3.5inch LCD Shield

Fully assembled 3.5inch display with 480×320 pixels TFT LCD and a resistive touch overlay. Just plug it on top of ODROID-C2/C1+/C0. XU4 needs the Shifter-Shield board to use this display.

You will need an official Ubuntu image to configure it. You can simply download the Kernel updates (via dist-upgrade) and configure your ODROID for this display shield.

Where to buy

Hardware Schematic

Schematic : 3.5inch_lcd_rev0.1.pdf

Specification and pin informations

Specs & pin info

Configuring framebuffer and Touch driver

- You need a HDMI connection or a Serial/SSH console connection to follow the instruction below.
- Operation confirmed with our
  - ODROID-C1+

        - Ubuntu Mate 16.04.5 + kernel 3.10.107-192
        - Ubuntu Mate 18.04.1 + kernel 3.10.107-11
    - ODROID-C2
        - Ubuntu Mate 16.04.4 + kernel 3.14.79-117
        - Ubuntu Mate 18.04 + kernel 3.16.57-23
    - ODROID-XU4 with Shifter Shield
        - Ubuntu Mate 16.04.4 + kernel 4.14.47-132
        - Ubuntu Mate 18.04 + kernel 4.14.52-145
    - ODROID-N2
        - In ODROID-N2, console mode works but GUI desktop mode doesn't for now.
        - Ubuntu Minimal 18.04.2 + kernel 4.9.162-19

0. Update your Kernel

Updating kernel is highly recommended.
And install **fbset** package to make sure it installed.

```
$ sudo apt update && sudo apt full-upgrade
$ sudo apt install fbset
```

---

1. Insert modules

ODROID-C1/C2

```
# Omitted "$" sign for convenience

sudo modprobe aml_i2c
sudo modprobe pwm-meson
sudo modprobe pwm-ctrl
sudo modprobe fbtft_device name=flexpfb rotate=270
sudo modprobe flexfb chip=ili9488
sudo modprobe sx865x
```

This warning message **can be ignored** since we need only one PWM output.

```
"pwm-ctrl pwm-ctrl: cannot export to PWM-1 : modprobe pwm-meson npwm=2"
```

And you have to activate LCD shield by turning PWM on.

```
echo 500000 | sudo tee /sys/devices/platform/pwm-ctrl/freq0
echo 1 | sudo tee /sys/devices/platform/pwm-ctrl/enable0
echo 1023 | sudo tee /sys/devices/platform/pwm-ctrl/duty0
```

If you're using **Ubuntu 18.04 on C2**, enter the following commands instead.

```
echo 500000 | sudo tee /sys/devices/pwm-ctrl/freq0
echo 1 | sudo tee /sys/devices/pwm-ctrl/enable0
echo 1023 | sudo tee /sys/devices/pwm-ctrl/duty0
```

ODROID-XU3/XU4

First, To enable sx865x touch driver on **ODROID-XU4** you need to edit a dtb as following commands.

- Kernel Version == 3.10.y

```
$ sudo apt install device-tree-compiler
$ sudo cp /media/boot/exynos5422-odroidxu3.dtb /media/boot/exynos5422-
odroidxu3.dtb.old
$ sudo fdtput -t s /media/boot/exynos5422-odroidxu3.dtb
/hsi2c@12CB0000/sx865x@49 status "okay"
$ sudo reboot
```

- Kernel Version >= 4.9.y
  - ODROID-**XU4** Board dtb file : exynos5422-odroidxu4.dtb
  - ODROID-**XU3** Board dtb file : exynos5422-odroidxu3.dtb

Following are **the example for ODROID XU4** users. If you try with ODROID XU3, edit entered dtb file name.

```
$ sudo apt install device-tree-compiler
$ sudo cp /media/boot/exynos5422-odroidxu4.dtb /media/boot/exynos5422-
odroidxu4.dtb.old
$ sudo fdtput -t s /media/boot/exynos5422-odroidxu4.dtb
/soc/i2c@12cb0000/sx865x@49 status "okay"
$ sudo reboot
```

After editing the device tree for your platform, then load the modules.

- If the device **/dev/fb*** is not created and the following messages appears from **dmesg**,

```
[  621.952761] flexfb_probe_common : ioremap gpiox register success!
[  621.957496] flexfb flexpfb.0: fbtft_request_gpios:
gpio_request_one('wr'=190) failed with -16
```

- Enter the following command to unload SPI modules and try again.

```
$ sudo modprobe -r spidev
```

```
# Omitted "$" sign for convenience

sudo modprobe fbtft_device name=flexpfb rotate=270
sudo modprobe flexfb chip=ili9488
```

ODROID-N2

First, To enable sx865x touch driver on **ODROID-N2** you need to edit a dtb using the following commands.

```
$ sudo apt install device-tree-compiler
$ sudo cp /media/boot/meson64_odroidn2.dtb
/media/boot/meson64_odroidn2.dtb.old
$ sudo fdtput -t s /media/boot/meson64_odroidn2.dtb
/soc/cbus@ffd00000/i2c@1c000/sx865x@49 status "okay"
$ sudo reboot
```

After editing the device tree for your platform, then load the module.

- If the device **/dev/fb*** is not created and the following messages appears from **dmesg**,

```
[   23.260649] flexfb flexpfb.0: fbtft_request_gpios:
gpio_request_one('wr'=486) failed with -16
[   23.265899] flexfb: probe of flexpfb.0 failed with error -16
```

- Enter the following command to unload SPI modules and try again.

```
$ sudo modprobe -r spidev spi_meson_spicc
```

```
# Omitted "$" sign for convenience

sudo modprobe fbtft_device name=flexpfb rotate=270
sudo modprobe flexfb chip=ili9488
```

---

2. Check if the modules loaded

You will have a new frame buffer **/dev/fbX**.
Find a node named **flexfb**.

- **FOR ODROID XU4 USERS**: It's different between two conditions.

    - When the HDMI connected and you're working with a monitor.
    - When any monitor isn't connected and you're working via SSH or serial connection.
    - If the monitor connected, the frame buffer device 0(**/dev/fb0**) would be assigned at the connected monitor. So the other frame buffer could be for a LCD shield, like **/dev/fb1**.
- **FOR ODROID C1/C2 USERS**: The frame buffer 0, 1 (**/dev/fb{0,1}**) always exists. Proceed with **/dev/fb2** or other named **flexfb**.
- **THE BEST WAY** is that find out which frame buffer device has a name **flexfb** by yourself.
- This guide assumes that you're working on **ODROID C2**. In this case, the frame buffer device for a LCD shield is **/dev/fb2**.

```
$ ls /dev/fb*
# results like..
/dev/fb2
$ cat /sys/class/graphics/fb2/name
# results..
flexfb
```

Run Console

1. Run con2fbmap

Map the console to the frame buffer device assigned at the LCD shield.
This guide assumes that you're working on **ODROID C2**, so the proper frame buffer device would be **/dev/fb2**.

```
# Usage: con2fbmap {CONSOLE} [FRAME_BUFFER]
# If the frame buffer isn't specified, it shows the currently mapped frame buffer.
$ sudo con2fbmap 1
# results
```

```
console 1 is mapped to framebuffer

$ sudo con2fbmap 1 2
$ sudo con2fbmap 1
console 1 is mapped to framebuffer 2
```

2. Change foreground virtual terminal

Change the foreground screen by using the command **chvt**. That means you will move to the selected console.
[The command chvt N makes /dev/ttyN the foreground.
Since the frame buffer device for the console #1 is assigned at the LCD shield, terminal screen shows up at the LCD shield after entering the command.
If you face a problem with enter **chvt** command, try again via **SSH or serial(UART)**.

```
# Usage: chvt {CONSOLE_NUM}
$ sudo chvt 1
```

Opt. Auto login on console

Edit tty1 service.

```
$ sudo systemctl edit getty@tty1
```

Add the following codes and save by pressing **Ctrl + K and X** when if JOE editor shown.

```
[Service]
ExecStart=
ExecStart=-/sbin/agetty -a odroid --noclear %I $TERM
```

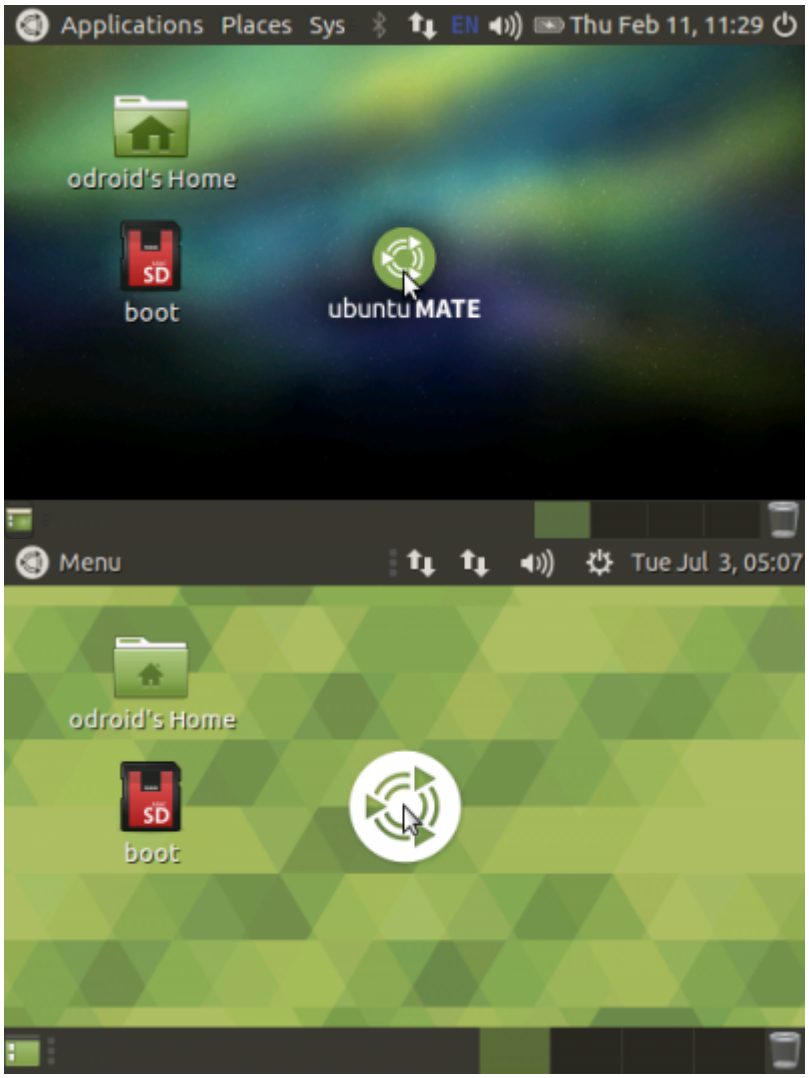
Check if it saved.

```
$ sudo systemctl cat getty@tty1 | grep Exec
# results
ExecStart=-/sbin/agetty --noclear %I $TERM
ExecStart=
ExecStart=-/sbin/agetty -a odroid --noclear %I $TERM
```

Restart tty1 service.

```
$ sudo systemctl restart getty@tty1
```

Run Xwindow



1. Create a new config file

```
$ sudo mv /etc/X11/xorg.conf /etc/X11/xorg.conf.bak

# Enter this command too if you're using XU4. This file might be there by default but to make sure it doesn't exist.
$ sudo rm /etc/X11/xorg.conf.d/exynos.conf
```

This guide assumes that you're working on **ODROID C2**, so the proper frame buffer device would be **/dev/fb2**.
This step is for set up the target frame buffer device to show the X-window (X11) screen. Since the frame buffer device **/dev/fb2** is assigned at the LCD shield, X-window screen will show up at the LCD shield.
Add following lines into the **/etc/X11/xorg.conf** file.

```
$ sudo vi /etc/X11/xorg.conf
```

```
Section "Device"
```

```
    Identifier    "C fbdev"
    Driver        "fbdev"
    Option        "fbdev" "/dev/fb2"
EndSection
```

If you're using **Ubuntu 18.04 on C2**, edit **/usr/share/X11/xorg.conf.d/99-odroidc2.conf** file instead.
Change **/dev/fb0** to **/dev/fb2**.

```
$ cd /usr/share/X11/xorg.conf.d
$ sudo cp 99-odroidc2.conf 99-odroidc2.conf.bak
$ sudo vi 99-odroidc2.conf
```

Tip : If you rotate the screen 180°, the touchscreen coordination must be rotated too.
You have add this line into “/usr/share/X11/xorg.conf.d/99-odroidc2.conf” file.

```
Option "TransformationMatrix" "-1 0 1 0 -1 1 0 0 1"
```

This nice tip was made by forum user @mad_ady.

2. Run con2fbmap

- If you have a problem with running X-window on your LCD module, please try again after following **“Create a new config file” (Edit /etc/X11/xorg.conf) instruction**.

Map the X-window console to the frame buffer device for the LCD shield.

```
# Usage: con2fbmap {CONSOLE} [FRAME_BUFFER]
# If the frame buffer isn't specified, it shows the currently mapped frame buffer.
$ sudo con2fbmap 7 2
$ sudo con2fbmap 7
# results
console 7 is mapped to framebuffer 2
```

3. Change foreground virtual terminal

Change the foreground screen by using the command **chvt**. That means you will move to the selected console.
The command chvt N makes /dev/ttyN the foreground.
Since the frame buffer device for the console #7 is assigned at the LCD shield, make the console that LCD shield displays to #7.
If the display manager ready, graphical desktop screen will appear.
If you face a problem with enter **chvt** command, try again via **SSH or serial(UART)**.

```
# Usage: chvt {CONSOLE_NUM}
$ sudo chvt 7
```

4. Restart display manager

Restart display manager to show up the screen on the LCD shield.

```
$ sudo service lightdm restart
```

Then you can see graphical desktop on your LCD shield.

Opt. Auto login on X-Window

Edit **/etc/lightdm/lightdm.conf** file.

```
$ sudo vi /etc/lightdm/lightdm.conf
```

Add the following lines.

```
[SeatDefaults]
autologin-user=odroid
autologin-user-timeout=0
```

Save and restart lightdm.

```
$ sudo service lightdm restart
```

Backlight control

- ODROID-C1/C2 only.
- ODROID-XU4 does not use PWM control.

Set PWM frequency and enable it.

```
$ echo 500000 | sudo tee /sys/devices/platform/pwm-ctrl/freq0
$ echo 1 | sudo tee /sys/devices/platform/pwm-ctrl/enable0
```

You can enter **0 ~ 1023**(total 1024 step) to the **duty0** file to adjust backlight brightness of the LCD shield.

```
$ echo 1023 | sudo tee /sys/devices/platform/pwm-ctrl/duty0
```

If you're using **Ubuntu 18.04 on C2**, be careful for the changed **pwm-ctrl path**.

```
$ echo 500000 | sudo tee /sys/devices/pwm-ctrl/freq0
$ echo 1 | sudo tee /sys/devices/pwm-ctrl/enable0
$ echo 1023 | sudo tee /sys/devices/pwm-ctrl/duty0
```

Touch Calibration

1. Setup

Install the calibrator.

```
$ sudo apt install xinput-calibrator xserver-xorg-input-evdev
```

2. Run calibrator

- If your display mode is portrait(rotate 0 or 180), run the following command.

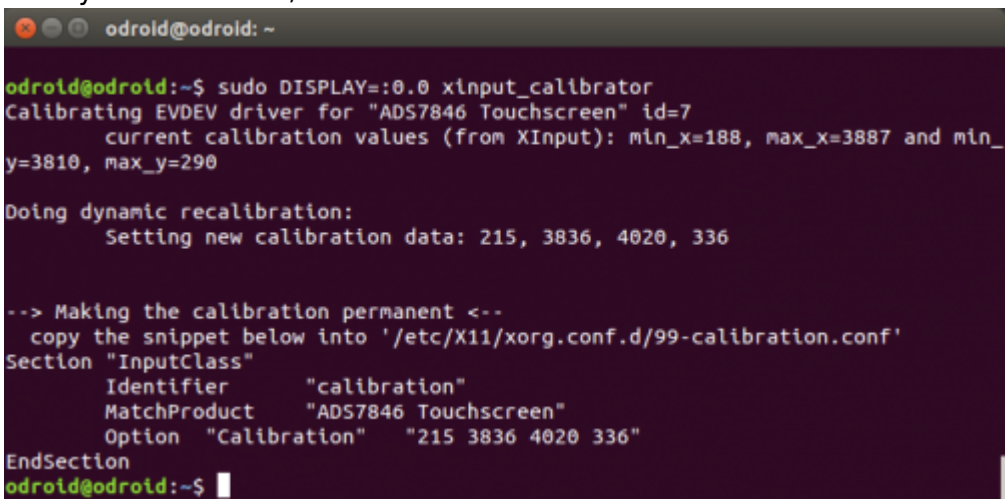
```
$ sudo DISPLAY=: xinput set-prop 'SX865X Touchscreen' 'Evdev Axes Swap'
```

- Here is further information about configuration for event driver on X.org.
  - https://www.x.org/releases/X11R7.5/doc/man/man4/evdev.4.html

Enter the command below and make sure you will enter via installed 3.5 inch LCD shield.

```
$ sudo DISPLAY=: xinput_calibrator
```

Follow the directions on your screen.
Then you will see like,



You should add a property **“Driver”** to the result yourself.
Then it will be like,

```
Section "InputClass"
    Identifier    "calibration"
    MatchProduct    "SX865X Touchscreen"
    Driver  "evdev"
    Option    "Calibration"    "89 3848 254 3854"
    Option    "SwapAxes"    "0"
EndSection
```

Copy that result to **/etc/X11/xorg.conf.d/99-calibration.conf**.

```
$ sudo mkdir /etc/X11/xorg.conf.d/
$ sudo vi /etc/X11/xorg.conf.d/99-calibration.conf

# Ubuntu 18.04 on C2
$ sudo vi /usr/share/X11/xorg.conf.d/99-calibration.conf
```

Restart display manager.

```
$ sudo service lightdm restart
```

How to use keypad

There are four tact switches on LCD shield. The switches will change a ADC value when you press these buttons.
Each sysfs nodes are:

- C2 - /sys/class/saradc/ch0
- C1 - /sys/class/saradc/saradc_ch0
- XU4 - /sys/devices/12d10000.adc/iio\:device0/in_voltage3_raw

SBC	switch #	ADC value
C2/C1	SW1	5 ±10
	SW2	515 ±10
	SW3	680 ±10
	SW4	770 ±10
XU4	SW1	0 ±100
	SW2	2030 ±100
	SW3	2695 ±100
	SW4	3014 ±100

Build Wiring Pi

```
$ sudo apt install git
$ git clone https://github.com/hardkernel/wiringPi
$ cd wiringPi
$ sudo ./build
```

Build a source code

4 Keys on the shield will be mapped to **SPACE, UP, DOWN, ENTER** input event.

ODROID-C2/C1

tftlcd35_key.c

```
$ wget https://dn.odroid.com/source_peripherals/lcd35/tftlcd35_key.c
```

Compile & Run.

```
$ gcc -o tftlcd35_key tftlcd35_key.c -lwiringPi -lwiringPiDev -lm -lpthread
-lrt -lcrypt
$ sudo ./tftlcd35_key &
```

ODROID-XU4

tftlcd35_key_xu4.c

```
$ wget https://dn.odroid.com/source_peripherals/lcd35/tftlcd35_key_xu4.c
```

Compile & Run.

```
$ gcc -o tftlcd35_key_xu4 tftlcd35_key_xu4.c -lwiringPi -lwiringPiDev -lm -
lpthread -lrt -lcrypt
$ sudo ./tftlcd35_key_xu4 &
```

Auto Run

- Automatically start console mode
- Automatically start desktop GUI mode

Applications

- Using the mplayer on framebuffer
- Multiclick button handler by mad_ady