



GEVA elettronica Italy  
[shop.gevaelettronica.it](http://shop.gevaelettronica.it)  
[email@gevaelettronica.it](mailto:email@gevaelettronica.it)  
<https://www.facebook.com/GEVAelettronica>

## IL MANUALE DI GEVINO TFT 7" v1.1

Compatible Arduino Zero  
Manual v1.3

GEVINO products have been designed for civil, domestic, corporate or industrial environments. They have high voltage and high current inputs and outputs, protected against disturbances, electrostatic discharges, magnetic fields, inductive loads, extra voltages and currents.

They comply with EC directives.

It is possible to customize them for more than 50 pieces.

To program Gevino simply connect it to your computer with a USB cable, write some instructions, and press the «Upload» button. It is very rich in libraries for an infinity of modules and sensors. The instructions are quite simple and understandable.

The programming software, also called IDE (Integrated Development Environment), contains many examples from which to take inspiration to write their own listings.

*I prodotti GEVINO sono stati progettati per ambienti civili, domestici, aziendali o industriali. Hanno ingressi e uscite ad alta tensione, alta corrente, protetti contro disturbi, scariche elettrostatiche, campi magnetici, carichi induttivi, extra tensioni e correnti.*

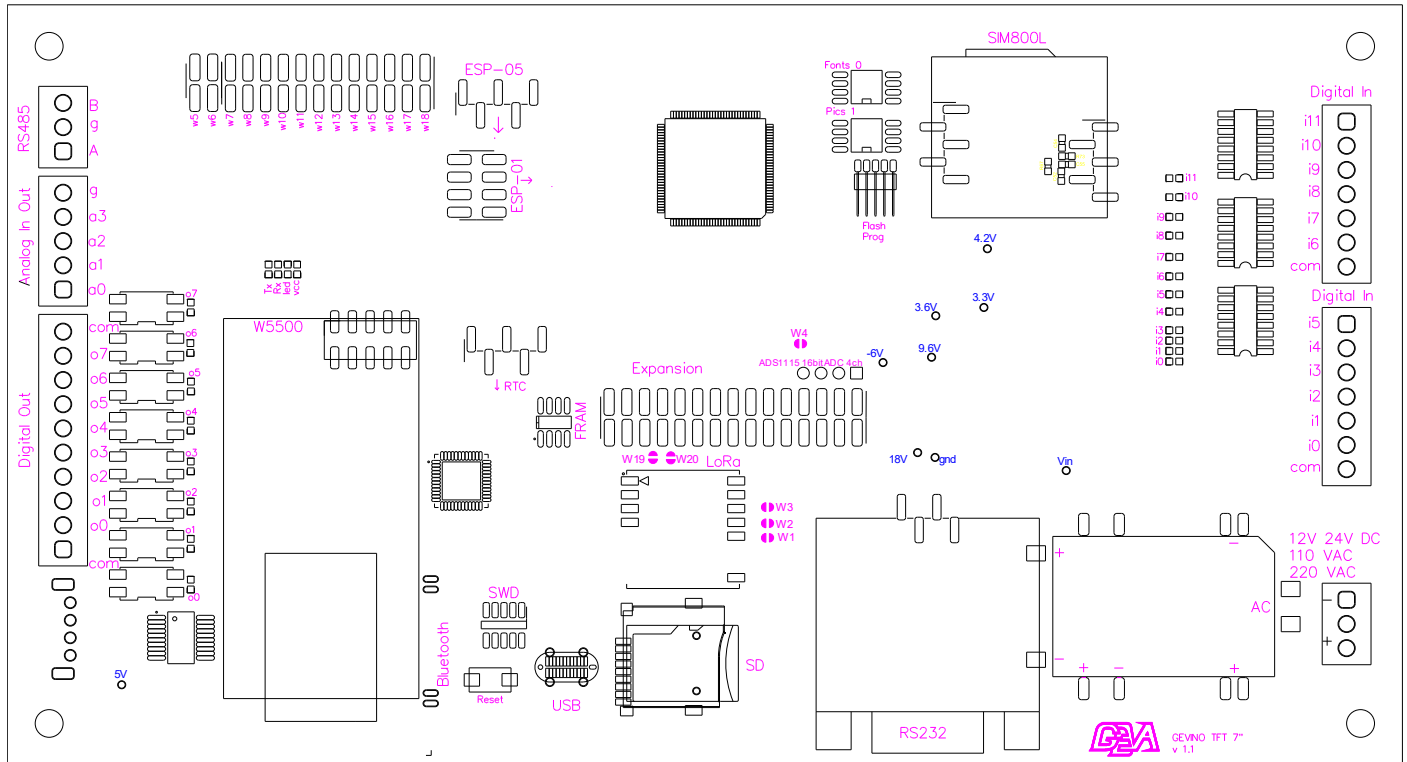
*Sono conformi alle direttive CE.*

*E' possibile personalizzarle per una quantità minima di 100 pezzi.*

*Per programmare Gevino è sufficiente collegarlo al proprio computer con un cavo USB, scrivere qualche istruzione, e premere il tasto «Upload». E' ricchissimo di librerie per un' infinità di moduli e sensori. Le istruzioni sono abbastanza semplici e comprensibili.*

*Il software di programmazione, chiamato anche IDE (Ambiente Integrato di Sviluppo), contiene numerosi esempi da cui prendere spunto per scrivere i propri listati.*

Board 110x200mm, Hole 98x188mm d4 mm



## JUMPER

## a0 Terminal

|                             |                          |
|-----------------------------|--------------------------|
| W8 + W9 = 20mA Output       | (Open W7 W10 W14 W18)    |
| W7 + W10 = 10V Output       | (Open W8 W9 W14 W18)     |
| W8 = 10V Input              | (Open W7 W9 W10 W14 W18) |
| W8 + W14 = 1V Input         | (Open W7 W9 W10 W18)     |
| W8 + W14 + W18 = 20mA Input | (Open W7 W9 W10)         |

## a1 Terminal

10V Input  
W13 = 1V Input (Open W17)  
W13 + W17 = 20mA Input

## a2 Terminal

10V Input  
W12 = 1V Input (Open W16)  
W12 + W16 = 20mA Input

### a3 Terminal

10V Input  
W11 = 1V Input (Open W15)  
W11 + W15 = 20mA Input

## LoRa-CS

W1 = D5 W2 = D13 W3 = A5

## TFT Backlight

W4 = Always On Otherwise PWM0 of RA8876

Input-CS

W19 = D2  
W20 = D6 = output-cs (Read problem on the example)

- 7" Capacitive Touch 1024x600 FT5316 16bpp
- Viewing Angle Left:Right:70° Up:60° Dw:70° , Brightness 400 Cd/m2
- Vcc from 10.5V to 42V 300mA 3.6W
- 8x Optoisolated Out 30V 1A, DC or AC. PNP or NPN (Common can be connected to GND, or VCC, or AC) Out0, are fast Output, 1Khz max frequency.
- 12x Optoisolated Input, High level 5 to 40V, -5 to -40V, Low level -2.5 to +2.5V, DC or AC. In0, In1, In2, In3, are fast input and can be interrupt. 10Khz max frequency
- 4x Not Isolated Analog Input, 12Bit, 10V or 1V or 20mA
- RS485 Half Duplex
- CPU ATSAM21G18 - Arduino ZERO M0 compatible.
- 48 Mhz, 32-bit ARM® Cortex®-M0+ processor.
- Flash 256KB, RAM 32K
- -40°C +85°C Working temperature

## Utilization / Utilizzo

---

Easy and Powerfull Graphic design via PC <https://www.impulseadventure.com/elec/guislice-gui.html>

ARDUINO IDE <https://www.arduino.cc/en/main/software>

Install zero Compiler <https://www.arduino.cc/en/Guide/ArduinoZero>

Choose Board «Arduino/Genuino Zero (Native USB Port)»

Use the Arduino Zero reference guide.

See also the GEVINO Common manual

### FRAME in Black Acrylic:

### CORNICE in Acrilico Nero:

---

In preparation frame in black acrylic, the same measures of the board (110x200mm), with 4 Studs M4, (98x188mm)  
For a water resistant mounting, it is recommended to cover the display panel with an polycarbonate adhesive.

*In preparazione cornice in acrilico nero, delle stesse misure della scheda (110x200mm), con 4 Prigionieri M4, (98x188mm), non protetto all'acqua.*

*Per un montaggio protetto dall'acqua, si consiglia di coprire il display con un pannello adesivo in polycarbonato.*

## ACCESSORIES / Accessori

---

- WiFi ESP-01 (internal antenna) or ESP-05 (External antenna)
- Bluetooth SPP-C / HC-06 / HC-07 / HC-08
- RS232
- SIM800L SIM GSM 2G
- Ethernet module W5500
- LoRa wireless Module RA-01 RA-02 410 ~ 525 MHz 18dBm 100mW
- RTC Battery Clock DS3231
- 110/220VAC power supply on PCB
- DC-DC isolation power supply 12V 24V
- Memory Stick Reader on front, CH376
- FRAM MB85RC256V I2C 256K-bits 32K-Byte
- Analog Output 10V or 20mA, 1x Not Isolated, loses an analog input
- Expansion Board with
  - 8 Isolated Digital Input
  - 8 Isolated Digital Output
  - 2 Analog Output
  - 4 Analog Input
  - 1 RS485

Six Serial Communication Interfaces, each configurable to operate as either:

- USART with full-duplex and single-wire half-duplex configuration
- I2C up to 3.4MHz
- SPI
- LIN slave

<https://www.arduino.cc/en/Tutorial/SamdSercom>

<https://learn.adafruit.com/using-atsamd21-sercom-to-add-more-spi-i2c-serial-ports/overview>

| SerialUSB | USB                     | CS Addr | Addr | Int | Tx En |
|-----------|-------------------------|---------|------|-----|-------|
| Serial    | ESP-WiFi                |         |      |     |       |
| Serial1   | RS485                   |         |      |     | 7     |
| Serial2   | RS232 or Bluetooth      |         |      |     |       |
| Serial3   | SIM800 or Bluetooth     |         |      |     |       |
| I2C       | RTC DS3231              |         | 0x68 |     |       |
| "         | FT5206 capacitive Touch |         | 0x32 | 38  |       |
| SPI       | SD card reader          | 12      |      |     |       |
| "         | LoRa RA-01              | 5 13 A5 |      |     |       |
| "         | CH376 USB Flash         | 45      |      | --- |       |
| "         | W5500 ETH               | 5       |      | --- |       |
| "         | RA8876 TFT              | 42      |      |     |       |
| "         | 74HC595 Shift Output    | 6       |      |     |       |
| " MISO=27 | 74HC165 Shift Input     | 2       |      |     |       |

## VERTICAL VIEW

---

Per passare alla modalità verticale, bisogna cambiare la mappa caratteri scritta nel chip flash presente sul PCB.

Per avere più memoria per le immagini si può saldare un ulteriore chip.

Partendo dalla cartella Arduino\libraries\GEVINI\Vertical

Copiare il file font.bin nella SD e metterlo nel gevino.

Copiare font.h e fontExt.h nella cartelle della libreria Arduino\libraries\GEVINI\

Caricare l'esempio GEVINO\_TFT\_Vertical.ino, oppure il test, al boot ci metterà diversi minuti per copiare il file, dalla SD al chip flash saldato.

Per ritornare alla visione Horizontale bisogna ripetere tutto dalla cartella Arduino\libraries\GEVINI\Horizontal

To switch to vertical mode, change the character map written in the flash chip on the PCB.

For more memory for images, it is possible weld one other chip.

Starting from the folder Arduino\libraries\GEVINI\Vertical

Copy the font.bin file to the SD and put it in the gevino.

Copy font.h and fontExt.h in the library folder Arduino\libraries\GEVINI\

Load GEVINO\_TFT\_Vertical.ino example, or the test, to boot will put several minutes to copy the file, from the SD to the welded flash chip.

To return to the horizontal vision you have to repeat everything from the Arduino\libraries\GEVINI\Horizontal folder.

## To see the RAM used by the program: [link](#)

---

edit the platform.txt on the ARM Compiler folder

Add this line after `v"compiler.libraries.ldflags=`

```
recipe.hooks.linking.postlink.1.pattern="{compiler.path}\arm-none-eabi-size.exe"  
"{build.path}\{build.project_name}.elf"
```

In the Arduino IDE Preferences, enabling the option:

Mostra un output dettagliato durante: ☒ compilazione ☐ caricamento

Show verbose output during: ☒ compilation ☐ upload

There will appear this.

| text  | data | bss   | dec    | hex   | filename |
|-------|------|-------|--------|-------|----------|
| 78900 | 664  | 27672 | 107236 | 1a2e4 | C:\Users |

Where:

- text = Program FLASH memory
- data = Is used for initialized data. This is best explained by adding the following (global/extern) variable to a program. It will increase 'data' by 4 bytes: `int32_t myVar = 0x12345678;`  
It counts for RAM and FLASH. The linker allocates the data in FLASH which then is copied from ROM to RAM in the startup code.
- bss: The 'bss' contains all the uninitialized data. This is what ends up in RAM.
- dec: Is the sum of text, data and bss.

## Flash and RA8876 RAM usage

---

RA8876M have 8,192,000 M-Byte of RAM

Framebuffer is  $1024 \times 600 \times 2 = 1,228,800$  M-Byte

Flash Chip is 16M-Byte

Font Array is 3.807932 M-Byte (on may 2020, see font.h for know actual)

All Font Array are copied on RA8876M-RAM at startup.

You can add Pictures or icon on Flash, after the Font Array.

## GuiSlice library:

---

Install from <https://github.com/ImpulseAdventure/GUISlice>

Or <https://www.impulseadventure.com/elec/guislice-gui.html>

### Edit

```
Arduino\libraries\GUISlice\src\GUISlice_config.h  
Uncomment  
#include "../configs/ard-shld-gevino_tft.h"
```

### Examples:

Copy, on the example folder, the file All\_76\_Pic\_65K.h

Add this line:

```
#include "gevino_tft_io.h" // Respect this sequence  
#include "All_76_Pic_65K.h" // Rispetta questa sequenza  
#include "gevino_tft_display.h"
```

```
void setup()  
{  
  gslc_tsElemRef* pElemRef = NULL;  
  
  // Initialize debug output  
  Serial.begin(9600);  
  gevino_tft_setup();  
  tft.init();  
}
```

The spectacular nature of this program, is that it draws in PC graphics, .ino Builder generates the code, it is complete with its own program, if you have to change the graphics, the builder modifies the .ino without corrupting its program. It 's almost like working with a "Visual C"

*La spettacolarità di questo programma, è che si disegna la grafica a PC, il Builder genera il codice .ino, lo si completa con il proprio programma, se si deve modificare la grafica, il builder modifica il .ino senza corrompere il proprio programma. E' quasi come lavorare con un "Visual C"*

Note:

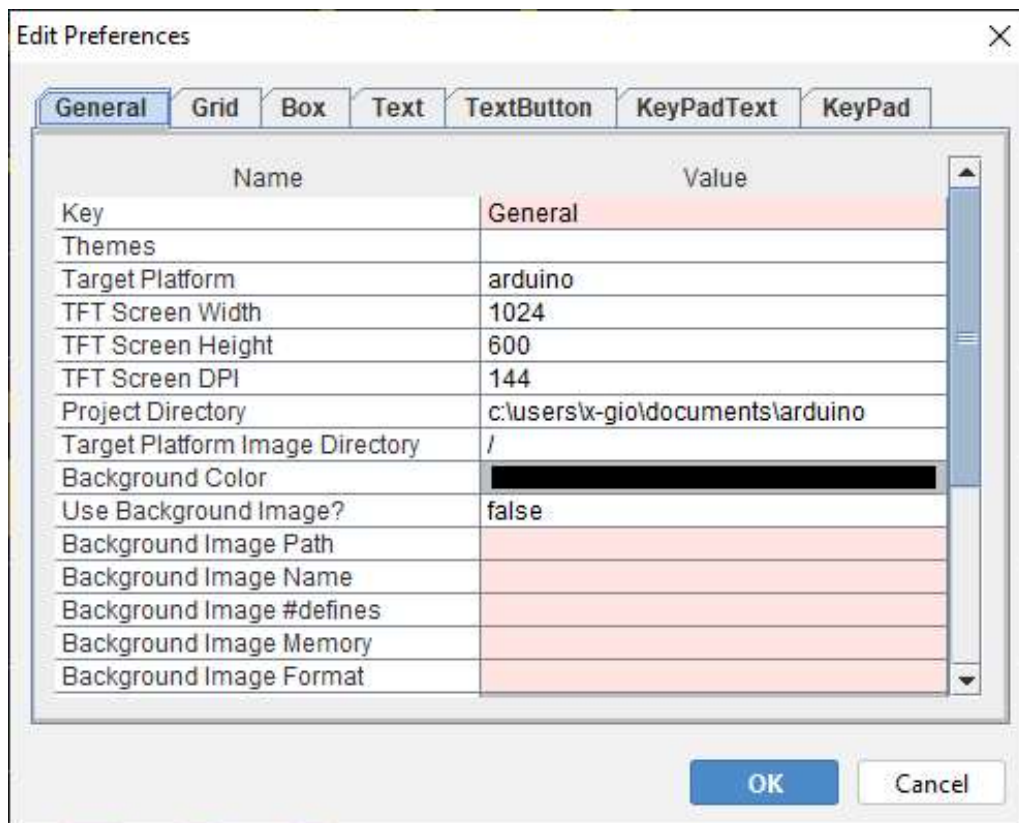
Flash API, uses little RAM, but the objects can not be changed by the code.

*Flash API, usa poca RAM, ma gli oggetti non possono essere modificati dal codice.*

|                |       |
|----------------|-------|
| Use Flash API? | false |
|----------------|-------|

Install from <https://github.com/ImpulseAdventure/GUISlice-Builder>

Copy file arduinofonts.csv on C:\Program Files (x86)\GUISliceBuilder\templates\

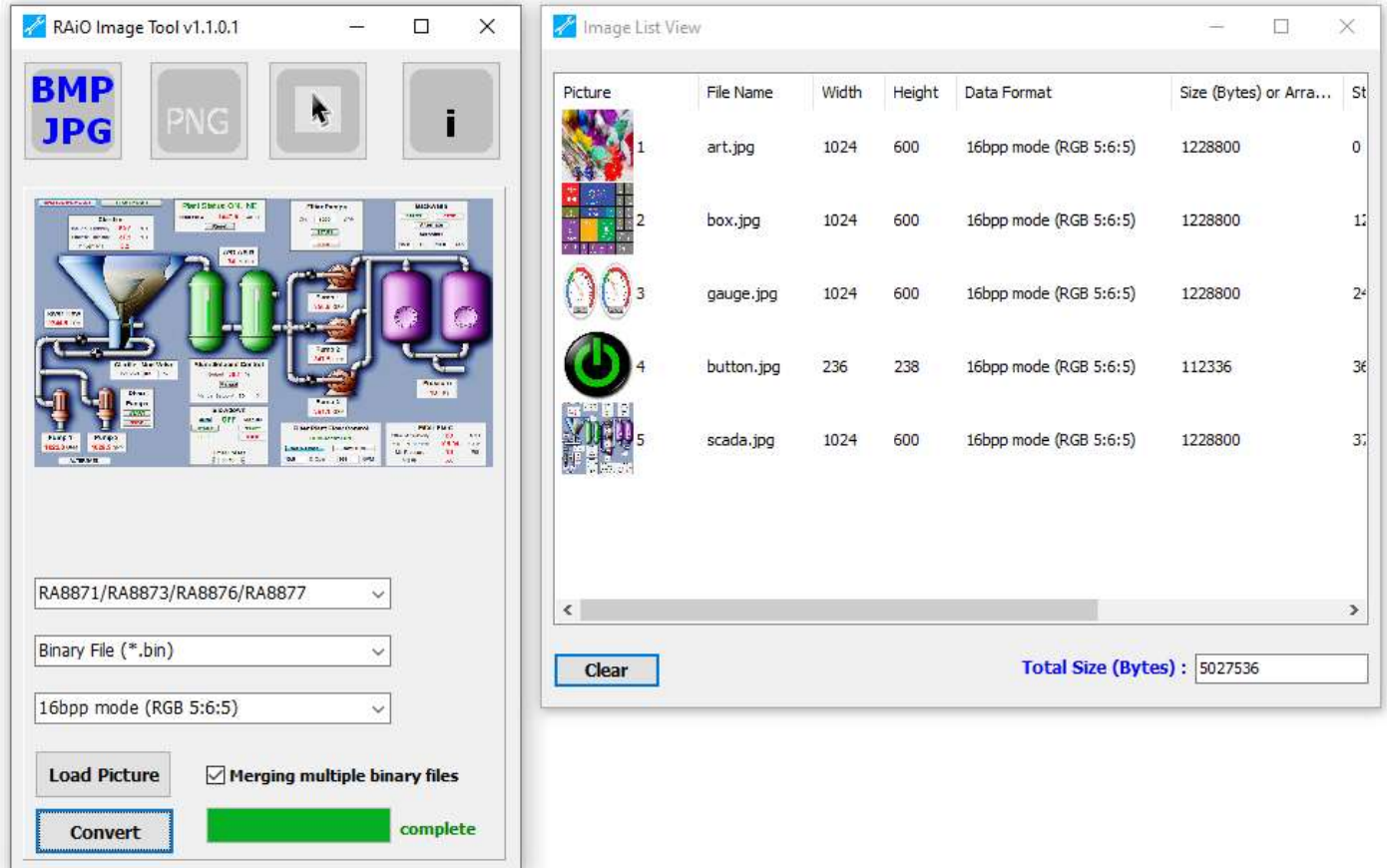


## IMAGE from chip flash DMA

For test example, load on the GEVINO TFT DMA\_Pics\_from\_flash.ino  
Copy on SD card, "All\_76\_Pic\_65K.bin", reboot and wait. Remove SD.

### CREATE YOUR CUSTOM IMAGE.

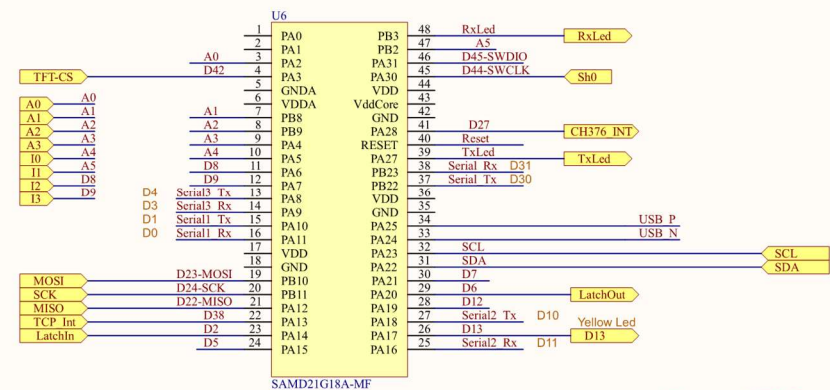
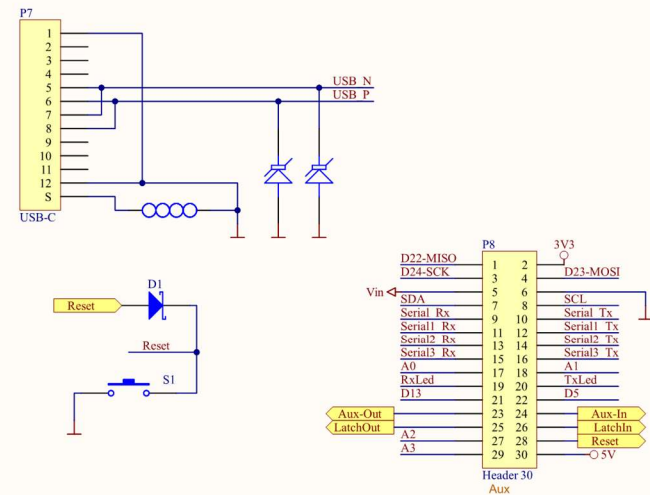
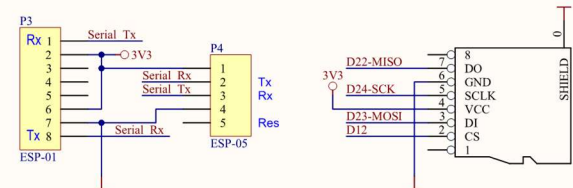
Open image with RAiO Image Tool (Max 1024x600 pixel)



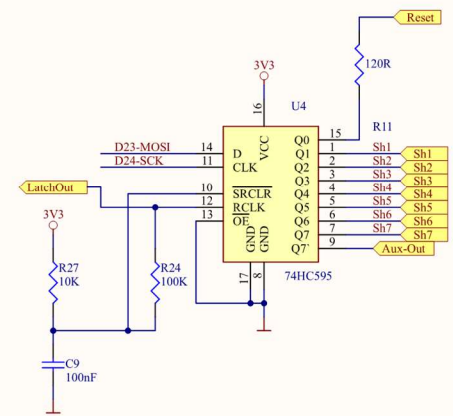
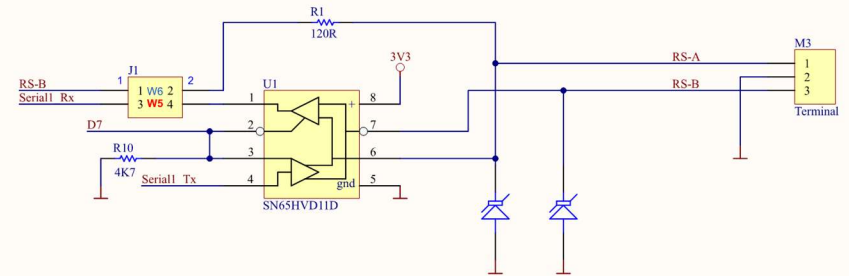
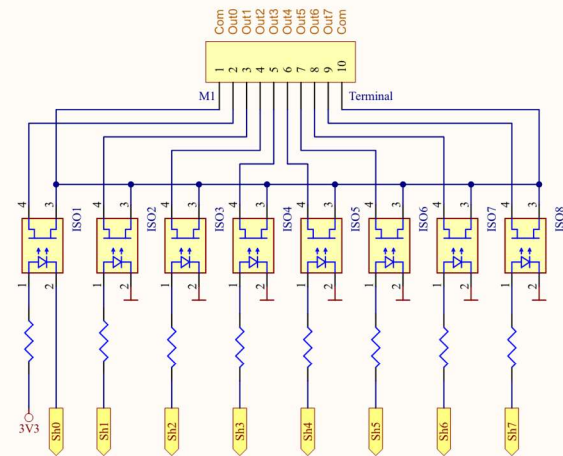
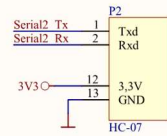
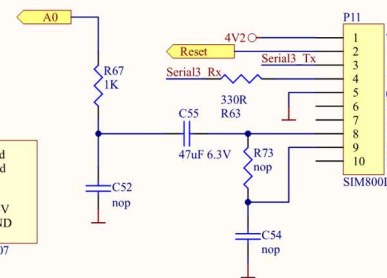
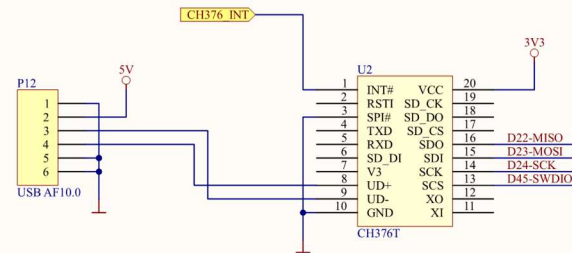
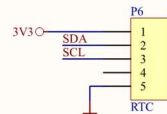
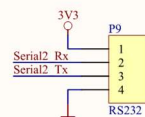
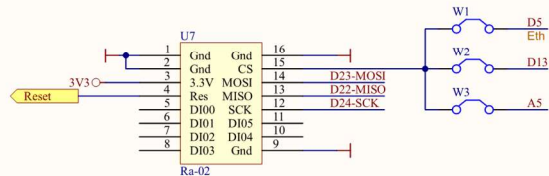
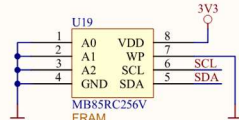
- Copy on SD card All\_76\_Pic\_65K.bin, and using DMA\_Pics\_from\_flash.ino, or using below function, copy content on the Flash Chip.  
`CopySDonFlash();`
- Show picture using command  
`tft.pic(picture_name,X,Y);`

See gevinoTFT\_RA8876.h

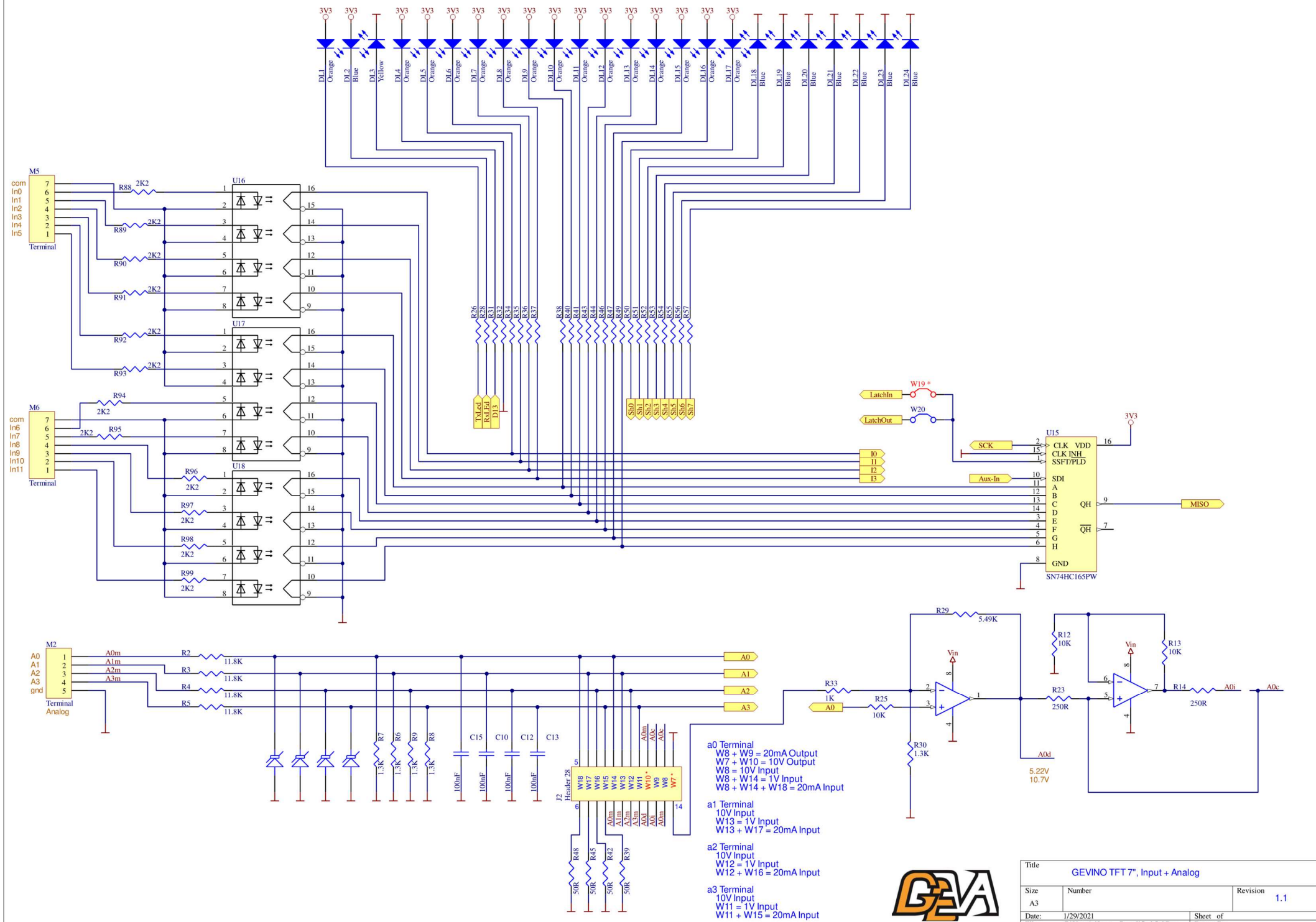
```
setTextMode();
setGraphicsMode();
getWidth();
getHeight();
colorBarTest(enabled);
drawPixel(x, y, color);
drawLine(x1, y1, x2, y2, color);
drawRect(x1, y1, x2, y2, color);
fillRect(x1, y1, x2, y2, color);
drawTriangle(x1, y1, x2, y2, x3, y3, color);
fillTriangle(x1, y1, x2, y2, x3, y3, color);
drawCircle(x, y, radius, color);
fillCircle(x, y, radius, color);
clearScreen(color);
// Text cursor
setCursor(x, y);
getCursorX();
getCursorY();
// Text
setFont();
selectInternalFont();
selectExternalFont();
getTextSizeY();
setTextColor(color);           // Clear background
setTextColor(color, bg);       // Color background
setTextScale(scale);
setTextScale(xScale, yScale);
write(c);
write(*buffer, size);
putChar( cr );
FontCanvasLoad( );
pic(picture_name,X,Y);
PWM0(on_off, Clock_Divided, Prescalar, Count_Buffer,Compare_Buffer);
```



Sercom 0 = Serial1  
Sercom 1 = Serial2  
Sercom 2 = Serial3  
Sercom 3 = I2C  
Sercom 4 = SPI  
Sercom 5 = Serial



|       |                                           |        |              |          |  |
|-------|-------------------------------------------|--------|--------------|----------|--|
| Title |                                           |        | GEVINO TFT 7 |          |  |
| Size  |                                           | Number |              | Revision |  |
| A3    |                                           |        |              | v1.1     |  |
| Date: | 8/30/2022                                 |        | Sheet of     |          |  |
| File: | D:\Document\GEVINO TFT 7 - Simplified\Doc |        | 1 of 1       |          |  |



|                                                |  |           |  |                               |                 |  |  |
|------------------------------------------------|--|-----------|--|-------------------------------|-----------------|--|--|
| Title                                          |  |           |  | GEVINO TFT 7", Input + Analog |                 |  |  |
| Size<br>A3                                     |  | Number    |  |                               | Revision<br>1.1 |  |  |
| Date:<br>1/29/2021                             |  | Sheet of  |  |                               |                 |  |  |
| File:<br>C:\Users\...Input - Simplified.SchDoc |  | Drawn By: |  |                               |                 |  |  |