Serial Dot Matrix Display

Technical Manual Rev 1r0

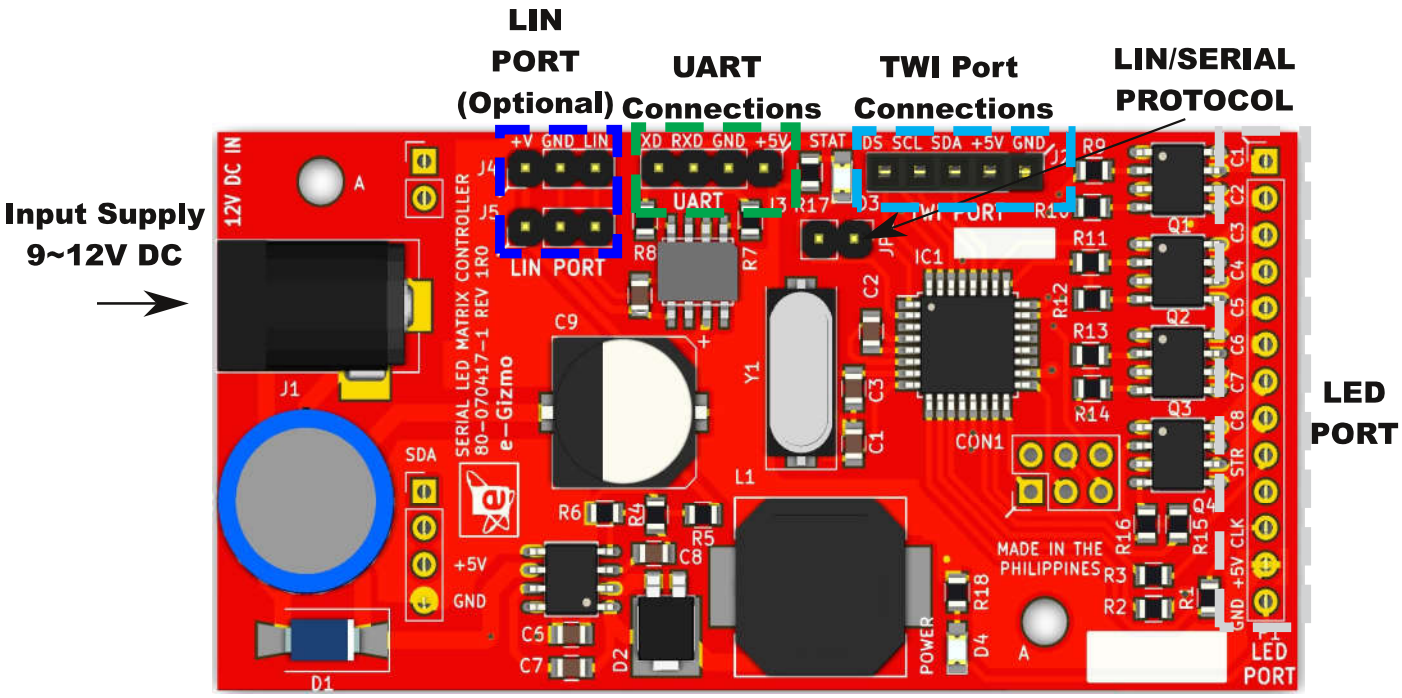
Serial Dot Matrix Display is a LED dot matrix controller for displaying messages thru serial UART communications ,where you can show your output from your sensors and time/date display. gizDuino and Arduino Compatible module.

Features:

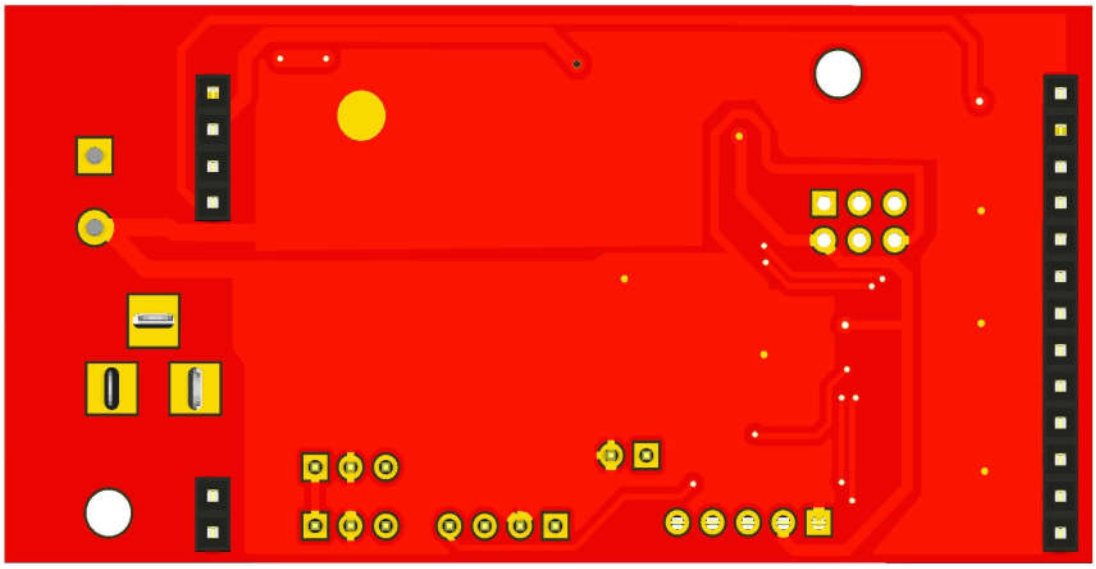
- *On board high efficiency DC/DC converter allows the display unit to work from 7.5-16V DC source.*
- *User determined display size, supports up to 4 e-Gizmo Dot Matrix panel display assembly.*
- *Simple serial UART display command set allows you to build a fully functional dot matrix unit with just a few lines of code.*
- *Arduino friendly. A single gizDuino/Arduino board can control more than 4 Serial Dot Matrix display units using SoftwareSerial configured ports.*

General Specifications:

Input supply voltage: 7.5~16VDC
Current: 1.5A Maximum
Maximum Panel: 4 eGizmo Dot Matrix Panel
Interface: UART
PCB Dimensions : 66mm x 38mm



Top View



Bottom View

Figure 1. Board pinouts presentation.

Wiring Connections:
USB-UART to Serial Matrix Controller

+5V	**+5V**
GND	**GND**
TXD	**RXD**
RXD	**TXD**

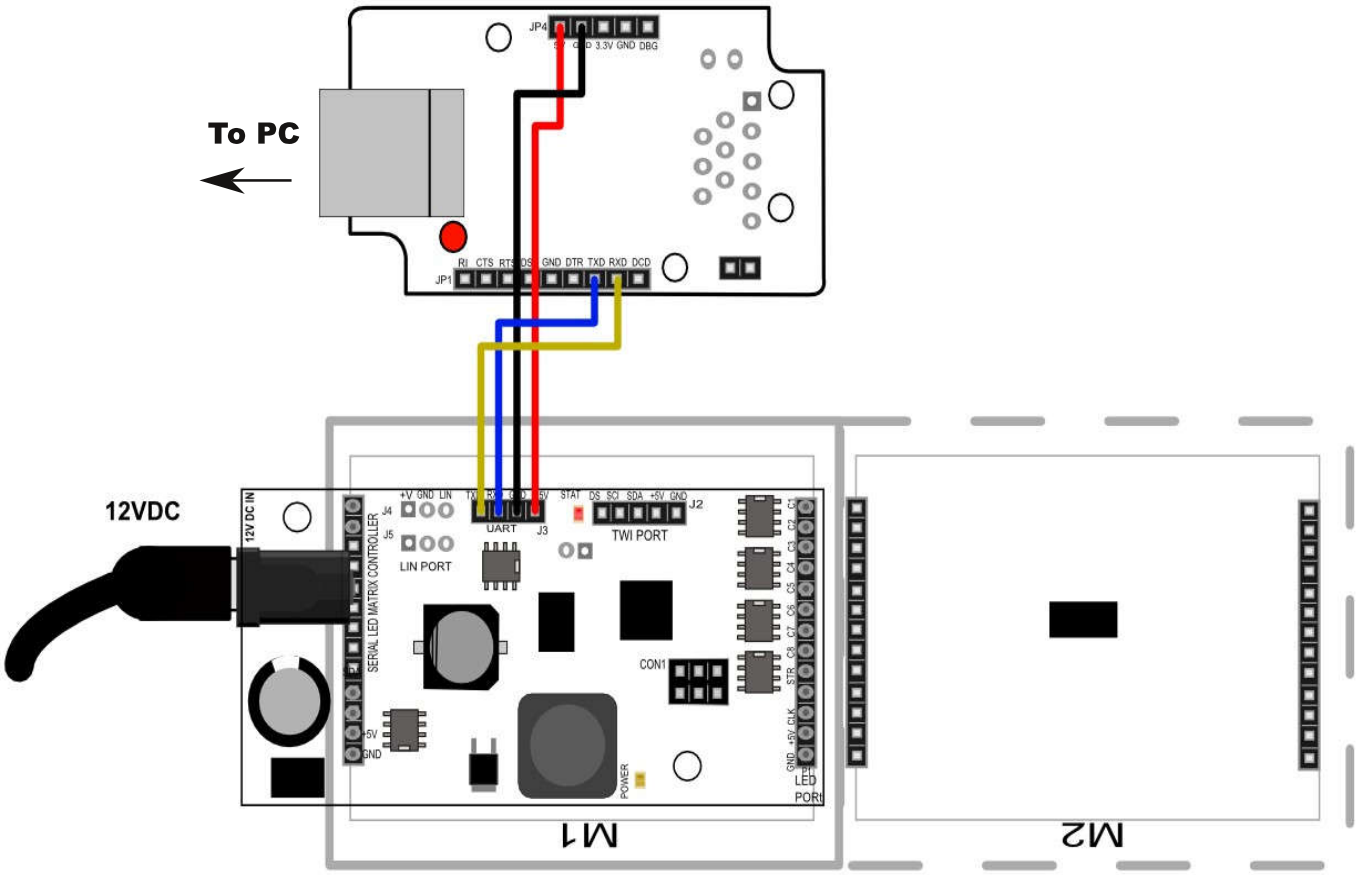


Figure 2. Sample connection using USB to UART converter.

You need to set up a few parameters before you can use the serial dot matrix. The most important parameters you need to set includes the number of display matrix panels used in the display unit, and the serial communications baud rate. These are most probably preset to their correct values if you ordered a complete kit. This section will guide you through the procedure if a need to change the parameters arises (e.g. you decided to install additional panels).

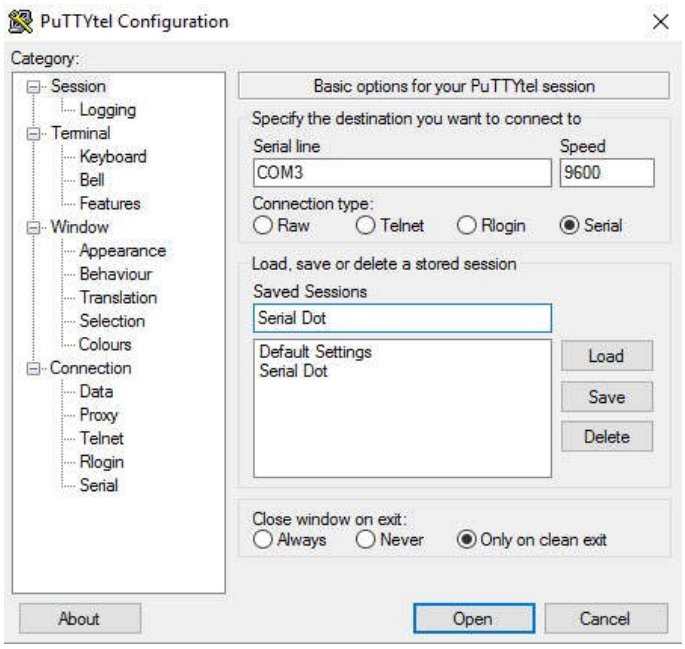
Equipment Needed

PC running a Terminal apps (e.g. PuTTY)
USB to UART TTL Converter board.

UART Settings

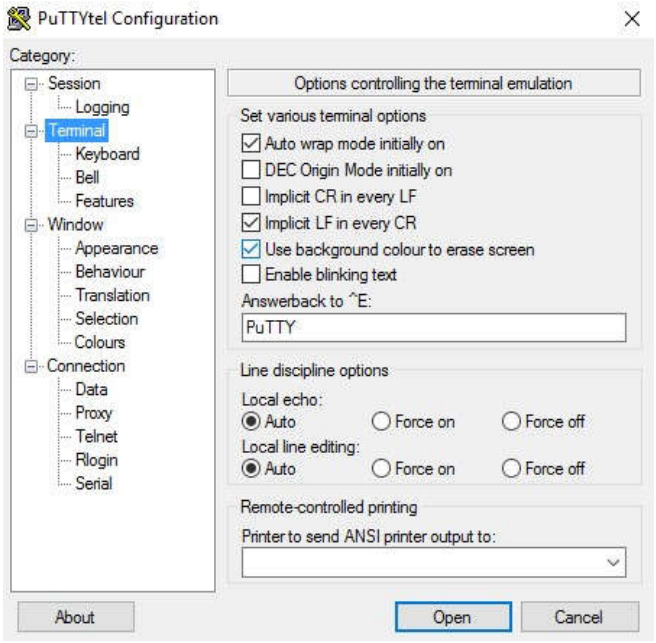
Following are the default UART settings.

Baud Rate - 9600 bps. This can be changed by the user through the Config menu.
Data - 8
Stop Bit - 1
Parity - None
Handshake - None



Category: Session

Serial line: COM #
Speed: 9600
Connection type: Serial
Saved Session: Type 'Serial Dot'
Click SAVE.
Close window on exit: Only on clean exit



Category: Terminal

Set Various terminal options:
~ Auto wrap mode initally on
~ Implicit LF in every CR
~ Use background colour to erase screen
Line discipline options: Auto
Local line ending: Auto
Click OPEN.(another window will open)

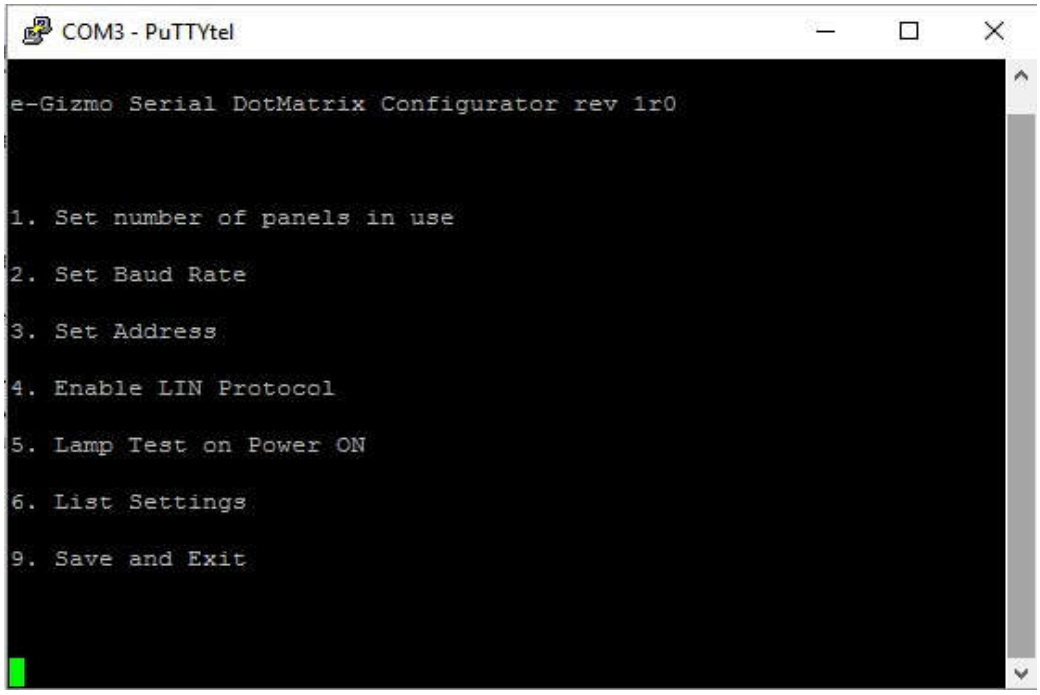


Figure 3. Putty terminal.(Showing the Configuration menu)

The config menu can be activated by entering at least **Press hold [CTRL]+ [C] 3x** keystroke and then hitting **[ENTER]** key.

Configuration Menu Functions

Note: Press the corresponding number key of the desired function to enter.

#1. Set Number of panels in use.

Set the number of dot matrix panels installed in the serial dot matrix controller. Default=2. The controller will work from 1 to 4 panels. You need to set the correct panels for the display to work properly.

Enter number of panels in use (1-4)... []

#2. Set Baud Rate.

The communications baud rate defaults at 9600 baud. You can change this to 4800, 38400, or 57600 baud if you so desire.

Baud Rate (0=9600, 1=4800, 2=38400,3=57600... []

#3. Set Address.

Currently not implement and has no effect on the functionality of the Serial Dot Matrix controller.

Enter Device Address (1-65535,0=No addressing)... []

#4. Enable LIN Protocol.

Currently not implement and has no effect on the functionality of the Serial Dot Matrix controller.

#5. Lamp Test on Power ON.

Enabling this function will result in all dots being lit (lamp test) on power up. The LEDs will stay lit until the host controller sends a command.

LAMP Test Enable? (Y/N)... []

#6. List Setting.

Display the current settings.

#9. Save and Exit.

Save all settings. This Serial Dot Matrix controller will run in this new settings and retain them even if power is removed.

Setting Saved!

New Baud rate (if applicable) will now be in effect.

OK

[]

Factory Reset

The factory reset option initialize all settings to their default values.

- *Number of Panels = 2*
- *Baud Rate = 0 (9600)*
- *Lamp Test ON = Enabled*

To initiate factory reset:

1. *Power OFF display unit.*
2. *Install a jumper (short) across JP1.*
3. *Power ON display unit.*
4. *Remove Jumper.*

Notation

[ESC] = ESC key character
[ESC]+D = Press (or send) ESC key followed by D key

MESSAGE ANIMATIONS

Prefix your message text with the following ESC sequence to display messages with the described animation effects.

Note: All commands must be terminated with a [carriage return] character. (***Press*** **[ENTER]** after typing in commands).

[ESC] + 0,1,2 = Drop In

Drop In animation displays the message by dropping one character at a time, starting from the beginning of the text. This function has 3 variants:

0 = Clear the display and then Drops In the message.

1 = Overwrite the old message by Dropping In the new text. Space characters are skipped.

2 = Overwrite the old message by Dropping In the new text, space characters included.

[ESC]+U = Scroll Up message. Scrolls in the new message from the bottom of the display in upwards direction. Old message is pushed out of the way in the process.

[ESC]+D = Scroll Down message. Same as above, except the direction is now downward starting from the top of the display.

[ESC]+S = Clear display and run in towards the left side to fill the display with the new message until complete message is displayed out.

[ESC]+A = Display message

Wiring Connections:

Gizduino to Serial Matrix Controller

D3	**RXD**
GND	**GND**
D9	**DS**

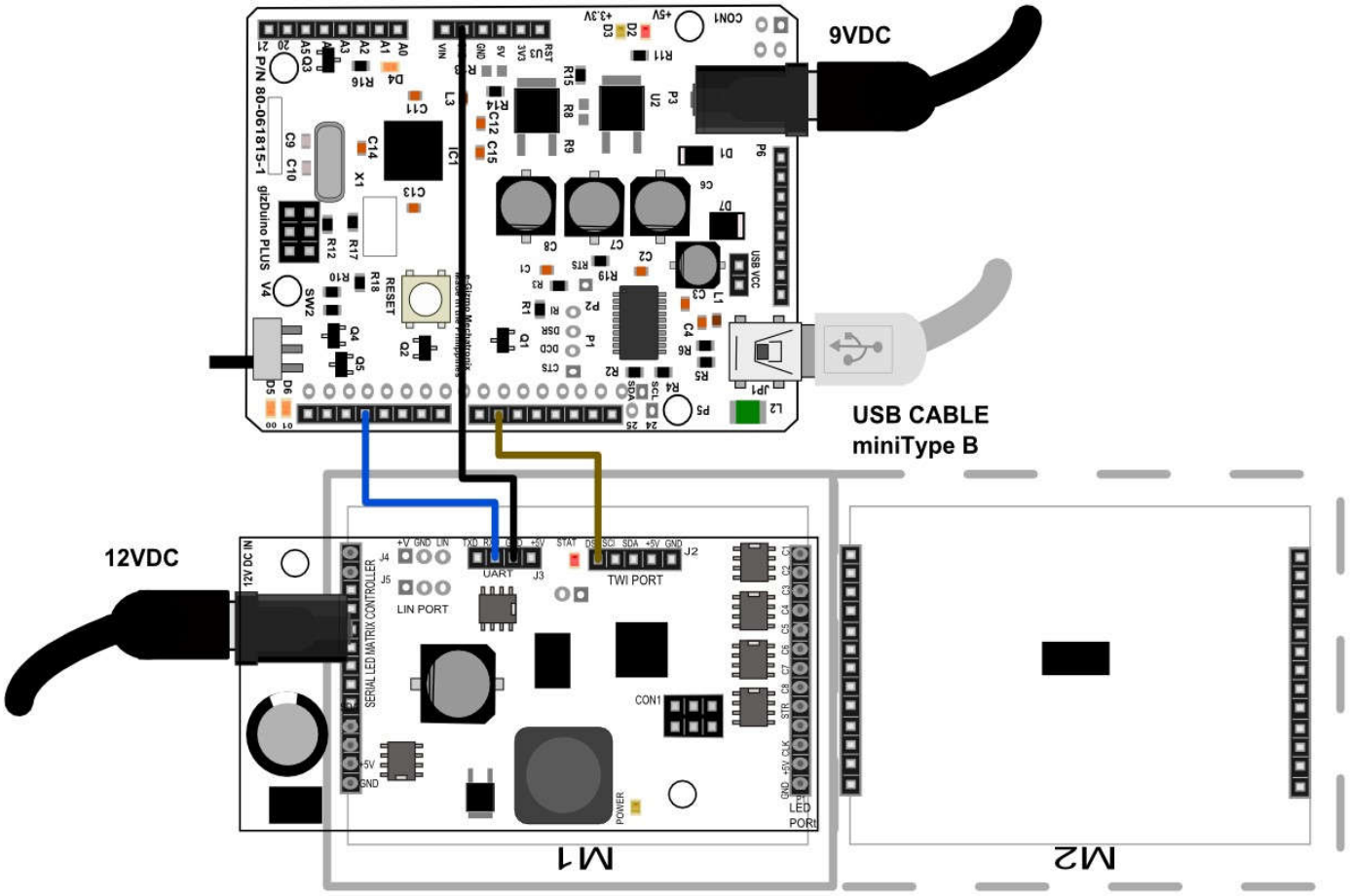


Figure 4. Sample connection using gizDuino PLUS ATmega644P.

POST MESSAGE ANIMATIONS

[**ESC]+s** = Run currently displayed message towards the left side of the display until complete message is displayed out.

[ESC]+d = Slide out downward currently displayed message.

[ESC]+u = Slide out upward currently displayed message.

[ESC]+f = Flash whole display

[ESC]+[ESC] = Clear display screen.

MESSAGE POSITIONING

[ESC]+C = Show suceeding messages centered on the display screen.

[ESC]+L = Show suceeding messages left justified on the display screen. (Default)

[ESC]+R = Show suceeding messages right justified on the display screen.

Note: *If the message is too long to fit in display screen, display positioning defaults to left justified.*

TEST

[ESC]+T = Display the entire character set.

Sample application using USB to UART conveter:

Messaging positioning

1. ***Open the Putty.*** Set the default UART settings. (See Figure)

2. On the Configure Menu. ***Press 1*** on your keyboard, Type 3.
In my example above i used ***'3 panels'***.

3. Then ***Press 9*** to **save** your settings and exit.
If Setting Saved! you may now do the Message animations.

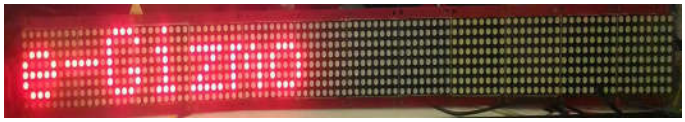
4. For Displaying a message.
example: "e-Gizmo",
then set it to center position.

Note:
- You do not need to hold the ESC + C. Just press them once in order.

Again to start:
a. ***Press [Esc], [C], and [Enter].*** - to set the position in center
b. ***Press [Esc], [A],*** [Type your message ex. ***"e-Gizmo"***], and ***[Enter]***

Note:
- You may also Turn OFF the Caps lock to display the message in small letters. After pressing the Message animation code.

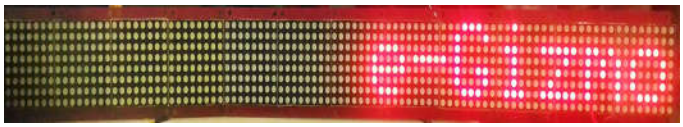
LEFT POSITION



5. For Displaying it to the left position.

a. ***Press [Esc],[Esc]*** and ***[Enter]***. Clear display screen.
b. ***Press [Esc], [L],*** and ***[Enter]***. - to set the position on the left side.
c. ***Press [Esc], [A],*** [Type your message ex. ***"e- Gizmo"***], and [***Enter].***

RIGHT POSITION



6. For Displaying it to the right position.

a. ***Press [Esc],[Esc]*** and ***[Enter].*** Clear display screen.
b. ***Press [Esc], [R],*** and ***[Enter].*** - to set the position on the right side.
c. ***Press [Esc], [A],*** [Type your message ex. "***e-Gizmo***"], and ***[Enter]***

POST MESSAGE ANIMATIONS

1. ***Press [Esc],[C]*** and ***[Enter]*** - to center the position example only.

2. ***Press [Esc]***, ***[A]***, [Type your message ex. "***e-Gizmo***"], and ***[Enter].***

2. Then ***Press [Esc],[s]*** and ***[Enter]***- run currently message towards the left side.

3. Do the no.2 Then **Press [Esc]**,**[d]** and **[Enter]** - slide out Downward.

4. Do the no.2 Then **Press [Esc]**,**[u]** and **[Enter]** - slide out Upward.

5. Do the no.2 Then **Press [Esc].[f]** and **[Enter]** - flashing display

6. If you want to steady again the display do the no.2 again.

7. To clear the display **Press the [Esc]**, ***[Esc]***, **[Enter].**

```
//#include <SoftwareSerial.h>

//SoftwareSerial mySerial(2,3);
#define       READYPIN    9       // pin 9 Ready input


/*  Serial Dot Matrix connections to gizDuino/Arduino
        gizDuinoPLUS->      Serial Dot Matrix
        pin 3               J3-RXD
        pin 9               J2-DS
        GND                 J3-GND

        J2- DS is Serial Dot Matrix output telling gizDuino/Arduino it has completed
                its task and is ready to receive new command (DS LOW=BUSY, DS
HIGH=READY).

 Note: (This example is for gizduinoPLUS ATMEGA644P with Hardware Serial1)
 If you are using gizDuino UNO, use the SoftwareSerial Library from Serial1 to mySerial.

COMMANDS:
 Message Animation
     KeyChar HEX
     ESC    1B
     0      00
     1      01
     2      02
     U      55
     D      44
     S      53
     A      41
 Post Message Animations
     s      73
     d      64
     u      75
     f      66
 Message Positioning
     C      43
     L      4C
     R      52
 Test  T     T or 54
ex. add "\x" before the HEX value of a key character
   for [ESC] =  "\x1B" then add carriage return character "\r" or newline character "\n"
   It shoud be like this: "Serial.print("\x1B\r");"

 Codes by e-Gizmo Mechatronix Central
 http://wwww.e-gizmo.com
 October 26, 2017
 */
```

```
void setup(){

  Serial1.begin(9600);
  delay(1000);
  // "\x1B" = ESC character
  Serial1.print("\x1B\x1B\r");          //[ESC] + [ESC] clear display
}

void loop(){

  //ESC +C = Messages center justified
  Serial1.print("\x1B\x43\r");
  delay(100);

  //ESC +A = Displaying the message
  Serial1.print("\x1B\x41");
  Serial1.print("e-Gizmo"); //Display your messages here
  Serial1.print("\r"); // [Enter] Send your command
  delay(2000);

  //[ESC] +u = Scroll up the message
  Serial1.print("\x1B\x75\r");
  delay(2000);

  Serial1.print("\x1B\x41");
  Serial1.print("Mechatronix"); //Display your messages here
  Serial1.print("\r"); // [Enter] Send your command
  delay(3000);

  //[ESC] +d = Scroll down the message
  Serial1.print("\x1B\x64\r");
  delay(2000);

  Serial1.print("\x1B\x41");
  Serial1.print("Central"); //Display your messages here
  Serial1.print("\r"); // [Enter] Send your command
  delay(1000);

  //[ESC] +f = flashing the message
  Serial1.print("\x1B\x66\r");
  delay(3000);

  Serial1.print("\x1B\x41");
  Serial1.print("Serial Dot Matrix"); //Display your messages here
  Serial1.print("\r"); // [Enter] Send your command
  delay(1000);
```

```
  //ESC+s = Run the message display until it cleared
  Serial1.print("\x1B\x73\r");
  delay(2000);

  Serial1.print("\x1B\x1B\r"); // [ESC] + [ESC]  Clear the display
  delay(100);

  //ESC+T = Test display the character set
  Serial1.print("\x1BT\r");
  wait_till_ready();
}


// This routine will wait until the display is ready
void    wait_till_ready(void){
  int     i;

  //  Give time for the display unit to activate busy output
  for(i=0;i<1000;i++){
    if(digitalRead(READYPIN)==0) break;
  }

  //wait until ready
  while(digitalRead(READYPIN==0));
}
```