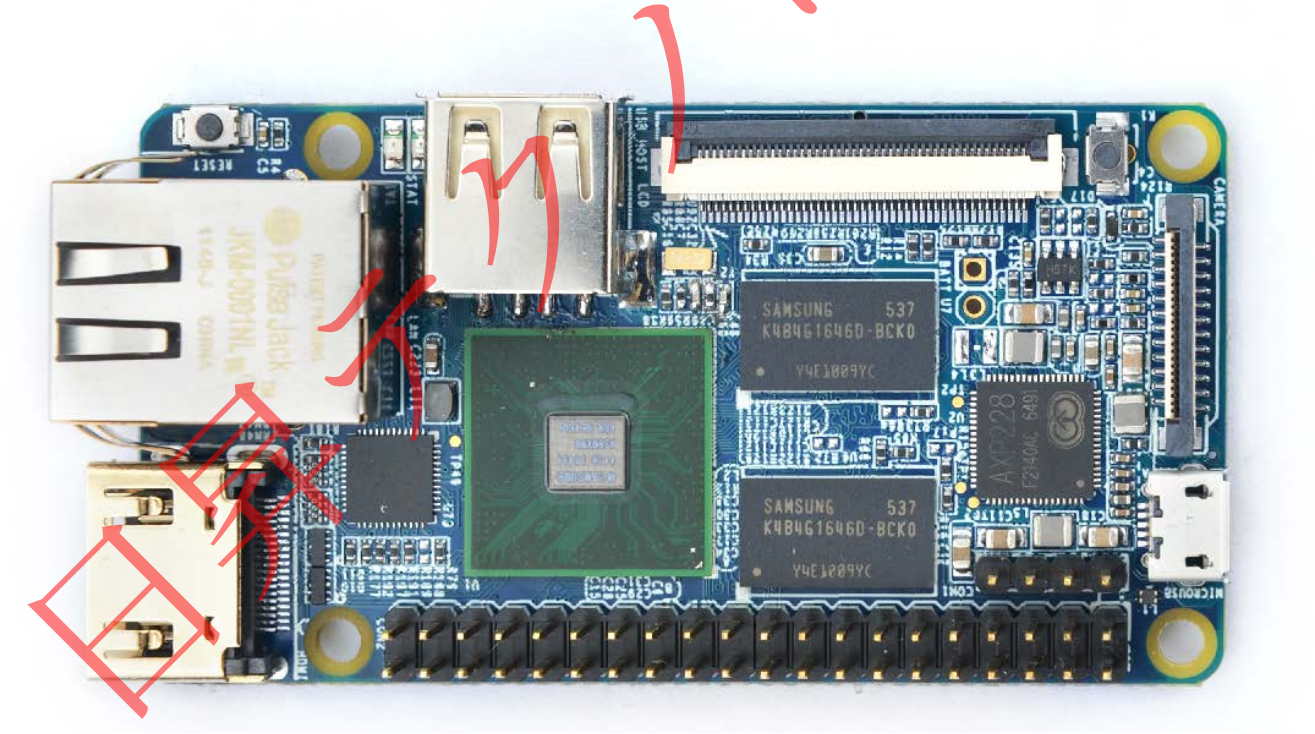
Cortex-A9 4コア S5P4418 ボード
NanoPi2 Fire 簡易マニュアル

株式会社日昇テクノロジー
http://www.csun.co.jp
info@csun.co.jp

作成日 2016/6/15



• 修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2016/06/15

目 次

1 紹介

NanoPi2 FireはIoT設計のために開発された高性能のARMマスタコントロールボードである。 SamsungのCortex-A9クアッド４コア S5P4418、1.4GHz、SoC 1G 32ビット DDR3 RAMを備えており、Gbpsイーサネットポートを搭載している。TFカードからandroidと Debianシステムを実行することができ、HMDIとLCD インターフェースを搭載。Raspberry PiのGIPOと互換性を持つ。 PCBのサイズは75×40mmである。

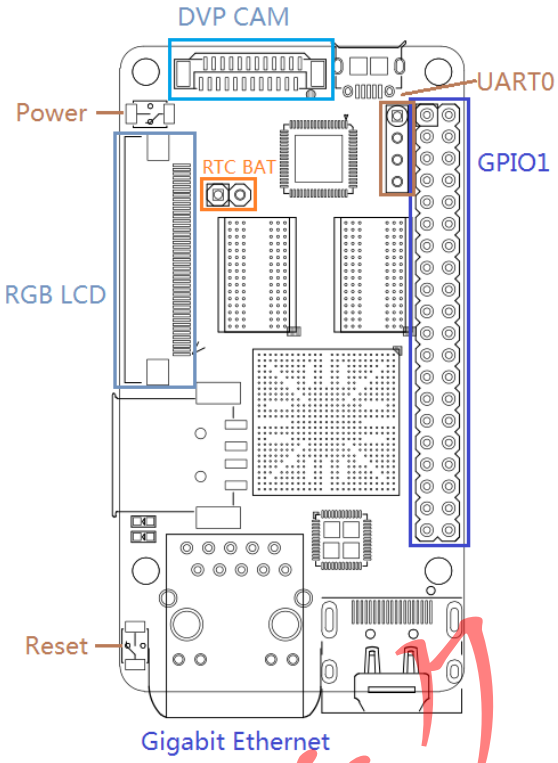
2 主な仕様

CPU：Samsung S5P4418、動作周波数1.4GHz
RAM：1GB DDR3
PMU電源管理：AXP228を搭載、ソフトウェアシャットダウン、スリープ、ウェイクアップをサポート
ネットワーク： ギガビットGbpsイーサネットポート
USB2.0 タイプA x1
デバッグ用シリアルポートUARTOx1
microSD Slot x1
microUSB x1：給電とデータ伝送
LCD I/F：0 . 5 mmピッチSMT FPCシート、フルカラーLCDをサポート（RGB：8-8-8）
HDMI: Type-Aコネクタ、1080P60出力をサポート
DVPカメラ I/F： 0.5mmピッチ省スペースタイプFPCソケット、ITU-R BT 601/656 8ビット、I2CおよびIOを含む。
GPIO：2.54mmピッチ、40ピン、RasberryPiのGPIOと互換性がある 。UART、SPI、I2C、IOなどを含む。
ボタン：電源ボタンx１、リセットボタンx１
LED: 電源LEDx1、システムLEDx1
PCBサイズ：75 x 40 mm
電源：DC 5V/2A
OS: Android、 Debian

3 インターフェースの配置及びサイズ

3.1 インターフェースの配置

3.1.1 GPIO1 ピン定義



Pin#	Name	Pin#	Name
1	SYS_3.3V	2	VDD_5V
3	I2C0_SDA	4	VDD_5V
5	I2C0_SCL	6	DGND
7	GPIOD8/PPM	8	UART3_TXD/GPIOD21
9	DGND	10	UART3_RXD/GPIOD17

11	UART4_TX/GPIOB29	12	GPIOD1/PWM0
13	GPIOB30	14	DGND
15	GPIOB31	16	GPIOC14/PWM2
17	SYS_3.3V	18	GPIOB27
19	SPI0_MOSI/GPIOC31	20	DGND
21	SPI0_MISO/GPIOD0	22	UART4_RX/GPIOB28
23	SPI0_CLK/GPIOC29	24	SPI0_CS/GPIOC30
25	DGND	26	GPIOB26
27	I2C1_SDA	28	I2C1_SCL
29	GPIOC8	30	DGND
31	GPIOC7	32	GPIOC28
33	GPIOC13/PWM1	34	DGND
35	SPI2_MISO/GPIOC11	36	SPI2_CS/GPIOC10
37	AliveGPIO3	38	SPI2_MOSI/GPIOC12
39	DGND	40	SPI2_CLK/GPIOC9

3.1.2 Debug Port CON1 (UART0)

Pin#	Name
1	DGND
2	VDD_5V
3	UART_TXD0
4	UART_RXD0

3.1.3 DVP Camera IF ピン定義

Pin#	Name
1, 2	SYS_3.3V
7,9,13,15,24	DGND
3	I2C0_SCL
4	I2C0_SDA
5	GPIOB14
6	GPIOB16
8,10	NC

11	VSYNC
12	HREF
14	PCLK
16-23	Data bit7-0

3.1.4 RGB LCD IF ピン定義

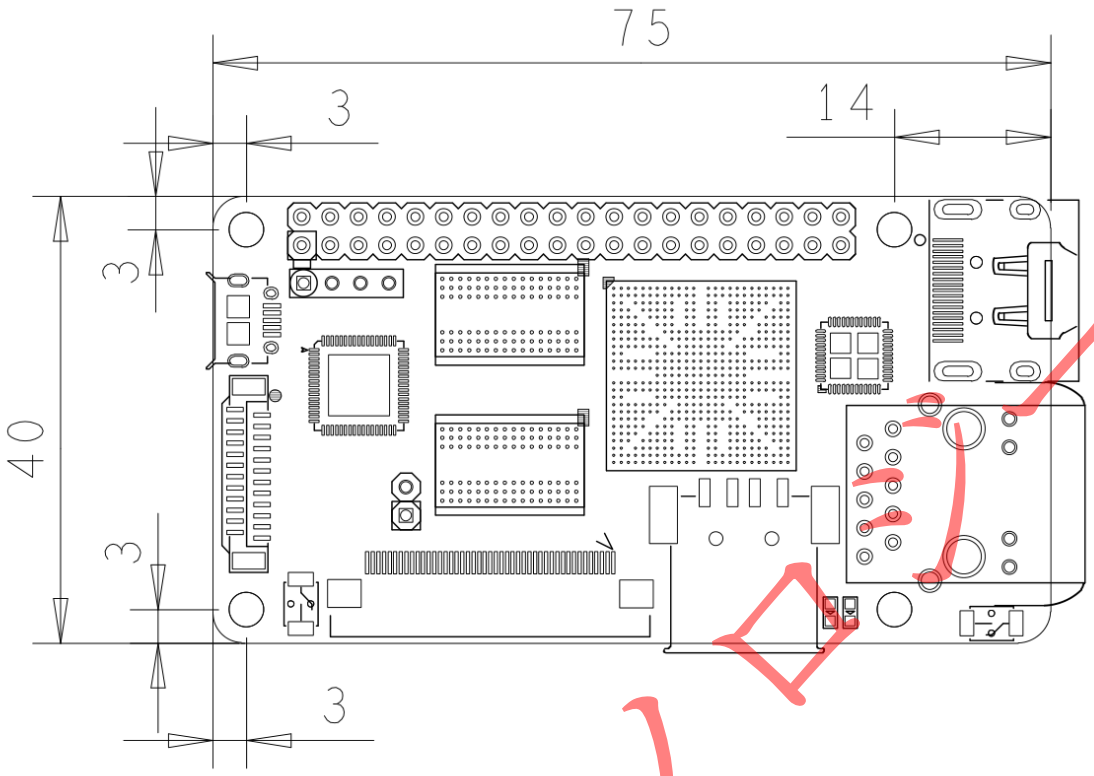
Pin#	Name	Description
1, 2	VDD_5V	5V output, LCD power
11, 20, 29, 37, 38, 39, 40, 45	DGND	ground
3-10	Blue LSB to MSB	RGB Blue
12-19	Green LSB to MSB	RGB Green
21-28	Red LSB to MSB	RGB Red
30	GPIOB25	available for users
31	GPIOC15	occupied by FriendlyARM one wire technology to recognize LCD models and control backlight and implement resistive touch, not

		applicable for users
32	XnRSTOUT Form CPU	low when system is reset
33	VDEN	signal the external LCD that data is valid on the data bus
34	VSYNC	vertical synchronization
35	HSYNC	horizontal synchronization
36	LCDCLK	LCD clock, Pixel frequency
41	I2C2_SCL	I2C2 clock signal, for capacitive touch's data transmission
42	I2C2_SDA	I2C2 data signal, for capacitive touch's data transmission
43	GPIOC16	interrupt pin for capacitive touch, used with I2C2
44	NC	not connected

説明

1. VDD_SYS_3.3V：3.3V電源の出力
2. VDD_5V：5V電源入力/出力。外部機器の電源がMicroUSBより高い場合、ボードに給電、それ以外の場合には、ボードの外部デバイスに給電。入力範囲：4.7～5.6V。
3. 詳細については回路図をご参照ください。

3.2 PCB サイズ



更に詳しい寸法についてはHPからダウンロードしてください。

4 クイックスタート

4.1 ハードウェアの準備

・NanoPi2 Fire
・microSDカード/ TFカード：Class10以上の8GBのSDHCカード
・microUSBインタフェースの外部電源、5V/2A
・HDMIモニター、またはLCD
・USBキーボード、USBマウス
・Ubuntu 14.04 64ビットシステムを実行するホスト

4.2 TFカードでテストする

NanoPi2 Fireを起動させたTFカードを作る時、クラス10かそれ以上の8GB SDHDカードを推奨する。以下は試験実績のある高速TFカード。

・Sandisk TF 8G クラス10 Micro/Sd高速TFカード

SanDisk 闪迪

Sandisk TF128G MicroSDXC TF 128G クラス10 48MB/S

4.3 実行システムを持つ microSD カードを作成する

4.3.1 Windows 環境での作成

弊社HPから下記のファイルをダウンロードする。

LCD 或いは HDMI の出力は下記のイメージファイルを使用

nanopi2-debian-sd4g.img.zip	Debian image files
nanopi2-android-sd4g.img.zip	Android image files
Flash Utility:	
win32diskimager.rar	Windows utility. Under Linux users can use "dd"

・上記のファイルを解凍する。SDカード（４G以上）をウィンドウズPCに挿入し、管理者としてwin32diskimagerユーティリティーを起動する。ユーティリティーのメイン画面で、SDカードのドライブやイメージファイルを選択し、SDカードの点滅開始のために［Write］をクリックする。
・このカードをNanoPi 2 Fireに挿入し、電源を入れる（ 5V/2A）。緑のLEDが点灯し、青のLED点滅した場合、NanoPi 2 Fireの正常に起動したことを示す。

4.3.2 Linux Desktop 環境での作成

1)microSDをUbuntuを起動しているホストに挿入　以下のコマンドでSDカードのデバイス名をチェックする
```
dmesg | tail
```
dmesgが「sdc：sdc1 sdc2」と類似した情報を出力する時、SDカード対応デバイス名は/dev/sdcになる。コマンドcat /proc/partitionsでも確認できる。
2)ファームウェアをダウンロードする
```
git clone https://github.com/friendlyarm/sd-fuse_nanopi2.git
cd sd-fuse_nanopi2
```
3)Androidの実行カードを作成する
```
su
./fusing.sh /dev/sdx
```
（注：/dev/sdxを実際のSDカードのデバイスファイル名に変えてください）
初めて使う際、ダウンロードするか確認が必要。Ｙを押してダウンロードし、Ｎ或いは10秒間入力無い場合は取り消しとなる。
4)Debianのファームウェアを SDカードに書き込む
```
./fusing.sh /dev/sdx debian
```

4.3.3 NanoPi2 の TF カードセクションを拡張

Debian/Ubuntu システムは初回起動時自動的に SD カードセクションを拡張する。

Android システム：
PC の端末ホストに以下のコマンドを実行する。
```
sudo umount /dev/sdx?
sudo parted /dev/sdx unit % resizepart 4 100 resizepart 7 100 unit MB print
```

```
sudo resize2fs -f /dev/sdx7
```

（[/dev/sdx] をシステムのデバイス名に置き換えが必要）

4.3.4 LCD/HDMI の解像度

システムが起動すると、uboot が LCD に接続されているかをチェックする。Uboot が LCD を認識した場合、その解像度を設定する。デフォルトではその表示を HDMI720P に設定する。LCD の解像度をリセットしたい場合、カーネル内の[arch/arm/plat-s5p4418/nanopi2/lcds.c] ファイルを修正し、リコンパイルすることができる。
NanoPi が HDMI モニターに接続され、Android を実行すると、解像度は自動的に［EDID］を確認し、相応の HDMI モードに設定される。

4.4 パソコンで SD カード上のシステムの更新

システムを実行する前に、少し変更したい場合は、本節の内容をご参照ください。
作成したmicroSDカードをLinuxのパソコンに挿入して、SDカードのboot、rootfsをマウントして内容を変更できる。下記の場合変更が必要：
1) カーネルのコマンドラインパラメータを更新したい場合は、「sd-fuse_nanopi2/tools」の下にある、「fw_setenv」ツールを使用することができる。
現在のコマンドラインを確認する。

```
cd sd-fuse_nanopi2/tools
./fw_printenv /dev/sdc | grep bootargs
```

現在の Android 5.1.1_r6 により SELinux が有効になる。デフォルトモードは enforcing となり、Command Line を通して変更することが可能。

```
./fw_setenv /dev/sdc bootargs XXX androidboot.selinux=permissive
```

直ぐに、permissive モードに変更でき、[XXX]は元の bootargs に置き換える必要がある。
2)カーネルの更新
新バージョンのUbootが起動時にLCDを認識した場合、SDカードのブートパーティションのuImage.hdmiを読み取る。
Androidにおいては、同じファイルであるため、直接新しいコンパイラのuImageで、SDカードのブートパーティションのファイルに交換する。
Debianにおいては、２つのファイルが異なるため、新しいコンパイラをサポートするLCD uImageで、直接SDカードのブートパーティションのファイルに交換する。HDMIのカーネルをサポートする場合は、uImage.hdmiに交換する。

4.5 Androidまたは Debian を実行する

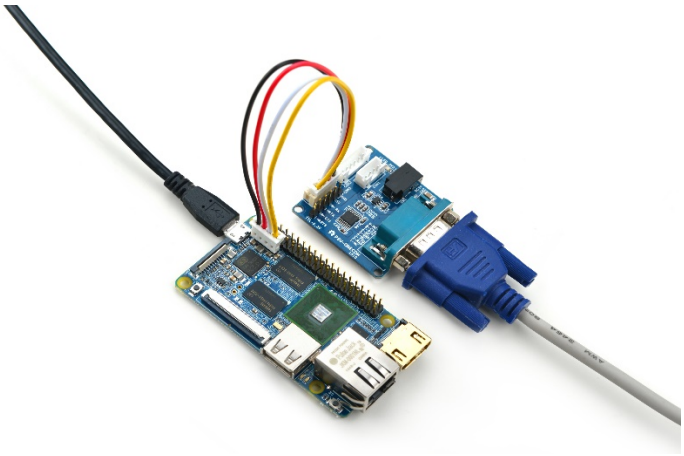
microSDカードをNanoPi2 Fireに挿入し、HDMIモニターと接続して、電源（5V/2A）に接続すると、NanoPi2 Fireは自動的に起動する。青色LEDライトの点灯でシステムが起動していることが確認でき、またHDMIモニターには起動画面が表示される。
1) NanoPi2 FireをHDMIモニターに接続する場合、USBマウスとUSBキーボードが必要である。もしLCDと接続

していれば、タッチパネルで操作可能。

2) カーネルを開発する場合、シリアルボードを装備すれば、シリアルの端末からNanoPi2 Fireを操作できる。

以下がシリアルケーブルを介してNanoPi2 FireをUbuntoとMnicomを起動中のPCに接続した状態。

Minicomの端末



パスワード入力の提示がある場合、Debianのrootユーザーのデフォルトのパスワードは[fa] である。

4.6 VNC と SSH 経由で Debian にログイン

NanoPi2 Firega

ディスプレイディバイスに接続されない場合、[VNCViewer]を PC あるいは携帯にダウンロード&インストールでき、VNC 経由で NanoPi2 にログインできる。ディフォルトのパスワードは[fa123456]。

NanoPi2 Fire へのユーザーログイン後のスクリーンショットは次のようになる。

[SSH -l root 192.168.8.1] 経由でもログイン可能。[root]のデフォルトパスワードは[fa]である。

```
iwconfig wlan0 power off
```

5 Debain システム

5.1 イーサネット接続

NanoPi2 Fireが以前にイーサネットでネットワークに接続されたことがある場合、電源を入れた後、自動的にIPを取得する。イーサーネット接続したことがなく、またはDHCPサービスもない場合、IPアドレスの取得に失敗することになる。因って、システムの起動には約15～60秒待つことになる。

1） MACアドレスを設定する
NanoPi2 Fireは、デフォルトでは有効なMACアドレスがない。ボードがネットワークに正常に接続されると、Pi2 Fireは自動的に［/etc/network/interfaces.d/eth0］のランダムなMACを生成する。ユーザーはそれを固定されたMACアドレスに変更することができる。

```
vi /etc/network/interfaces.d/wlan0
```

設置内容は次のようになる。

```
auto eth0
allow-hotplug eth0
iface eth0 inet dhcp
hwaddress 76:92:d4:85:f3:0f
```

[hwaddress]は MAC アドレスを指定する。ここで、[76:92:d4:85:f3:0f]はランダム MAC。有効なものに変更することを推奨する。[注意:MAC をリセットするとき、予期せぬ問題の発生を防うため、MAC が IEEE の定義に満たしていることを確認してください。]

変更後、終了する。ネットワークを再起動するために、ボードを再起動するか、または、下記のコマンドを実行する。

```
systemctl restart networking
```

5.2 Debian のパッケージソフトをインストールする

提供しているのは標準的なDebian jessieシステムである。apt-getなどのコマンドでパッケージソフトをインストールすることができる。初めてインストールする場合、まず以下のコマンドでパッケージソフトリストを更新する必要がある。

```
apt-get update
```

その後、パッケージソフトをインストールすることができる。例えばFTPサーバーをインストールするには以下のコマンドを使用する。

```
apt-get install vsftpd
```

/etc/apt/sources.listを編集することで、ダウンロードサーバーを変更することができる。
http://www.debian.org/mirror/lisから全てのサーバーリストが取得可能。［armhf］が付くリストを選択することが必要。

6 システムのコンパイル方法

6.1 クロスコンパイラをインストールする

先ず、コンパイラをダウンロードして解凍する。

```
git clone https://github.com/friendlyarm/prebuilts.git
sudo mkdir -p /opt/FriendlyARM/toolchain
sudo tar xf prebuilts/gcc-x64/arm-cortexa9-linux-gnueabihf-4.9.3.tar.xz -C /opt/FriendlyARM/toolchain
```

コンパイラのパスをPATHに追加する。viでvi ~/.bashrcを実行して、末尾に以下の内容を追加する。

```
export PATH=/opt/FriendlyARM/toolchain/4.9.3/bin:$PATH
export GCC_COLORS=auto
```

~/.bashrcスクリプトを実行してカレントshellで有効にする。"."の後ろにスペースがある。

```
. ~/.bashrc
```

コンパイラは64ビットのため、32ビットのLinuxでは実行できない。
インストールの完了後、インストールが成功したかを確認できる。

```
arm-linux-gcc -v
Using built-in specs.
COLLECT_GCC=arm-linux-gcc
COLLECT_LTO_WRAPPER=/opt/FriendlyARM/toolchain/4.9.3/libexec/gcc/arm-cortexa9-linux-gnueabihf/4.9.3/lto-wrapper
Target: arm-cortexa9-linux-gnueabihf
Configured with: /work/toolchain/build/src/gcc-4.9.3/configure --build=x86_64-build_pc-linux-gnu
--host=x86_64-build_pc-linux-gnu --target=arm-cortexa9-linux-gnueabihf --prefix=/opt/FriendlyARM/toolchain/4.9.3
--with-sysroot=/opt/FriendlyARM/toolchain/4.9.3/arm-cortexa9-linux-gnueabihf/sys-root --enable-languages=c,c++
--with-arch=armv7-a --with-tune=cortex-a9 --with-fpu=vfpv3 --with-float=hard
...
Thread model: posix
gcc version 4.9.3 (ctng-1.21.0-229g-FA)
```

6.2 U-Boot のコンパイル

U-Bootソースコードをダウンロードし、コンパイルする。ブランチは[nanopi2-lollipop-mr1]であることに注意する。

```
git clone https://github.com/friendlyarm/uboot_nanopi2.git
cd uboot_nanopi2
git checkout nanopi2-lollipop-mr1
make s5p4418_nanopi2_config
make CROSS_COMPILE=arm-linux-
```

コンパイルに成功した後、u-boot.binを取得する。Fastbootで、NanoPi2のSDカードのUbootを更新する。

手順は下記の通り：

1）PCでコマンド [sudo apt-get install android-tools-fastboot]でfastbootツールをインストールする。

2）シリアルデバッグセットでNanoPi2 FireとPCを接続する。起動後２秒以内、シリアル端末でEnterを押して、u-bootのコマンドラインモードに入る。

3）u-bootのコマンドラインモードでfastbootコマンドを入力し、Enterを押してfastbootモードに入る。

4）microUSBケーブルでNanoPi2 FireとPCを接続する。PC側で下記コマンドを入力してu-boot.binを書き込む。

```
fastboot flash bootloader u-boot.bin
```

注意点：直接ddコマンドでSDカードを更新してはいけない。正常に起動できなくなる可能性がある。

6.3 mkimage を用意する

カーネルをコンパイルするには u-boot の mkimage ツールが必要。因って、カーネル uImage をコンパイルする前に、PC 側で実行できることの確認が必要。

直接 sudo apt-get install u-boot-tools コマンドでインストールできる。或いは自分でコンパイルしてインストールする。

```
cd uboot_nanopi2
make CROSS_COMPILE=arm-linux- tools
sudo mkdir -p /usr/local/sbin && sudo cp -v tools/mkimage /usr/local/sbin
```

6.4 Linux kernel のコンパイル

6.4.1 カーネルのコンパイル

1）カーネルのソースコードをダウンロードする。

NanoPi2 Fireのカーネルのソースコードは[nanopi2-lollipop-mr1]ブランチにある。

```
git clone https://github.com/friendlyarm/linux-3.4.y.git
cd linux-3.4.y
git checkout nanopi2-lollipop-mr1
```

2）Androidカーネルをコンパイルする。

```
make nanopi2_android_defconfig
touch .scmversion
make uImage
```

3）Debianカーネルをコンパイルする。

```
make nanopi2_linux_defconfig
touch .scmversion
make uImage
```

コンパイル成功後、uImageが[arch/arm/boot/uImage]ディレクトリに生成される。このカーネルはHDMI出力をサポートする。既存のuImageと置き換えることができる。

LCDをサポートするカーネル生成する方法は次のようになる。

```
touch .scmversion
make nanopi2_linux_defconfig
make menuconfig
  Device Drivers -->
    Graphics support -->
      Nexell Graphics -->
        [*] LCD
        [ ] HDMI
make uImage
```

コンパイル成功後、uImageがLCD用に生成される。それを使って、既存のuImageに置き換えることができる。

6.4.2 カーネルモジュールのコンパイル

Androidはカーネルモジュールを含んでいる。場所はsystemセクションの/lib/modules/である。新しいカーネルモジュール或いはカーネルモジュールの設定が変更した場合、再コンパイルが必要である。

先ず、カーネルソースのモジュールをコンパイルする。

```
cd linux-3.4.y
make CROSS_COMPILE=arm-linux- modules
```

またAndroidのソースに２つのカーネルモジュールのソースがある。下記コマンドでコンパイルする：

```
cd /opt/FriendlyARM/s5p4418/android
./vendor/friendly-arm/build/common/build-modules.sh
```

“/opt/FriendlyARM/s5p4418/android”はAndroidのソースのTOPフォルダである、[-h]パラメータでヘルプ内容を確認できる。

コンパイル成功後、生成したカーネルモジュールが表示される。

6.5 Andriod システムのコンパイル

6.5.1 コンパイル環境の構築

64ビットのUbuntu 14.04を推奨する。必要なパッケージをインストールすれば良い。

```
sudo apt-get install zlib1g-dev:i386
sudo apt-get install bison g++-multilib git gperf libxml2-utils make python-networkx zip
sudo apt-get install flex libncurses5-dev zlib1g-dev gawk minicom
```

詳細内容は下記 URL をご参照ください。

https://source.android.com/source/initializing.html

6.5.2 ソースコードをダウンロードする

Android のソースコードをダウンロードするには repo が必要、インストール方法及び使用方法は下記 URL をご参照ください。https://source.android.com/source/downloading.html

```
mkdir android && cd android
repo init -u https://github.com/friendlyarm/android_manifest.git -b nanopi2-lollipop-mr1
repo sync
```

上記の“android”はワークフォルダのことある。

6.5.3 システムをコンパイルする

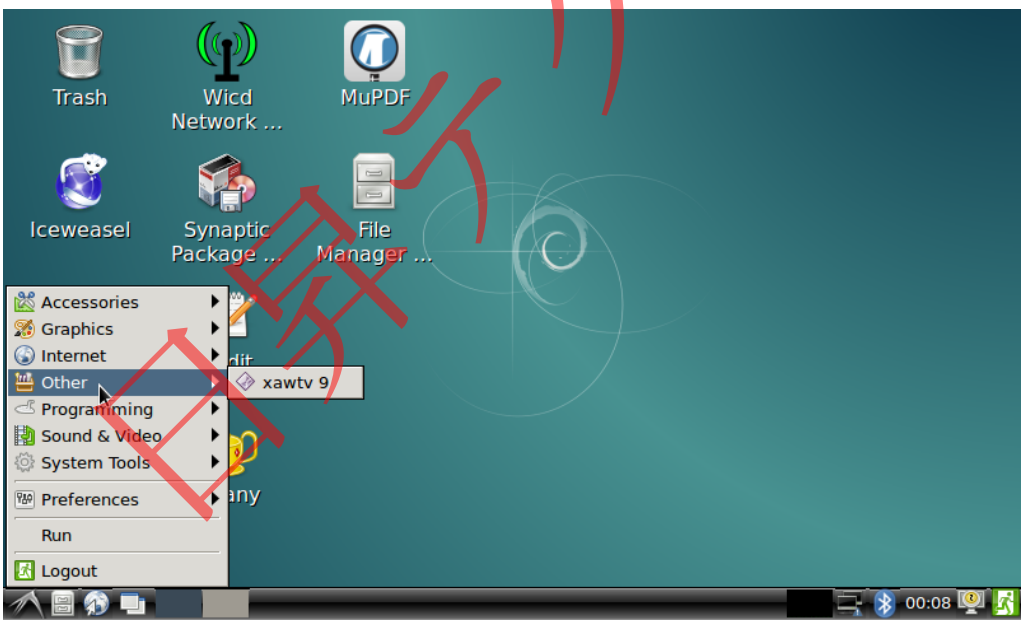
```
source build/envsetup.sh
lunch aosp_nanopi2-userdebug
make -j8
```

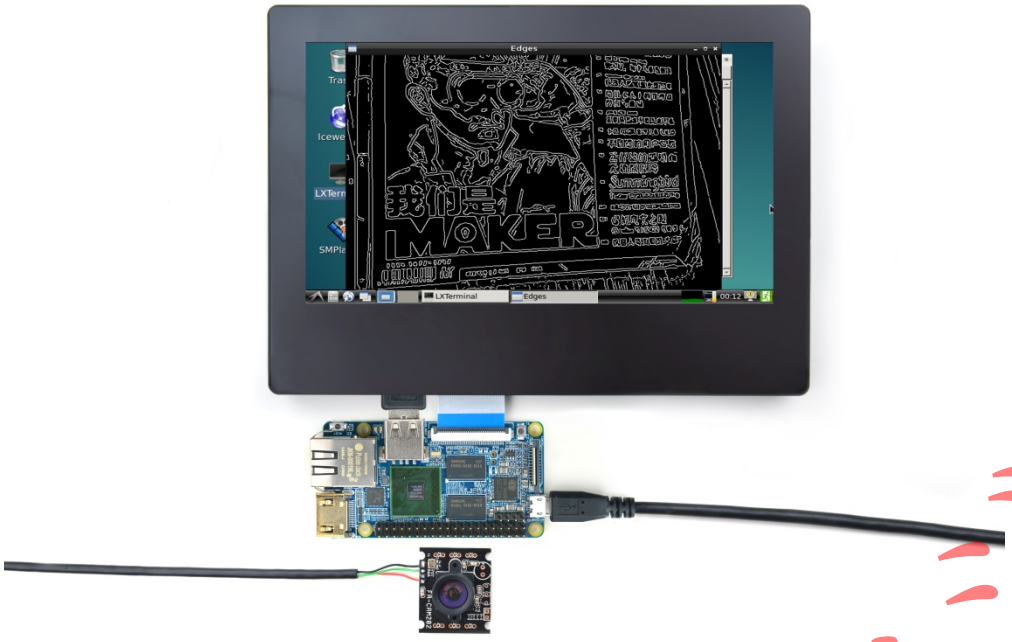
コンパイル終了後、out/target/product/nanopi2/のフォルダにイメージファイルが生成される。

7 カメラモジュールを接続する

7.1 USB カメラモジュール(200 万画素)

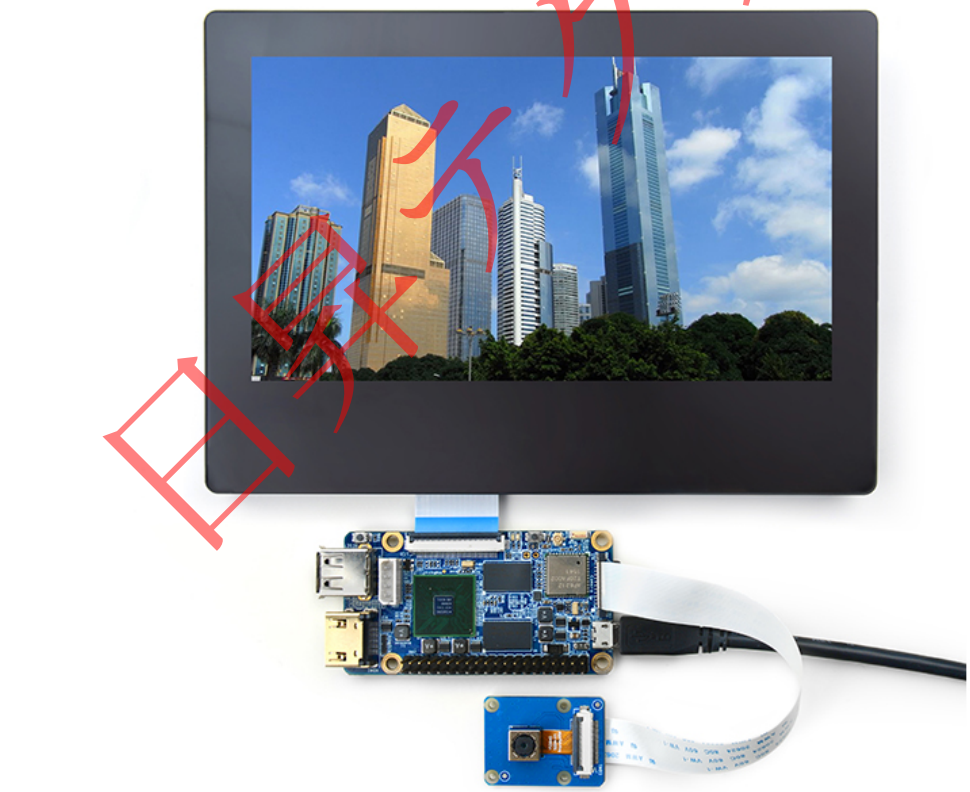
本テストでは、NanoPi2 Fire は Debian を実行する。Debian が完全にロードされた後、お使いの NanoPi2 Fire を LCD に接続し、GUI の左ボタンにある "other"-->"xawtv" をクリックすると、USB カメラのアプリケーションが起動する。[welcome to xawtv !] ウィンドウが表示され、 [OK] をクリックする。





7.2 OV5640 カメラモジュール(500 万画素)

Android5.1 システムを実行して “camera”アイコンをクリックする。



7.3 USB カメラを接続して OpenCV を使う

OpenCV は Open Sorece Computer Vision Library の略で、クロスプラットフォームビジョンライブラリである。NanoPi Fire を実行すると、Debian ユーザは USB カメラデバイスにアクセスするために OpenCV APIs を使用できる。

C++ on the NanoPi 2 Fire で OpenCV を使う方法についてのガイドラインは次のようになる。

1. 準備

```
 ---Firstly you need to make sure your NanoPi 2 Fire is connected to the internet.
  Login your NanoPi 2 Fire via a serial terminal or SSH. After login please type
your username(root) and password(fa):
 ---Run the following commands:
```

```
 #apt-get update
 (The OS images we provide for the NanoPi 2 Fire by default have the vi utility. However
we suggest you install the vim utility)
#apt-get install vim
 #apt-get install libcv-dev libopencv-dev
```

2. USB カメラが NanoPi2 Fire で動作することを確認する。NanoPi2 Fire のカメラユーティリティでカメラをテストすることが可能。
3. カメラのデバイスをチェックする。

```
#ls /dev/video + "Tab" key (This lists available USB camera devices. In our
test case video0 was available)
```

4. OpenCV のコードサンプル

```
#cd /home/fa
#vim test.cpp
```

```
 #include "opencv2/opencv.hpp"

using namespace cv;

int main(int, char**)
{
    VideoCapture cap(0); // open the default camera
    if(!cap.isOpened())  // check if we succeeded
        return -1;

    Mat edges;
    namedWindow("edges",1);
    for(;;)
    {
        Mat frame;
```

```
        cap >> frame; // get a new frame from camera
        cvtColor(frame, edges, CV_BGR2GRAY);
        GaussianBlur(edges, edges, Size(7,7), 1.5, 1.5);
        Canny(edges, edges, 0, 30, 3);
        imshow("edges", edges);
        if(waitKey(30) >= 0) break;
    }
    // the camera will be deinitialized automatically in VideoCapture destructor
    return 0;
}
```

コードサンプルをコンパイルする

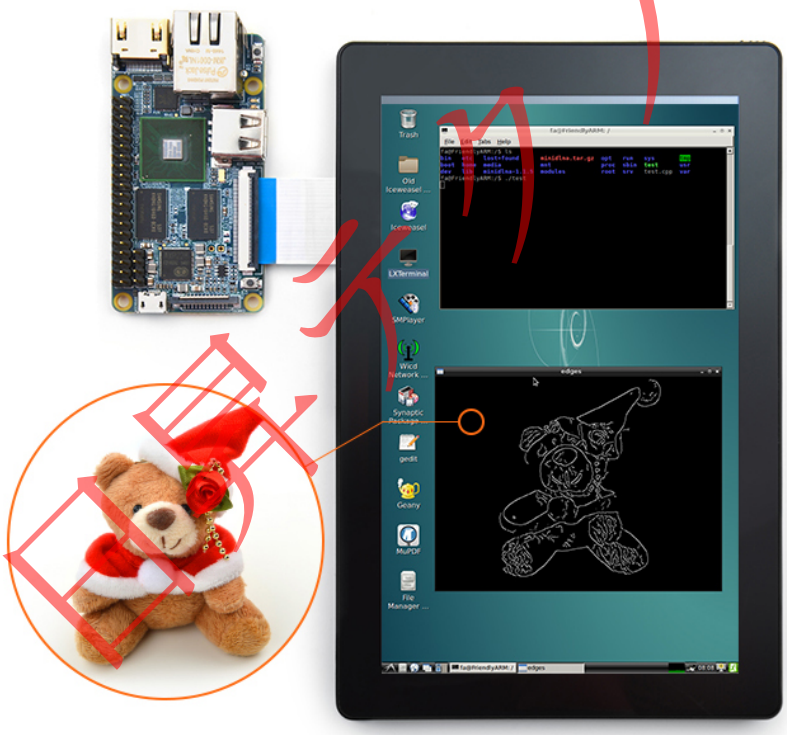
```
#g++ test.cpp -o test -lopencv_core -lopencv_highgui -lopencv_imgproc
```

コンパイルが成功すると、［テスト］実行可能ファイルが生成される。

5. NanoPi 2 Fire を USB キーボードに接続し、次のコマンドを実行する。

```
#./test
```

以下の画像参照。



以上。