2,color:0~65535	Draw a straight line [From x1,y1to x2,y2;color: color value]
BPS(bps);\r\n	W	bps:2400,4800,9600,38400, 115200,etc **Default:115200**	Set the baud rate [communication protocol: 8 data bits, 1 stop bit, no parity], with a maximum baud rate of 1,500,000.
CLR(color);\r\n	W	color:0~65535	Clear screen with the specified color [color].
DIR(angle);\r\n	W	Angle: 0~3	Set the viewing angle [angle=0: 6 o'clock; 1: 9 o'clock; 2: 12 o'clock; 3: 3 o'clock].
BOX(x1,y1,x2,y2,color);\r\n	W	x1,y1,x2,y2,color:0~65535	Draw a hollow rectangular frame with the starting coordinates (x1, y1), ending coordinates (x2, y2), and the line color set to [color].

BOXF(x1,y1,x2,y2,color);\r\n	W	x1,y1,x2,y2,color:0~65535;	Draw a solid rectangular frame with the starting coordinates (x1, y1), ending coordinates (x2, y2), and the fill color set to [color].
CIR(x,y,r,color));\r\n	W	x,y,color:0~65535;	Draw a hollow circle with the center coordinates (x, y), radius (r), and the line color set to [color].
CIRF(x,y,r,color);\r\n	W	x,y,color:0~65535;	Draw a solid circle with the center coordinates (x, y), radius (r), and the fill color set to [color].
FSIMG(Paddr,x,y,w,h,mode);\r\n	W	Paddr:32bit width; x,y,w,h:0~65535; mode:0~3;	Display an image from FLASH with the following parameters: Paddr: Starting address of the image stored in FLASH. x, y: Coordinates of the top-left corner where the image will be displayed on the screen. w: Width of the image. h: Height of the image. Mode: Mode=0: Display the original UI image; fastest speed. Mode=1: Transparent mode; transparent black backgrounds will be filtered out, pixels match the UI original image. Mode=2: Background color is set to Bcolor for character display, pixels match the UI original image. Mode=3: Background color is set to Bcolor, pixels are set to Ccolor. Mode=4: Transparent display; background matches the base image color, pixels are set to Ccolor. Note: Except for Mode 0, other Modes are suitable for overlaying small images. Transparent backgrounds are based on the previous full-screen base image (such as using FSIMG/CLR/BOXF commands for full-screen display). Mode 2/3/4 are suitable for icons that require dynamic color modification.
BTN(x,y,w,h, Cstring,Style,Fcolor, Ccolor,Bcolor); \r\n	W	x,y,w,h:0~65535;Cstring: string; Style: 0~4; Fcolor,Ccolor,Bcolor: 0~65536;	Set a button with character display with the following parameters: x, y: Coordinates of the top-left corner of the button. w: Width of the button. h: Height of the button. Cstring: Content of the string inside the button. Fcolor: Color of the button's outer frame. Ccolor: Color of the characters. Bcolor: Background color. Style: Button style (0: No border; 1: Button pressed; 2: Button raised; 4: Solid color border).
FRM(x,y,w,h,Fcolor); \r\n	W	x,y,w,h,Fcolor:0~65535;	Set a window with the following parameters: x, y: Coordinates of the top-left corner of the window. w: Width of the window. h: Height of the window. Fcolor: Background color of the window.
SEC(xC,yC,radius,angleS,angleE,Width,color);\r\n	W	xC,yC,radius,angleS,anglE, Width,color: 0~65535	Draw an arc with the following parameters: xC, yC: Coordinates of the center of the arc. radius: Radius of the arc. angleS, angleE: Starting and ending angles of the arc. Width: Width of the arc in pixels. color: Color of the arc. Note: This command includes rounded ends for the arc. It can be conveniently used to create circular progress bars and similar visual elements.
SCHC(Ccolor,Bcolor);\r\n	W	Ccolor,Bcolor:0~65535	Set font colors with the following parameters: Ccolor: Font color. Bcolor: Background color. Note: This command is used to specify the colors for displaying characters.
CADR(Caddr,Taddr,Cw,Ch);\r\n	W	Caddr,Taddr:32bit width; Cw,Ch:0~255;	Set the starting addresses in FLASH for ASCII characters and Chinese characters, along with the pixel width and height for ASCII characters, with the following parameters: Caddr: Starting address in FLASH for ASCII characters. Taddr: Starting address in FLASH for Chinese characters (GBK).

			Cw: Pixel width of ASCII characters. Ch: Pixel height of ASCII characters. Note: When creating font libraries, ensure that ASCII characters and Chinese (GBK) characters have a 1:2 width-to-height ratio and are of equal height. Common font sizes include: ASCII - 8x16, 12x24, 16x32, 24x48; Chinese (GBK) - 16x16, 24x24, 32x32, 48x48. If there are 4 pairs of font libraries, there will be 8 flash addresses, and configuration should be done in pairs. Only the last set of addresses configured will be effective. You can choose to create and configure either ASCII or Chinese (GBK) font libraries, and the other address parameter can be set to any value. Before using font libraries, you need to create and store the font libraries in FLASH, and then use this command to set the addresses of the font libraries in FLASH. This is necessary for using DCHR/DART commands. If you do not change the font library or only have one type of font library, you need to set the address only once. If you have multiple different font libraries to switch between, you should use this command to set the corresponding addresses before using DCHR/DART commands.
DCHR(x,y,Cstring,Ccolor,Bcolor,mode);\r\n	W	x,y,Ccolor,Bcolor:0~65535; Cstring:string; mode:0~1;	Display a string with the following parameters: x, y: Coordinates of the top-left corner of the string. Cstring: The string to be displayed. Ccolor: Font color. Bcolor: Background color. Mode: 0 for opaque background, 1 for transparent background. Note: The string should not contain half-width commas. If you have a GBK font library, you can use full-width commas. Transparent background is based on the entire base image. If the character you need is not present in the 95 ASCII characters, you can replace an unused character in the ASCII set with your custom character at the corresponding code position. This is a useful workaround.
DART(x,y,Cstring,Ccolor,Bcolor,mode);\r\n	W	x,y,Ccolor,Bcolor:0~65535; Cstring:ASCII artistic numerical string mode:0~2;	Display custom artistic numbers with the following parameters: x, y: Coordinates of the top-left corner of the string. Cstring: ASCII number string. Ccolor: Font color. Bcolor: Background color. Mode: Mode=0: Display the same as the original UI image; fastest speed. Mode=1: Background color set to Bcolor, pixels match the UI original image. Mode=2: Background color set to Bcolor, pixels set to Ccolor. Mode=3: Transparent display; background matches the base image color, pixels match the UI original image. Mode=4: Transparent display; background matches the base image color, pixels set to Ccolor. Note: To display ASCII-encoded artistic characters, you need to first set the address of the artistic font in FLASH using the <CADR> command. Artistic fonts are actually images, and the underlying code uses the FSIMG command to implement them. When creating artistic numbers, each type of artistic number is limited to the characters [0123456789 -.:%^@#&*<>], up to 22 characters in total. Except for " - .:", all characters must be of equal width, with an even pixel width. The " - .:" characters should be half-width and of equal height compared to the other characters. The first 11 characters (including half-width spaces) are mandatory to build the font library, and subsequent characters can be added as needed. If a character like "@" is used, the characters " - .:%" must also be defined. Each type of artistic number character must be stored continuously in the specified order.
SVAR(Saddr,offset,var1,var2,var3,var4);\r\n	W	Saddr:32bit width; offset:0~65536; var1,var2,var3,var4:0~255;	Save variable parameters with the following parameters: Saddr: Starting address of the sector where variable parameters are saved. offset: Offset in bytes relative to Saddr. var1, var2, var3, var4: Four parameter values to be written at once. Note: One sector in FLASH contains 4096 bytes (depending on the onboard FLASH). It is recommended to use the last sector to save variable parameters, with a maximum of 4096 bytes for parameter storage.

RVAR(Saddr,offset,readnum);\r\n	R	Saddr:32bit width; offset:0~65536; readnum:0~1024;	Read variable parameters with the following parameters: Saddr: Starting address of the sector where variable parameters are saved. offset: Offset in bytes relative to Saddr. readnum: Number of bytes to be read. Note: You can read variable parameters stored in a specific sector starting from the given address and offset, with the specified number of bytes to be read.
TPCB();\r\n	W	No Parameter	Initiate a touchscreen calibration once by touching five specified points. After completion, the "TPOK" end character will be sent. Note: If the touchscreen is continuously pressed for 15 seconds, it will automatically enter the touchscreen calibration interface.
5A,A5,cmd,Px_H8, ,Px_L8, Py_H8, ,Py_L8, 5B,B5	R	Each parameter is 8 bits width	5AA5: Two bytes for the frame header. 5BB5: Two bytes for the frame footer. Cmd: Control byte (0x73 = key pressed; 0x72 = key released). Px_H8: High 8 bits of the x-coordinate. Px_L8: Low 8 bits of the x-coordinate. Py_H8: High 8 bits of the y-coordinate. Py_L8: Low 8 bits of the y-coordinate. Note: If the touch is continuously held, the 0x73 touch command will be continuously sent with an approximate interval of 130ms.
5A A5 Length0 Length1 Length2 Length3 5B B5 data0……datan checksum	W	Each parameter is 8 bits wide, and there are four parameters in total. DATA0~3 : four parameters together form a 32-bit value,	The UART2FLASH command is used to update the SPI FLASH through UART communication. Here is the communication flow between the external MCU and UART LCM: 1, The external MCU sends the UART2FLASH command, which consists of 8 bytes. The UART LCM is in a serial waiting state to receive the UART LCM's flash formatting indicator. 2, Upon receiving the command, the UART LCM enters the process of formatting the entire flash chip. After formatting is complete, the UART LCM outputs two bytes: 0X4F 0X4B. It then enters a waiting state for the external MCU to send data. 3, The external MCU receives 0X4F 0X4B, indicating that the UART LCM is ready for data transfer. The MCU then sequentially sends data through the serial port to the UART LCM. The last byte sent is the checksum (cumulative sum of data0 to datan). After sending data, the MCU waits for the UART LCM to indicate data reception completion. 4, Upon receiving all the data, the UART LCM checks if the data size is greater than or equal to what was specified in the command and confirms the checksum is correct. If everything is in order, the UART LCM sends a reception completion indicator to the external MCU: 0X4F 0X56. 5, The external MCU receives 0X4F 0X56, indicating that the entire flash update process is complete. Note 1: The UART LCM formats the SPI FLASH as a whole chip, which may take a considerable amount of time depending on the flash size. For example, a 25Q32 flash may take approximately 12 seconds or longer to format. The formatting time increases with larger flash sizes. Data is written sequentially, starting from address 0. Note 2: It's essential to send data from the external MCU in framed packets, with a maximum of 512 bytes per frame. It is recommended to have at least a 5-byte pause between frames. Note 3: The UART LCM supports a maximum baud rate of 1500000. Before sending the command, the external MCU can modify the UART LCM's baud rate to 921600 or 1500000 for high-speed data transmission.。

4. Send Commands

To send commands to the UART LCM, configure the serial communication protocol as shown in the figure below and paste the following commands into a serial debugging assistant. Ensure that you send the commands as strings, not in HEX format. When sending commands from the

MCU, you can send a maximum of 512 bytes at a time, and it supports command chaining. For example:

UART_Send_NChars("CADR(0,1520,8,16);DCHR(0,0,Small URAT Display,63488,65504,1);\r\n");

2 commands are included.

The UART LCM will provide response strings after executing the commands. The response can vary but will always end with "OK" (except for touch screen and flash update commands). It is recommended to check for the "OK" flag in the response before sending the next command. If you do not check for the "OK" flag and rely on delays, consider the number of commands in the command chain and test with the maximum transparent icon as standard

```
BL(255);\r\n
PS(80,80,65535);\r\n
PL(0,0,100,100,65535);\r\n
BPS(115200);\r\n
CLR(63488);\r\n
DIR(0);\r\n
BOX(40,100,100,140,2016);\r\n
BOXF(140,160,180,200,63488);\r\n
CIR(160,120,30,65535);\r\n
CIRF(240,120,20,31);\r\n
FSIMG(2495840,0,0,320,240,0);\r\n
BTN(60,60,60,36,串口屏,2,63390,63488,00000);\r\n
FRM(40,40,160,136,63390);\r\n
SEC(80,80,60,0,300,6,63488);\r\n
SCHC(63390,63488);\r\n
CADR(0,6080,8,16);\r\n
DCHR(75,30,NEW PANEL-小尺寸串口屏,63488,65504,0);\r\n
DART(0,120,12:34,63488,65504,3);\r\n
SVAR(8384512,100,2,2,3,4);\r\n
RVAR(8384512,100,20);\r\n
TPCB();\r\n
```