3.5 Inch Spi Display

User Manual V1.0

1.1 Overview:

3.5 Inch Spi Display is a display module can be applied to Raspberry pi ,it can be used as raspberry pi x window display terminals .It use the Raspberry pi high-speed SPI interface transfer Linux frame buffer data to the LCD module. The SPI clock can reach as high as 63Mbps. The 3.5 LCD panel with this module have 16bit/pixel 320x480 high-resolution.

This module equipped with high accuracy DS3231 hardware RTC unit make your Raspberry system time no longer depend on the network.

The LCD Module can equip with resistive or capacitive touch screen. Two types of touch screen module can use the same software package.

The LCD module used SPI, I2C and some GPIO signal from Raspberry pi 40PIN interface. Other resource in 40PIN interface can be defined by users.

The Module have the same size as Raspberry pi and can be directly plugged on the Raspberry pi.

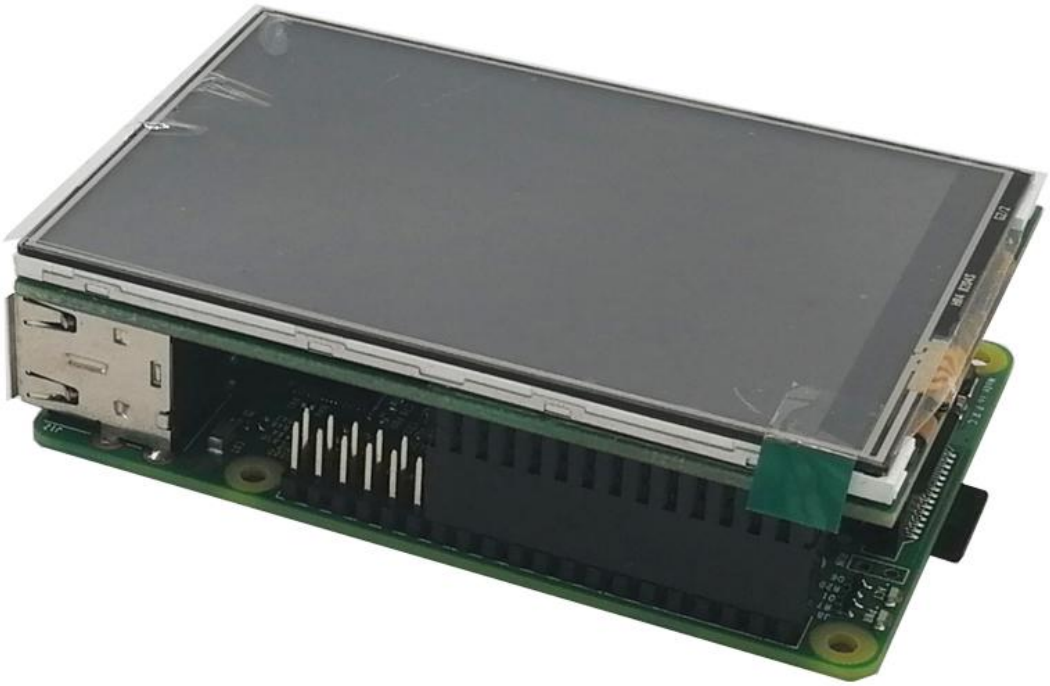
1.2 Features：

1, 16bit/pixel, 320x480 resolution and large view angle.
2, with both resistive and capacitive touch specifications, the software fully compatible.
3, Providing preset raspbian-jessie system image and a separate installation package.
4, Provide accurate DS3231 RTC unit to ensure that the system time is accurate.
5, LCD Backlight can be controlled.
6, Hardware and software is compatible with any version of the raspberry pi.
7, It have the same size as standard raspberry pi.
8, Reserved ID EEPROM location on board.

1.3 Hardware connection

The LCD modules used 28 pins out of raspberry pi 40 pin.

When installing the module attention to align the first leg of the raspberry pi and LCD module.



1.4 Software Installation

We offer two software installation methods：

1.4.1, Install the software image we preset.

Prepare a capacity of more than 8GB TF card and card reader decompression the package：LCD35-Raspbian-jessie-170705.zip and we can get IMG files then open Win32DiskImager.exe and select IMG click 「write」wait for it complete .Inset the TF card into the Raspberry pi then power on the system.

1.4.2, install stand-alone package we provide

First we download the OS image from https://www.raspberrypi.org/downloads/raspbian/ and Install the OS image into your TF card.
Here we are with 2017-07-05-raspbian-jessie.img, for example：

Step1， First we use Win32DiskImager tool flush the OS image into TF Card.

Step2， Copy the stand-alone package file setup_rpi.tar.bz into your USB flash disk（because setup.tar file size exceeds the remaining capacity of the boot partitions so we can't copy setup.tar directly into boot partitions）.

Step3， Insert the TF Card into your raspberry pi and power on the system.

Notice：

A、If you want to use the Debug serial port terminal to interface with the raspberry pi, due to the raspberry pi3 serial debugging is not enabled by default you can't get the debug information from the terminal and will not be able to interface with the system.

How To: Copy the file cmdline.txt and config.txt from *pi3 debug enable* directory and overwrite the corresponding files in TF card boot partition.

B、If you need to install the package on the system you are using, it is strongly recommended that you back up all of the TF content in order to avoid data loss caused by improper operation.

Step4, After system booting finished plug the USB flash drive into the USB port and copy the setup-rpi.tar file into the current user directory:

```
pi@raspberrypi:~$ cp /media/pi/disk/setup_rpi.tar    ./
pi@raspberrypi:~$ pwd
/home/pi
pi@raspberrypi:~$ ls
 Desktop      Downloads   Pictures   python_games     Templates
Documents    Music         Public      setup_rpi.tar   Videos
```

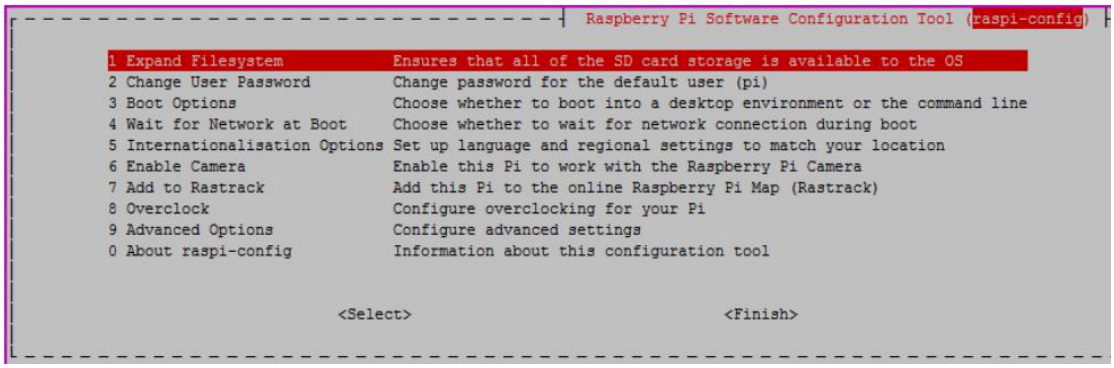
Step5 See TF card free space：

```
pi@raspberrypi:~$ df -lh
Filesystem          Size   Used Avail Use% Mounted on
/dev/root           4.2G   3.9G     66M   99% /
devtmpfs            458M        0   458M     0% /dev
tmpfs               462M        0   462M     0% /dev/shm
tmpfs               462M   6.3M   456M     2% /run
tmpfs               5.0M   4.0K   5.0M     1% /run/lock
tmpfs               462M        0   462M     0% /sys/fs/cgroup
/dev/mmcblk0p1       42M     21M     21M    51% /boot
tmpfs                93M        0     93M     0% /run/user/1000
/dev/sda1           3.5G   3.1G   372M    90% /media/pi/disk
```

Obviously, we have only 66MB of space left for use, we need to extend the root file system partition to the whole TF card. There are two ways we can select:

A Method 1,

$sudo raspi-config



Select 1，Expand File systems　select and confirm then execute.

$df –lh

Then you can see the capacity of the root file system has the same size.

However Raspberry pi.org the latest release of software image when use we found the first item Expand File system has been removed, we can't use such method any more.

B　Method 2,

```
pi@raspberrypi:~$ cat /sys/block/mmcblk0/mmcblk0p2/start
94208

pi@raspberrypi:~$ sudo fdisk /dev/mmcblk0

Welcome to fdisk (util-linux 2.25.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help): d
Partition number (1,2, default 2): 2

Partition 2 has been deleted.

Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 2): 2
First sector (2048-15564799, default 2048): 94208
Last sector, +sectors or +size{K,M,G,T,P} (94208-15564799, default 15564799):

Created a new partition 2 of type 'Linux' and of size 7.4 GiB.

Command (m for help): w
The partition table has been altered.
```

```
Calling ioctl() to re-read partition table.
Re-reading the partition table failed.: Device or resource busy
```

The kernel still uses the old table. The new table will be used at the next reboot or after you run partprobe(8) or kpartx(8).

Reboot system:

```
pi@raspberrypi:~$ sudo reboot
```

After rebooting the system we also need to perform resize2fs command.

```
pi@raspberrypi:~$ sudo resize2fs /dev/mmcblk0p2
resize2fs 1.43.3 (04-Sep-2016)
Filesystem at /dev/mmcblk0p2 is mounted on /; on-line resizing required
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/mmcblk0p2 is now 1933824 (4k) blocks long.
```

Use the df –lh command again and we found the system free space has become larger.

```
pi@raspberrypi:~$ df -lh
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       7.2G  3.9G  3.0G  57% /
devtmpfs        458M     0  458M   0% /dev
tmpfs           462M     0  462M   0% /dev/shm
tmpfs           462M  6.3M  456M   2% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           462M     0  462M   0% /sys/fs/cgroup
/dev/mmcblk0p1   42M   21M   21M  51% /boot
tmpfs            93M     0   93M   0% /run/user/1000
/dev/sda1       3.5G  3.1G  372M  90% /media/pi/disk
```

Step6, unzip the package ：

```
pi@raspberrypi:~$ tar   -xvf setup_rpi.tar
```

Notice： due to the release of the package content is new, if your system time is not updated the following warning may appear.

```
tar: setup/firmware/setup.tar: time stamp 2017-07-26 09:41:50 is 1804436.157124564 s in the future
```

How To：you can set the system time to the current time.

```
pi@raspberrypi:~$ sudo date 072611282017.30
Wed 26 Jul 11:28:30 UTC 2017

pi@raspberrypi:~$ ls
Desktop    Downloads  Pictures  python_games  setup_rpi.tar  Videos
Documents  Music      Public    setup         Templates
```

Into the installation package：

```
pi@raspberrypi:~$ cd ./setup/
pi@raspberrypi:~/setup$ ls
firmware  lcd_setup
```

```
pi@raspberrypi:~/setup$ cd ./firmware/
pi@raspberrypi:~/setup/firmware$ ls
setup_rpi.sh    setup.tar
```

step7, Updating system firmware：

```
pi@raspberrypi:~/setup/firmware$ ./setup_rpi.sh
```

step8,Config your LCD ：

```
pi@raspberrypi:~/setup/firmware$
pi@raspberrypi:~/setup/firmware$ cd ..
pi@raspberrypi:~/setup$ ls
firmware    lcd_setup
pi@raspberrypi:~/setup$ cd lcd_setup/
pi@raspberrypi:~/setup/lcd_setup$ ls
boot    etc    input_rule    overlays    rtc    setup.sh    user-app    usr
pi@raspberrypi:~/setup/lcd_setup$ ./setup.sh
Setup    3.5 inch TFT and touch panel on RPI!
Configure the LCD display 0 degrees
System setup is complete, restart the system!
```

Taking into account the needs of different display scene, we can set the LCD screen to rotate 0 90 180 270 degree, executing installation and configuration instructions as follow:

```
./lcd_setup          :LCD screen display properly.
./lcd_setup   90     :LCD screen display is rotated 90 degrees.
./lcd_setup   180    : LCD screen display is rotated 180 degrees.
./lcd_setup   180    : LCD screen display is rotated 270 degrees.
```

Notice:

If the system LCD display properly but the touch screen didn’t work normal (while click you even feel the coordinates are reversed) indicate：

The system need to install *Xorg-input-edev* .

How TO：

First connect your RPI to the internet

```
sudo apt-get install xserver-xorg-input-evdev
sudo cp -rf /usr/share/X11/xorg.conf.d/10-evdev.conf /usr/share/X11/xorg.conf.d/45-evdev.conf
```

Reboot your system then the touch screen can work now.

```
pi@raspberrypi:~$ sudo apt-get install xserver-xorg-input-evdev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
```

```
        xserver-xorg-input-evdev
    0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
    Need to get 116 kB of archives.
    After this operation, 168 kB of additional disk space will be used.
    Get:1 http://archive.raspberrypi.org/debian/ jessie/main xserver-xorg-input-evdev
armhf 1:2.10.3-1 [116 kB]
    Get:2 http://archive.raspberrypi.org/debian/ jessie/main xserver-xorg-input-evdev
armhf 1:2.10.3-1 [116 kB]
    Fetched 33.6 kB in 32s (1,021 B/s)
    Selecting previously unselected package xserver-xorg-input-evdev.
    (Reading database ... 115404 files and directories currently installed.)
    Preparing to unpack .../xserver-xorg-input-evdev_1%3a2.10.3-1_armhf.deb ...
    Unpacking xserver-xorg-input-evdev (1:2.10.3-1) ...
    Processing triggers for man-db (2.7.5-1~bpo8+1) ...
Setting up xserver-xorg-input-evdev (1:2.10.3-1) ...
```

1.5 Touch screen calibration program install

Touch screen we have calibrated by default, does not need to calibrate any more, if for some reason you need to calibrate the touch screen you can carry out by *xinput_calibration* program.

How to install:

Step 1

```
pi@raspberrypi:~/setup/lcd_setup/user-app$ sudo dpkg -i -B
xinput-calibrator_0.7.5-1_armhf.deb
Selecting previously unselected package xinput-calibrator.
(Reading database ... 115395 files and directories currently installed.)
Preparing to unpack xinput-calibrator_0.7.5-1_armhf.deb ...
Unpacking xinput-calibrator (0.7.5-1) ...
Setting up xinput-calibrator (0.7.5-1) ...
Processing triggers for gnome-menus (3.13.3-6) ...
Processing triggers for desktop-file-utils (0.22-1) ...
Processing triggers for mime-support (3.58) ...
Processing triggers for man-db (2.7.5-1~bpo8+1) ...
```

Click on the system tray Menu（RPI icon）,Select Preferences→Calibrate Touch screen.

After calibration there will be a pop-up window displays the coordinates of the calibrated value: Option “Calibration” “xxx xxx xxx xxx”

In order to make the touch screen calibration effective when next time power up you need to update those coordinates values to file:

/etc/X11/xorg.conf.d/99-calibration.conf .

1.6 Set the hardware RTC

Due to cost and size Raspberry pi didn't put hardware RTC on board. Updates the system time need to connect to the Internet via NTP time service. When there isn't Internet can access the system time will not be accurate. This module designed an accurate DS3231 RTC module.

Setting as follow:

step1

```
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo apt-get -y remove fake-hwclock
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  fake-hwclock
0 upgraded, 0 newly installed, 1 to remove and 0 not upgraded.
After this operation, 74.8 kB disk space will be freed.
(Reading database ... 115401 files and directories currently installed.)
Removing fake-hwclock (0.9) ...
Processing triggers for man-db (2.7.5-1~bpo8+1) ...
```

Step2

```
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo update-rc.d –f fake-hwclock remove
pi@raspberrypi:~/setup/lcd_setup/rtc$ ls
hwclock-set    setup.sh
```

step3

```
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo cp hwclock-set    /lib/udev/hwclock-set
pi@raspberrypi:~/setup/lcd_setup/rtc$
```

Set System Time: For example, on July 26, 2017 at 16:22 30s

```
sudo date 072616222017.30
```

Time Value set to the hardware RTC.

```
sudo hwclock –w
```

```
sudo hwclock –D –r
```

sudo hwclock –r you can view the hardware RTC time.

Power-off and reboot the system to check whether the system time have been successfully updated by RTC unit.

```
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo date 072616222017.30
Wed 26 Jul 16:22:30 UTC 2017
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo hwclock  -w
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo hwclock
Wed 26 Jul 2017 16:22:39 UTC  -0.524764 seconds
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo hwclock
Wed 26 Jul 2017 16:23:02 UTC  -0.492084 seconds
pi@raspberrypi:~/setup/lcd_setup/rtc$ sudo hwclock -D -r
hwclock from util-linux 2.25.2
Using the /dev interface to the clock.
Last drift adjustment done at 1501086155 seconds after 1969
Last calibration done at 1501086155 seconds after 1969
Hardware clock is on UTC time
Assuming hardware clock is kept in UTC time.
Waiting for clock tick...
/dev/rtc does not have interrupt functions. Waiting in loop for time from /dev/rtc to change
...got clock tick
Time read from Hardware Clock: 2017/07/26 16:23:44
Hw clock time : 2017/07/26 16:23:44 = 1501086224 seconds since 1969
Wed 26 Jul 2017 16:23:44 UTC  -0.442868 seconds
pi@raspberrypi:~/setup/lcd_setup/rtc$
```

1.7 LCD backlight control

LCD Module support backlight control, it use GPIO18 hardware resource, the backlight can be controlled through the following ways:

Turn off the backlight： echo 1 | sudo tee /sys/class/backlight/fb_ili9486/bl_power

Turn on the backlight： echo 0 | sudo tee /sys/class/backlight/fb_ili9486/bl_power