Vacuum Fluorescent Display Module

[GU-7000 series module]

Application Note

APN200 R2.1 August 19, 2016

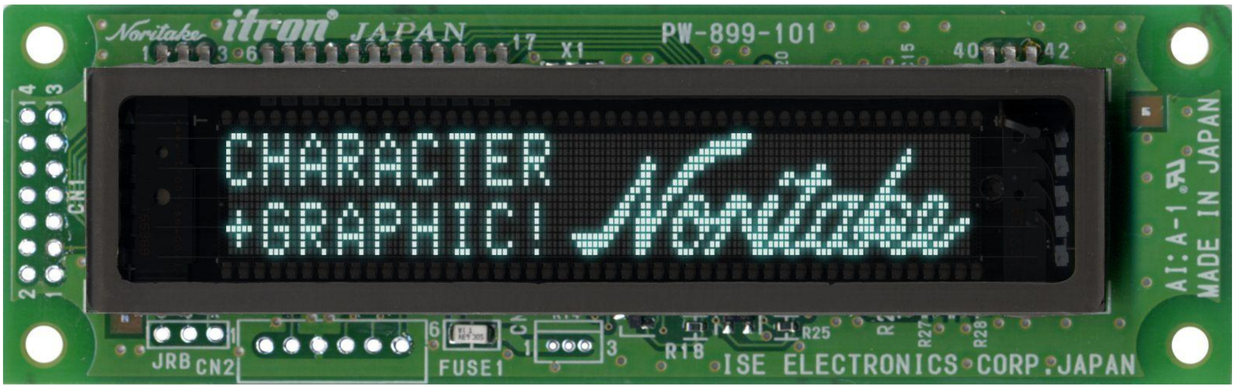
itron is the registered trademark of the NORITAKE Vacuum Fluorescent Display (VFD). VFD is the display device of lighting blue-green, easy to read and self-luminous. It features high visibility, a wide operating temperature range, etc, compared with other display devices such as liquid crystal (LCD) and LED. GU-7000 series is a VFD module which can enable a character display using the graphic type VFD 'dot matrix'. Moreover, they are designed to be used as a functional enhancement version of CU-U series by being similar to the general character type LCD modules. This document is prepared for technical support data when using a GU-7000 series display module.

Table of contents

1 Scope

GU-7000 series VFD module is a series of subsystem of display which consists of vacuum fluorescent display with high readability and reliability, a set of power converter, a controller and flash memory. The display module can be controlled by a host machine sending commands. CJK font models are also available.
Several items have PWB size compatibility with a character LCD and CU-U series modules. Please refer to Photo.1 as an example.

PHOTO.1 GU140X16G-7000



GU-7000 series has character display function with embedded firmware, so it can be used as a character module.

1.1 Character size

VFD is a light emitting display which uses phosphor, it has higher readability than LCD which must use reflections and penetrates light. With VFDs, users are able to read characters from longer distances when using the same character size display. Some VFDs have even smaller font size while utilizing the same reading distance of an LCD. Please see our demonstrations and samples.

1.2 GU-7000B series

GU-7000B series is a model to replace GU-7000. GU-7000B series has some additional command with later version of built in controller. Command set is up word compatible.
However, please test before starting to use GU-7000B, because there are some differences of response timing.

	GU-7000/7003/7900	GU-7000B/7003B/7900B
Serial input buffer size	12 bytes	60 bytes
Display address set by dot	Nothing	Available Reference:[5.6.5 Command table5 Additional command sequence for GU-7xxxB Display by dot]

2 VFD module lineup

2.1 Item number

Outline of specification by item number

Example: GU128X32D-7003

GU : Graphic display

128X32 : 128 x 32 dots

D : Dot size height 0.4~0.49mm / dot

C=0.3~0.39mm

D=0.4~0.49mm

E=0.5~0.59mm

F=0.6~0.69mm

G=0.7~0.79mm

J=1.0~1.09mm

-7003 : 7000 series

Indicate interface and embedded font.

7x3x＝3.3V power requirement

70xx＝8 bit (1 byte) character only

72xx＝8 bit character code with more function

79xx＝8 Bit & 16 Bit character code available

(Japanese, Chinese, Korean font)

7xx0＝RS232C & parallel interface

7xx1= USB interface

7xx2＝C-MOS serial& parallel interface

7xx3＝C-MOS serial interface

77xx= Custom item

Example: GU128X32D-7003B

B at the end indicates means upper version.

Same series items use the same commands and major features. This table shows general information. Please refer to each specification for details.

2.2 Distribution diagram for standard series

For displays other than GU-7000, please refer to applicable application notes and specifications.

```
Noritake Itron VFD module
 ├ Custom module
 ├ Standard module
 └ Standard GU series
        |    Graphic display series
        ├ GU-300 series, GU9300 series
        |
        ├ GU-800 series
        |       ├ GU-800, GU-800B
        |       |    Graphic mode only
        |       └ GU-820A, GU-820B
        |            Embedded Japanese font
        ├ GU-7000 series   Outline compatibility with LCD
        |       ├ GU-7000
        |       |    Small size graphic and character display module
        |       ├ GU-7032
        |       |    3.3V input voltage version
        |       ├ GU-7900
        |       |    GU-70000 with embedded Japanese, Chinese and Korean font
        |       ├ GU-7000B
        |       |    Replacement for 7000
        |       ├ GU-7900B
        |       |    Replacement for 7900
        |       └ GU-7900BX
        |            High brightness version of 7900B
        └ GU-3000 series
                ├ GU-3100
                |    Graphic and character display module
                |    Embedded 16x16 & 32x32 dot Japanese font
                |    Stand-alone control by macro function and embedded FROM
                ├ GU-3900
                |    Graphic and character display module
                |    Embedded Japanese, simplified Chinese, traditional Chinese, Korean
                |    font
                |    Stand-alone control by macro function and embedded FROM
                └ GU-3900B
                     Graphic and character display module
                     Upper compatible of GU-3100 and GU-3900
```

3 Applicable item numbers: GU-7000 series as of June. 2012

This application note is applicable for the following item numbers.

Other standard items which are not listed on this list are the same or have similar basic functions.

GU112X16G-7000,	GU112X16G-7000B,	GU112X16G-7003,	GU112X16G-7003B
GU112X16G-7900	GU112X16G-7900B		
GU128X32D-7000,	GU128X32D-7000B,	GU128X32D-7003,	GU128X32D-7003B
GU128X32D-7900,	GU128X32D-7900B,	GU128X32D-7050,	GU128X32D-7950
GU128X32D-7901			
GU128X64D-7000,	GU128X64D-7003,	GU128X64D-7900	
GU128X64F-7000,	GU128X64F-7003,	GU128X64F-7900	
GU128X64F-7900BX			
GU128X128D-7203			
GU140X16G-7000,	GU140X16G-7000B,	GU140X16G-7003,	GU140X16G-7003B
GU140X16G-7040A,	GU140X16G-7900,	GU140X16G-7900B,	GU140X16G-7903
GU140X16J-7000,	GU140X16J-7000B,	GU140X16J-7003,	GU140X16J-7003B
GU140X16J-7900B			
GU140X32F-7000,	GU140X32F-7000B,	GU140X32F-7003,	GU140X32F-7003B
GU140X32F-7032			
GU140X32F-7900,	GU140X32F-7900B,	GU140X32F-7903	
GU140X32F-7050,	GU140X32F-7053,	GU140X32F-7950	
GU144X16D-7053B			
GU256X64C-7000,	GU256X64C-7003,	GU256X64C-7900	
GU256X64D-7000,	GU256X64D-7900		
GU280X16G-7000,	GU280X16G-7003		

Please check our site for the latest product list or contact our sales office.

http:/www.noritake-itron.jp/eng/ (English)

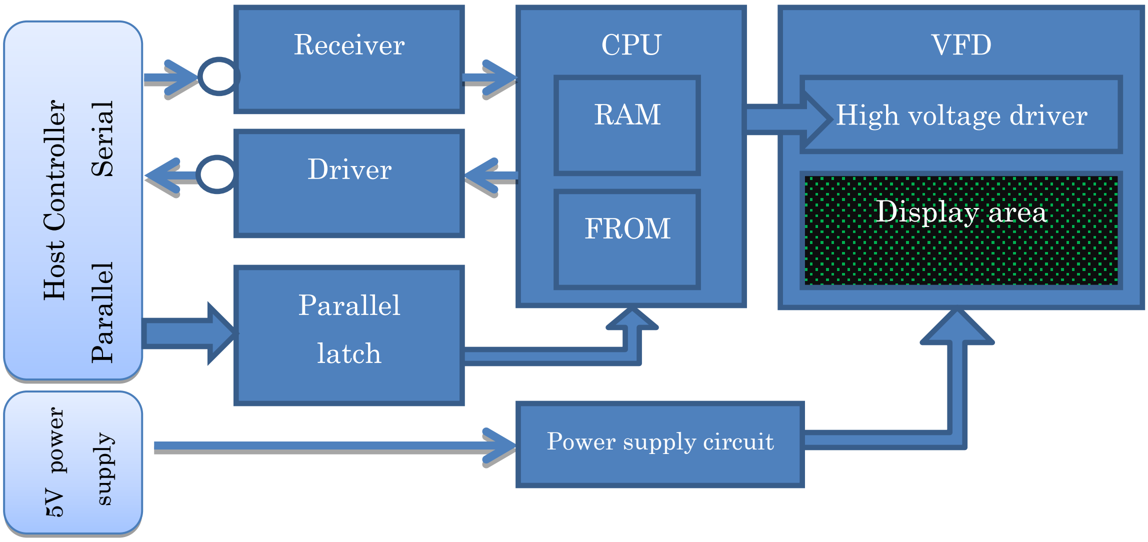
http:/www.noritake-itron.jp/ (Japanese)

4 Hardware

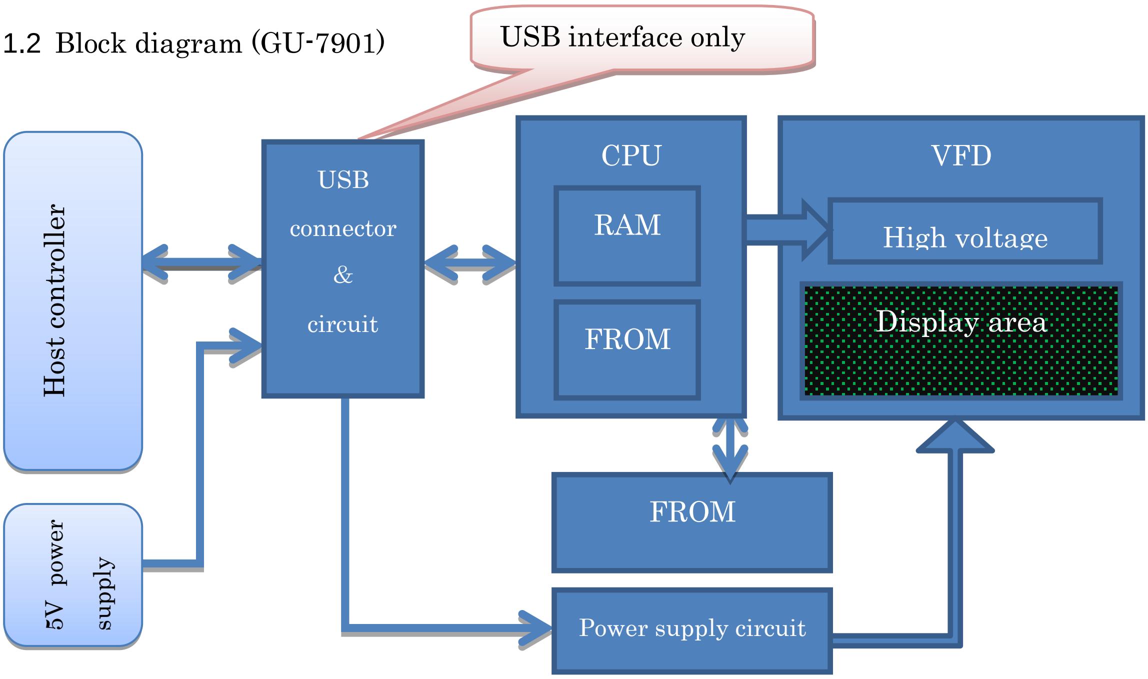
4.1 Block diagram

This module consists of input-output circuit, CPU (control circuit), power supply circuit, and a vacuum fluorescent display tube.

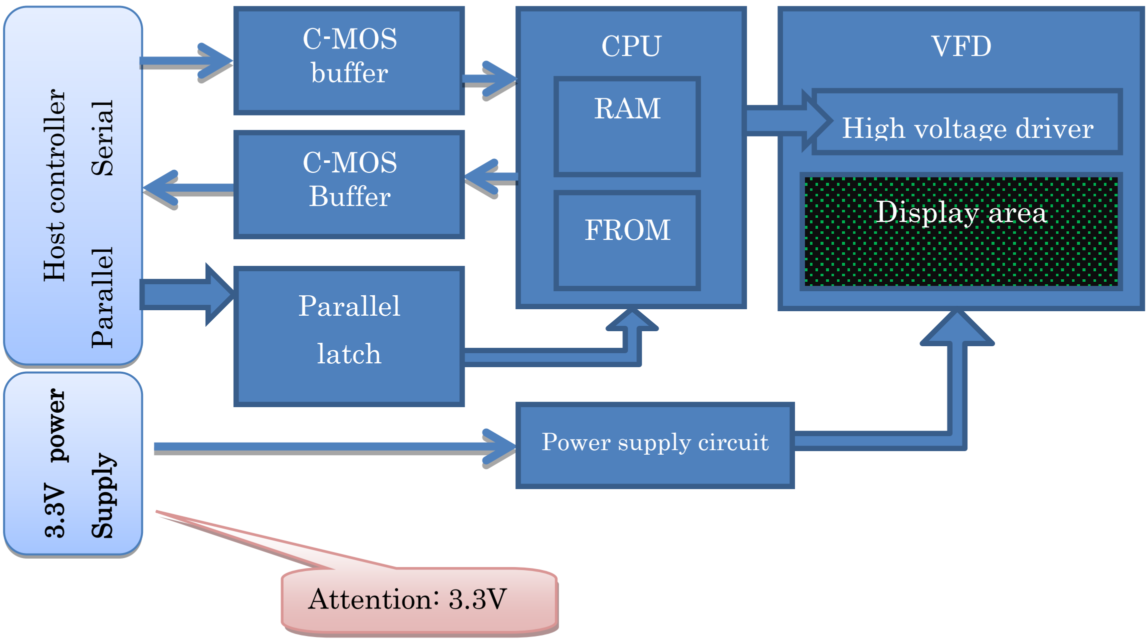
4.1.1 Block diagram (GU-7000)



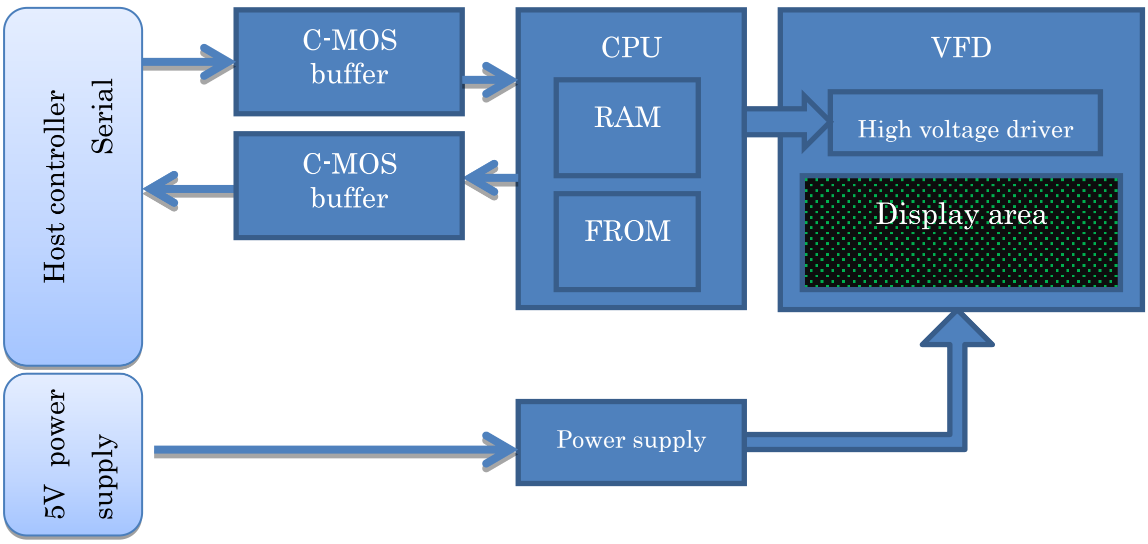
4.1.2 Block diagram (GU-7901)



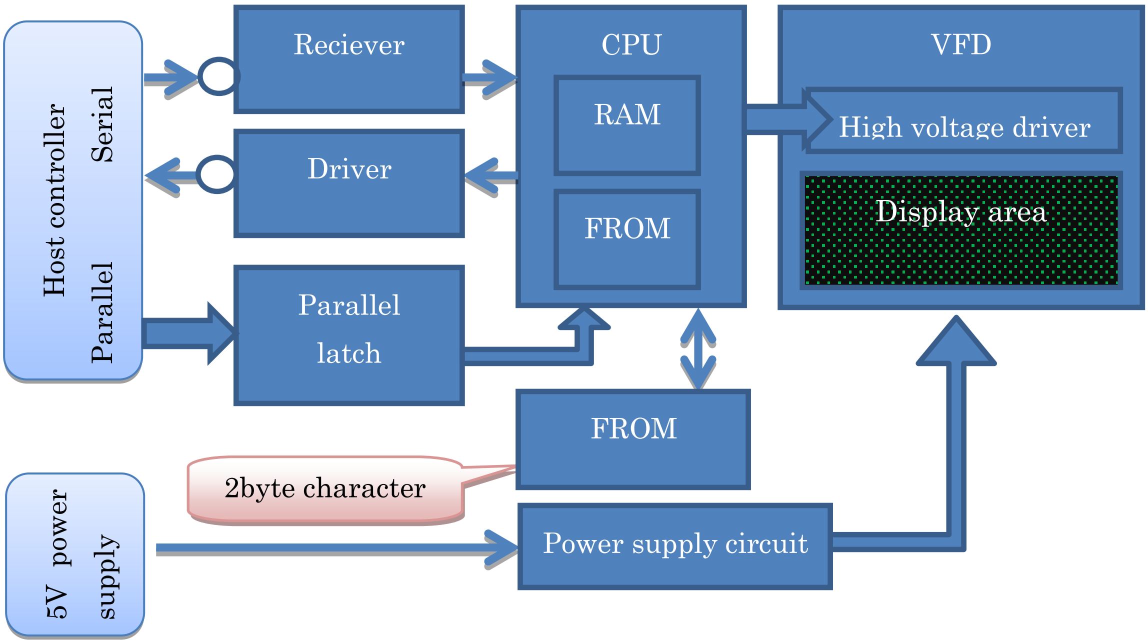
4.1.3 Block diagram (GU7032)



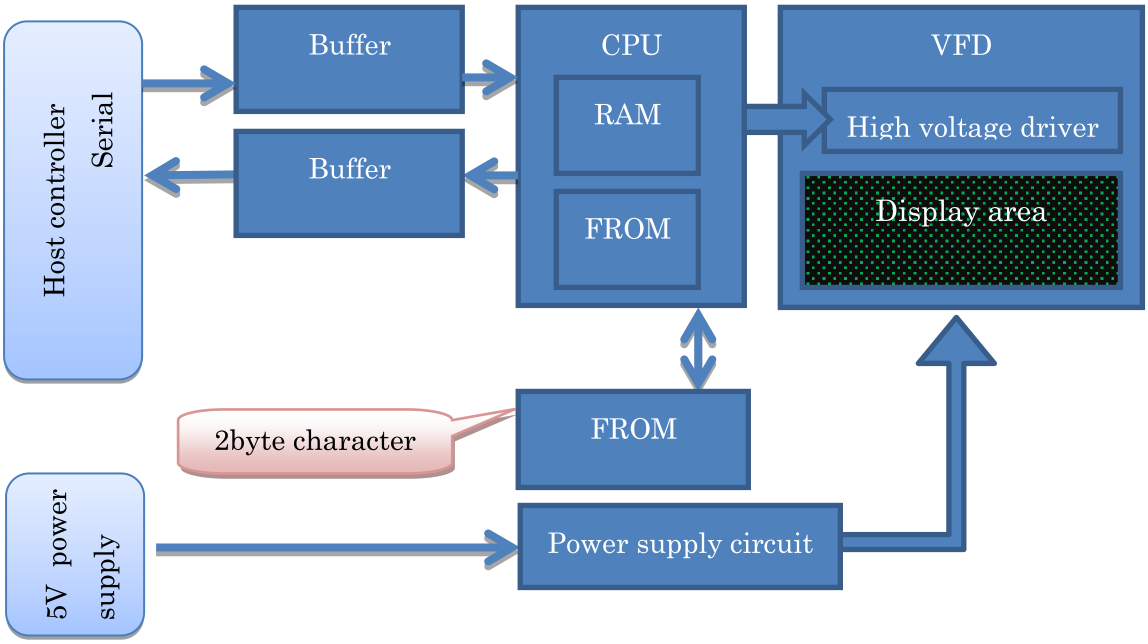
4.1.4 Block diagram (GU-7003)



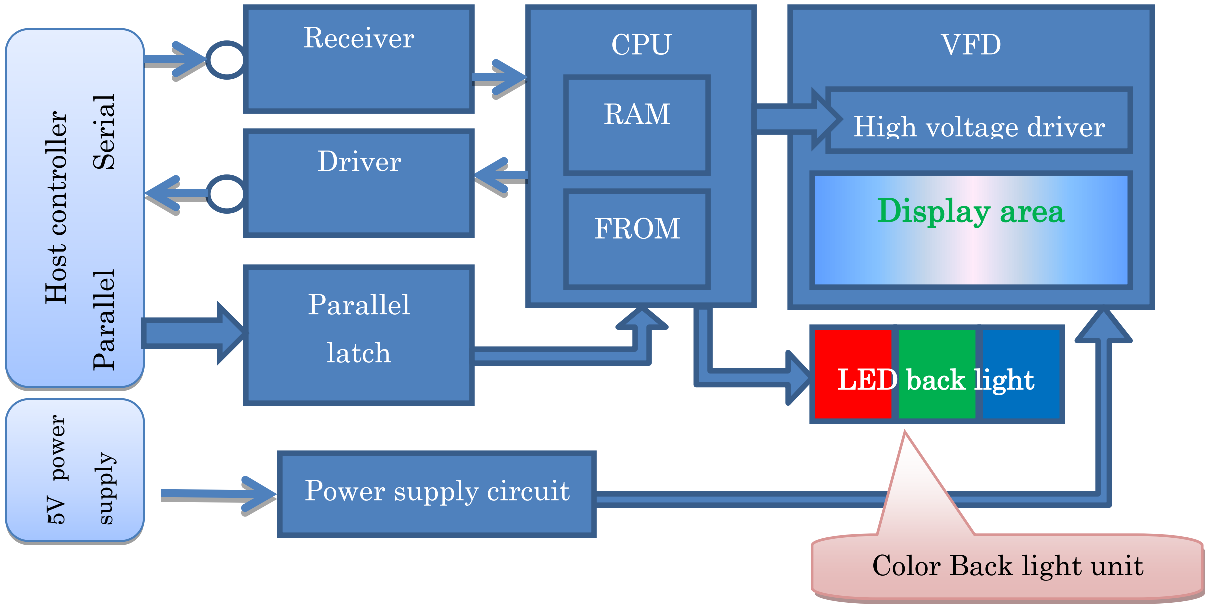
4.1.5 Block diagram (GU-7900)



4.1.6 Block diagram (GU-7903)



4.1.7 Block diagram (GU-7040)



4.2 Connecter

4.3

4.3.1 GU-7xx0

GU-7000 series do not mounted interface connecters on PWB

Please solder a connecter before using.

Please note that GU-7xxx series in not suitable for re-flow soldering.

4.3.1 GU-7901

GU128X32D-7901 includes a USB Mini-B connector and can be connected to a PC directly with the commercial USB cable.

4.3.2 GU-7xx3

GU-7xx3 series do not mounted interface connecters on PWB

Please solder a connecter before using.

Please note that GU-7xxx series in not suitable for re-flow soldering.

4.4 Host interface

The last digit of the part number indicates the interface specification.

0: RS-232C level asynchronous serial input & 8 bit parallel
1: USB
2: C-MOS level synchronous or asynchronous serial input & 8bit parallel
3: C-MOS level synchronous or asynchronous serial input

“0” or “3” of last number are standard.
Please select based on a type of host interface.

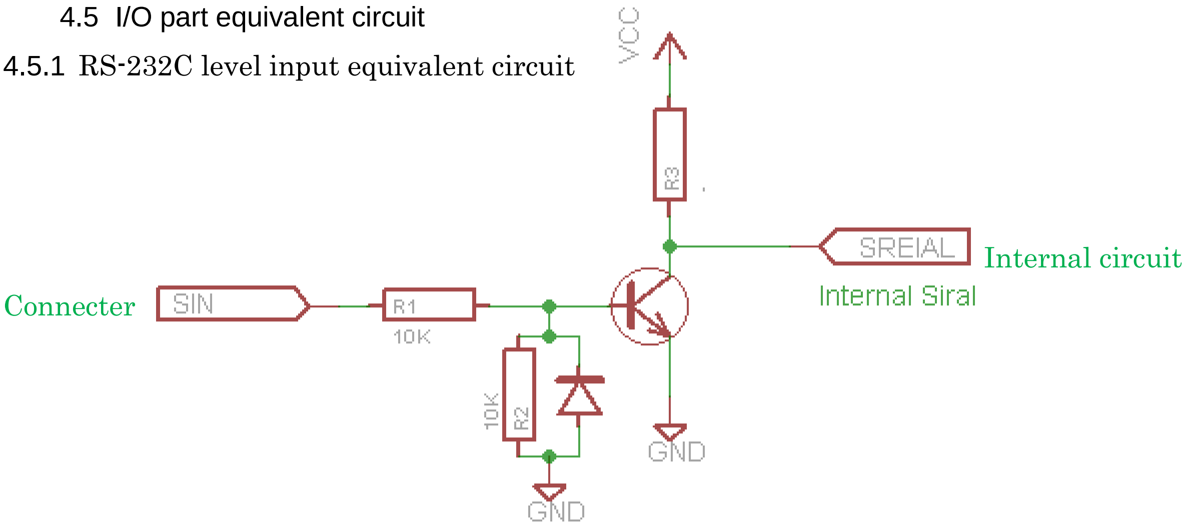
Regarding USB type:

Applicable item: GU-7xx1

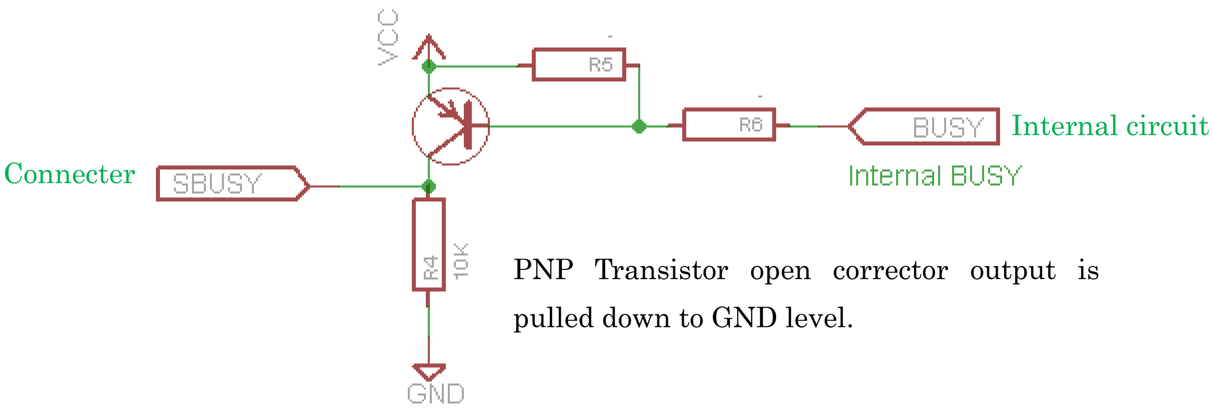
GU128X32D-7901 is a USB interface type. "COMEMO" employs GU128X32D-7901 with a small enclosure. (Currently available in Japan only)

4.5 I/O part equivalent circuit

4.5.1 RS-232C level input equivalent circuit

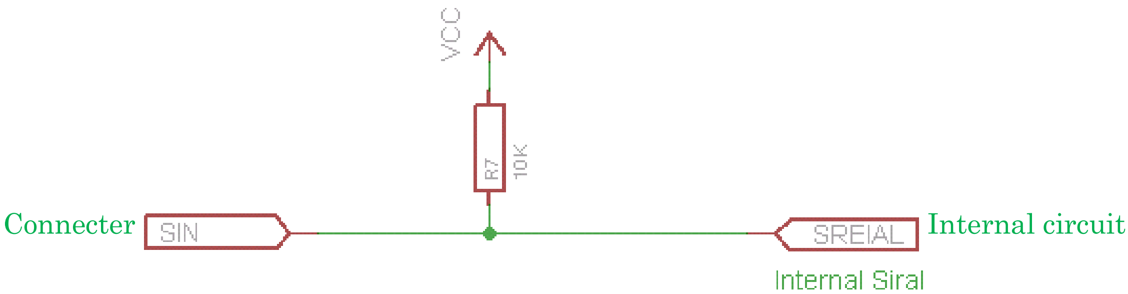


4.5.2 RS-232C level output equivalent circuit

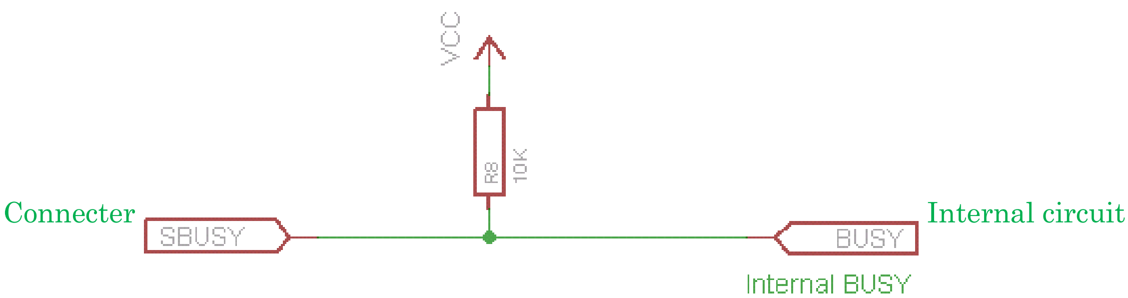


PNP Transistor open corrector output is pulled down to GND level.

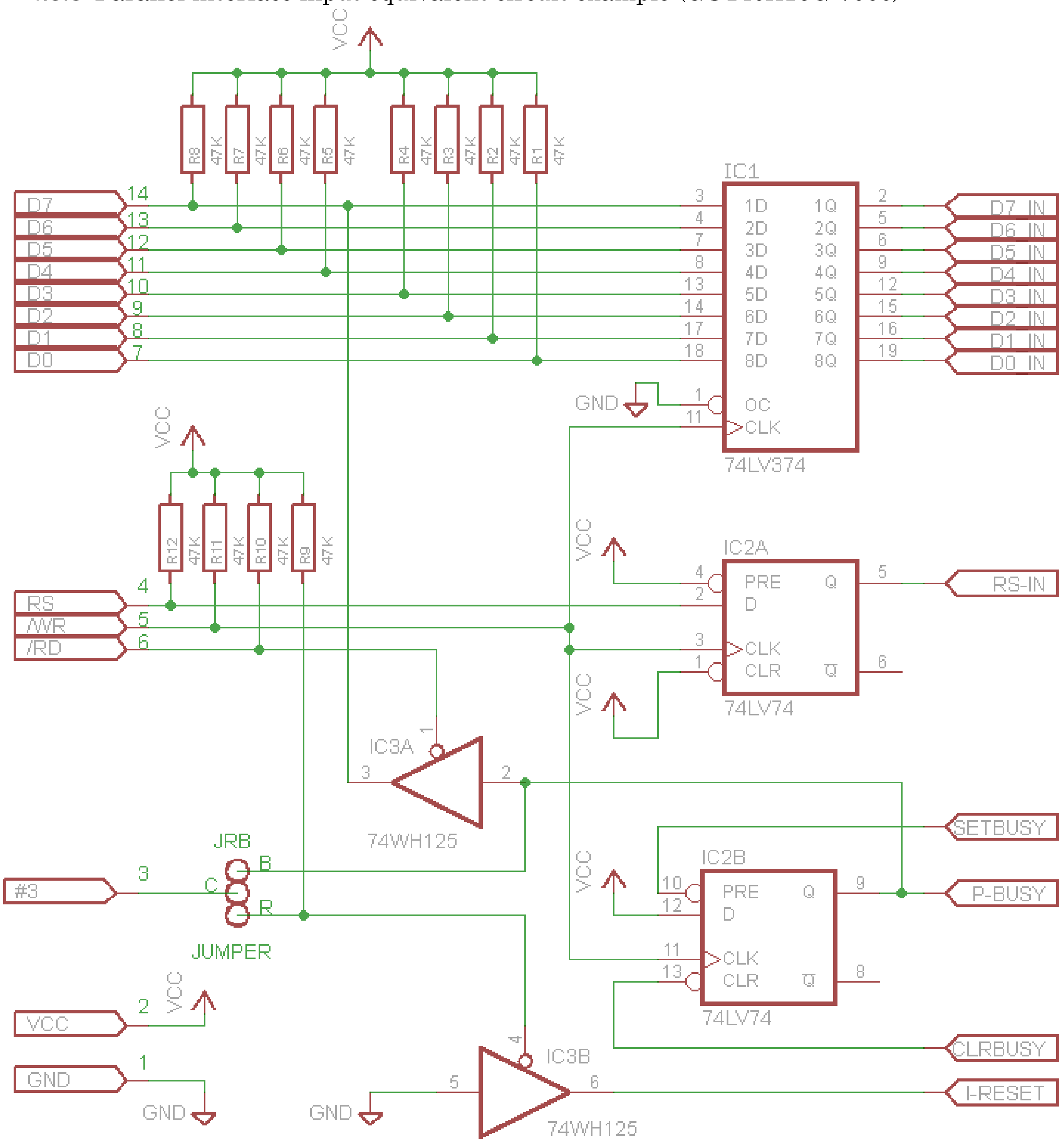
4.5.3 C-MOS level input equivalent circuit



4.5.4 C-MOS level output equivalent circuit



4.5.5 Parallel interface input equivalent circuit example (GU140X16G-7000)



Handling of unused terminals:

Since reset signal is internally pulled up as shown in the equivalent circuit, you may keep input open. However, since it may malfunction in the noise induced to wiring, please avoid running unconnected wiring.

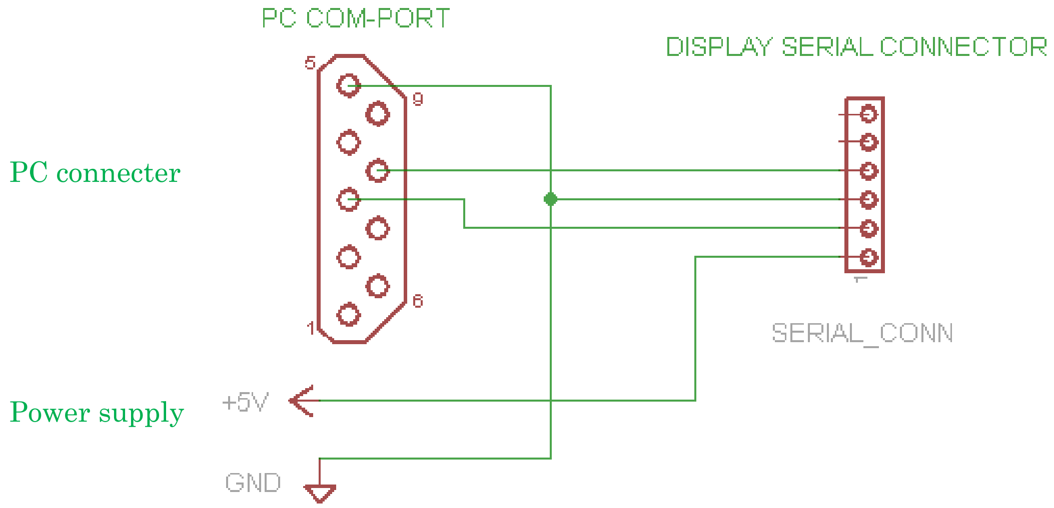
4.6 Examples of host interface connection

In order to use scrolling, waiting and other commands effectively, please use hardware handshake with a BUSY signal. (As noted in the specification)

Note: Interface examples are references only. Please validate the circuit design to meet the host circuit's specification.

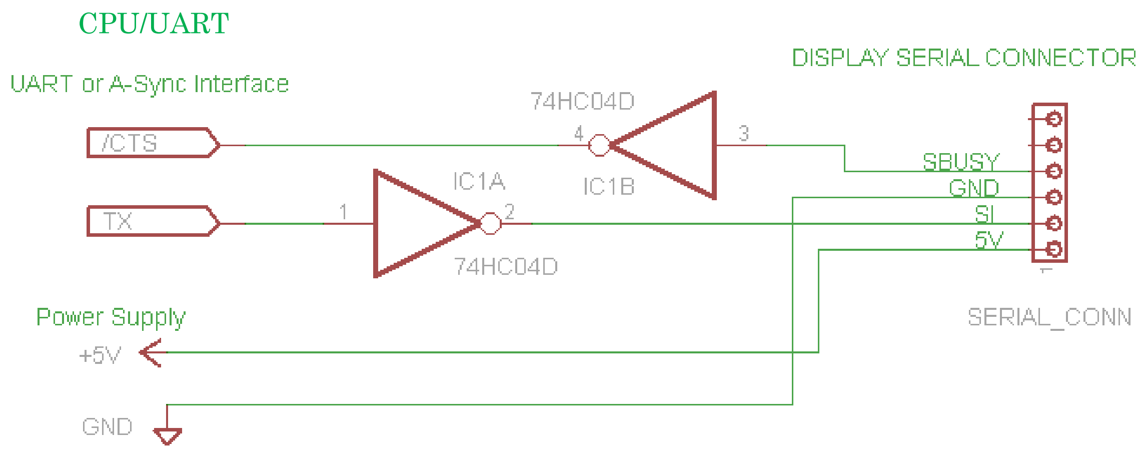
4.6.1 Example of RS-232C serial connection to PC

Applicable model: GU-7xx0



4.6.2 Example 1 of embedded CPU connection by RS-232C

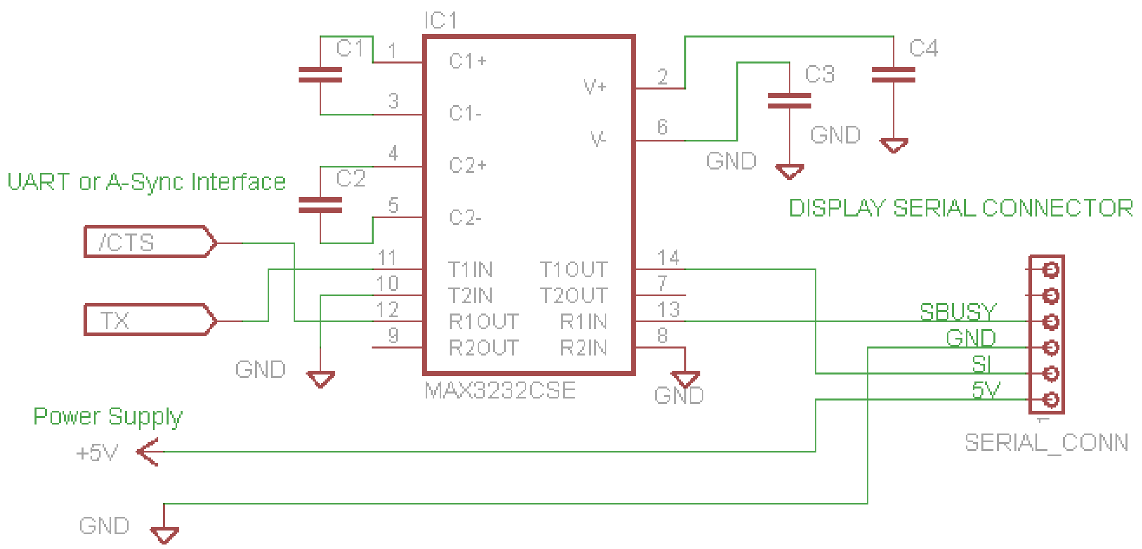
Applicable model: GU-7xx0



Please use a regular input port, when hardware handshake port is not available. And program to check /CTS signal before sending data.

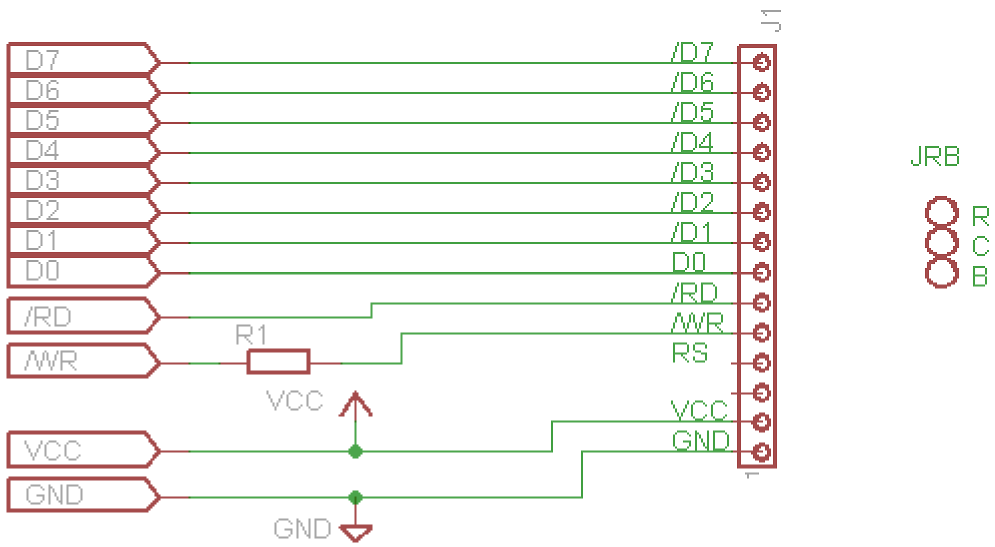
4.6.3 Example 2 of embedded CPU connection by RS-232C

Applicable with GU-7xx0



4.6.4 Example 1 of Parallel input connection

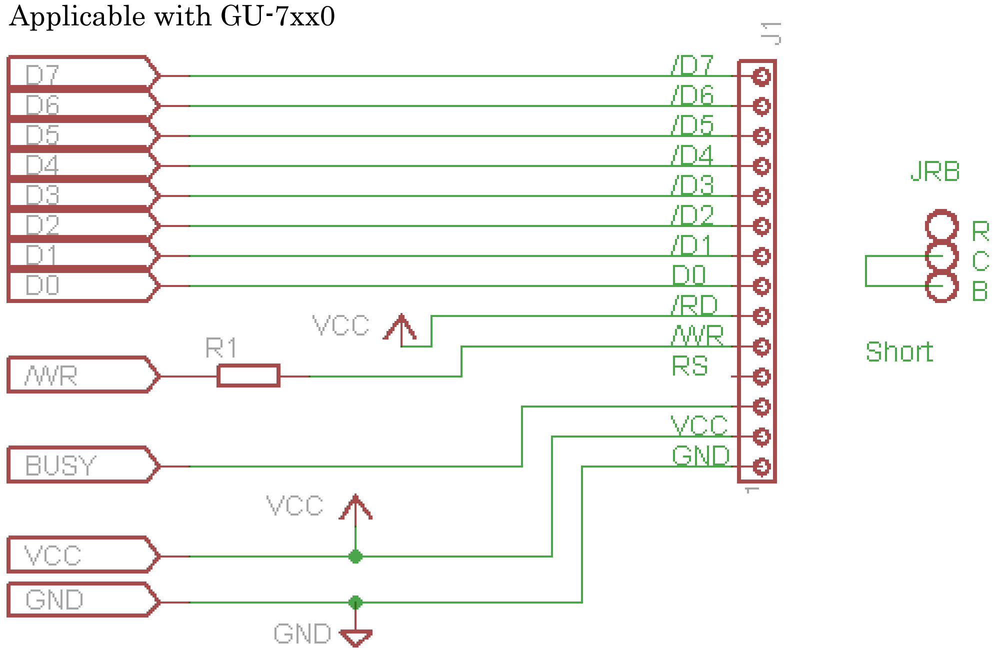
Applicable with GU-7xx0



Notes: About R1, in cases of over or under shooting on the /WR signal, adding R1 (50 to 200 ohms) might reduce the risk of malfunctioning.

4.6.5 Example 2 of Parallel for output BUSY signal connection

Applicable with GU-7xx0

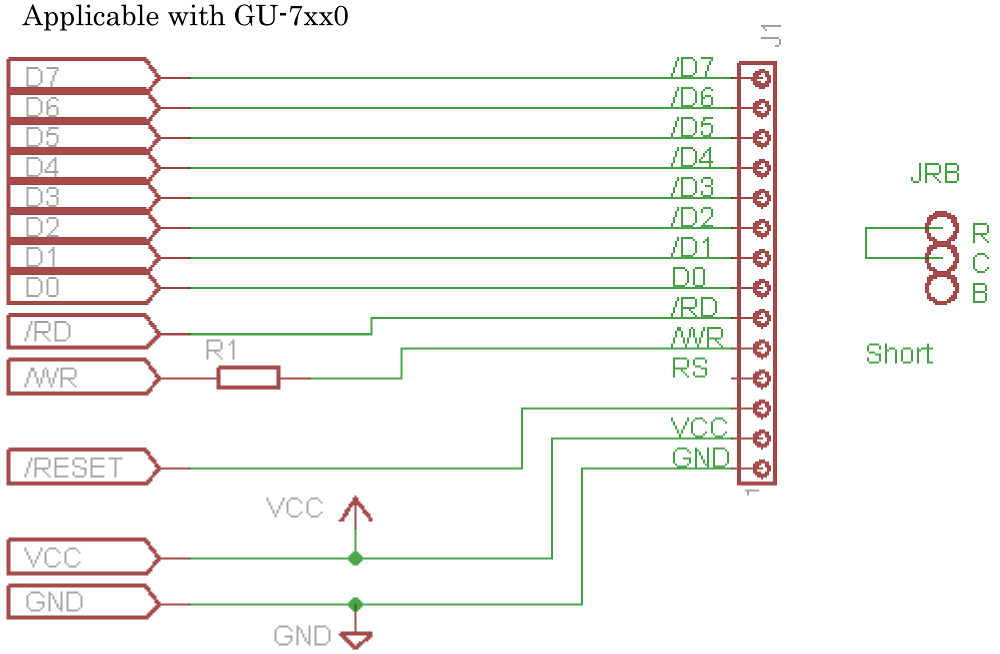


Notes: About R1, in cases of over or under shooting on the /WR signal, adding R1 (50 to 200 ohms) might reduce the risk of malfunctioning.

Program code example is available at "5.10".

4.6.6 Example 3 of Parallel using RESET signal

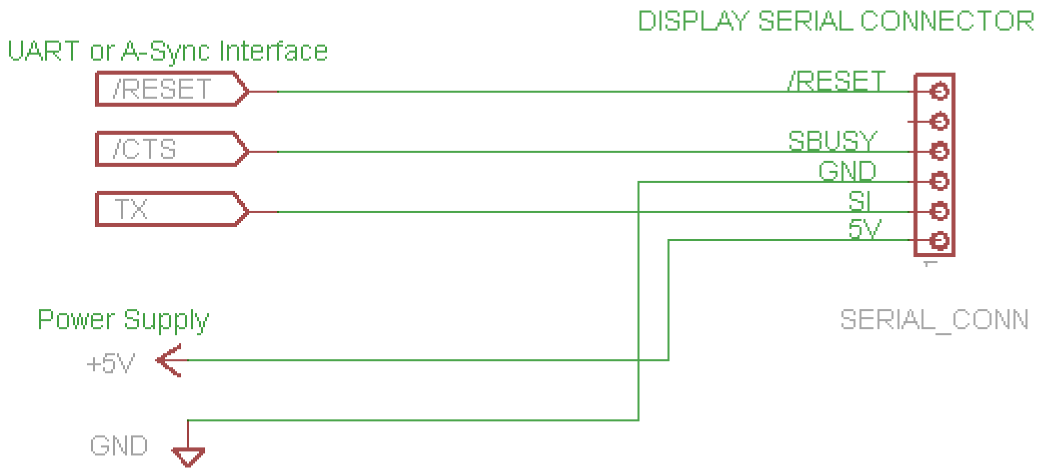
Applicable with GU-7xx0



Notes: About R1. in cases of over or under shooting on the /WR signal, adding R1 (50 to 200 ohms) might be reduce the risk of malfunctioning.

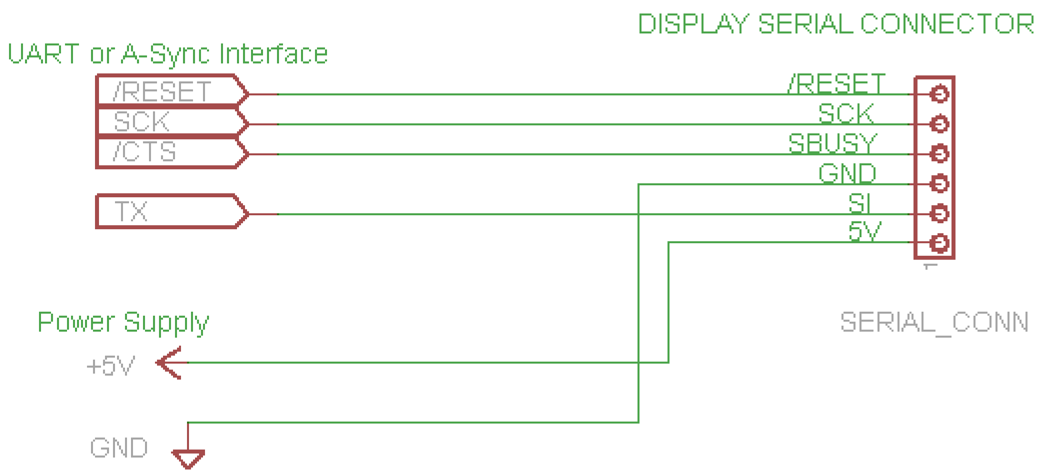
4.6.7 Example of embedded CPU connection by Asynchronous C-MOS serial

Applicable with GU-7xx3



4.6.8 Example of embedded CPU connection by Synchronous C-MOS

Applicable with GU-7xx3



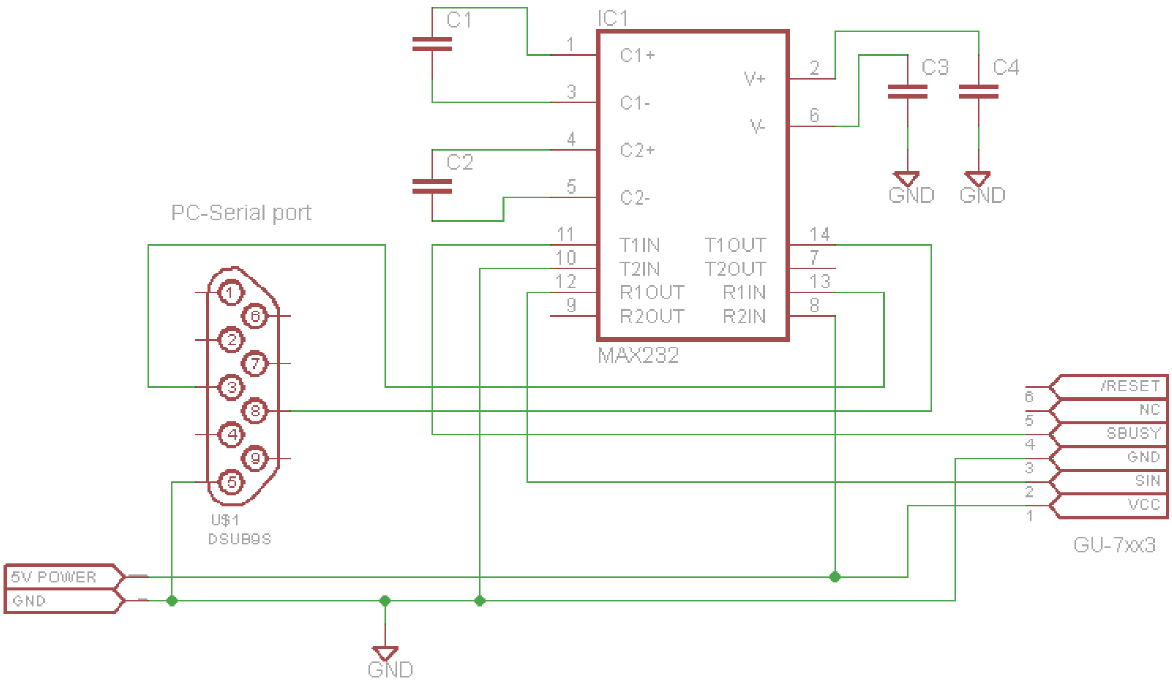
4.6.9 Output of BUSY signal and additional input of hardware reset

Applicable with GU-7xxx

Connecting the JBR jumper set the parallel connector No.3 pin as "BUSY OUT" or "/RESET IN".

4.6.10 Example of connection to PC by Asynchronous C-MOS serial

Applicable with GU-7xx3



5 Software

5.1 Default setting and input Protocol

A communication protocol is non-procedural with hardware handshake. The display module can use a basic function with default settings by the internal power on reset without any special initialize command.
A few seconds after turning the module on, try to send a character code to the display. The display accepts popular ASCII codes. Commands for using various functions use the extended sequence which starts in ESC code. Please perform the write control by hardware handshake at write timing.

In order to understand commands and functions of the GU-7000 series, please try "GUD-10K" as the development support tool. Since GUD-10K has a tutorial navigator, user can experiment major functions. GUD-10K software is available at our home page in Japan, or contact local sales office.

5.2 Display memory (RAM)

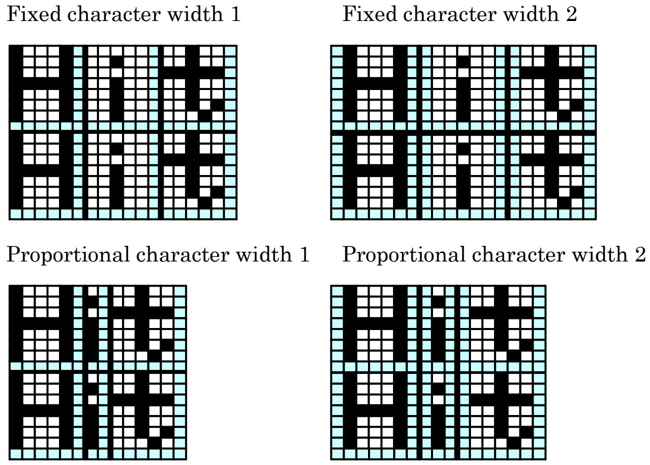
Written data and commands are translated into graphic images, and stored in the graphic RAM. Since the size of graphic RAM is larger than the size of display area, then a part of the RAM data appear on screen. In other words, graphic RAM is divided into the display area and hidden area.

RAM

The hidden area can be used as working area to prepare images, and help display functions such as picture scrolling.

5.3 Proportional font

The proportional font command is a function to narrow a character pitch. You can increase the average number of characters that can be displayed within the same area.



5.4 Embedded font tables (7xxxB are the same as 7xxx)

Item number	Embedded font
GU112X16G-7000	5x7dot ANK & International
GU112X16G-7003	5x7dot ANK & International
GU112X16G-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU128X32D-7000	5x7dot ANK & International
GU128X32D-7003	5x7dot ANK & International
GU128X32D-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU128X32D-7901	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU128X32D-7050	5x7dot ANK & International
GU128X32D-7950	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU128X64D-7000	5x7dot ANK & International
GU128X64D-7003	5x7dot ANK & International
GU128X64D-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU128X64F-7000	5x7dot ANK & International
GU128X64F-7003	5x7dot ANK & International
GU128X64F-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU128X128D-7203	6x8, 8x16, 12x24, 16x32dot ANK & International
GU140X16G-7040A	5x7dot ANK & International
GU140X16G-7000	5x7dot ANK & International
GU140X16G-7003	5x7dot ANK & International
GU140X16G-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU140X16G-7903	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU140X16J-7000	5x7dot ANK & International
GU140X16J-7003	5x7dot ANK & International
GU140X16J-7900B	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU140X32F-7000	5x7dot ANK & International
GU140X32F-7003	5x7dot ANK & International
GU140X32F-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU140X32F-7903	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU140X32F-7032	5x7dot ANK & International
GU140X32F-7050	5x7dot ANK & International
GU140X32F-7053	5x7dot ANK & International
GU140X32F-7950	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU256X64C-7000	5x7dot ANK & International
GU256X64C-7003	5x7dot ANK & International
GU256X64C-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU256X64D-7000	5x7dot ANK & International
GU256X64D-7900	5x7dot ANK & International, 16x16dot Japanese, Simplified/Traditional Chinese, Korean
GU280X16G-7000	5x7dot ANK & International
GU280X16G-7003	5x7dot ANK & International

5.5 Font table

A concept is shown here. Please refer to the specifications for entire font data.

5.5.1 5x7dot ANK (1byte character)

<table>
<tr><th></th><th>0_H</th><th>1_H</th><th>2_H</th><th>3_H</th><th>4_H</th><th>5_H</th><th>6_H</th><th>7_H</th><th>8_H</th><th>9_H</th><th>A_H</th><th>B_H</th><th>C_H</th><th>D_H</th><th>E_H</th><th>F_H</th></tr>
<tr><td>_0H</td><td colspan="2" rowspan="16">Command</td><td colspan="6" rowspan="16">Common Font area
ASCII</td><td colspan="8" rowspan="16">Extend Font area
Select by ESC t n</td></tr>
<tr><td>_1H</td></tr>
<tr><td>_2H</td></tr>
<tr><td>_3H</td></tr>
<tr><td>_4H</td></tr>
<tr><td>_5H</td></tr>
<tr><td>_6H</td></tr>
<tr><td>_7H</td></tr>
<tr><td>_8H</td></tr>
<tr><td>_9H</td></tr>
<tr><td>_AH</td></tr>
<tr><td>_BH</td></tr>
<tr><td>_CH</td></tr>
<tr><td>_DH</td></tr>
<tr><td>_EH</td></tr>
<tr><td>_FH</td></tr>
</table>

Single-byte characters are divided into three groups.

00Hex~1FHex: Functions have been assigned. For example, when 0DHex (CR) is written in, the cursor position will move to the left end of a screen.

20Hex~7FHex: Alphanumeric character font area of ASCII conformity.

80Hex~FFHex: Extended font area, one font table out of 10 can be chosen by writing a "ESC t n" command. "ESC t n" command does not affect to characters which written before the command.

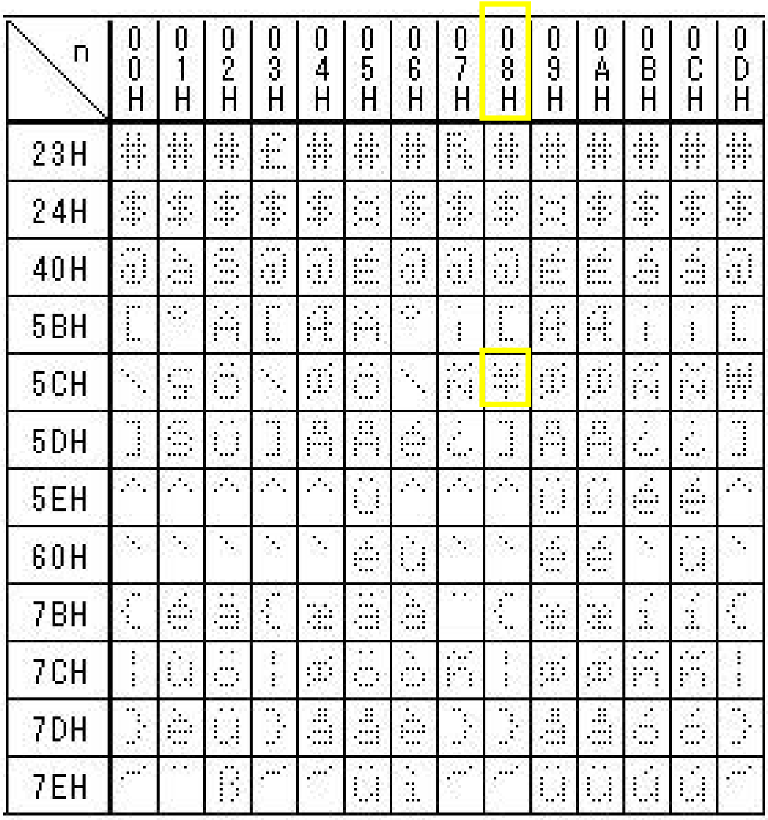
n	Character set
0	PC437(USA: Standard Europe)
1	Katakana
2	PC850(Multilingual)
3	PC860(Portuguese)
4	PC863(Canadian-French)
5	PC865(Nordic)
16	WPC1252
17	PC866(Cyrillic #2)
18	PC852(Latin 2)
19	PC858

5.5.2 International font set

International font set command replace some fonts with designated fonts in basic font area (20Hex~7FHex). For example, a command, “ESC R n=08h (Japan)” replace “ \ “ with “ ¥ “.

Command code is “ESC R n”

n	Language
0	USA
1	France
2	Germany
3	England
4	Denmark 1
5	Sweden
6	Italy
7	Spain1
8	Japan
9	Norway
10(0AH)	Denmark2
11(0BH)	Spain2
12(0CH)	Latin America
13(0DH)	Korea



Characters written before this command are not affected.

5.5.3 16x16dot JIS, Simplified Traditional Chinese, Korean (GU-79xx only)

(2byte character)

GU-79xx model has 16x16 dot bitmap font.

These characters have 2 byte code, please send character codes after the following command. Because displayed characters are not changed by this command, you can display several languages at same time.

Example display 2byte characters

Example of Japanese

1FH, 28H, 67H, 01H, 02H	‘ 8x16 dot font size select
1FH, 28H, 67H, 02H, 01H	‘ set 2byte character mode
1FH, 28H, 67H, 0FH, 00H	‘ Japanese
88H, A2H	‘ Displaying “阿”

Example of Korean

1FH, 28H, 67H, 01H, 02H	‘ 8x16 dot font size select
1FH, 28H, 67H, 02H, 01H	‘ set 2byte character mode
1FH, 28H, 67H, 0FH, 01H	‘ Korean
B0H, A1H	‘ Displaying “가”

Example of Simplified Chinese

1FH, 28H, 67H, 01H, 02H	‘ 8x16 dot font size select
1FH, 28H, 67H, 02H, 01H	set 2byte character mode
1FH, 28H, 67H, 0FH, 02H	‘ Simplified Chinese
B0h, A1H	‘ Displaying “啊”

Example of Traditional Chinese

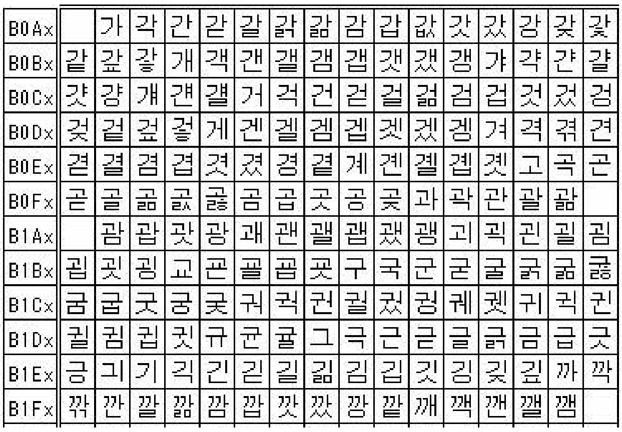
1FH, 28H, 67H, 01H, 02H	‘ 8x16 dot font size select
1FH, 28H, 67H, 02H, 01H	‘ set 2byte character mode
1FH, 28H, 67H, 0FH, 03H	‘ Traditional Chinese
A4h, 41H	‘ Displaying “乙”

The code ranges of each language are as follows;

Font	Standard	2byte code area
Japanese JIS Kanji	JISX208 (Shift-JIS)	8140H~9FF0H, E040~EFFCH
Korean Hangul	KSX5601-87	A1A1H~FEFEH
Simplified Chinese	GB2312-80	A1A1H~FEFEH
Traditional Chinese	Big-5	A140H~FEFEH

Example of JIS font

Example of Korean font



Example of Simplified Chinese font

Example of Traditional Chinese font

Please refer to the specifications for the complete font tables.

5.6 Command tables

5.6.1 Common command table1 for GU-7xxx series

Name	Code	Function
BS	08h	Cursor moves to the left by one character.
HT	09h	Cursor moves to the right by one character.
LF	0Ah	Cursor moves to next lower line.
HOM	0Bh	Cursor moves to home position (top left).
CR	0Dh	Cursor moves to left end of current line.
CLR	0Ch	Display screen is cleared and cursor moves to home position.
ESC	1Bh,,	Start extended command sequence.
US	1Fh	Start extended command sequence.

5.6.2 Common command table2 Expand command sequences for GU-7xxxx

Name	Code	Function
Initialize	1Bh 40h	Settings return to default values. Jumper settings are not re-loaded.
Cursor set	1Fh 24h xL xH yL yH	Cursor moves to the specified (X, Y) position on Display Memory.
Cursor display	1Fh 43h n	Cursor display setting. n = 00h: Cursor display OFF (Default) n = 01h: Cursor display ON
Font Width	1Fh 28h 67h 03h w	Set font width for 1byte characters. w = 00h: 1 dot space on right side w = 01h: 1 dot space on right side and left side w = 02h: 1 dot space on right side w = 03h: 1 dot space on right side and left side
Font Magnification	1Fh 28h 67h 40h x y	Set character magnification 'x' times to the right and 'y' times downward.

Download character definition	1Bh 26h a C1 C2 [data]	A maximum of 16 download characters can be defined. To display downloaded characters, Download character ON command is required. If a currently-displayed download character is re-defined, there is no effect on the currently-displayed character. It is effective only for newly entered characters.
Download character delete	1Bh 3Fh a c	Delete defined download character.
Download character ON/OFF	1Bh 25h n	Enable or disable display of download characters. Changing this setting only affects subsequent data. Contents already displayed are not affected. n = 01h: Enable (ON) n = 00h: Disable (OFF)
International font set	1B 52h n	Character code of parts of 20h~7Fh is replaced for each country. Characters already displayed are not affected.
Character table type	1Bh 74h n	Character code of 80h~FFh is replaced for each font tables. Characters already displayed are not affected.
Over-write mode	1Fh 01h	Display mode set to Over-write mode.
Vertical scroll mode	1Fh 02h	Display mode set to Vertical scroll mode.
Horizontal scroll mode	1Fh 03h	Display mode set to Horizontal scroll mode.
Horizontal scroll speed	1Fh 73h n	Set speed for Horizontal scroll mode. It takes (T * (n-1)) msec to move one dot. n=0 : Fastest, n=1 : T/2 msec. T is approximately 10~20 msec and changes with item numbers.
Reverse display	1Fh 72h n	Reverse display ON/OFF for character and image display. Changing this setting only affects subsequent data. Content already displayed is not affected.
Write mixture display mode	1Fh 77h n	Newly-written characters and images are combined with current display contents in Display Memory. n = 00h: Over write (Default) n = 01h: OR display write n = 02h: AND display write n = 03h: EX-OR display write

Brightness level setting	1Fh 58h n	Set display brightness level. Brightness: n/8, n=1~8, n=8: 100%
Wait	1Fh 28h 61h 01h t	Waits for the specified time. (Command and data processing is stopped) Wait time = t × approximately 0.5s, n=0~255
Scroll display action	1Fh 28h 61h 10h wL wH cL cH s n	Shift the display screen. It becomes a scroll action by setting up two or more shifts.
Blink	1Fh 28h 61h 11h p t1 t2 c	Blink display action. Blink pattern specified by 'p'.
Screen saver	1Fh 28h 61h 40h p	Control Power ON or OFF, and Start Screen saver mode. This Screen saver mode setting is cancelled when next data is written. p = 00h: Display power OFF (Power save mode) p = 01h: Display power ON p = 02h: All dot OFF p = 03h: All dot ON p = 04h: Repeat blink display with normal and Reverse display
User Window define / cancel	1Fh 28h 77h n a b [Window address]	Define or cancel User-Window. Display contents are not changed by this command.
Window select	1Fh 28h 77h 01h a	A current window is chosen from one of the user windows and base window. a=0 : Base window(all area) a=1 to 4 : User window 1 to 4
Window select shortcut	10h or 11h or 12h or 13h or 14h	Contracted form of current window select, using just 1byte.
Write screen mode select	1Fh 28h 77h n a	Set base window area mode. a=0 : Set display area as base window. a=1 : Set all RAM area as base window. A cursor moves in the range of a base window.
Real-time bit image display	1Fh 28h 66h 11h xL xH yL yH g d1…dk	Display the bit image data at the cursor position in real-time.

5.6.3 Command table3 Expand command sequence to control FROM for GU-79xxx

Caution1: Please minimize the use of this command, due to the write/erase life time of FROM.
Caution2: Never turn OFF the power supply in rewrite mode, because data and firmware stored in F-ROM could be damaged.
Caution3: Do not use FROM rewrite command [R k n d(1) … d(32768)] and FROM erase command [C k n], these are for factory use only.

Name	Code	Function
Memory re-write mode	1Ch 7Ch 4Dh D0h d1…d6	Shift to "Memory re-write mode" from "Normal mode".
FROM bit image definition	42h k n d1…d32768	Define user bit image to the FROM. This command is only valid in Memory re-write mode.
FROM SUM compare	53h k d1…d4 dm	Compare SUM of FROM re-write data (FROM bit image definition data) with d1 – d4. If SUM is not equal to d1 – d4, an error message appears on the display screen, and BUSY signal at writing data is kept BUSY for 2 sec.
Memory re-write mode END	45h k	End Memory re-write mode and return to normal mode with Initialize Display command.

5.6.4 Command table4 Expand command sequence of 2 bytes character for GU-79xx

Name	Code	Function
Font size select	1Fh 28h 67h 01h m	Sets the font size for 1-byte characters. m = 1h : 5x7dot 2h : 8x16dot
2-byte character	1Fh 28h 67h 02h m	Sets 2-byte character ON/OFF. m = 0h : OFF 1h : ON
2-byte character type	1Fh 28h 67h 0Fh m	Sets 2-byte character type. m = 0h : Japanese 1h : Korean 2h : Simplified Chinese 3h : Traditional Chinese
Downloaded bit image display	1Fh 28h 66h 10h m aL aH aE ySL ySH xL xH yL yH g	Display the defined FROM bit image at cursor position.

5.6.5 Command table5 Additional command sequences for GU-7xxxB Display by dot
(Additional commands for GU-7000B, 7900B series, not for GU-7000, 7900)

The GU-7000B series can specify address by dot (pixel) although the GU-7000 series specifies the vertical address by 8 dot unit.

Name	Code	Function
Dot unit downloaded bit image display	1Fh 28h 64h 20h xPL xPH yPL yPH m aL aH aE ySL ySH xOL xOH yOL yOH xL xH	GU-7000B: Display the bit image defined in Display Memory at the specified (x,y) position. Display position, display size, and image data offset are specified in unit of 1 dot. GU-7900B: Display the bit image defined in FROM or Display memory at the specified (x,y) position. Display position, display size, and image data offset are specified in unit of 1 dot.
Dot unit real-time bit image display	1Fh 28h 64h 21h xPL xPH yPL yPH xL xH yL yH g d(1)…d(k)	Display the bit image data at the specified (x,y) position in real-time manner. Display position and display size are specified in units of 1 dot.
Dot unit character display	1Fh 28h 64h 30h xPL xPH yPL yPH m bLen d(1)…'d(bLen)	Display the specified text characters at the specified (x,y) position. Display position is specified in units of 1 dot. For display position xP=FFFFh, write position continues from previous writes done using this command. Character magnification and bold settings are not used.

5.7 Moving cursor and display mode

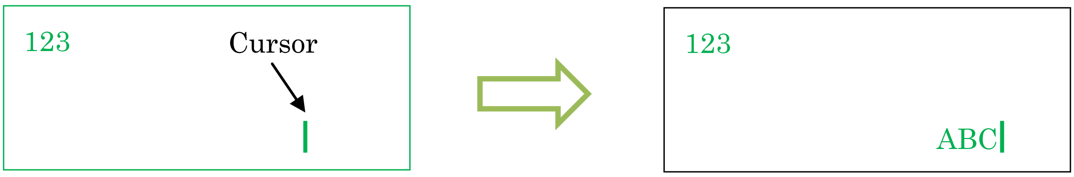
This section describes the display mode of extended sequence commands.

Display mode specifies how to move a cursor at end of the line..

Over-write mode	1Fh 01h	Over-writes, or replaces, existing data.
Vertical scroll mode	1Fh 02h	Scrolls cursor up 1 line
Horizontal scroll mode	1Fh 03h	Scrolls cursor horizontally 1 space

When a character is written, the character is displayed at the cursor position, and the cursor moves forward one character. For example, a motion of a cursor when written as "ABC" is as follows;

Before written in After written in



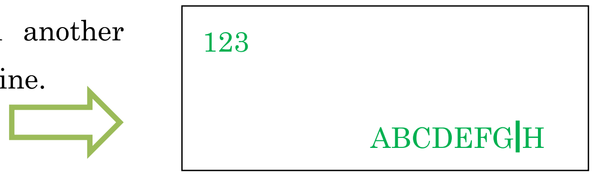
Display mode command affects next write-in operation.

Operation in each mode when written as "DEFGH" is as follows;

Overwrite mode: Return to the upper left and “FGH” is over-written on “123”.

Vertical scroll mode: Scroll up entire screen to make lowest line empty, then “FGH” is written to the bottom line.

Horizontal scroll mode: Scroll the line to the left. Scrolling speed can be specified in another command. Scrolling will occur on any line.



5.8 Program examples of Microsoft Visual Studio 2010 on Windows PC

5.8.1 Connect to serial port by Visual C# 2010

This is a sample program to display characters on GU-7000 series modules.

This sample can be run in C#2010 Express Edition.

First, install the C # 2010 from the site, Microsoft Corp.

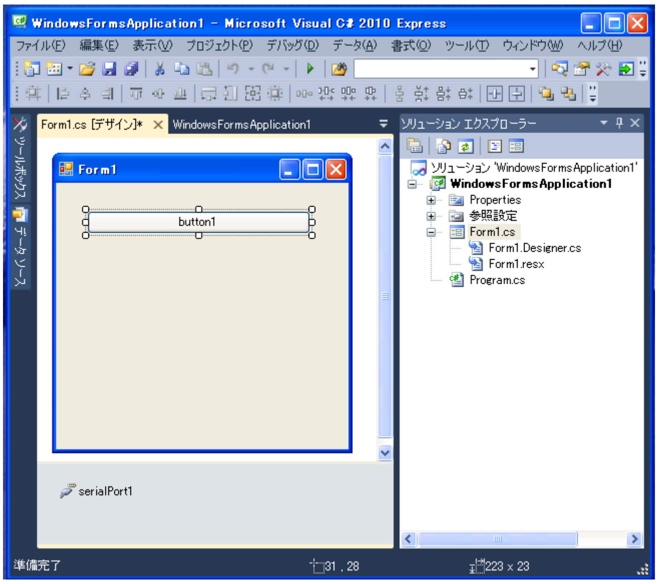
"New project" > double click on "Windows form application" to make a new application.

Double-click "Serial Port" in the tool box.

Double-click "Button" in the tool box.

There is a button in the upper left corner of the window of "Form1.cs". Move its button in the center.

The right hand figure is a screen image.



When you double-click Form1 dialog beside button1, it comes to be able to enter the definition of "Form1_Load". Then add serial port settings as follows.

Change COM3 to a suitable serial port number according to the system you want to use. Visual studio generates gray colored lines already.

------ Start of list -----------

```
private void Form1_Load(object sender, EventArgs e)
        {
            this.serialPort1.PortName = "COM3";
            this.serialPort1.BaudRate = 38400;
            this.serialPort1.DataBits = 8;
            this.serialPort1.StopBits = System.IO.Ports.StopBits.One;
            this.serialPort1.Parity = System.IO.Ports.Parity.None;
            this.serialPort1.DiscardNull = false;
            this.serialPort1.Handshake = System.IO.Ports.Handshake.RequestToSend;
            if (this.serialPort1.IsOpen) { this.serialPort1.Close(); }
        }
```

------ End of list -----------

Click the [design] tab of "Form1.cs" so that it returns to the design screen.

When Button1 is double-clicked, the processing input of button1_click will be added. After that, type the program code in as follows;

------ Start of list -----------

```
private void button1_Click(object sender, EventArgs e)
{
    this.serialPort1.Open();
    if (this.serialPort1.IsOpen)
    {
        this.serialPort1.Write("Hello World");
        this.serialPort1.Close();
    }
}
```

------ End of list -----------

The whole program becomes like as follows;

------ Start of list -----------

LIST C# sample program

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            this.serialPort1.PortName = "COM3"; //Change port name according to the system
            this.serialPort1.BaudRate = 38400;
            this.serialPort1.DataBits = 8;
            this.serialPort1.StopBits = System.IO.Ports.StopBits.One;
            this.serialPort1.Parity = System.IO.Ports.Parity.None;
            this.serialPort1.DiscardNull = false;
            this.serialPort1.Handshake = System.IO.Ports.Handshake.RequestToSend;
            if (this.serialPort1.IsOpen) { this.serialPort1.Close(); }
        }
        private void button1_Click(object sender, EventArgs e)
        {
            this.serialPort1.Open();
```

Check Reference *

```
                if (this.serialPort1.IsOpen)
                {
                    this.serialPort1.Write("Hello World");
                    this.serialPort1.Close();
                }
            }
        }
}
```

----- End of list -----------

*: Please change COM3 to the port name of use.

```
this.serialPort1.PortName = "COM3";
```

Press the F5 key when an input finishes. After a while, compiling will finish. Form1 window should be opened, then click "Button1". "Hello World" should be displayed on the VFD module."

5.9 Program examples

The previous example explained how to display "Hello World" using C #.
This next example will show the various functions written in C #.

5.9.1 Clear display

```
private void ClearScreen()
{
            byte[] bb = new byte[1];
            bb[0] =(byte) 0x0c;

            this.serialPort1.Open();
            if (this.serialPort1.IsOpen) {
                this.serialPort1.Write(bb, 0, 1);
                this.serialPort1.Close(); }

}
```

5.9.2 Moving Cursor

A cursor moves to (int X, int Y). Please note that Y is in bytes.

```
/* Move Cursot to (X, Y).   Y is in Byte. */
        private void moveCursor(int X, int Y)
        {

            byte[] bb = new byte[6];

            bb[0] = (byte)0x1f;
            bb[1] = (byte)0x24;
            bb[2] = (byte)(X % 0x100);
            bb[3] = (byte)(X / 0x100);
            bb[4] = (byte)(Y % 0x100);
            bb[5] = (byte)(Y / 0x100);

            this.serialPort1.Open();
            if (this.serialPort1.IsOpen)
```

```
        {
            this.serialPort1.Write(bb, 0, 6);
            this.serialPort1.Close();
        }

    }
```

5.9.3 Magnified font

Magnified font is a function to enlarge a character to vertical and horizontal directions.

```
    /*  @Font Magnified */
    private void fontMagnified(int X, int Y)
    {
        byte[] bb = new byte[6];

        bb[0] = (byte)0x1f;
        bb[1] = (byte)0x28;
        bb[2] = (byte)0x67;
        bb[3] = (byte)0x40;
        bb[4] = (byte)X;
        bb[5] = (byte)Y;

        this.serialPort1.Open();
        if (this.serialPort1.IsOpen)
        {
            this.serialPort1.Write(bb, 0, 6);
            this.serialPort1.Close();
        }

    }
```

5.9.4 Proportional ASCII

Change the character spacing. Using with a proportional font, the number of average characters which can be displayed will increase.

```
    /* Set Font Size **
     **
     **    w=0: Fixed Font Size with 1 dot space
     **    w=1: Fixed Font Size with 2 dot space
     **    w=2: Proportional Font Size with 1 dot space
     **    w=3: Proportional Font Size with 2 dot space
     */
    private void FontWidth(int w)
    {
        byte[] bb = new byte[5];

        bb[0] = (byte)0x1f;
        bb[1] = (byte)0x28;
        bb[2] = (byte)0x67;
        bb[3] = (byte)0x03;
        bb[4] = (byte)(w % 0x100);

        this.serialPort1.Open();
        if (this.serialPort1.IsOpen)
        {
            this.serialPort1.Write(bb, 0, 5);
            this.serialPort1.Close();
        }
```

```
}
```

5.9.5 Set display 2-byte font (for 7900 only)

Set up CJK (Chinese-Japanese-Korean) font.

```
/*  Setup Kanji (2 byte font in Japanese)
 **
 **      Set up Kanji Display mode.
 **
 **      cjk=0:Japanese,
 **          1:Korean,
 **          2:Simplified Chinese
 **          3:Traditional Chinese
 */

private void CJK_setup( int cjk)
{

    byte[] bb = new byte[15];

    bb[0] = (byte)0x1f;
    bb[1] = (byte)0x28;
    bb[2] = (byte)0x67;
    bb[3] = (byte)0x01;
    bb[4] = (byte)0x02;
    bb[5] = (byte)0x1f;
    bb[6] = (byte)0x28;
    bb[7] = (byte)0x67;
    bb[8] = (byte)0x02;
    bb[9] = (byte)0x01;
    bb[10] = (byte)0x1f;
    bb[11] = (byte)0x28;
    bb[12] = (byte)0x67;
    bb[13] = (byte)0x0f;
    bb[14] = (byte)cjk;

    this.serialPort1.Open();
    if (this.serialPort1.IsOpen)
    {
        this.serialPort1.Write(bb, 0, 15);
        this.serialPort1.Close();
    }

}
```

5.9.6 Using Shift-JIS code character display

Write character code for display module is Shift-JIS in Japanese. Since the internal code in C# is Unicode, it is necessary to write by encoded characters to Shift-JIS. The example to display at the click of “button2” by Japanese in C# is as follows;

```
private void button2_Click(object sender, EventArgs e)
    {
        const int JIS = 0;
        CJK_setup(JIS); /*Setup JIS Kanji*/

        string str = "日本語表示します";
        Encoding sjisEnc = Encoding.GetEncoding("Shift_JIS");
        int NumberOfBytes = sjisEnc.GetByteCount(str);
        byte[] bytes = sjisEnc.GetBytes(str);
```

```
        this.serialPort1.Open();
        if (this.serialPort1.IsOpen)
        {
            this.serialPort1.Write(bytes,0,NumberOfBytes);
            this.serialPort1.Close();
        }
    }
```

5.9.7 Graphic display

The bitmapped image converted into byte strings is written in VFD module. The image is displayed immediately at the display area. When you write an image to a hidden area, the image appears on the screen by the scroll display command.

```
    /* Realtime Bitimage Display
     *
     *   image: bitmap image
     *   X       : Horizontal size in Bit
     *   Y       : Vertial size in Byte (8Bit)
     *
     */
    private void DrawBitmap(byte[] image, int X, int Y )
    {
        byte[] bb = new byte[9];

        bb[0] = (byte)0x1f;
        bb[1] = (byte)0x28;
        bb[2] = (byte)0x66;
        bb[3] = (byte)0x11;
        bb[4] = (byte)(X % 256);
        bb[5] = (byte)(X / 256);
        bb[6] = (byte)(Y % 256);
        bb[7] = (byte)(Y / 256);
        bb[8] = (byte)0x01;

        this.serialPort1.Open();
        if (this.serialPort1.IsOpen)
        {
            this.serialPort1.Write(bb, 0, 9);
            this.serialPort1.Close();
        }
        this.serialPort1.Open();
        if (this.serialPort1.IsOpen)
        {
            this.serialPort1.Write(image, 0, X * Y);
            this.serialPort1.Close();
        }
    }
```

5.9.8 Graphic display example

The following is a sample program that opens the graphic file and calls "DrawBitmap()". This program is added "Button5" and run as a click event. Please note that you cannot display which is larger than display are size. Moreover, when the vertical resolution of the bitmap image is not a multiple of 8, its remainder part is not displayed.

```
    private void button5_Click(object sender, EventArgs e)
    {
       int i, j;
       byte b2;
```

```
        OpenFileDialog openDia = new OpenFileDialog();

        if (openDia.ShowDialog() == System.Windows.Forms.DialogResult.OK)
        {
            Bitmap bmp = new Bitmap(openDia.FileName);

            //  @Show Bitmap on window
            pictureBox1.Image = bmp;

            // Transform Bitmap file into byte array
            Byte[] bb = new Byte[bmp.Width * (bmp.Height/8)];

            for (i = 0; i < bmp.Width; i++)
            {
                for (j = 0; j < (bmp.Height / 8); j++)
                {
                    b2 = 0;
                    if (bmp.GetPixel(i, j * 8).G < 128) { b2 = 1; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 1).G < 128) { b2++; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 2).G < 128) { b2++; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 3).G < 128) { b2++; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 4).G < 128) { b2++; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 5).G < 128) { b2++; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 6).G < 128) { b2++; } b2 += b2;
                    if (bmp.GetPixel(i, j * 8 + 7).G < 128) { b2++; }

                    bb[i * (bmp.Height / 8) + j] = b2;
                }
            }


            // Move Cursor to Home
            moveCursor(0, 0);

            // Call Realtime bitmap display
            DrawBitmap(bb, bmp.Width, bmp.Height / 8);

        }
        openDia.Dispose();
    }
```

5.9.9 Graphic scroll

The display module performs the scroll of the screen by moving data in Display RAM.

```
/*
 *   Graphics Hrizontal scroll
 *
 *
 */
    private void GraphicsHrizontalScroll(int skip, int number, int speed)
    {

        byte[] bb = new byte[9];

        bb[0] = (byte)0x1f;
        bb[1] = (byte)0x28;
        bb[2] = (byte)0x61;
        bb[3] = (byte)0x10;

        bb[4] = (byte)(skip % 256);
        bb[5] = (byte)(skip / 256);
        bb[6] = (byte)(number % 256);
        bb[7] = (byte)(number / 256);
        bb[8] = (byte)speed;

        this.serialPort1.Open();
        if (this.serialPort1.IsOpen)
        {
            this.serialPort1.Write(bb, 0, 9);
            this.serialPort1.Close();
        }
    }
```

5.9.10 Graphic scroll example

The following is a sample program that is added and run as a click event of “button6”.
Please rewrite the X and Y size to accommodate your display module.

```
private void button6_Click(object sender, EventArgs e)
{
    const int Xsize = 256; /* Horizontal screen size of display*/
    const int Ysize = 64; /* Vertical screen size of display*/
    const int speed = 1;

    /*
     * Scroll entire display
     */

    GraphicsHrizontalScroll(Ysize / 8, Xsize, speed);

}
```

5.9.11 Display hidden area

The hidden area of RAM appears on the display area using with the scroll command "5.9.9."

The following example is of a calling program implemented as a click event for “Button7”. Rewrite the X and Y size to accommodate your display module.

```
private void button7_Click(object sender, EventArgs e)
{
```

```
        const int Xsize = 256; /* Horizontal screen size of display*/
        const int Ysize = 64; /* Vertical screen size of display*/
        const int speed = 1;

        /*
         * Show hidden area
         */

        GraphicsHrizontalScroll(Ysize/8 * Xsize, 1, speed);

    }
```

5.9.12 Character scroll

Character scrolling is performed when the character is written.
The horizontal scrolling is performed under the following conditions:

Condition 1: After setting the horizontal scroll mode,
Condition 2: when the cursor reaches the right side of the screen,
Condition 3: This display will initiate a scroll action by one character at next character writing.

The scroll speed also needs to be set.

```
/*
 * Horizontal Scroll Mode
 */
private void HorizontalScrollMD3()
{
    byte[] bb = new byte[2];

    bb[0] = (byte)0x1f;
    bb[1] = (byte)0x03;

    this.serialPort1.Open();
    if (this.serialPort1.IsOpen)
    {
        this.serialPort1.Write(bb, 0, 2);
        this.serialPort1.Close();
    }
}

/*
 * Horizontal Scroll Speed
 */
private void HorizontalScrollSpeed(int speed)
{
    byte[i] bb = new byte[3];

    bb[0] = (byte)0x1f;
    bb[1] = (byte)0x73;
    bb[2] = (byte)(speed % 32);

    this.serialPort1.Open();
    if (this.serialPort1.IsOpen)
    {
        this.serialPort1.Write(bb, 0, 3);
        this.serialPort1.Close();
    }
}
```

5.9.13 Character scroll example

Example illustrates a use of this command. It is a function as a click event of "button8".

```
private void button8_Click(object sender, EventArgs e)
{
 const int speed = 2;

    HorizontalScrollMD3();
    HorizontalScrollSpeed(speed);

    this.serialPort1.Open();
    if (this.serialPort1.IsOpen)
    {
        this.serialPort1.Write("Horizontal Scroll Mode Test.............................");
        this.serialPort1.Close();
    }

}
```

5.9.14 Subdivision of a screen in the user window.

A user window can be separated into four sections and commands are performed within the user window. You use User window setting, delete, and select command.

User window define

```
private void DefineUserWindow(int a, int X, int Y, int W, int H)
{
    byte[] bb = new byte[14];

    bb[0] = (byte)0x1f;
    bb[1] = (byte)0x28;
    bb[2] = (byte)0x77;
    bb[3] = (byte)0x02;
    bb[4] = (byte)a;
    bb[5] = (byte)1;
    bb[6] = (byte)(X % 256);
    bb[7] = (byte)(X / 256);
    bb[8] = (byte)(Y % 256);
    bb[9] = (byte)(Y / 256);
    bb[10] = (byte)(W % 256);
    bb[11] = (byte)(W / 256);
    bb[12] = (byte)(H % 256);
    bb[13] = (byte)(H / 256);

    this.serialPort1.Open();
    if (this.serialPort1.IsOpen)
    {
        this.serialPort1.Write(bb, 0, 14);
        this.serialPort1.Close();
    }
}
```

User Window delete

```
private void CancelUserWindow(int a)
{
    byte[] bb = new byte[6];

    bb[0] = (byte)0x1f;
    bb[1] = (byte)0x28;
    bb[2] = (byte)0x77;
    bb[3] = (byte)0x02;
    bb[4] = (byte)a;
```

```
            bb[5] = (byte)0;

            this.serialPort1.Open();
            if (this.serialPort1.IsOpen)
            {
                this.serialPort1.Write(bb, 0, 6);
                this.serialPort1.Close();
            }
        }
```

User window select

```
        private void SelectCurrentUserWindow(int a)
        {
            byte[] bb = new byte[5];

            bb[0] = (byte)0x1f;
            bb[1] = (byte)0x28;
            bb[2] = (byte)0x77;
            bb[3] = (byte)0x01;
            bb[4] = (byte)a;

            this.serialPort1.Open();
            if (this.serialPort1.IsOpen)
            {
                this.serialPort1.Write(bb, 0, 5);
                this.serialPort1.Close();
            }
        }
```

5.9.15 User window example

The following is a sample program that assign to a click event of "Button9".

A 50x16 pixels user window is set to the upper left corner of the screen, and it is displayed there as "Window". When character scrolling and a graphic display are performed without a break, its data is written in this user window.

```
        private void button9_Click(object sender, EventArgs e)
        {
            const int UserWindow1 = 1;

            DefineUserWindow(UserWindow1, 0, 0, 50, 2);
            SelectCurrentUserWindow(UserWindow1);

            this.serialPort1.Open();
            if (this.serialPort1.IsOpen)
            {
                this.serialPort1.Write("Window");
                this.serialPort1.Close();
            }
        }
```

Use the "SelectCurrentUserWindow(0); function" when you stop using the User window.

5.10 Parallel interface program examples

The following is a sample program that writes 1 byte for use with 8Bit parallel interface of embedded MPU. Add a wait time if necessary.

This circuit assumes the case of "4.5.5 Example 2 of Parallel for output BUSY signal connection".

```
--------- Program example of Data input -----------------
Void GU7000_out( char data)
{
        do {} while ( BUSY == 1); /* Wait for ready */
        /* wait 1.5uSec, if necessary */
        DataLines = data;        /* Output data */

        WR = 0; /* set WR Low */
        /* wait 100nS, if necessary*/
        WR = 1; /* set WR High */
}
--------------------------------------------------------------
```

Please take a wait time when a host is too high speed.

Please take a wait time when a host is too high speed

6 Troubleshooting Tips

6.1 BUSY signal

BUSY for parallel interface is different than SBUSY for serial interface.

Use short between C and B pin of JRB jumper when you use a BUSY signal for parallel. Default setting is no connection made at factory.

6.2 Reset

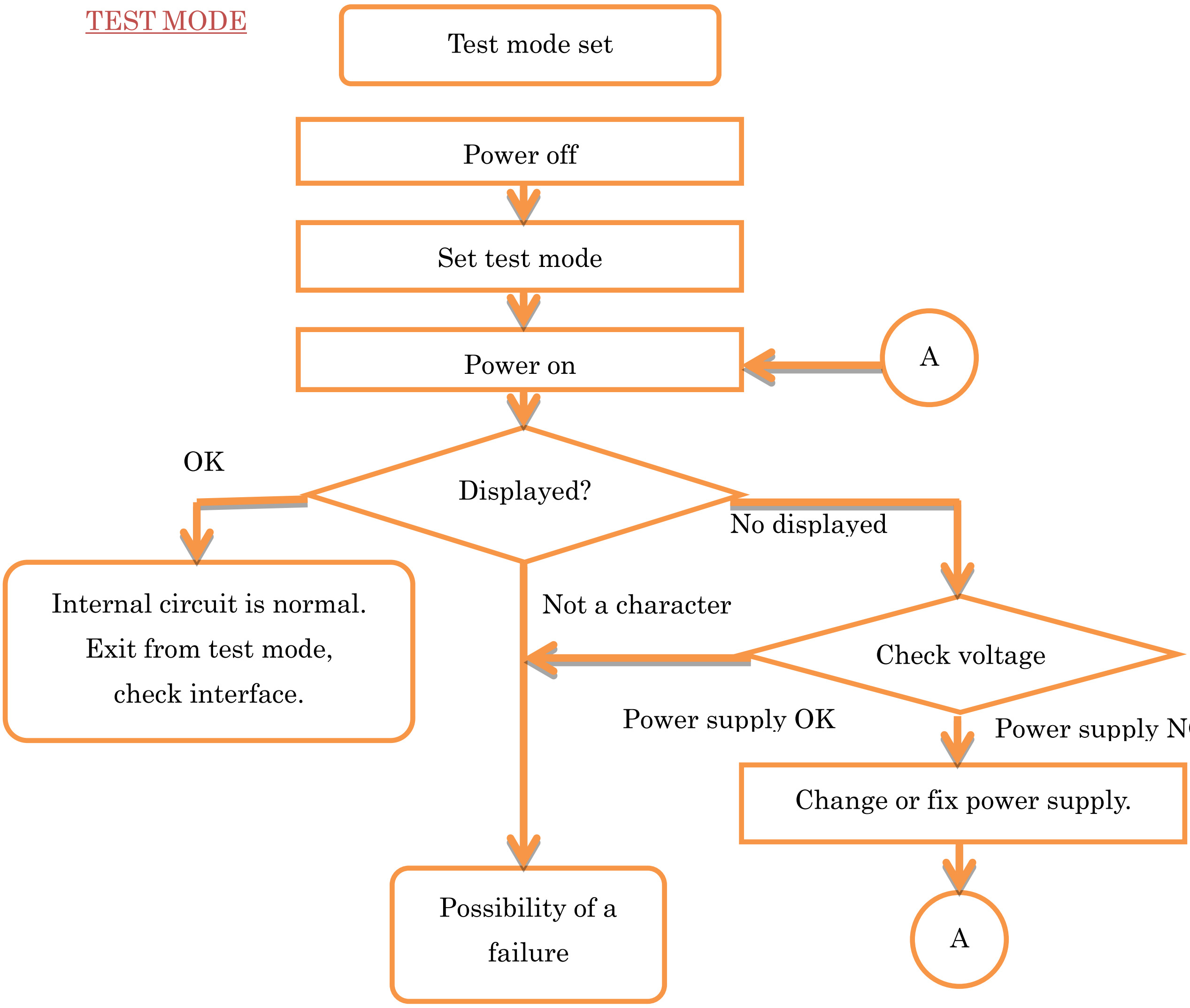
The display controller cannot accept data or commands during the internal power up initialization. During this initialization, BUSY output is set to “BUSY”, then do not write data or commands during this period.

6.3 Why is not it displayed at all? Self test mode

GU-7000 series does not require initialization to start to show some characters. If the display does not turn on at all, isolate the cause of failure such as wrong circuit drive or a display module failure. As a solution, GU-7000 series has a self-diagnostic function to displays a test pattern automatically at turning on.

Please refer to "6.4 How to set test mode" for each item numbers.

TEST MODE

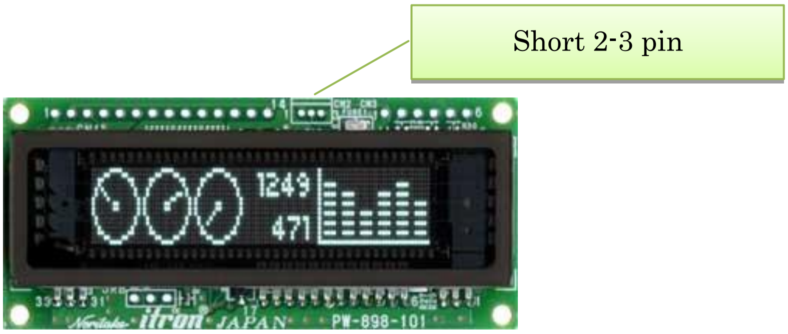


6.4 How to set test mode

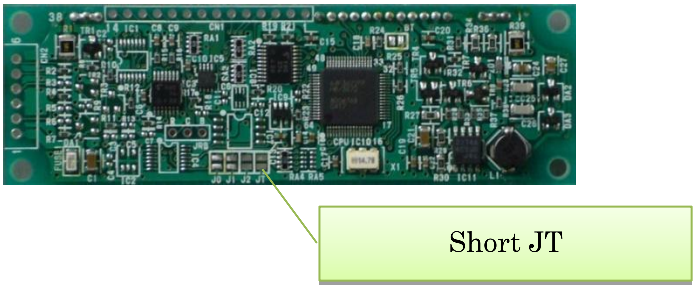
Starting the self test mode varies for each item. Set the test mode by connecting the center pin of the 3-pin connector to GND. Set other models without a 3pin connecter by using the short of Jumper J-T or DIP-SW.

Items are not to scale

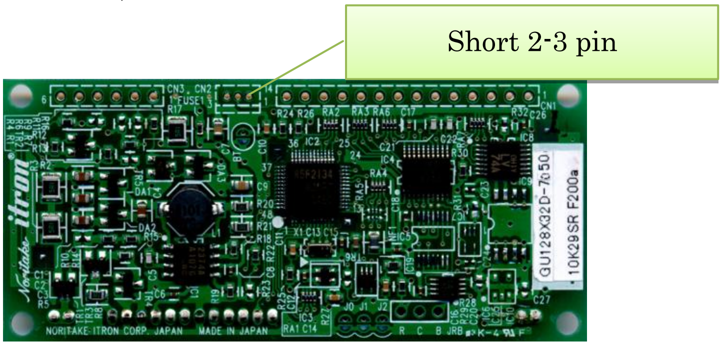
6.4.1 GU112X16G-7000, -7003, -7900, -7000B, -7003B, -7900B



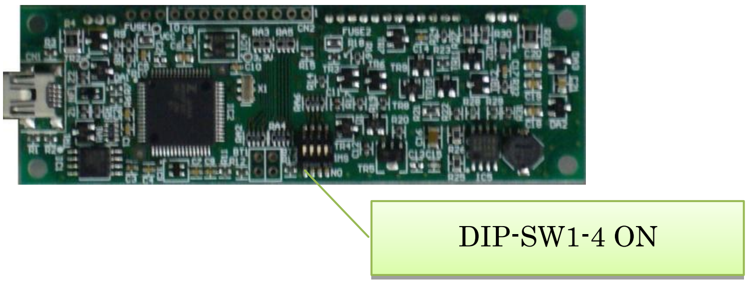
6.4.2 GU128X32D-7000, -7003, -7900, -7000B, -7003B, -7900B



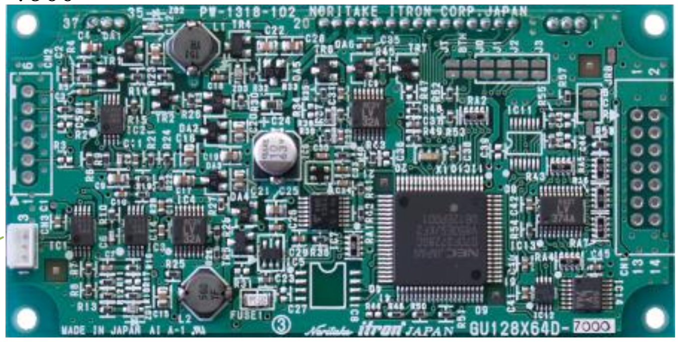
6.4.3 GU128X32D-7050, -7950



6.4.4 GU128X32D-7901

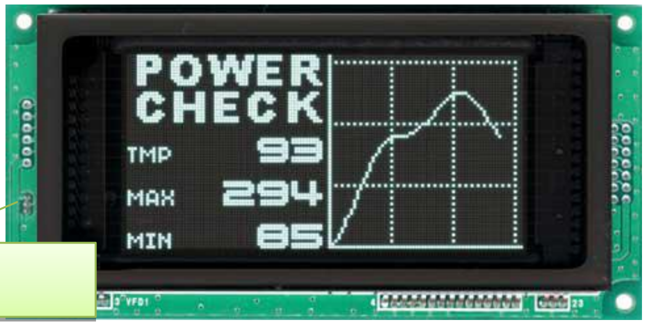


6.4.5 GU128X64D-7000, -7003, -7900



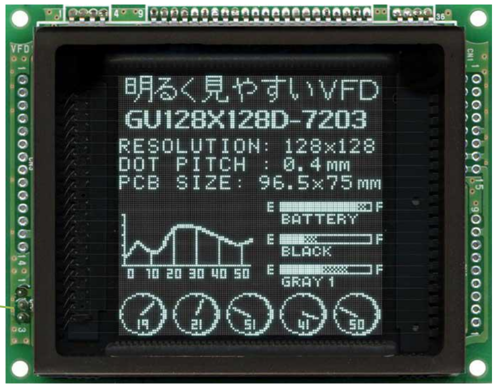
Short 2-3 pin

6.4.6 GU128X64F-7000, -7003, -7900, -7900BX



Short 2-3 pin

6.4.7 GU128X128D-7203



Short 2-3 pin

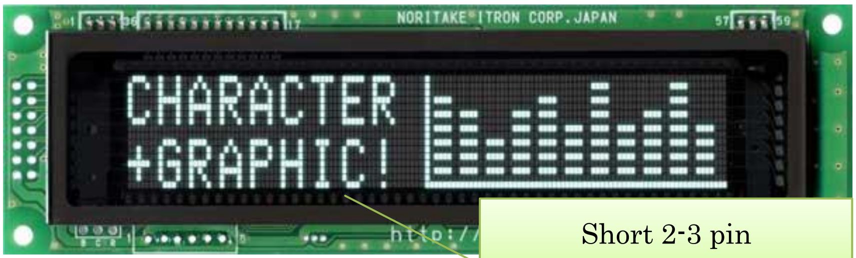
6.4.8 GU140X16G-7040A

Short 2-3 pin

6.4.9 GU140X16G-7000, -7003, -7900, -7903, -7042, 7000B, -7003B, -7900B

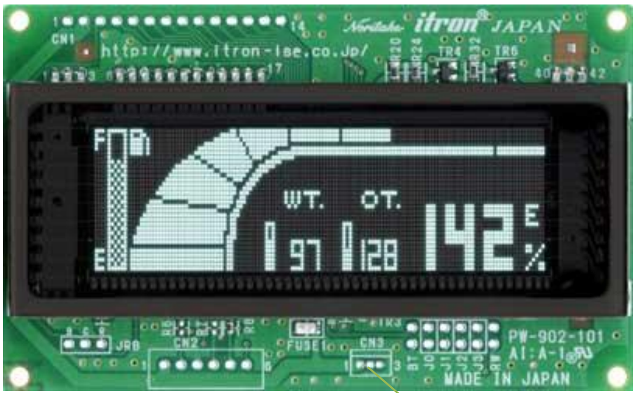
Short 2-3 pin

6.4.10 GU140X16J-7000, -7003, -7000B, -7003B, -7900B



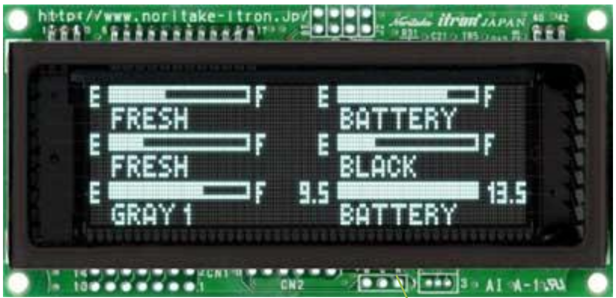
Short 2-3 pin

6.4.11 GU140X32F-7000, -7003, -7900, -7903, -7032, -7000B, -7003B, -7900B



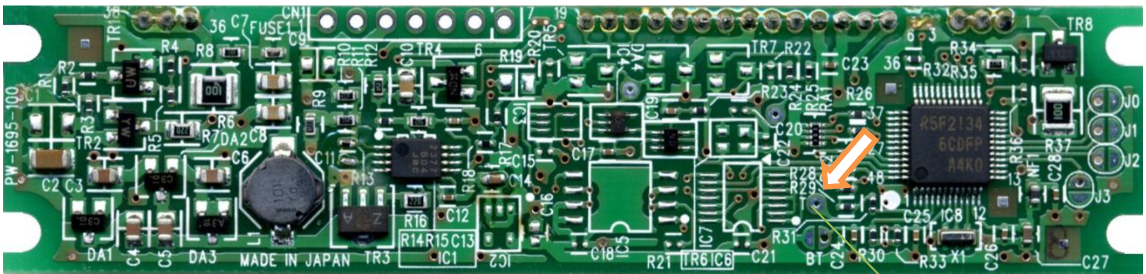
Short 2-3 pin

6.4.12 GU140X32F-7050, -7053, -7950



Short 2-3 pin

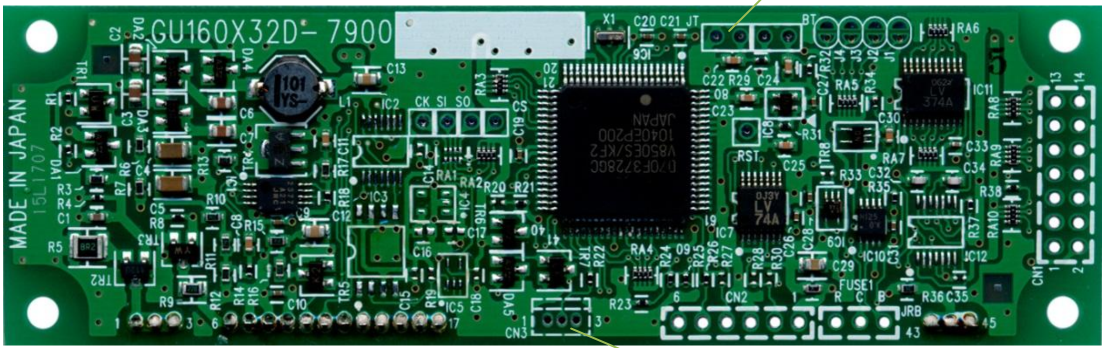
6.4.13 GU144X16D-7053B



Connect Pad T to GND

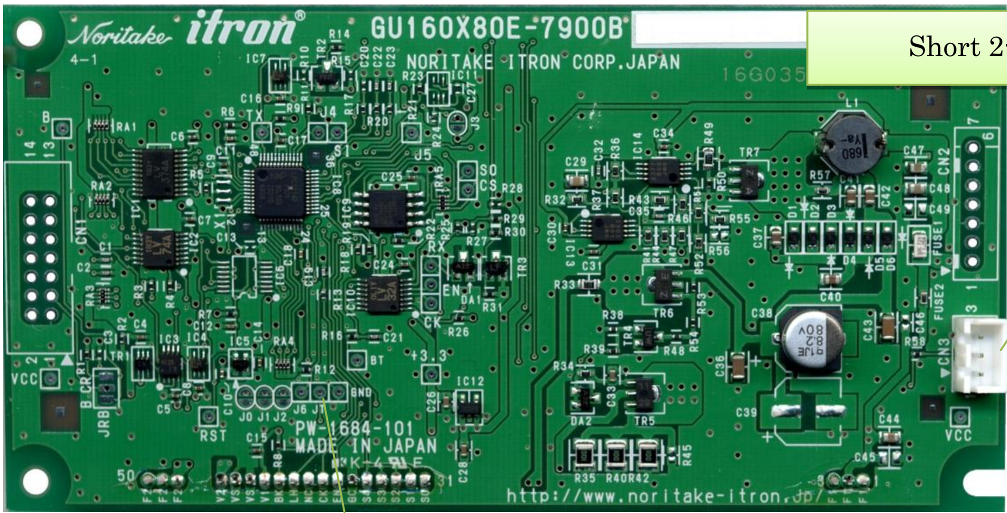
6.4.14 GU160X32D-7050, -7950

The test mode will start with one or the other.



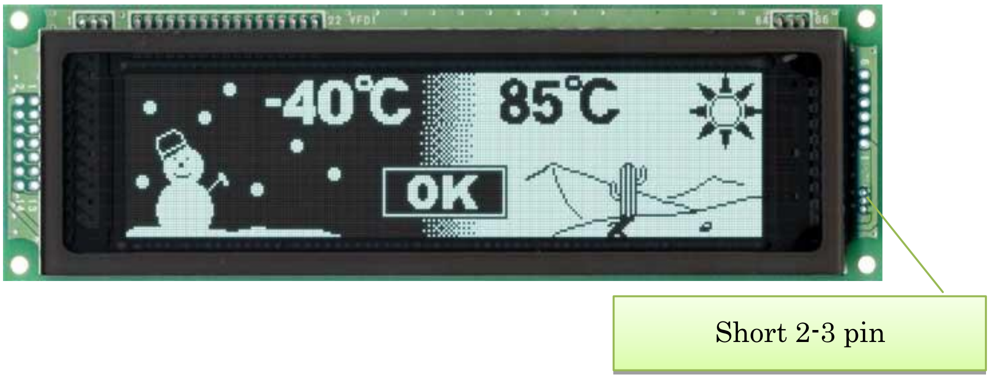
6.4.15 GU160X80E-7900B

The test mode will start with one or the other.



6.4.16 GU256X64C-7000, -7003, -7900

6.4.17 GU256X64D-7000, -7900



6.4.18 GU280X16G-7000, -7003

7 Support TOOL

We can offer support tool for you, please contact from our website.

Homepage (Japanese):

http://www.noritake-itron.jp/

(English):

http://www.noritake-itron.jp/eng/

(Chinese):

http://www.noritake-itron.jp/chinese/

7.1 GUD10K [Tutorial]

Platform: Windows PC

- GUD10K is tutorial Software that offers guidance in creating, programming, sequencing and displaying messages with the Noritake GU-7000 Series.
- This tool offers a list of command sequences to perform complex operations with the display screen.

7.2 C library 7000sample_v10.zip

Platform: H8/325 , Renesas Technology Corporation

Command library of subroutines to use GU-7000 series smoothly.

7.3 Sub routine list

Platform: C language

Ann example of the function definitions which use the 7000 series functions.

This is easy to use if you understand command names and routine names.

7.4 Visual Basic 2008 sample code

A Form application of Visual Basic. Also can be used as a form for software development.

8 Environment

We are committed to the understanding of environmentally hazardous substances established by the Green Procurement Guideline.

We have also certification of ISO14000 and are keeping striving for the correspondence to an environmental issue.

8.1 RoHS Compliance

All standard module products including GU-7000 series are compatible with the **RoHS Directive**.

GU-7000 series is exempt from lead-containing component RoHS as follows:

- Lead oxide in glass used in Vacuum Fluorescent display.
- Lead oxide contained in the ceramic and glass in electronic components.
- Lead in high melting point solder components used to connect the internal semiconductor products.

9 Safety standard

Printed circuit board materials are certified UL :94-V0 flame retardant grade.

The UL certification number is indicated on the printed circuit board.

10 Disclaimers and limitations

The contents provided in this document are carefully created and managed by our Company, but we cannot guarantee to use these contents on all platforms. If you have any problems or questions, please consult us.

Noritake sample codes are only for the use of Noritake products. Please carry out operating verification for final products such as application software on your responsibility. Support tools provided in a form of the installer may contain programs that are used under license. These must not be reproduced, modified, or integrated, etc. as reverse engineered, reverse compiled, or reverse assembled.

11 Contact us

If you have any questions or requests, please consult our sales office or customer support desk (cs@noritake-itron.jp)

End of document