Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

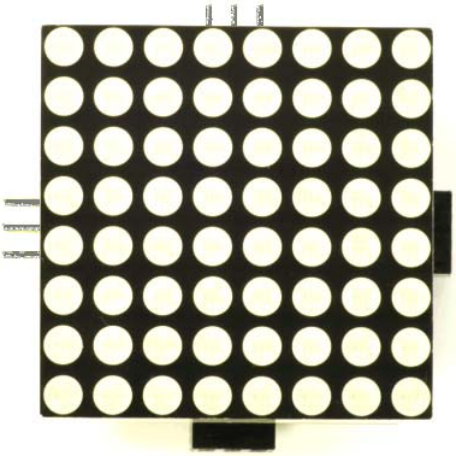
Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

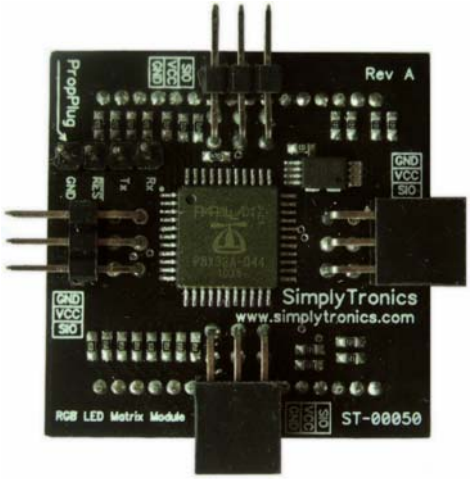
www.simplytronics.com
sales@simplytronics.com
support@simplytronics.com

8x8 RGB LED Matrix Display Module #ST-00050

Add a full-color 8 x 8 LED display to your next project! The RGB LED Matrix Display Module, based on Parallax's multicore Propeller microcontroller, comes pre-loaded with factory firmware that supports a detailed command set. The module may be controlled with most microcontrollers, such as the BASIC Stamp, Propeller, Arduino, AVR, PIC, etc.

The RGB LED Matrix Display Module is fully customizable: you can re-write your own firmware, define your own protocol, or modify the Spin and C firmware available for download on our website. Or, you can use the factory firmware and your own custom computer software to configure the modules via the 4-pin header for Parallax's Prop Plug.





Product Features

- Built-in 5 x 8 ASCII characters
- 16,777,216 different colors available for each pixel (RGB-888)
- Easy interfacing via UART(Baud Rate: 2400 to 115200 bps)
- Control with a microcontroller through a single digital I/O pin
- Save a pixel pattern to the module's EEPROM
- Reprogrammable with Propeller Spin or C firmware, available for free download and customization
- 4-pin header for the Prop Plug (Parallax #32201, not included) for firmware updates or direct configuration via a custom computer interface.
- Factory pre-programmed with C (PropGCC) firmware
- Connect multiple modules to create a large screen
- Dual-row 3-pin headers provide stable connections between modules
- Firmware supports 8 x 16 pixel ASCII characters for multiple modules

Technical Details

- Parallax Propeller P8X32A
- 10 MHz crystal for up to 80 MHz operation
- 32 KB EEPROM
- 3.3 V on-board regulator
- Communication/Interface: UART/TTL, 2400~115200 bps, 8N1
- Operating temperature: 0 °C to 70 °C
- Dimensions: 3.8 x 3.8 x 1.8 cm

Suggested Applications

- Advertisement display
- Video wall
- Public/transit information displays
- Electronic art installations

Warning: Voltage- and Static-sensitive Device

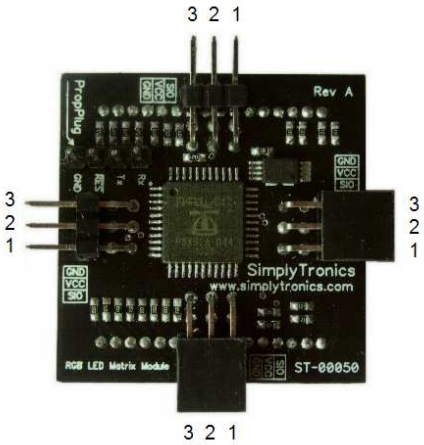
- RGB LED Module is a delicate, static-sensitive device requiring proper voltage
- Observe proper anti-static techniques when handling them.
- Disconnect all power before connecting or disconnecting the module to a circuit.
- Do not reverse polarity of the power connections, it could destroy the device.

Specifications

Symbol	Quantity	Minimum	Typical	Maximum	Units
VCC	Supply Voltage	4.2	5.0	7.5	V
ICC(Sb)	Standby supply current	50	55	65	mA
ICC(Su)	Startup supply current	145	150	180	mA
ICC(Rt)	Run-time supply current	50	55	80	mA
GND	Ground reference connection	—	0		V
VIH	Signal high receive (SIO)	2.4	3.3	5.0	V
VIL	Signal low receive (SIO)	- 0.3	GND	0.3	V

Pin Definitions

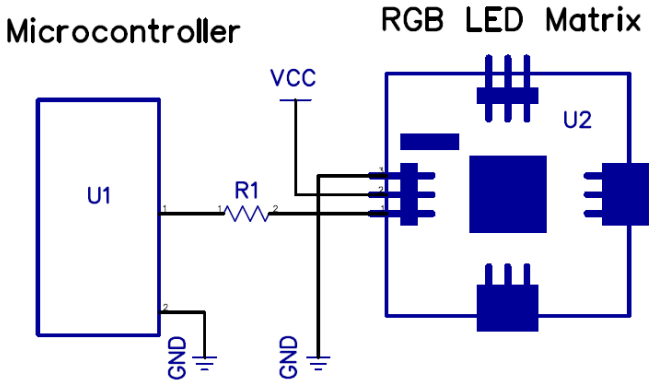
Read pin labels on the board carefully when making connections. When read from the edge of the board, the top and bottom headers have the opposite pin sequence. So do the two side headers. This allows modules to be connected together. Only connect power to one VCC pin at a time. Only one GND connection is needed.



Pin	Name	Type	Function
1	SIO	Input	Serial data input
2	VCC	Power	Supply voltage
3	GND	Ground	Ground (0 V)

Microcontroller Connection

You may connect the RGB LED Matrix module to a microcontroller through any of the module's SIO pins. Do not connect to more than one SIO pin at once. Note: the male 3-pin headers may be covered with removable protective caps.



VCC = 5V to 7.5V

R1 = 4.7k for 5V microcontrollers

R1 = 0 (not required) for 3.3V microcontrollers

Example Programs

Download the latest version of this documentation, microcontroller example code, firmware, and software from the RGB LED Matrix product page. Search ST-05000 at http://www.simplytronics.com

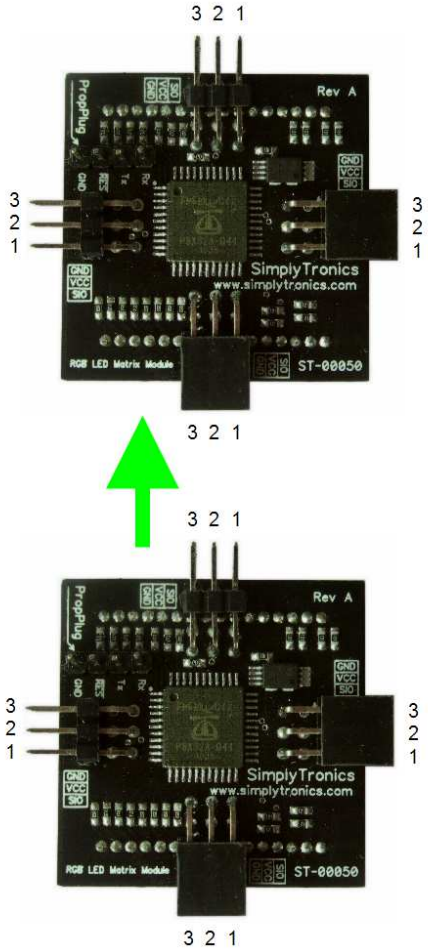
Connecting Multiple Units

Advanced users may connect several modules in a row or grid to create a larger display screen.

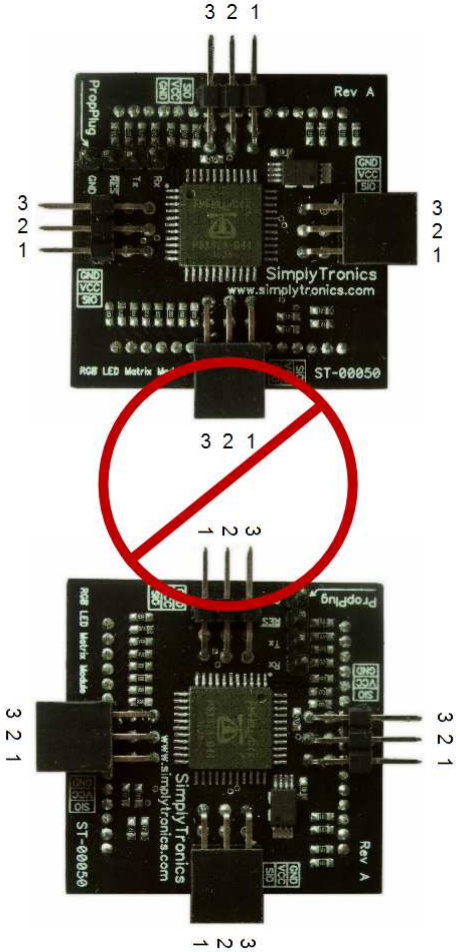
1) To connect modules together, orient the modules LED-side down in a row or grid. All the male headers should point toward two sides of the grid (such as top and left) while all female headers should point the other two sides of the grid (such as bottom and right).
2) Make sure the male 3-pin header on one module has the same pin sequence as the female header of the module next to it or above it before connecting them together (see diagrams below).

CAUTION: DO NOT PLUG A (GND, VCC, SIO) HEADER INTO A (SIO, VCC, GND) SOCKET.

CORRECT.



WRONG!



Notes:

1) Before connecting multiple units together in a row or grid to make a larger screen, each one must be assigned its own Device ID, and all modules must be assigned the same PSS value. See Device ID and PSS on page 6 for more information.
2) Connect power to any one—and only one—of the modules in the screen. Be sure to supply enough current for all modules in the screen.
3) To control a screen of multiple units, connect a microcontroller I/O pin to an SIO pin on any one—and only one—of the modules. Or, connect to a module via a Prop Plug to configure the screen from custom computer software.

Communication Protocol

UART Serial

Default baud rate 19200 bps, max 115200 bps, 8N1, no parity. Baud rate will not be stored to EEPROM.

Command Packet

The modules receive command from host by packets.

"PMTX"	Command	Parameter/Value
4 bytes	1 byte	4 bytes

Color Data Format

The modules receive color data from host. Each pixel needs 3 bytes of data to store the RGB-888 color.

1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	…
Blue	Green	Red	Blue	Green	Red	…

Command Set

Notes:

1) Wait for 2 seconds after start or restart before sending any commands or data. The RGB LED Matrix Module takes about 2~3 seconds to initialize.
2) Make sure the data you send out are NOT less than Data_Length. That means, if you tell the Module you're about to send 512 bytes, you need to send 512 or more bytes before the next command. For best results, send (Data_Length + 1) bytes.
3) There are 6 Matrix Graphic RAMs inside the module, GDRAM, wGDRAM and 3 bxGDRAMs. The Module always gets data from GDRAM to display, wGDRAM is the working GDRAM. If you want to display the data from bxGDRAM, you need to copy the data to wGDRAM and then update to GDRAM.
4) This command set is subject to change with firmware updates. Please go to http://www.simplytronics.com and search ST-00050 for the latest firmware.

Command	Index	Parameter	Description
_UpdateScreen	$00	$xxxxxxxx	Update Screen, copy the data from wGDRAM to GDRAM.
_SetPointerX	$03	Pointer X	Move the Cursor to the address given by this command.
_SetPointerY	$04	Pointer Y	Move the Cursor to the address given by this command.
_SendData	$05	Data Length	Send the RGB data to LED Matrix, current support RGB-888, 24 bit/pixel. After sending this command packet, send Data_Length bytes of RGB data to feed the modules.
_SetWindowsBoundStartX	$06	Windows Bound X Start	Set the start of the X boundary of the operating area.
_SetWindowsBoundStartY	$07	Windows Bound Y Start	Set the start of the Y boundary of operating area.
_SetWindowsBoundEndX	$08	Windows Bound X End	Set the end of the X boundary of the operating area.
_SetWindowsBoundEndY	$09	Windows Bound Y End	Set the end of the Y boundary of the operating area.
_Copy2bxGDRAM	$0A	x=0~3	Copy Data from GDRAM to bxGDRAM.
_DisplaybxGDRAM	$0B	x=0~3	Copy Data from bxGDRAM to GDRAM.
_fillGDRAM	$0C	Color	Fill the GDRAM with the specified color.
_SetForegroundColor	$11	Color	Set the character color.
_SetBackgroundColor	$12	Color	Set the character background color.

Command	Index	Parameter	Description
_DisplayChar0508	$13	ASCII Character	Display a 5x8 ASCII character. The character will take 8x8 dots, with background set by user.
_DisplayChar0816	$14	ASCII Character	Display an 8x16 character. The character will take 8x16 dots, with background set by user.
_DisplayChar0508NBG	$15	ASCII Character	Display a 5x8 character. The character will take 8*8 dots, without background.
_DisplayChar0816NBG	$16	ASCII Character	Display an 8x16 character. The character will take 8x16 dots, without background.
_DisplayPropellerHat	$17	$xxxxxxxx	Display Propeller Hat logo in 16x16 mode, with background .
_DisplayPropellerHatNBG	$18	$xxxxxxxx	Display Propeller Hat logo in 16x16 mode, without background.
_DisplayParallaxLogo	$19	$xxxxxxxx	Display Parallax logo in 96x16 mode, with Background.
_DisplayParallaxLogoNBG	$1A	$xxxxxxxx	Display Parallax logo in 96x16 mode, without background.
_DisplayCustomerBMP1	$1B	$xxxxxxxx	Display built-in BMP picture 1. Requires replacing _CustomerLOGO1.bmp file in firmware folder and recompiling and uploading firmware.
_DisplayCustomerBMP2	$1C	$xxxxxxxx	Display built-in BMP picture 2. Requires replacing _CustomerLOGO2.bmp file in firmware folder and recompiling and uploading firmware.
_DrawDot	$1D	$xxxxxxxx	Draw a dot on the screen.
_SetLineEndX	$1E	Line End X	Set the Line End X, the line starting from Pointer X.
_SetLineEndY	$1F	Line End Y	Set the Line End Y, the line starting from Pointer Y.
_DrawLine	$20	Line Color	After set the start and end of the line, draw it on the screen.
_SetRadius	$21	Circle Radius	Set the radius of the circle.
_DrawCircle	$22	Circle Color	Draw a circle at(PointX, PointY).
_SavePictureEeprom	$23	$00000000	Save the screen to EEPROM.
_LoadPictureEeprom	$24	$00000000	Load the picture to screen from EEPROM.
_ScreenColorInverse	$30	$xxxxxxxx	Inverse the color of the screen.
_ScreenOverLay	$31	$xxxxxxxx	Over lay
_ScreenMinusLay	$32	$xxxxxxxx	Minus lay
_Reset	$A0	$xxxxxxxx	Resets the module. After reset, wait 2 seconds for the module to start up before sending another command.
_SetBaudRate	$A1	NewBaudrate	Change to a new baud rate.
_SetMode	$A2	Mode	5 modes supported.
_GetScreenSize	$A7	$xxxxxxxx	Return PSS.
_ChangeScreenSize	$A8	PSS	Change Screen size to (PSS+1)*8, this command needs to reset the module.
_GetDeviceID	$A9	$xxxxxxxx	Return Device ID.
_ChangeDeviceID	$AA	new Device ID	Change to a new Device ID, this command needs to reset the module.
_GetSystemConfiguration	$AB	$xxxxxxxx	Return Device ID and PSS.

Device ID and PSS

When connecting several modules into a row or grid to make a larger screen, each module needs to be assigned its own Device ID to indicate its current position. The advanced test code was designed based on the Device ID table below, up to 16 x 16 modules, a 128 x 128 pixel screen.

Each module in a screen also needs to be assigned the correct Pseudo Screen Size (PSS) value. This value indicates the size of the screen, and all modules in a screen use the same value. The Real Screen Size (RSS) can be calculated using the following formula:

RSS = (PSS + 1) * 8

PSS also can be seen as the Device ID of the last module in the first row.

Device ID Table

↙LED dot(0,0) LED dot(127,0)↘

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

↖LED dot(0,127) LED dot(127,127)↗