LK-RZG-A02
カメラモジュール（VS-CAM-01）の使用方法

Rev1.2　2023/10/02

目　次

表記

●バージョンに関する表記

弊社提供のソース等に関しては、弊社の管理するバージョン番号がファイル名やフォルダ名に付いている場合があります。
そのバージョン番号に関しては、本ドキュメントでは、『**X**』を使用して表現しております。
そのため、以下のような表記になりますので、その部分は読み替えてください。

例：

以下の表記がある場合

helloworld-X.X.tar.bz2

Ver1.0 での実際のファイル名は、以下になります。

helloworld-1.0.tar.bz2

●コマンドラインの表記

本ドキュメントには、コマンドラインで入力する操作手順が記載されております。
操作は PC 及び XG ボードで行います。それぞれの記述について以下に記載します。

ゲスト OS(Ubuntu)での操作

プロンプトは、『**$**』で記載します。
実際のプロンプトには、カレントディレクトリ等が表示されますが、本ドキュメントでは省略します。

AP-RZG-0A ボード上の Linux での操作

プロンプトは、『**#**』で記載します。
実際のプロンプトには、カレントディレクトリ等が表示されますが、本ドキュメントでは省略します。

本ドキュメント中での入力では、以下のように表現し、入力の最後には、  があります。

例：ゲスト OS(Ubuntu)上で make コマンドを実行する場合の表記

```
$ make
```

コマンドによっては 1 つのコマンドが複数行で記載されている場合もあります。
その場合には、2 行目以降の入力では ENTER キーを押さずに続けて入力し、  の表記がある行の最後で ENTER キーを入力してそのコマンドを実行してください。

例：2 行続いてコマンド入力がある表記

```
$ cd ~/build/tmp/work/aprzg0a-poky-linux-gnueabi/linux-renesas/3.10+git34547b2a5032ce6dca24b745d608d2f3baac187f-r0/git
```

1.　概要

1.1　はじめに

本ドキュメントでは、AP-RZG-0A ボードにカメラモジュール（VS-CAM-01）を接続し、動作確認するための設定について説明します。

・カーネル対応方法
・テストアプリケーションのビルド、実行方法

1.2　開発環境

本ドキュメントでは、Yocto/Poky 開発環境が Ubuntu にインストールされていることが前提となっています。

カメラモジュールのドライバを Linux カーネルに組み込むには、カーネルのビルド環境が必要となります。
カーネルのビルド環境設定は以下のドキュメントを参照してください。

・LK-RZG-A02 Install Manual
・LK-RZG-A02 Software Manual

2.　Linux カーネルへの対応方法

Linux カーネルのデフォルトでは、カメラモジュールを使用する設定になっておりませんので、Linux カーネルを再作成する必要があります。

再作成する手順を以下に説明します。

本手順では、開発環境が、コマンド『**bitbake core-image-weston**』にて作成されていることを前提で説明します。
コマンドに関しての詳細は、AP-RZG-0A の Linux 開発キット（LK-RZG-A02）のソフトウェアマニュアルでご確認ください。

また、動作サンプルアプリでは、タッチパネル LCD キットが必要となりますので、タッチパネル LCD キットのアプリケーションノートを参照し使用できるよう、あらかじめ設定してください。

2.1　Linux カーネルへのドライバ追加

Linux カーネルのデフォルトは、カメラモジュールを使用する設定になっておりませんので、以下の手順で Linux カーネルにカメラモジュールのドライバを組み込むように変更します。

①　ビルド環境の設定をします。

```
$ cd ~/user_work    入力
$ source poky/oe-init-build-env    入力
```

環境設定が終了すると、カレントディレクトリは~/user_work/build に移動します。

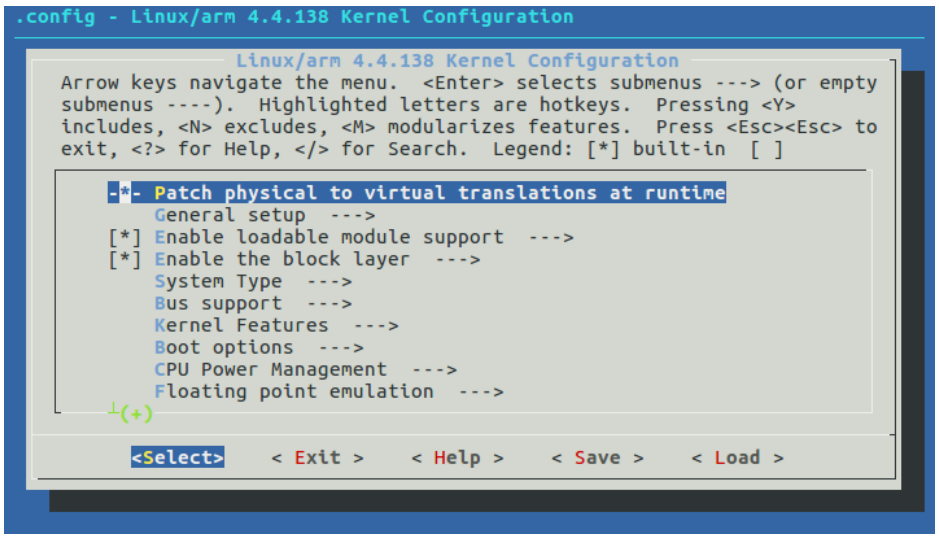
カーネルのコンフィギュレーションを初期化する場合は、以下のコマンド実行します。

bitbake -c configure linux-renesas --force

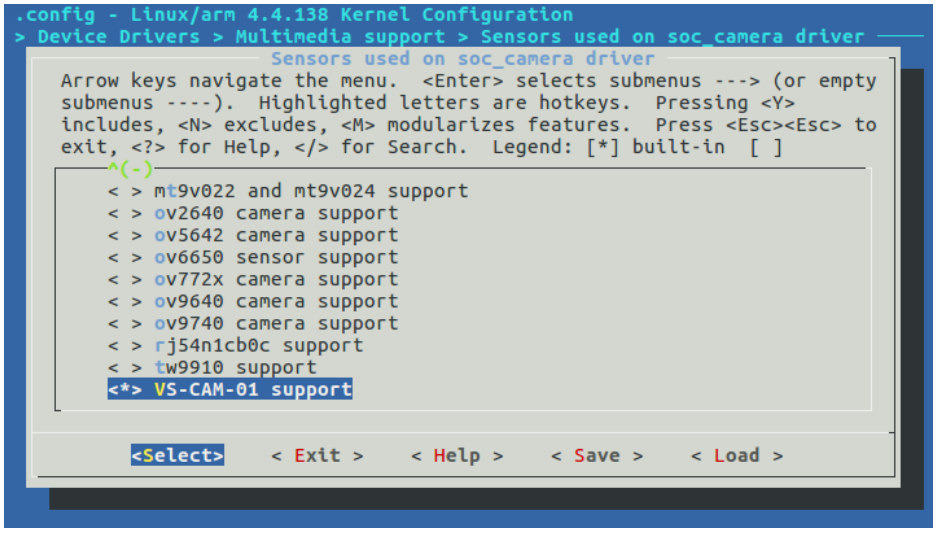
このコマンドを実行すると以前に行われた menuconfig による設定変更、およびドライバなどのソースの変更は全て初期化されます。

② カーネルのカスタマイズをするため、設定画面を開きます。

```
$ bitbake -c menuconfig linux-renesas
```

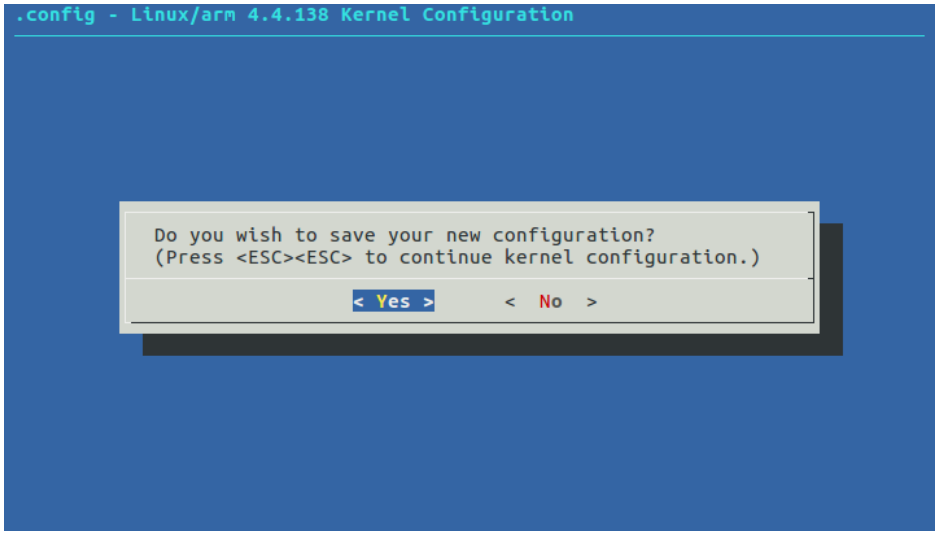


③ Sensors used on soc_camera driver メニューに移動します。
[Device Drivers] - [Multimedia support] - [Sensors used on soc_camera driver]の順に開いていき、
『**VS-CAM-01 support**』を選択します。



上記の画面で何も項目が表示されない場合は、一つ上の階層の『**Multimedia support**』がチェックされているかご確認ください。

④　ESC キーを複数回押し、トップメニューに戻り、設定値を保存します。



2.2　Linux カーネルの作成

Linux カーネルのみ再ビルドする手順を説明します。

①　カーネルのビルドをします。

```
$ bitbake -c compile linux-renesas --force
Loading cache: 100% |#####################################################################| ETA:  00:00:00
Loaded 1846 entries from dependency cache.

          :
      ＜途中省略＞
          :

NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 226 tasks of which 225 didn't need to be rerun and all succeeded.

Summary: There was 1 WARNING message shown.
```

②　カーネルのビルドが成功したら、作成されたカーネルをデプロイ(配布)します。

```
$ bitbake -c deploy linux-renesas
Loading cache: 100% |#####################################################################| ETA:  00:00:00
Loaded 1846 entries from dependency cache.

          :
      ＜途中省略＞
          :

NOTE: Tasks Summary: Attempted 233 tasks of which 227 didn't need to be rerun and all succeeded.

Summary: There was 1 WARNING message shown.
```

③　正常に完了しますと、ディレクトリ『./tmp/deploy/images/aprzg0a』にファイルが生成されます。
カメラモジュールの追加で更新するファイルは、以下となります。

ファイル名	内容
uImage	カーネルイメージファイル

Table 2.2-1 更新ファイル

作成された Linux カーネルを microSD カードにコピーして動作できるようにする必要があります。
その手順に関しては、AP-RZG-0A の Linux 開発キット（LK-RZG-A02）のソフトウェアマニュアルでご確認ください。

3.　カメラサンプルアプリのビルド

カメラモジュールの動作確認を行うためのサンプルアプリのビルド方法を説明します。

作成のための準備

① 作業用ディレクトリ『**aprzg0a-app**』をホームディレクトリに作成します。
すでに作成されている場合は、手順②にお進みください。

```
$ mkdir ~/aprzg0a-app
```
入力

② ディレクトリ『**aprzg0a-app**』に移動します。

```
$ cd ~/aprzg0a-app
```
入力

③ 作業用ディレクトリに付属 DVD-ROM 内の以下の 1 つのファイルをコピーします。
手順④～⑥で例として DVD-ROM から直接コピーする方法を記述します。他の方法でコピーする場合には、コピー作業完了後に、手順⑦にお進みください。

vscam-X.X.tar.bz2

※『X.X』にはバージョン番号が入ります。Ver1.0 の場合は、『1.0』

④ DVD-ROM をドライブに挿入します。
デフォルトでは、自動でマウントされますが、マウントされない場合は、以下のコマンドを実行します。

```
$ gvfs-mount -d /dev/sr0
```
入力

マウントされているかどうかは、『**mount**』コマンドで確認できます。
以下のように、『**/dev/sr0**』が表示されている場合は、すでにマウントされています。
（『*******』は、DVD-ROM のボリュームラベルになります。）

```
$ mount
入力
          :
     ＜途中省略＞
          :
/dev/sr0 on /media/******* type udf (ro,nosuid,nodev,uhelper=udisks,uid=1000,
gid=1000,iocharset=utf8,umask=0077)
```

⑤ 1 つのファイルをコピーします。コマンド途中の『*******』は、DVD-ROM のボリュームラベルになります。
そのため、その部分は挿入した DVD-ROM に合わせて入力してください。

```
$ cp /media/guest/*******/sample/vscam-X.X.tar.bz2 .
```
入力

⑥ DVD-ROM をアンマウントします。

```
$ umount /dev/sr0
```
入力

⑦ サンプルソースを展開します。

```
$ tar -xjpf vscam-X.X.tar.bz2
```
入力

サンプルアプリケーションのビルド

サンプルアプリケーションのビルド手順を説明します。
サンプルプログラムは、LCD-KIT-B01/C01/C02 用と LCD-KIT-D01/D02 用の 2 種類存在します。
内容は同じのため、以下では、LCD-KIT-B01 を使用した手順で説明します。

① SDKの環境を設定します。

```
$ . /opt/poky/2.4.2/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
```

最初の『.』と『**/opt/...**』の間には、半角スペースが必要ですので、ご注意ください。

『**. /opt/...**』の環境設定コマンドは、各ターミナル毎に行う必要があります。
同一のターミナルで 2 回実行する必要はありませんが、別のターミナルを起動した場合には、再度実行する必要があります。

② 準備作業で展開した作業用ディレクトリの『**vscam**』へ移動します。

```
$ cd ~/aprzg0a-app/vscam_app/vscam
```

LCD-KIT-D01/D02 の場合は、『**~/aprzg0a-app/vscam_app/vscam_lcdkitd0x**』となります。

③ サンプルアプリケーションをビルドします。

```
$ make
# make tool will compile sample app
make -f Makefile_Cloud
make[1]: ディレクトリ '/home/guest/app/vscam_app/vscam' に入ります

          :
     <途中省略>
          :

arm-poky-linux-gnueabi-g++  -march=armv7ve -mfpu=neon -mfloat-abi=hard -mcpu=cortex-a7
 --sysroot=/opt/poky/2.4.2/sysroots/cortexa7hf-neon-poky-linux-gnueabi -Wl,-O1 -Wl,--h
ash-style=gnu -Wl,--as-needed -Wl,-O1 -o vscam01_test main.o opencv_image.o qrc_qml.o
  -L/home/guest/user_work/build/tmp/work/cortexa7hf-neon-poky-linux-gnueabi/opencv/3.3
+gitAUTOINC+87c27a074d_2a9d1b22ed_a62e20676a_34e4206aef_fccf7cd6a4-r0/package/usr/lib
-lopencv_imgproc -lopencv_core -lopencv_highgui -lopencv_objdetect -lstdc++ -lopencv_v
ideoio -lQt5Quick -lQt5Gui -lQt5Qml -lQt5Network -lQt5Core -lGLESv2 -lpthread
make[1]: ディレクトリ '/home/guest/aprzg0a-app/vscam_app/vscam' から出ます
```

ビルドが成功しますと、『**vscam01_test**』が作成されます。

LCD-KIT-D01/D02 の場合は、『**vscam01_test_lcdkitd0x**』となります。

4.　動作確認

本章では、カメラモジュールの動作確認方法について説明します。

4.1　Linux の起動

『**2.　Linux カーネルへの対応方法**』で作成した Linux カーネルの動作確認方法を説明します。
なお、ルートファイルシステムは、LK-RZG-A02 ソフトウェアマニュアルで作成した microSD を使用した方法で説明します。

① AP-RZG-0A ボードに、以下のようにタッチパネル LCD キットとカメラモジュールを接続します。
詳しい接続方法に関しては、使用するタッチパネル LCD キット および カメラモジュールの『**ハードウェアマニュアル**』でご確認ください。

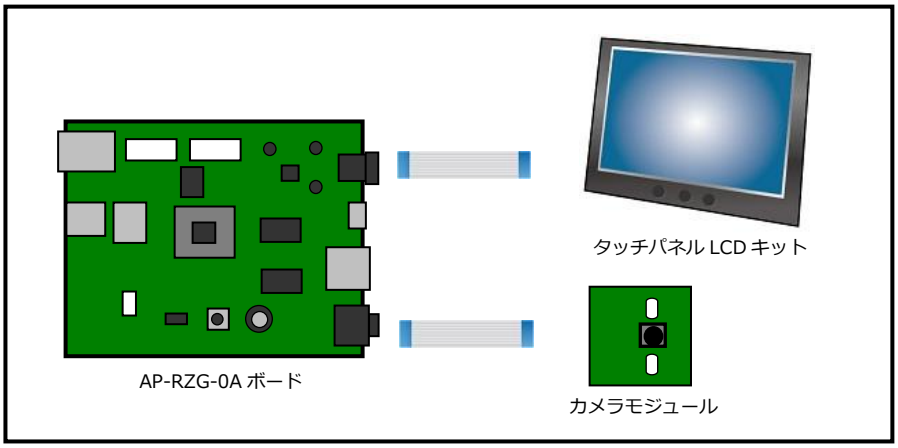


Fig 4.1-1　AP-RZG-0A とタッチパネル LCD キット／カメラモジュールの接続

② 『**2.　Linux カーネルへの対応方法**』の Linux カーネルが書き込まれた microSD カードをスロットに挿入した後に、電源を入れます。

4.2　カメラテストアプリの実行

作成したカメラテストアプリを使用して、カメラモジュールの動作確認を行います。
以下の手順では、LCD-KIT-B01 を例とした動作確認方法となります。

① 『**3.　カメラサンプルアプリのビルド**』で作成した vscam01_test をターゲットボードのファイルシステムにコピーします。

② vcam01_test を実行します

```
# ./vscam01_test
```
入力

画面をタッチするとプログラムが終了します。
（コンソールから終了するときは、『**Ctrl + C**』キーを押し、アプリを終了します）

ご注意

・本文書の著作権は、株式会社アルファプロジェクトが保有します。
・本文書の内容を無断で転載することは一切禁止します。
・本文書に記載されているサンプルプログラムの著作権は、株式会社アルファプロジェクトが保有します。
・本文書に記載されている内容およびサンプルプログラムについての技術サポートは一切受け付けておりません。
・本文書の内容およびサンプルプログラムに基づき、アプリケーションを運用した結果、万一損害が発生しても、弊社では一切責任を負いませんのでご了承下さい。
・本文書の内容については、万全を期して作成いたしましたが、万一ご不審な点、誤りなどお気付きの点がありましたら弊社までご連絡下さい。
・本文書の内容は、将来予告なしに変更されることがあります。

商標について

・Linux は、Linus Torvalds の米国およびその他の国における登録商標または商標です。
・Yocto Project は、Linux Foundation の登録商標です。

・その他の会社名、製品名は、各社の登録商標または商標です。

ALPHA PROJECT
株式会社アルファプロジェクト
〒431-3114
静岡県浜松市中央区積志町 834
https://www.apnet.co.jp
E-Mail: query@apnet.co.jp