

# DSO Nano Manual

PCB Version: V1.50

Software Verion: V1.40



## Intro

*DSO mobile* is a pocket size digital storage oscilloscope fulfills basic electronic engineering requirements. It is base on ARM Cortex™-M3 compatible 32 bit platform, equipped with 320\*240 color display, SD card capability, USB connection, and chargeable batteries.

## Features

- Super portable and lightweight
- 2.8" color 320\*240 display
- Micro SD card Waveform Storage
- Basic 1Mps sample rate with 12bit resolution
- Various measurement markers
- Various trigger mode
- Build-in test signal
- USB chargeable battery
- Open source

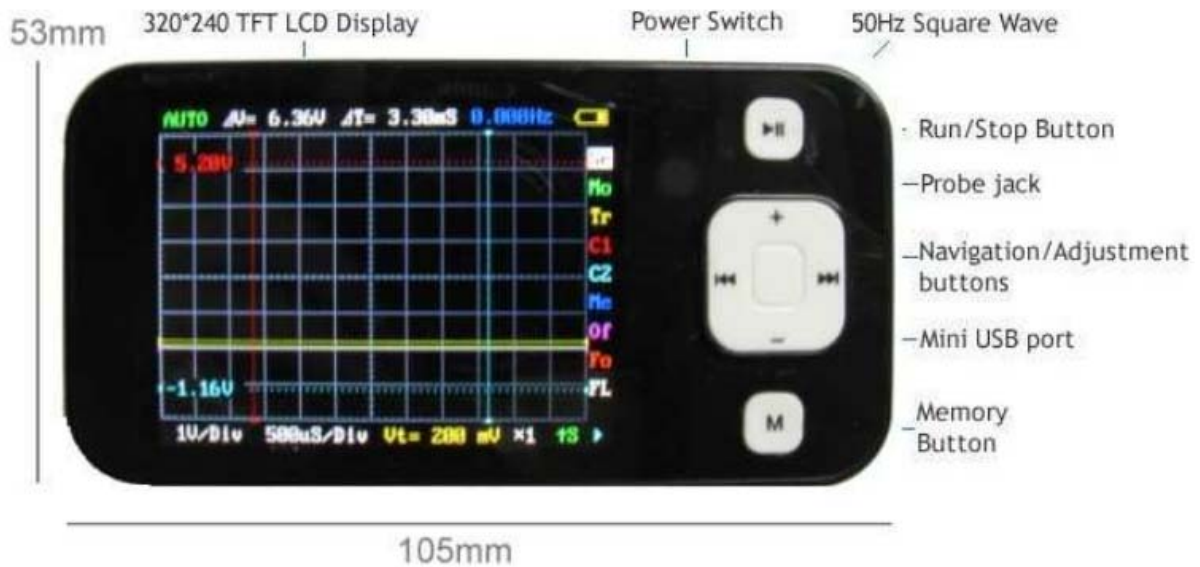


## Specification

Display	2.8" Color TFT LCD
Display Resolution	320×240
Display Color	65K
Analog bandwidth	0 - 1MHz
Max sample rate	1Msps 12Bits
Sample memory depth	4096 Point
Horizontal sensitivity	1uS/Div~10S/Div (1-2-5 Step)
Horizontal position	adjustable with indicator
Vertical sensitivity	10mV/Div~10V/Div (with ×1 probe)
	0.5V/Div~10V/Div (with ×10 probe)
Vertical position	adjustable with indicator
Input impedance	>500KΩ
Max input voltage	80Vpp (by ×1 probe)
Coupling	DC
Trig modes	Auto, Norma, Single, None and Scan
Functionalities:	Automatic measurement: frequency, cycle, duty, Vpp, Vram, Vavg and DC voltage
	Precise vertical measurement with markers
	Precise horizontal measurement with markers
	Rising/falling edge trigger
	Trig level adjustable with indicator
	Trig sensitivity adjustable with indicator
	Hold/run feature
Test signal	Built-in 10Hz~1MHz (1-2-5 Step)
Waveform storage	SD card
PC connection via USB	as SD card reader
Upgrade	by bootloader via USB
Power supply	3.7V Chargeable Lithium battery / USB
Dimension (w/o probe)	105mm X 53mm X 8mm

# Instructions

## User interface



**SE settings: vertical stalls, stalls level, multiple probe**

Click the up and down, left and right gear to achieve vertical and horizontal adjustment of stalls;  
Long by up and down, left and right position adjustment to achieve;  
Long press M button to select multiple probes;  
Play starts scanning short or suspended by scanning,  
Press and Play show or hide all measurements.

**MO settings: scan mode, trigger mode**

Select trigger mode by up and down, left and right select the scan mode;  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

**Tr settings: trigger level, trigger sensitivity**

Adjusted by the upper and lower trigger level, so adjust the trigger sensitivity.  
Press and M blanking trigger line;  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

**C1 set: cross cursor 1 position**

Cursor up and down to adjust the level by 1, so adjust the vertical cursor 1.  
"M" blanking press and move the cursor on the last instruction value -> blanking on the last move of the cursor and the direction value -> display the last move of the cursor on and Direct numerical;  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

**C2 settings: Cross Cursor 2 Position**

Cursor up and down to adjust the level by 2, so adjust the vertical cursor 2.  
"M" blanking press and move the cursor on the last instruction value -> blanking on the last move of the cursor and the direction value -> display the last move of the cursor on and Direct numerical;  
Long press Play button to return to cross cursor 1 Menu

**Me Select: measurement mode, set the measurement display mode**

Selected by measuring the content of the upper and lower,  
Long right-click shows all measurements,  
Long press left hidden;  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

**Of set: x-axis offset, y-axis offset**

Press down to adjust the vertical offset, so adjust the level of migration.  
M blanking offset by the level of long reference line;  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

**Fo settings: the output pulse frequency**

Press down to adjust the output frequency;  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

**FL: File Operations**

Right click choose save, left click to select the reading, press down the selection process number,  
press "M" the implementation of selected operations;  
Long press down the button, move the reference waveform locations,  
Long press the left button, hidden reference waveform,  
Long press the right button to display the reference waveform.  
Play starts scanning short or suspended by scanning,  
Press and Play directly to the SE menu.

Click the "M" switching table; a long time does not operate to automatically back to SE menu.

Save method parameters: Set a good start after at least 5 minutes to ensure, or can function in non-SE long press PLAY button under the table until you are prompted;

Hold down the "M" key to boot, you can use the default settings.

Time increment between two time the cursor into the frequency display function:

C1 or C2 in the next menu, click the down button can display time, display frequency, automatically switching between three modes alternately.

Voltage correction to use:

- a) zero correction, Hold down the left-turn, short the probe, open the measurement display. Press down button to select the correct short-stalls, short press the left or Right to show the average voltage closest to 0V.
  
- b) Gain correction  
Prepare a reference power, hold down the right boot, then the probe base power, open the measurement display. Press down button to select short. Correction of stalls, short press the left or right to display the average voltage closest to the reference power value.  
To achieve the best results, repeat this process several times, after a short press "M" key once, then press play button to save long.

Hold down the "M" key as the boot, will not use the information stored in the correction.

To avoid the slow USB charging oscilloscope display, the normal boot will be shielded USB communications, PC will be displayed at this time does not recognize the USB device


To use USB communications, press and hold the **PLAY** key to boot.



# Firmware upgrade

It's easy to upgrade firmware with USB bootloader.

1. Download "DfuSe USB Device Firmware Upgrade" from <http://www.st.com/stonline/products/support/micro/files/um0412.zip> and install. Instruction available at <http://www.st.com/mcu/familiesdocs-110.html#Application%20Note>.

2. Connect Oscilloscope with PC, press and hold  switch on power, until oscilloscope displays:

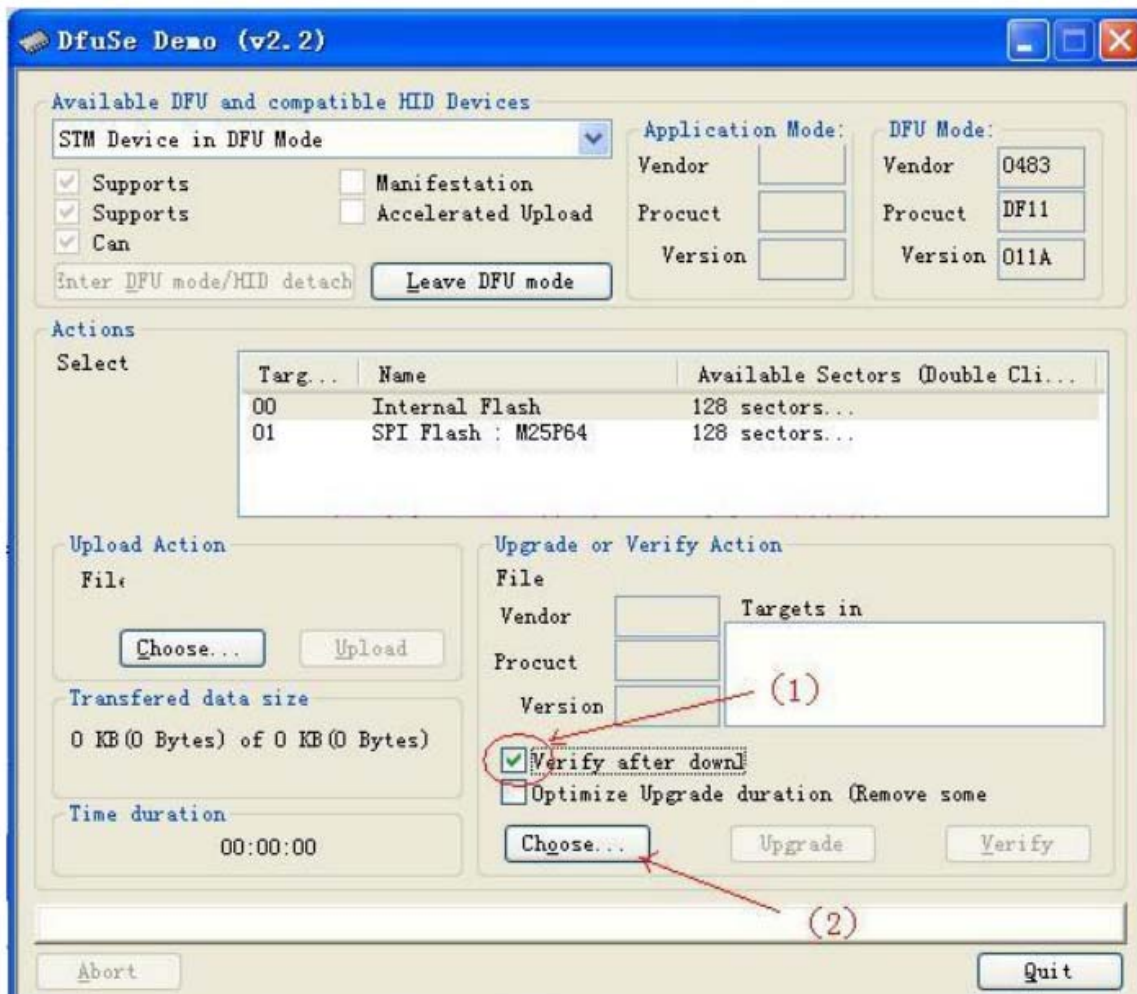
**"Please Connect to USB Host!"**  
**"DS0201 Device Firmware Upgrade Ver 2.0"**

When PC connection is detected,

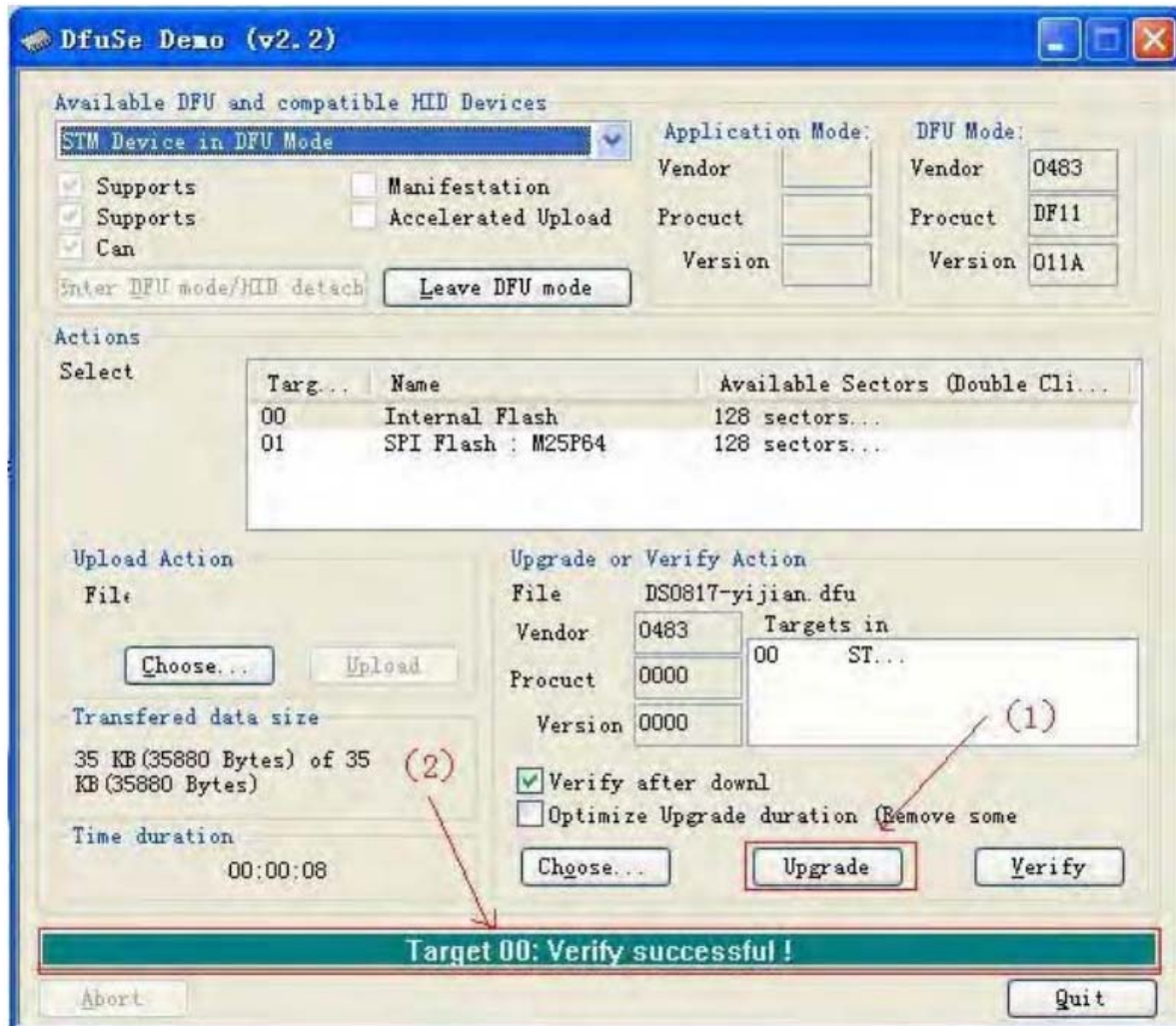
**"Firmware Upgrading..."**  
**"Please Wait"**

**"DS0201 Device Firmware Upgrade Ver 2.0"**

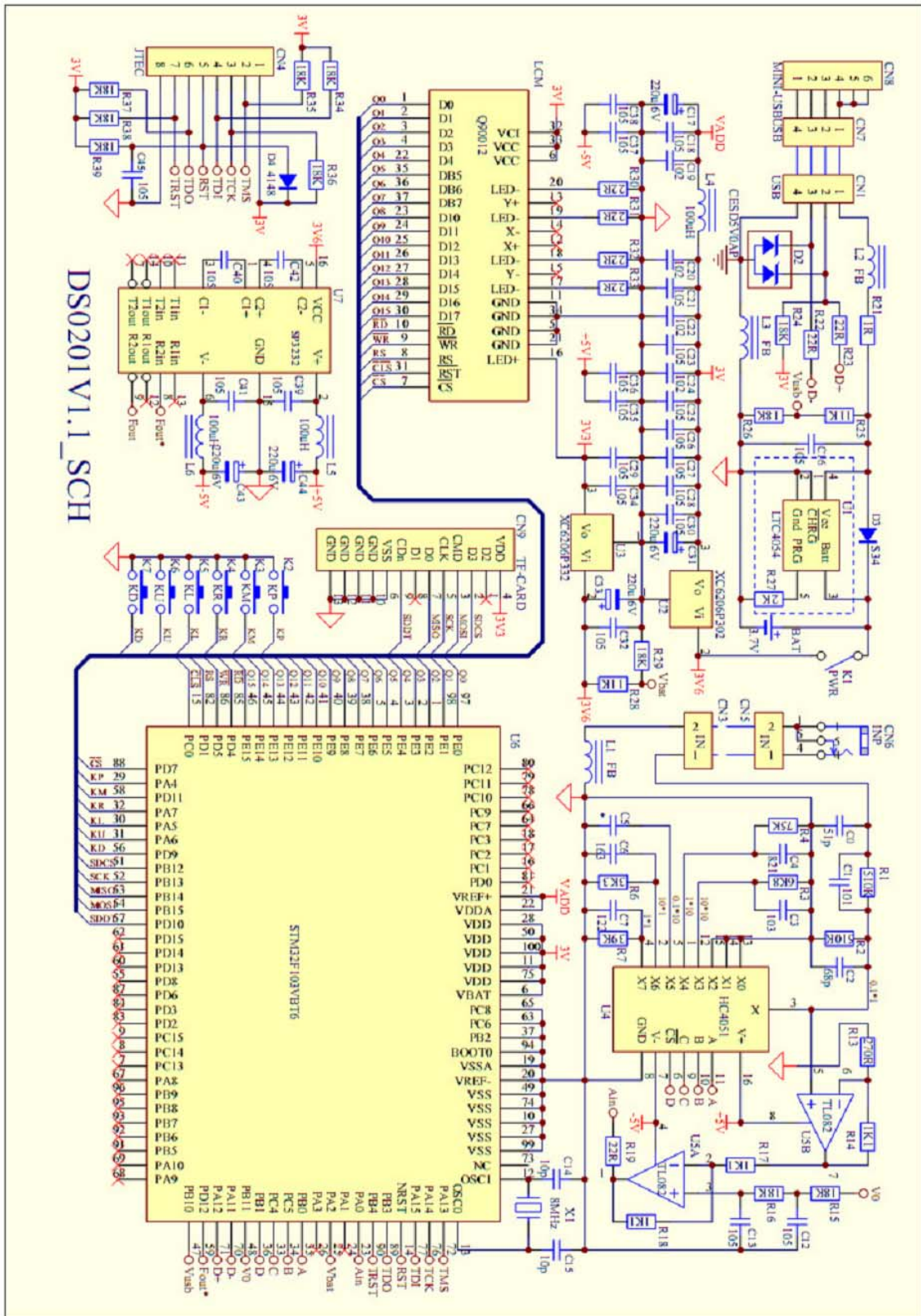
3. Run "Dfuse Demo" on PC, check (1) , select firmware to be uploaded (e.g."DS0201\_FW\_V2.00.DFU") at (2)



- In the next screen, press (1) "Upgrade", when upgrade finishes successfully, status bar will notify (2)



- Shut down and reactivate power to use new firmware.

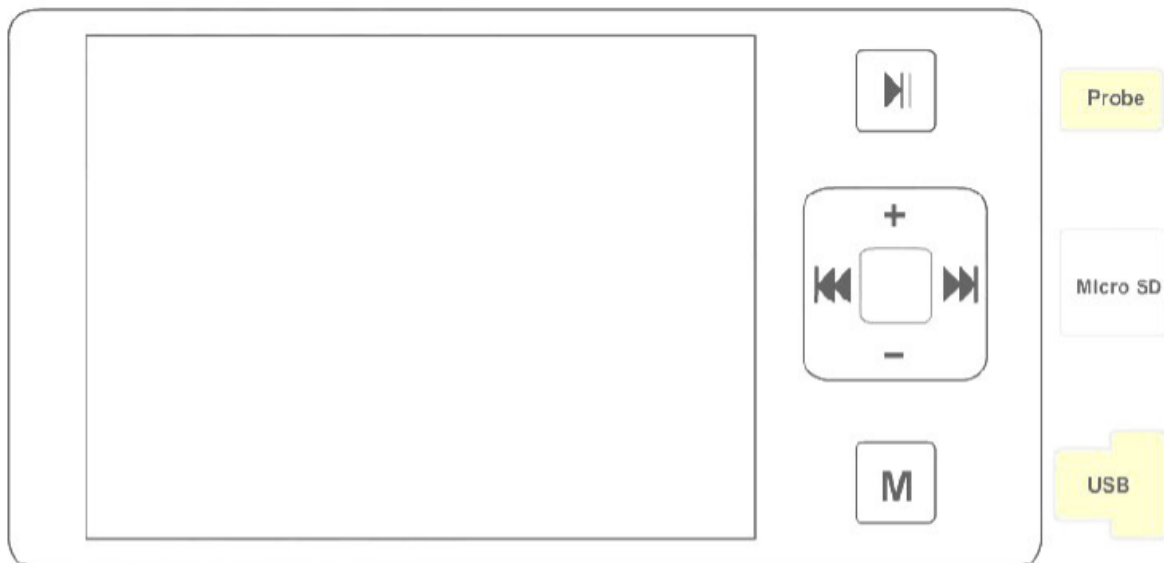






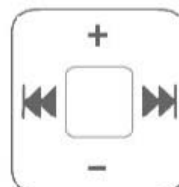
## ARM DSO Nano - Pocket-Sized Digital Oscilloscope

Firmware is updated.



 Run/Stop

 Menu



Value / Frequency  
(Div/Khz/Vo/us/ms/....)

### How to Charger the Battery ?

Turn off the power of DSO Nano , Connected the USB with PC or USB power adapter

### How to Connected with PC ?

Turn off the power ,Connected with USB to PC. Hold the Key "▶" and Turn on the power

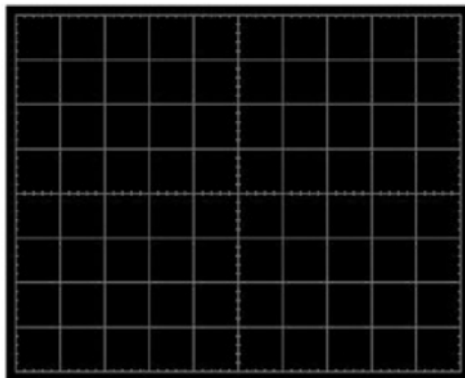
# Attachment:

## Customizing DSO Nano User Interface

DSO Nano is open source based, customizing the user interface is much easier than you thought. By linking the new UI to default variables and constants, you can build a individual interface yourself.

### Grid

The background of the DSO is grid , it help user to measure the waveform on the screen. Now the DSO have 8X10 grids, and each grid is 25X25 pixel. The entire grid is 300X200 pixels, here is where the screen display the waveform. To make more space for display another information, you can modify the grid into smaller size, but it need to be the times of 50 pixel.



You can modify the grid in the FUNction.h . Just change the definition of the below you can move the grid to any position and change its size:

```
#define X_SIZE 250
#define Y_SIZE 200
#define MIN_X 3
#define MIN_Y 24
#define MAX_X (X_SIZE + MIN_X)
#define MAX_Y (Y_SIZE + MIN_Y)
```

### Character

DSO has a font library in Lcd.c , just modify the font library you can change another typeface of character to display , and you can add more figure into the library to display , like "→".

```
//===== font library
=====
unsigned const short Char_Dot[760] =//744 12x6
0x0000,0x0000,0x0000,0x001C,0x0020,0x0040,/*" */
```

```
0x0040,0x0040,0x0020,0x001C,0x0000,0x0000,/*# */
0x0000,0x0000,0x0000,0xE000,0x1000,0x0800,/*$ */
0x60C0,0x9300,0x6D80,0x3240,0xC180,0x0000,/*% */
```

## Character string display function

You can find this function in the Lcd.c and use it you can show a string on the screen with any color and on any position.

```

/*****
Display_Str: display the string in assigned position
Input: X,Y,Color,Displaymode,string
*****/
void Display_Str(unsigned short x0, unsigned short y0, unsigned
short Color, unsigned char Mode, unsigned const char *s)
{
    .....
    while (*s!=0) {
        unsigned const short *scanline=Char_Dot+((*s-0x22)*6);
        for(i=0;i<6;++i){
            .....
            for(j=0;j<12;++j){
                if(b&16) {
                    .....
                    if(*s==0x21) x0 +=3;
                    else x0 += 6;
                    .....
                }
            }
        }
    }
}

```

unsigned short x0: X coordinate (0~319)

unsigned short y0: Y coordinate, (0~239)

unsigned short Color: color (will be reference in next section)

unsigned char Mode: display mode (PRN: normal; INV :Invert)

unsigned const char \*s: String

## Dot

You can use the two functions to drop a dot with any color in any position.

```
Point_SCR(x, y); // X coordinate (0~319) Y coordinate, (0~239)
```

```
Set_Pixel(color); //drop a color in the pixel set above
```

## Menu

The entire button signal is put into the Key\_Buffer. And you can read the key value to judge which key be pressed.

- ◆ KEYCODE\_PLAY
- ◆ KEYCODE\_LEFT
- ◆ KEYCODE\_RIGHT
- ◆ KEYCODE\_DOWN
- ◆ KEYCODE\_UP
- ◆ KEYCODE\_MANU

In the main.c , you can fine the code below. And modify the code here you can define a new menu for your DSO. The menu is made up by a loop.

```
Switch(Item) {
    case SYNC_MODE: //Menu name
```

```

if(Key_Buffer==KEYCODE_LEFT) Item=Y_VERNIER_2; //to forward menu
if(Key_Buffer==KEYCODE_RIGHT) Item=Y_SENSITIVITY //to next menu
if(Key_Buffer==KEYCODE_DOWN){ //the operation
    if .....
if(Key_Buffer==KEYCODE_UP){ .....//the operation
..... }
break;

case Y_SENSITIVITY:
break;
}

```

In the Function.h you can fine the define for every menu, the operation function is in the Function.c. You can add the new menu to create new function for you DSO .

```

#define SYNC_MODE 0
#define Y_SENSITIVITY 1
#define X_SENSITIVITY 2
#define Y_POSITION 3
#define MEASUR_KIND 4
#define POWER_INFOMATION 5
#define TRIG_SENSITIVITY 6
#define TRIG_SLOPE 7
#define INPUT_ATTENUATOR 8
#define SAVE_WAVE_CURVE 9
#define LOAD_WAVE_CURVE 10
#define OUTPUT_FREQUENCY 11
#define X_VERNIER_2 12
#define X_VERNIER_1 13
#define X_POSITION 14
#define RUNNING_STATUS 15
#define DELTA_T 16
#define Y_VERNIER_2 17
#define Y_VERNIER_1 18
#define TRIG_LEVEL 19
#define VERNIERS 20
#define WINDOW_AREA 21

```



## Color

In the Ldc.h you can find the definition for every color, just modify the definition you can change the color, and you can add the color define here to make your interface more colorful.

```
#define WHITE 0xFFFF //white: B = F800, G = 07E0, R = 001F
#define PANEL 0xFFE0 //panel: B = F800, G = 07E0, R = 0000
#define RED 0x001F //red: B = 0000, G = 0000, R = 001F
#define GRN 0x07E0 //green: B = 0000, G = 07E0, R = 0000
#define YEL 0x07FF //yellow: B = 0000, G = 07E0, R = 001F

#define GRID 0x738E //gray: B = 7000, G = 0380, R = 000E
#define CURVE 0x0F8E // green: B = 0001, G = 0780, R = 000E
#define MODEL 0xC05E // purple: B = C000, G = 0040, R = 001E
#define LINE 0xE79F // white: B = E000, G = 0780, R = 001F
#define BLACK 0x0000 //black: B = 0000, G = 0000, R = 0000
```

For example:

```
Point_SCR(2, 1);
```

```
Set_Pixel(YEL);
```

Will drop a yellow dot in the (2,1) pixel .

And modify the

```
#define GRID 0x07E0 //green
```

The grid will be change to green color.

Ok , now you can start working with your individual interface now. Have fun!