

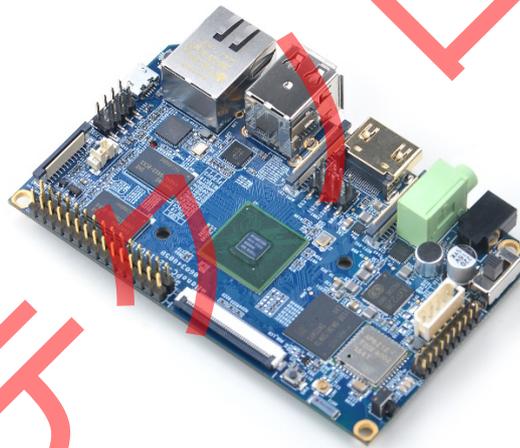
Cortex-A9 4 コア S5P4418 ボード NanoPC-T2 簡易マニュアル

株式会社日昇テクノロジー

<http://www.csun.co.jp>

info@csun.co.jp

作成日 2016/3/17



copyright@2016

• 修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2016/3/17

※ この文書の情報は、文書を改善するため、事前の通知なく変更されることがあります。最新版は弊社ホームページからご参照ください。[<http://www.csun.co.jp>]

※ (株)日昇テクノロジーの書面による許可のない複製は、いかなる形態においても厳重に禁じられています。

日昇テクノロジー

目次

1 紹介	5
2 主な仕様	5
3 インターフェースの配置及びサイズ	6
3.1 インターフェースの配置	6
3.1.1 30Pin GPIO ピン定義	7
3.1.2 20ピン LVDS インターフェイス ピン定義	8
3.1.3 DVP カメラ IF ピン定義	9
3.1.4 RGB LCD IF ピン定義	9
3.1.5 MIPI-CSI インタフェース ピン定義	13
3.2 PCB サイズ	15
4 クイックスタート	16
4.1 ハードウェアの準備	16
4.2 実行システムを持つ microSD カードを作成する	16
4.2.1 SD カードから NanoPC-T2 を高速でスタート	16
4.2.2 eMMC から NanoTC-2 を起動する	16
4.2.3 Linux Desktop 環境での作成	18
4.2.4 LCD/HDMI の解像度	18
4.3 パソコンで SD カード上のイメージファイルの更新	18
4.4 Android または Debian を実行する	19
4.5 VNC と SSH 経由で Debian にログイン	20
5 Debain システム	20
5.1 イーサネットに接続する。	20
5.2 無線ネットワークに接続する	20
5.3 Wi-Fi 無線ホットスポットの配置	22
5.4 Bluetooth を使ってファイルを転送する	22
5.5 Debian のパッケージソフトをインストールする	23
6 システムのコンパイル方法	24
6.1 クロスコンパイラをインストールする	24
6.2 U-Boot のコンパイル	24
6.3 mkimage を用意する	25

6.4 Linux kernel のコンパイル	25
6.4.1 カーネルのコンパイル	25
6.4.2 カーネルモジュールのコンパイル	26
6.5 Andriod システムのコンパイル	26
6.5.1 コンパイル環境の構築	26
6.5.2 ソースコードをダウンロードする	26
6.5.3 システムをコンパイルする	27
7 カメラモジュールを接続する	27
7.1 NanoPC-T2 を USB (FA-CAM202) に接続する	27
7.2 NanoPC-2 を CMOS 5M ピクセルのカメラに接続する	28
7.3 OpenCV を使用して USB カメラにアクセスする	29
8. NanoPC-T2 の SD カードパーティションを拡張	30
8.1. Debian 用	30
8.2. Android 用	30

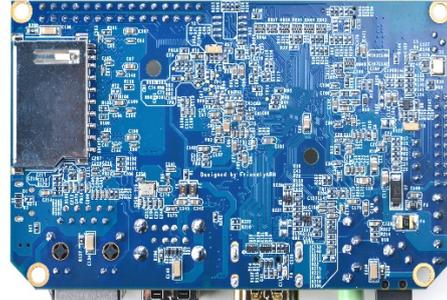
1 紹介

NanoPC-T2はIoT設計者やビジネスユーザーのために設計・開発したクワッドコア・Cortex9のシングルボードである。Samsungのクワッドコア・Cortex-A9・S5P4418 SoC、1.4GHzを備えている。既存の4418ベースのボードNanoPi2に比べて、NanoPC-T2は、8G eMMCが組み込まれており、オーディオジャック、ビデオ入力/出力インターフェースが搭載されている。前のNanoPCT1と比較すると、NanoPCT2はWiFiとBluetooth、Gbpsイーサネット・ポートを搭載している。また、NanoPC-T2は パワーマネージメント機能をそなえている (NanoPC-T1 がサポートしない)。オーバーヒートの問題を避けるために、NanoPC-T2は取り付け穴付きヒートシンクを備えている。

・NanoPC-T2は豊富なビデオインターフェースとHDMI1080用のサポートは人気のあるディスプレイデバイスのみならず、抵抗式タッチと静電容量式タッチの様々なLCDを作動させる。

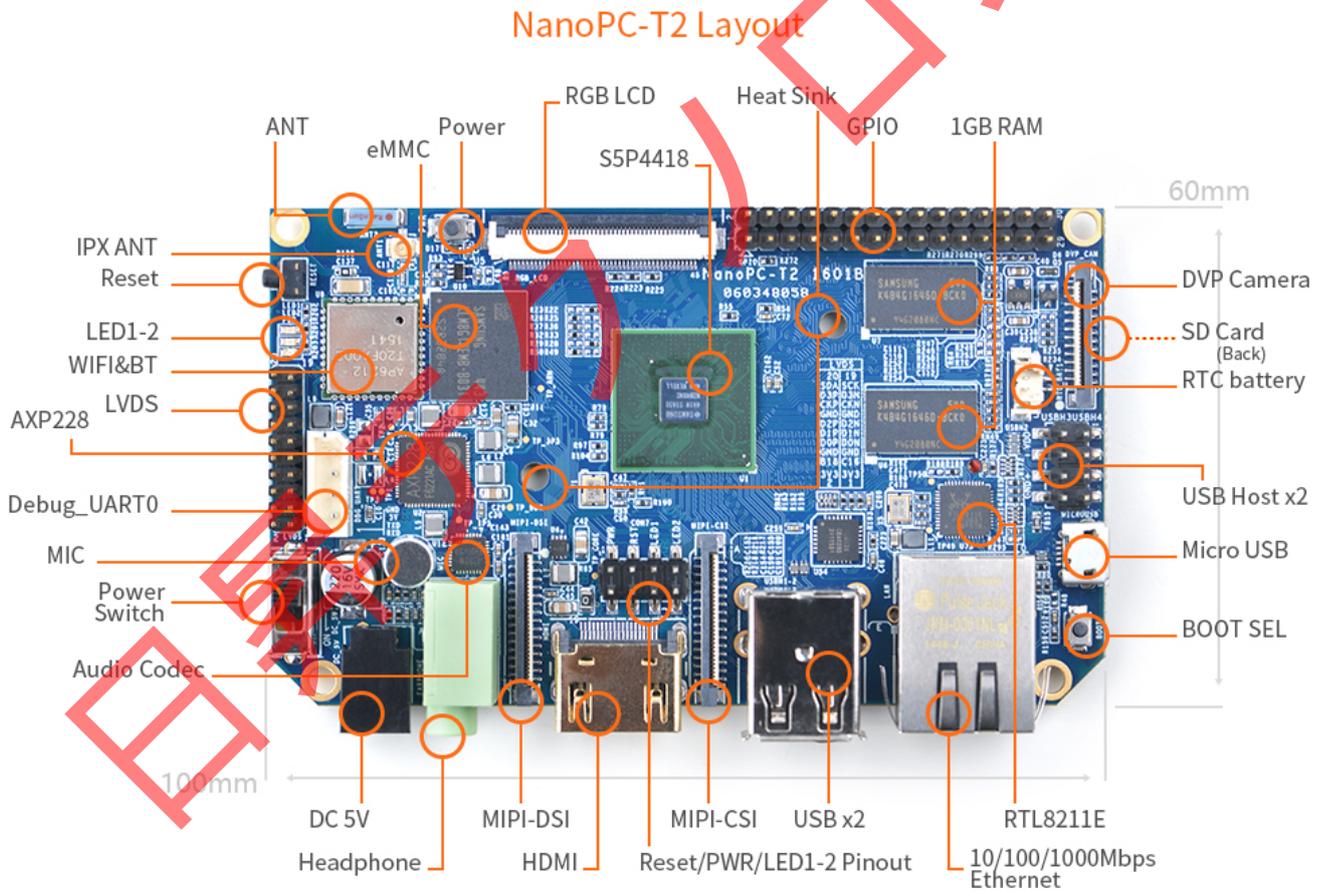
2 主な仕様

- ・SoC：動作周波数1.4GHzまでスケールアップするSamsung S5P4418クワッドコアのCortex-A9
- ・電力マネージメントユニット：AXP228 PMU ソフトウェアのパワーオフとウェイクアップをサポートする。
- ・システムメモリ：1GB32ビットDDR RAM
- ・SDIO WiFi/Bluetoothモジュール
- ・USB2.0 タイプA x1
- ・eMMC：8GB
- ・ストレージ：1 x SD カードソケット
- ・イーサネット：Gbps イーサネットポート (RTL8211E)
- ・WiFi：802.11b/g/n
- ・Bluetooth：4.0 デュアルモード
- ・アンテナ：磁器アンテナ IPXインターフェース
- ・ビデオ入力：DVP カメラ/MIPI-CSI (2つのカメラインターフェース)
- ・ビデオ出力：HDMIタイプA/LVDS/LCD/MIPI-DSI (4つのビデオ出力インターフェース)
- ・オーディオ：3.5ミリオーディオジャック/HDMI経由
- ・マイク：オンボードマイクx1
- ・USB：USB2.0ホスト×4、標準タイプAポート×2と2.54ミリピッチのピンヘッダ×2
- ・LCDインターフェース：0.5 mmピッチ45ピンFPCシート、フルカラー (RGB：8-8-8)
- ・マイクロUSB：1×マイクロUSB2.0クライアント、タイプA
- ・HDMI：1.4A、タイプA、1080P
- ・DVPカメラ：0.5ミリピッチ、45ピン、FPC用シート
- ・GPIO：2.54ミリピッチ、30ピンヘッダー
- ・シリアルデバッグポート：2.54ミリピッチ、4ピンヘッダー
- ・LED：電源LED×1、GPIO LEDx2
- ・PCB：6層
- ・PCBサイズ：100mm×60mm
- ・OS/ソフトウェア：Uブート、Android5.1、Debian 8



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 30Pin GPIO ピン定義

ピン	名前	ピン	名前
1	SYS_3.3V	2	DGND
3	UART2_TX/GPIOD20	4	UART2_RX/GPIOD16
5	I2C0_SCL	6	I2C0_SDA
7	SPI0_MOSI/GPIOC3 1	8	SPI0_MISO/GPIOD0
9	SPI0_CLK/GPIOC29	10	SPI0_CS/GPIOC30
11	UART3_TX/GPIOD21	12	UART3_RX/GPIOD17
13	UART4_TX/GPIOB29	14	UART4_RX/GPIOB28
15	GPIOB31	16	GPIOB30
17	GPIOC4	18	GPIOC7
19	GPIOC8	20	GPIOC24
21	GPIOC28	22	GPIOB26
23	GPIOD1/PWM0	24	GPIOD8/PPM
25	GPIOC13/PWM1	26	AliveGPIO3

27	GPIOC14/PWM2	28	AliveGPIO5
29	VDD_5V	30	DGND

3.1.2 20ピン LVDS インターフェイス ピン定義

ピン	名前	ピン	名前
1	SYS_3.3V	2	SYS_3.3V
3	GPIOC16	4	GPIOB18
5	DGND	6	DGND
7	LVDS_D0-	8	LVDS_D0+
9	LVDS_D1-	10	LVDS_D1+
11	LVDS_D2-	12	LVDS_D2+
13	DGND	14	DGND
15	LVDS_CLK-	16	LVDS_CLK+
17	LVDS_D3-	18	LVDS_D3+
19	I2C2_SCL	20	I2C2_SDA

3.1.3 DVP カメラ IF ピン定義

ピン	名前
1, 2	SYS_3.3V
7, 9, 13, 15, 24	DGND
3	I2CO_SCL
4	I2CO_SDA
5	GPIOB14
6	GPIOB16
8, 10	NC
11	VSYNC
12	HREF
14	PCLK
16-23	データ bit7-0

3.1.4 RGB LCD IF ピン定義

ピン番号	名前	説明
1, 2	VDD_5V	5V 出力 — LCD モジュールに電源を供給するために使うことが可能

11, 20, 29, 37, 38, 39, 40, 45	DGND	グラウンド
3-10	ブルーの LSB から MSB へ	RGB のブルー信号
12-19	グリーンの LSB か ら MSB へ	RGB のグリーン信号
21-28	赤の LSB から MSB へ	RGB の赤信号
30	GPIOB25	ユーザーがコントロール可能
31	GPIOC15	LCD モジュールと制御バックライトを認識し、抵抗式タッチを実現するワンワイヤ技術はユーザーに適用できない。
32	XnRSTOUT Form CPU	システムがリセットされる時、出力のレベルは低い
33	VDEN	RGB 信号が有効な信号であることを示す
34	VSYNC	垂直同期
35	HSYNC	水平同期
36	LCDCLK	LCD クロック, ピクセル周波数
41	I2C2_SCL	静電容量式タッチのデータ伝送のための I2C2 クロック信号
42	I2C2_SDA	静電容量式タッチのデータ伝送のための I2C2 データ信号

43	GPIOC16	静電容量式タッチ用のピンを中断・I2C2 と使用される
44	NC	接続されていない

• 3.1.4 MIPI-DSI インタフェース ピン定義

ピン番号	名前
1、2、3	VDD_5V
4	DGND
5	I2C2_SDA
6	I2C2_SCL
7	DGND
8	GPIOC0
9	DGND
10	GPIOC1
11	DGND
12	GPIOA28
13	nRESETOUT

14、15	DGND
16	MIPIDSI_DN3
17	MIPIDSI_DP3
18	DGND
19	MIPIDSI_DN2
20	MIPIDSI_DP2
21	DGND
22	MIPIDSI_DN1
23	MIPIDSI_DP1
24	DGND
25	MIPIDSI_DN0
26	MIPIDSI_DP0
27	DGND
28	MIPIDSI_DNCLK
29	MIPIDSI_DPCLK

30	DGND
----	------

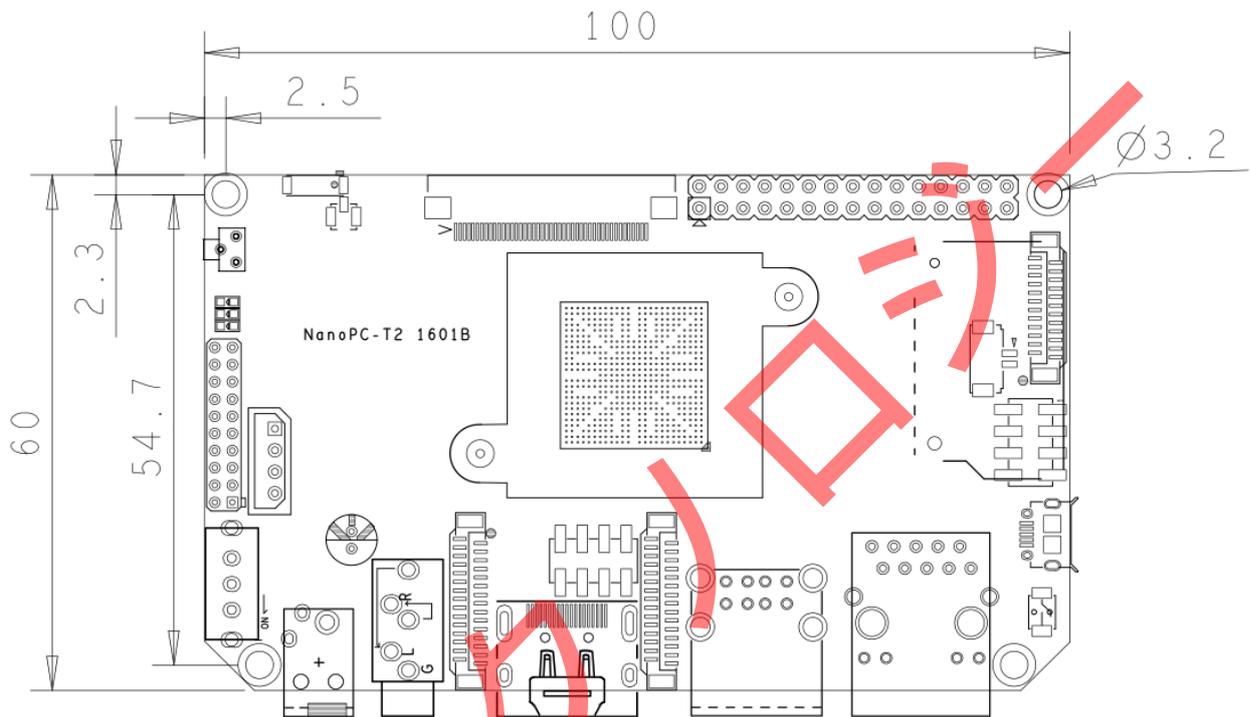
3.1.5 MIPI-CSI インタフェース ピン定義

Pin#	Name
1, 2, 3	VDD_5V
4	DGND
5	I2C2_SDA
6	I2C2_SCL
7	DGND
8	GPIOC0
9	DGND
10	GPIOC1
11	DGND
12	GPIOA28
13	nRESETOUT
14, 15	DGND

16	MIPIDSI_DN3
17	MIPIDSI_DP3
18	DGND
19	MIPIDSI_DN2
20	MIPIDSI_DP2
21	DGND
22	MIPIDSI_DN1
23	MIPIDSI_DP1
24	DGND
25	MIPIDSI_DN0
26	MIPIDSI_DP0
27	DGND
28	MIPIDSI_DNCLK
29	MIPIDSI_DPCLK
30	DGND

1. VDD_SYS_3.3V : 3.3V電源の出力
2. VDD_5V : 5V電源入力/出力。電圧がMicroUSBより高い場合、ボードに給電、そうでない場合、ボードはMicroUSBから電源を取る。入力範囲 : 4.7~5.6V。
3. 更に詳しい情報については回路図をチェックしてください。

3.2 PCB サイズ



4 クイックスタート

4.1 ハードウェアの準備

- NanoPC-T2ボード
- SDカード/ TFカード：Class10以上の8GBのSDHCカードが必要
- DCインタフェースの外部電源、5V/2A
- HDMI入力サポートするディスプレイまたはモニタ（またはLCD）
- USBキーボード、USBマウス、同時に使う場合はUSB HUB（またはシリアルボードへのTTL）
- Ubuntu 14.04 64ビットシステムを推奨する。

4.2 実行システムを持つ microSD カードを作成する

4.2.1 SD カードから NanoPC-T2 を高速でスタート

次のファイルをダウンロードしてください。

LCD または HDMI 出力の場合は、次のファイルを使用してください：	
nanopi2-debian-sd4g.img.zip	Debian のイメージファイル
nanopi2-android-sd4g.img.zip	Android のイメージファイル
フラッシュユーティリティ：	
win32diskimager.zip	Windows ユーティリティ。Linux でユーザーは“dd”を使用できる。

- これらのファイルを解凍する。SDカードをWindowsのPCに挿入し、win32diskimager.exeを右クリックして「管理者として実行」をクリックして起動する。ユーティリティのメインウィンドウでSDカードのドライブとイメージファイルを選択し、【Write】をクリックする。
- このカードをNanoPC-T2のブートソケットに挿入し、ブートキーを押して保持し、電源をオンにする（5V/2Aの電源）。PWR LEDがオンでLED1が点滅している場合、NanoPC-T2が正常に起動していることを示します。

4.2.2 eMMC から NanoTC-2 を起動する

- RAW Image をダウンロードする
 イメージファイル【nanopi2-eflasher-sd4g.img.zip】とウィンドウズユーティリティ【win32diskimager.zip】を取得する。

- RAWImageをSDカードにフラッシュする
 WindowsPCにSDカード（4G以上）を挿入し、win32diskimager.exeを右クリックして「管理者として実行」をクリックして起動する。ユーティリティのメイン画面で、あなたのSDカードのドライブとイメージファイルを選択し、[Write]をクリックする。
 AndroidとDebianのイメージファイル（System-image-files-for-eMMC）をダウンロードする。ダウンロード

した後、".tgz" 圧縮フォルダを解凍し、それをSDカードにコピーする。

OS	Image	Files	コピー先
Android 5.1	android-lollipop-images.tgz android-lollipop-images.tgz.hash.md5	boot.img system.img userdata.img cache.img partmap.txt	images¥android
Debian (Jessie)	debian-jessie-images.tgz debian-jessie-images.tgz.hash.md5	boot.img rootfs.img partmap.txt	images¥debian

・OSを指定する

デフォルトではSDカードの設定ファイル[images¥FriendlyARM.ini]はAndroidがEMMCにフラッシュされるように指定する。Debian をインストールしたい場合は次のように変更する。

```
OS = Debian
```

#はコメント

・NanoPC-T2のEMMCにImageをフラッシュする

このカードをあなたのNanoPC-T2のeMMCに挿入する。ボードをHDMIまたはLCDに接続し、ブートキーを押したままにし、インストールを始めるためにボードの電源をオンにする（5V/2A電源）。HDMIまたはLCDからインストールの全過程をみることが出来る。次のメッセージがポップアップされた場合、インストールは成功したことになる。

```
Android is fused successfully.
```

```
All done.
```

インストールが完了したら、eMMC

からのボードを起動するために、[リセット]あるいはボードの電源をオフにする。

インストールの過程をモニターするためにLEDのステータスもチェックできる。

ED ステータス	インストールステータス
LED1 が連続で2回点滅 LED2 はオフ	電源がオンの正常な状態 インストールが実行されず、LED1 がこのような状態を保持する場合は、LEDはOFFとなる。J will keep behaving this way and LED2 will be off

LED1 と LED2 は三秒ごとに交互に点滅	インストール実行中
LED1 と LED2 が 1.2 秒ごとに交互に点滅	インストール成功
LED1 と LED2 が同時に点滅	インストール失敗

4.2.3 Linux Desktop 環境での作成

1) microSDをUbuntuのパソコンに挿入 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「sdcc : 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) Linuxのスクリプト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SD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する。

4) DebianのSDカードを作成する

```
./fusing.sh /dev/sdx debian
```

4.2.4 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を認識した場合には、その解像度を設定する。デフォルトでは、ubootはHDMI720Pへのディスプレイを設定する。LCDの解像度をリセットしたい場合は、カーネル内の[arch/arm/plat-s5p4418/nanopi2/lcds.c] ファイルを修正し、再コンパイルできる。NanoPC-T2がHDMIモニターに接続され、Androidを起動した場合、「EDID」をチェックすることで自動的に適切なHDMIモードに解像度を設定する。NanoPC-T2がHDMIモニターに接続され、Debianを起動した場合、デフォルトでHDMI720Pへの解像度をセットする。この場合、カーネルの設定を変更することで1080Pまでセットできる。

4.3 パソコンでSDカード上のイメージファイルの更新

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の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。例えば、LCDがHD700であれば下記の方法で変更することができる。

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。

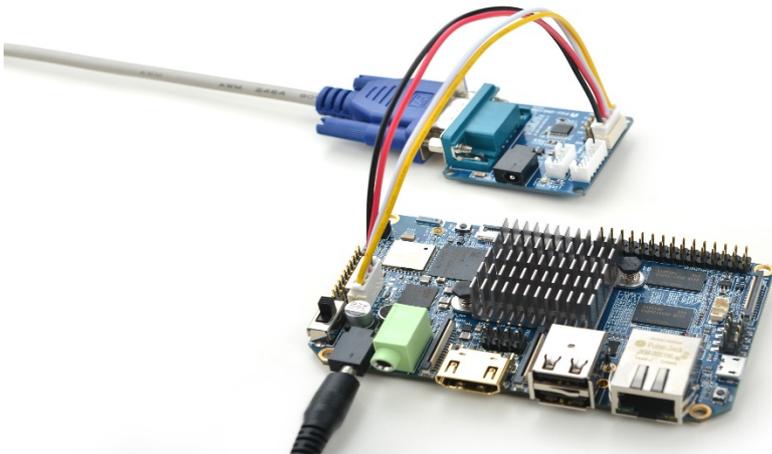
Androidにおいては、同じファイルであるため、直接新しいコンパイラのuImageで、SDカードのブートパーティションのファイルに交換する。

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。

4.4 Android または Debian を実行する

microSDカードをNanoPC-T2に挿入し、HDMIモニターと接続して、電源（5V/2A）に接続すると、NanoPC-T2はSDカードから起動する。PWRLEDとLED 1 が点滅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AndroidとDebianが確認できる。

- 1) NanoPC-T2をHDMIモニターに接続したい場合、USBマウス、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、シリアルデバッグポートに接続すれば、端末からNanoPC-T2を操作できる。シリアルケーブル経由でUbuntuとMinicomの実行しているPCにNanoPC-T2を接続する場合は次のようになります。



Debianでの[root]のパスワードは[fa]である。

4.5 VNC と SSH 経由で Debian にログイン

NanoPC-T2 を LCD・HDMI に接続せずに、[-wifiap.img]のイメージファイルを実行した場合、WIFI 経由で携帯等の他のデバイスから NanoPC-T2 の [nanopi2-wifiap]の NanoPi2 にログインできる。ホットスポット wifiap のデフォルトパスワードは[123456789]。正常に NanoPC-T2 に接続した後、以下の [URL](#) から[VNCViewer]をダウンロード&インストールできる。VNC 経由で NanoPi2 にログインするには、IP アドレスとポートを 192.168.8.1:5901 に設定する必要がある。デフォルトのパスワードは[fa123456]。

ユーザーログイン後のスクリーンショット



[SSH-1 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。SSH をスムーズにするには、WIFIの省電力モードをオフする。

```
iwconfig wlan0 power off
```

5 Debain システム

5.1 イーサネットに接続する。

NanoPC-T2 が電源を入れる前にイーサネット経由でネット接続された場合、電源をいれた後、PC-T2 は自動的に IP を取得する。

5.2 無線ネットワークに接続する

次のセクションはHDMIまたはLCDに接続されたNanoPC-T2にのみ適用する。

Debianがロードされた後、GUIの右上にあるネットワークアイコンをクリックすると、自動的に近くのWiFiホットスポットが検索される。リストからスポットを選択し、[Properties]をクリックする。

パスワードを入力、保存し、[Connect]をクリックする。



次の内容は[-wifiap. img]ファイルで実行されるNanoPC-T2のみに適用される。デフォルトではWiFiのAP(アクセスポイント)モードはオンになっているため、無線ルーターに接続できない。以下の手順でWiFiのAPモードをオフにする。

接続する対象となるWiFiルーターを設定する(SSH経由でNanoPC-2にログイン)。次のコマンドを実行し、WiFiデバイスを確認する。[wlan]で始まるものがWiFiデバイスである。

```
ifconfig -a
```

デフォルトで[wlan0]は、WiFiデバイスである。[/etc/network/interfaces.d/]内に同じ名前のコンフィギュレーションファイル(例:[wlan0]ファイル等)を作成する必要がある。

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

wlan0の内容は次のようになる。

```
auto wlan0
iface wlan0 inet dhcp
wpa-driver nl80211
wpa-ssid YourWiFiESSID
wpa-ap-scan 1
wpa-psk YourWiFiPassword
```

上記の中で、[YourWiFiESSID]と[YourWiFiPassword]を実際のESSIDとパスワードに置き換える必要がある。最後に、下記コマンドでホットスポットモードをオフにする。rootユーザーとして実行する必要。コマンド実行後ボードを再起動する。再起動したら、上記設定の通り自動的にWiFiに接続する。

```
su
turn-wifi-into-apmode no
```

5.3 Wi-Fi 無線ホットスポットの配置

WiFiホットスポットの配置を以下の手順で行う。

```
turn-wifi-into-apmode yes
```

システムを再起動する。デフォルトのホットスポット名は[nanopi2-wifiap]で、パスワードは123456789。PCホストから[nanopi2-wifiap]に接続可能になる。接続が成功すれば、SSHをを介して192.168.8.1でNanoPi2に登録できる。

```
ssh root@192.168.8.1
```

パスワードは[fa]である。次のコマンドで無線LANモードを確認できる。

sshのログインをスムーズ、且つ速くするために次のコマンドを起動し、WiFi無線のパワーセービングモードをオフにする。

```
iwconfig wlan0 power off
```

次のコマンドでWiFiモードをチェックできる

```
cat /sys/module/bcmhdhd/parameters/op_mode
```

出力する数字が2であれば、現在無線ホットスポットモードとして機能していることを示す。ステーションモードに切り替えたい場合、以下のコマンドを入力する。：

```
turn-wifi-into-apmode no
```

5.4 Bluetooth を使ってファイルを転送する

GUIの右上にあるBluetoothのアイコンをクリックすると、メニューが表示される。[Make Discoverable]によってNanoPC-T2 が他のBluetoothデバイスから検出可能になる。Devices...は検索画面を開き、近くのBluetoothデバイスを検索する([Make Discoverable]は先に有効にする必要がある)。[Send Files to Divices]でNanoPi2が別のBluetoothデバイス(NanoPC-T2とペア)にファイルを送ることができる。



5.5 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/list>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

まず、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-gnueabi/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-gnueabi/4.9.3/lto-wrapper
Target: arm-cortexa9-linux-gnueabi
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-gnueabi --prefix=/opt/FriendlyARM/toolchain/4.9.3 --with-sysroot=/opt/FriendlyARM/toolchain/4.9.3/arm-cortexa9-linux-gnueabi/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NanoPC-T2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NanoPC-T2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。

NanoPC-T2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、新しく生成したファイルはarch/arm/boot/uImage、HDMI出力をサポートする。SDカードのbootセクションにある同じファイル名のファイルと置き換えれば良い。

LCD表示をサポートするイメージファイルを作成するには設定を変更する必要。

```
touch .scmversion
make nanopi2_linux_defconfig
```

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

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。

コンパイル成功した後、生成したカーネルモジュールが表示される。

6.5 Android 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”はワークフォルダーのことである。

6.5.3 システムをコンパイルする

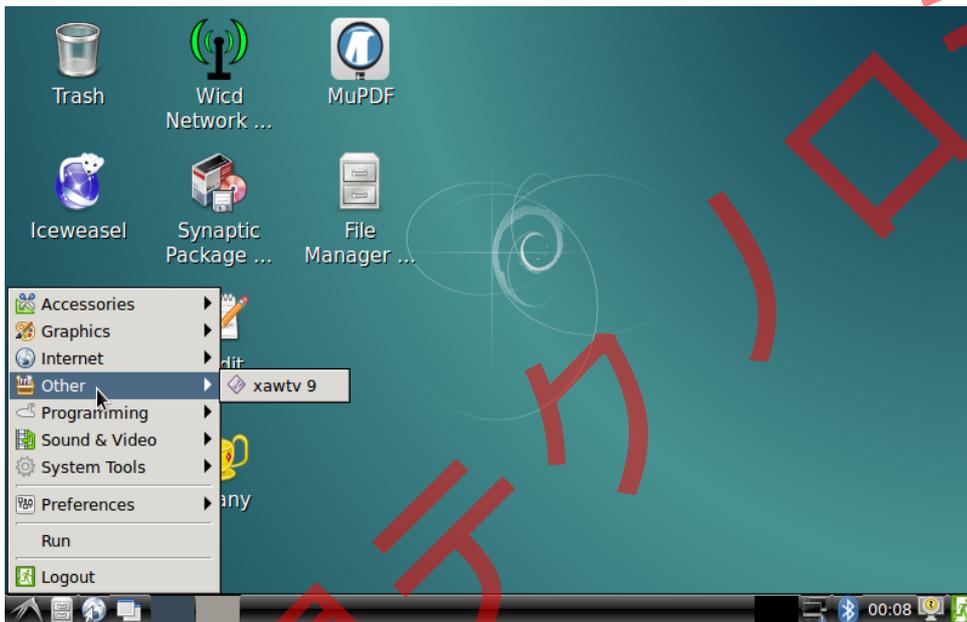
```
source build/envsetup.sh  
lunch aosp_nanopi2-userdebug  
make -j8
```

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。

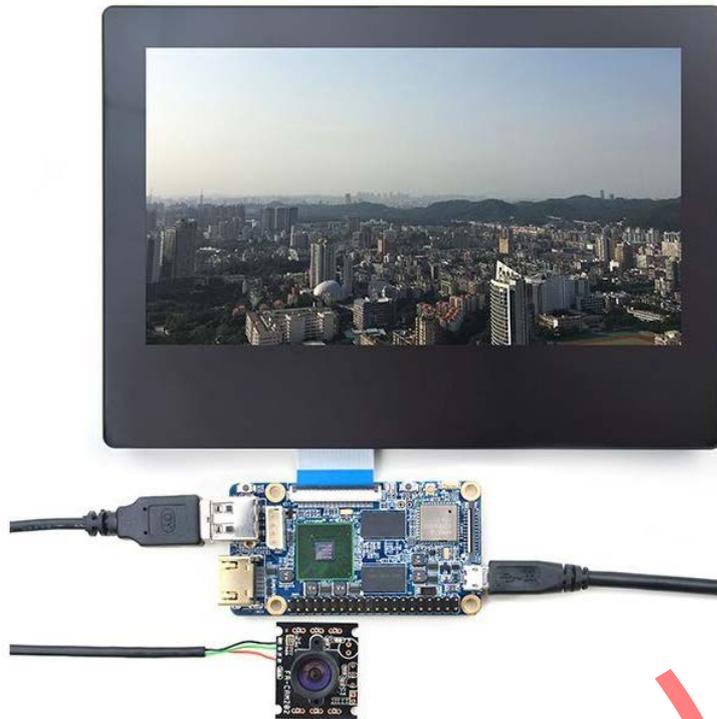
7 カメラモジュールを接続する

7.1 NanoPC-T2 を USB (FA-CAM202) に接続する

このケースでは、NanoPC-T2 は Debian を実行する。Debian が完全にロードされた後、NanoPC-T2 を当社の LCD または HDMI モニターに接続する場合は、GUI の左下のメニューボタン上の [other (その他)] → [xawtv9] をクリックする。USB カメラのアプリケーションが起動し始める。[welcome to rawtv!] を入力後、写真を撮るために [OK] をクリックする。



NanoPi 2接LCD屏使用USB Camera拍照



7.2 NanoPC-2 を CMOS 5M ピクセルのカメラに接続する

この場合は NanoPC-T2 は Android5.1 を実行する。あなたの NanoPC-T2 に LCD または HDMI モニターを接続し、[Camera (カメラ)] のアイコンをクリックすると、アプリケーションがスタートする。



7.3 OpenCV を使用して USB カメラにアクセスする

OpenCV はオープンソースのコンピュータ向けライブラリ であり、クロスプラットフォーム・ビジョンライブラリである。

NanoPC-T2 が実行されると、Debian ユーザーは USB カメラデバイスにアクセスするために OpenCV の API を使用することができる。

- 次に紹介しているのは NanoPC-T2 に C++ で OpenCV を使用方法についてのガイドラインである。
 - ・まず、NanoPC-T2 がシリアル端末または SSH 経由で internet.Login に接続されていることを確認する必要がある。ログイン後、ユーザーネーム (root) とパスワード (fa) を入力する。
 - ・次のコマンドを実行する。

```
apt-get update  
apt-get install libcv-dev libopencv-dev
```

- USB カメラが NanoPC-T2 で作動していることを確認する。NanoPC-T2 のカメラユーティリティを使用し、カメラのテストができます。
- 使用しているカメラのデバイスを確認する。

```
ls /dev/video*
```

注意：弊社のテストケースにおいて、video9 に利用可能で video0 から video8 は専有された

- OpenCV のコードサンプル (C++における公式コード) は /home/fa/Documents/opencv-demo の下にある。次のコマンドでコードサンプルをコンパイルする。

```
cd /home/fa/Documents/opencv-demo  
make
```

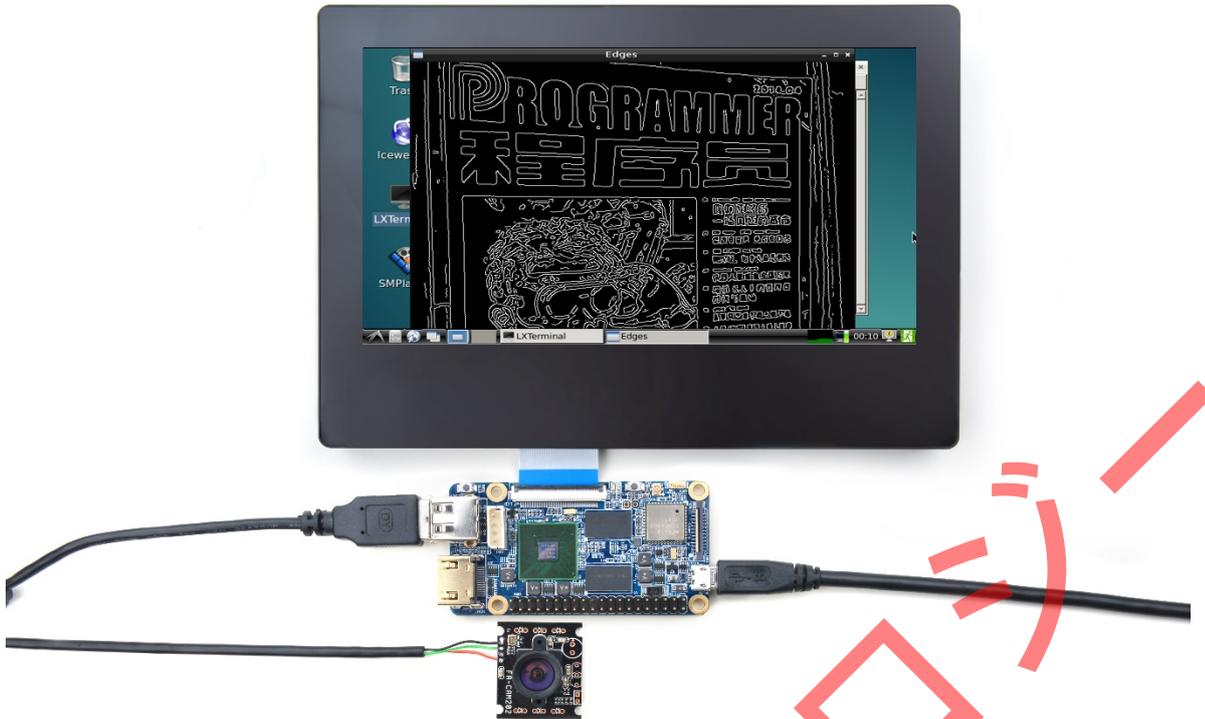
正常にコンパイルが完了した [demo] 実行ファイルが生成される。

- 注意：NanoPC-T2 には 9 つのビデオデバイスがある。しかし、この Open CV のコードサンプルは最大 8 台までしかアクセスできないため、一台のビデオを取り除く必要がある。次のように video0 を削除した。

```
rm /dev/video0  
mv /dev/video9 /dev/video0
```

- NanoPC-T2 を USB キーボードに接続し、次のコマンドを実行する。

```
./demo
```



8. NanoPC-T2 の SD カードパーティションを拡張

8.1. Debian 用

PC ホストの端末に以下のコマンドを実行する。

```
sudo umount /dev/sdx?  
sudo parted /dev/sdx unit % resizepart 2 100 unit MB print  
sudo resize2fs -f /dev/sdx2
```

8.2. Android 用

```
sudo umount /dev/sdx  
sudo parted /dev/sdx unit % resizepart 4 100 resizepart 7 100 unit MB print  
sudo resize2fs -f /dev/sdx7
```

(注意： [/dev/sdx] をお使いのシステムのデバイスネームと置き換える必要がある)