PetaLinux使用笔记（UG1144 2018.2~2018.3）

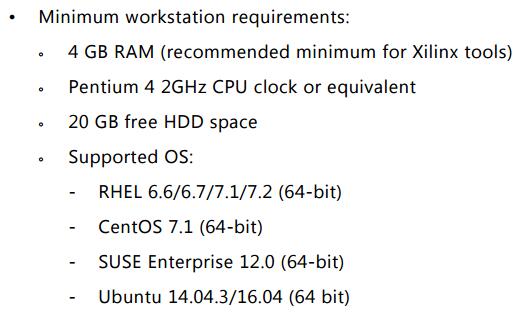
## 概述：

PetaLinux是一个嵌入式Linux系统开发工具包，针对的是基于Xilinx FPGA的系统芯片设计。本文主要讲述了简单的petalinux安装及使用方法，详细介绍请参阅官方文件，推荐ug1144.

**重要:PetaLinux v2018.2只适用于Vivado 2018.2。即PetaLinux与Vivado的版本应该相同，不然会出现bug。**

## 安装要求

### （一）PetaLinux工具的最小安装要求是:



**其中需要注意的就是尽可能的将硬盘空间分配的稍微大一些，后期每生成一个工程就需要6G左右的空间，如果条件允许就尽量将空间放大。内存方面，4G内存亲测可用，如果是在虚拟机下运行，可以将虚拟机的处理器个数分配的多一点，这样避免在build阶段消耗太长时间。亲测，在处理器个数只有一个的时候，其中build中有一步要花费1小时+，而整个工程需要2小时+。后来将处理器个数改为4之后，总体时间约40分钟就差不多了。**

### （二）必要的支持工具库

以下是列表，最左边是支持包的名称，右侧是各个操作系统下，该支持包的安装包名称（其实就是同一个功能在不同系统下有不同的叫法），只需要关注你所用的操作系统的那一列就可以了。记得在安装之前把这些支持库全部安装好。

安装命令 ：apt-get install 库名

例如： apt-get install make (安装时，有些后缀版本号不要加上)

如果系统有更新可以先更新apt-get, cmd: sudo apt-get update

**IMPORTANT:** PetaLinux tools require your host system "/bin/sh" is bash. If you are using Ubuntu distribution and your "/bin/sh" is dash, you can consult your system administrator to change your default with sudo dpkg-reconfigure dash command.

(在sudo权限下使用dpkg-reconfigure dash命令改成bash)

**查看与使用**

先用命令ls -l /bin/sh 看看

结果是： /bin/sh -> dash

**我们会发现Ubuntu默认采用的是 dash**

**如果要修改默认的sh，可以采用命令**

sudo dpkg-reconfigure dash

然后选择【否】

成功后再执行ls -l /bin/sh 看看

结果是： /bin/sh -> bash

修改成功！

|  |  |  |
| --- | --- | --- |
| **Tool / Library** | **Ubuntu 16.04.3** | **Command** |
| dos2unix | tofrodos\_1.7.13+ds-2.debian.tar.xz | apt-get install tofrodos |
| ip | iproute2 4.3.0-1ubuntu3 | apt-get install iproute2 |
| gawk | gawk (1:4.1.3+dfsg-0.1) | apt-get install gawk |
| gcc | - |  |
| g++  (gcc-c++) | - |  |
| xvfb | xvfb (2:1.18.3-1ubuntu2.3) | apt-get install xvfb |
| git | git 1.7.1 or above | apt-get install git |
| make | make 3.81 | apt-get install make |
| netstat | net-tools | apt-get install net-tools |
| ncurses devel | libncurses5-dev | apt-get install libncurses5-dev |
| tftp server | tftpd | apt-get install tftpd |
| zlib devel (also, install 32-bit of this version) | i386/zlib1g-dev/1:1.2.8.dfsg -2ubuntu4-dev | apt-get install zlib1g-dev:i386 |
| openssl devel | libssl-dev | apt-get install libssl-dev |
| flex | flex | apt-get install flex |
| bison | bison | apt-get install bison |
| libselinux | libselinux1 | apt-get install libselinux1 |
| gnupg | gnupg | apt-get install gnupg |
| wget | wget | apt-get install wget |
| diffstat | diffstat | apt-get install diffstat |
| chrpath | chrpath | apt-get install chrpath |
| socat | socat | apt-get install socat |
| xterm | xterm | apt-get install xterm |
| autoconf | autoconf | apt-get install autoconf |
| libtool | libtool | apt-get install libtool |
| tar | tar:1.24 | apt-get install tar |
| unzip | unzip | apt-get install unzip |
| texinfo | texinfo | apt-get install texinfo |
| zlib1g-dev | zlib1g-dev | apt-get install zlib1g-dev |
| gcc-multilib | gcc-multilib | apt-get install gcc-multilib |
| build-essential | build-essential | apt-get install build-essential |
| libsdl1.2-dev | libsdl1.2-dev | apt-get install libsdl1.2-dev |
|  |  | apt-get install libglib2.0-dev |
|  |  | apt-get install screen |
|  |  | apt-get install pax |
|  |  | apt-get install gzip |

## 安装

Petalinux工具的安装十分简单，只需要一条命令就可以解决，不需要选择其他选项，就可以把PetaLinux安装到指定目录之中。

但是有几点需要注意：

1 确保安装之前把所需要先支持的工具库安装好，具体见上文。

2 安装时不需要root权限，不要获取权限

3 安装后目录无法更改，确定好安装目录再安装。

接下来就是安装过程：

* + 1. 预备条件——安装Petalinux工具

1. 首先要确保所有支持库和tool工具都安装了，参考table1-3。
2. 下载Petalinux发布版：要找到合适的vivado下载界面下下载对应的Petalinux才好。
   * 1. 运行Petalinux工具安装包，这个安装包在制定好安装路径后，一次性安装完，不需要配置的。如果缺少库或者支持工具，可能安装不成功。同时你不能用root权限安装。但是不用root权限，不能安装到root根目录下的，所以如果要安装到根目录，还要chmod目录权限为755才行。所以干脆就安装到home目录下。
3. 建立安装文件夹 mkdir -p /home/petalinux
4. ./petalinux-vxxxx.run /home/petalinux

按照提示，需要输入y查看license，然后看到最后点击q，退出后再按照提示输入y接受license。一共有三个协议要同意。

* + 1. 设置工作环境（见ug1144）。

1. 假定Petalinux已经被安装，bash已经被设定。
2. 建立Petalinux工作环境：
   1. For Bash as user login shell:

$ source <path-to-installed-PetaLinux>/settings.sh

* 1. For C shell as user login shell:

$ source <path-to-installed-PetaLinux>/settings.csh

1. 校验工作环境是否完成：

$ echo $PETALINUX

显示：/opt/pkg/petalinux 表示OK，这个路径根据你的实际安装路径来显示。

## 设计流程概述：

通常，Petalinux工具是按照一系列流程来使用的。下面表格提供了一个DEMO演示这个工作流程，并演示了每个环节需要用到的工具和命令。

|  |  |  |
| --- | --- | --- |
| **Design Flow Step** | **Tool / Workflow** | **中文概述** |
| Hardware Platform Creation | Vivado | 创建硬件平台 |
| Create PetaLinux Project | petalinux-create -t project | 创建Petalinux |
| Initialize PetaLinux Project | petalinux-config --get-hw-description |  |
| Configure System-Level Options | petalinux-config |  |
| Create User Components | petalinux-create -t COMPONENT |  |
| Configure the Linux Kernel | petalinux-config -c kernel |  |
| Configure the Root Filesystem | petalinux-config -c rootfs |  |
| Build the System | petalinux-build |  |
| Deploy the System | petalinux-package |  |
| Test the System | petalinux-boot |  |

下面将详细介绍下Petalinux部分的创建流程：

## 打开并建立一个工程：

如果使用一些官方板卡，可以从官网下载相应的BSP支持包。下面的步骤是定制自己的平台，并生成一些列的支持文件，直到搭建Petalinux完成。

## 配置Petalinux的硬件平台

用vivado建立基础的硬件平台。要把管脚约束啥的都做好。下面说下这个平台必要的几个模块：

.至少64MB的外部存储，zynq有DDR就行了。

.至少一个UART作为串口终端

.存储器 QSPI FLASH SD卡等

.以太网(可选，主要上OS了，起码可以联网吧)

.One Triple Timer Counter (TTC) (required)。Zynq PS还需要一个TTC。如果板卡上没有分配管脚，直接勾选就行了，不一定要分配管脚。（If multiple TTCs are enabled, the Zynq-7000 Linux kernel uses the first TTC block from the device tree. Please make sure the TTC is not used by others.）

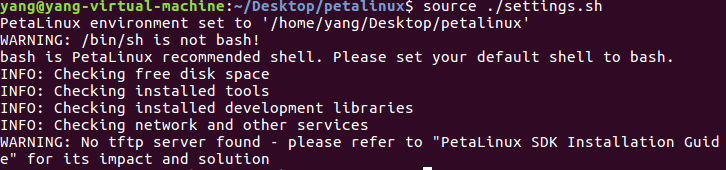
## 从vivado中导出硬件平台信息（Export Hardware Platform to PetaLinux Project）

在vivado中生成bitstream后，执行“**Export Hardware**”后在vivado的工程路径下的xxx.sdk文件夹下会有类似于system\_wrapper.hdf的硬件描述文件生成。 需要把这个文件拷贝到linux下（下面细说）。（貌似也能用DSA文件，类似SDSOC中使用的）

Petalinux工具可以根据这个hdf文件生成“设备树源文件，uboot配置头文件，并使能一些Xilinx IP核的驱动”详见ug1144的Appendix B, PetaLinux Project Structure.。

## 启动PetaLinux工作环境

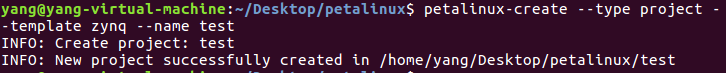
PetaLinux的启动非常简单，进入到petalinux的安装目录后，在命令窗口输入以下命令： source ./settings.sh 即可启动



## 创建一个Petalinux自定义工程

1、当petalinux启动之后，就可以使用以下命令创建一个工程了，注：test为工程名，随意修改:

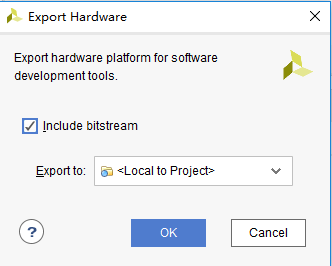
petalinux-create --type project --template zynq --name test



## 获取硬件信息

1）从已经完成好的vivado的工程中获取xxx.hdf

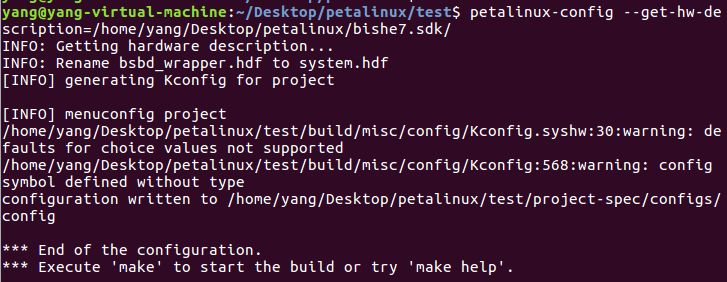
将vivado工程生成比特流文件后,选择File->Export->Export Hardware,此时xxx.hdf文件已经在xxxxx.sdk中了，将整个xxxxx.sdk文件夹拷进虚拟机。本质上就要一个hdf文件。如果已经有SDK的工程，可以不拷贝过来。(如果是针对Sdsoc的则不需要勾选include bitstream,毕竟硬件加速会生成新的bit的)



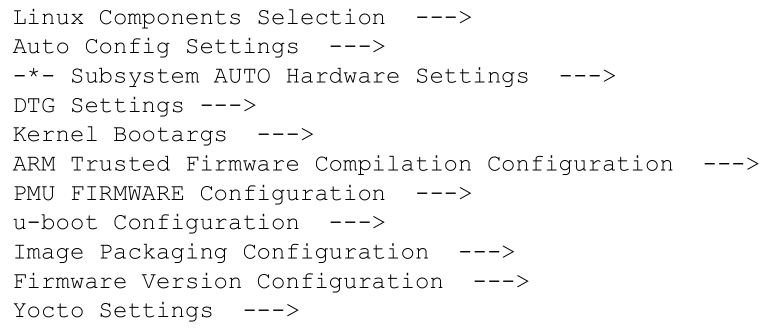
2)在命令窗口输入 cd test（test为工程名），进入工程文件夹，此步非常关键，如果不进入会提示找不到指令之类的。

3）进入工程文件夹后，在命令窗口输入以下指令，petalinux会自动读取硬件信息，定位到包含hdf的文件路径就行。

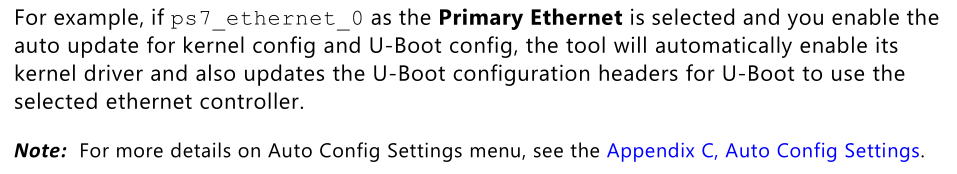
petalinux-config --get-hw-description=/home/yang/Desktop/petalinux/bishe7.sdk/



此时会出现以下图形化界面（左右选择保存或退出 上下选择设置选项）此步可以跳过 直接退出也可以。（这个界面只有在第一次执行该命令，或者hdf有变化时候才会出现。）



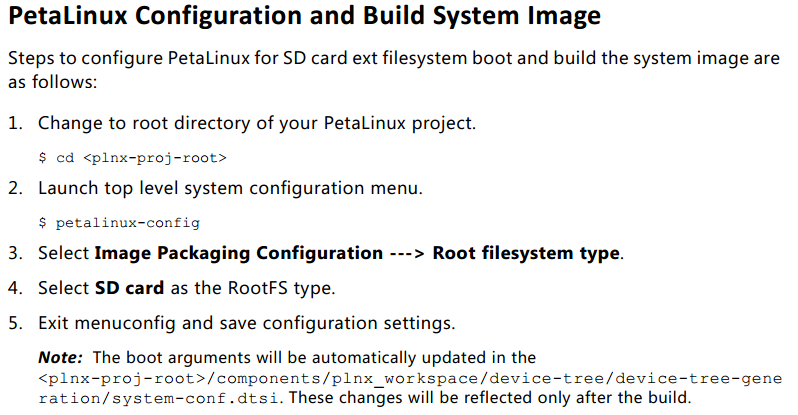
（以上是Petalinux的系统配置菜单，例如“Subsystem AUTO Hardware Settings”“Auto Config Settings”他们直接影响包含设备树、内核配置和uboot等配置步骤，。但是这里其实有很多配置可以增删改查，例如USB摄像头的驱动支持V4L2配置, USB转串口等等，这些很关键，待以后慢慢细说）。



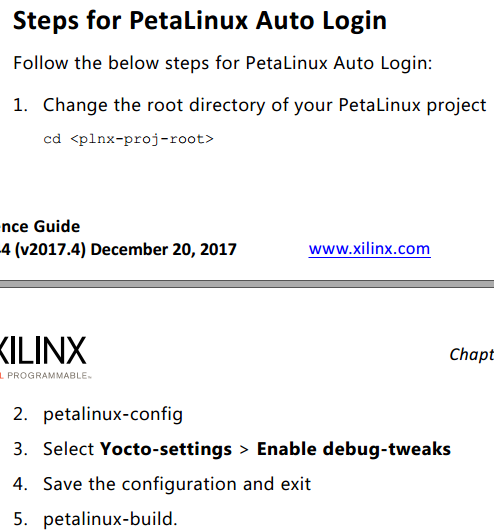
## 下面说下SD卡扩展分区启动方式的配置和自动登录（免linux登录密码）的配置。在ug1144中有提到，不过比较靠后，这里根据实际使用需求先说一下，方便系统生成移植。以后如果还有要增删改查的配置，将在文档后面继续补充。步骤还是参考6、7、8、9、10步操作。

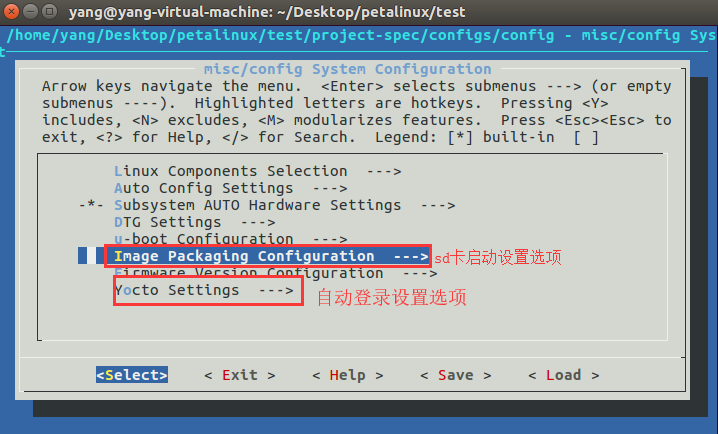
1. **Configuring SD Card ext filesystem Boot（ug1144）**：

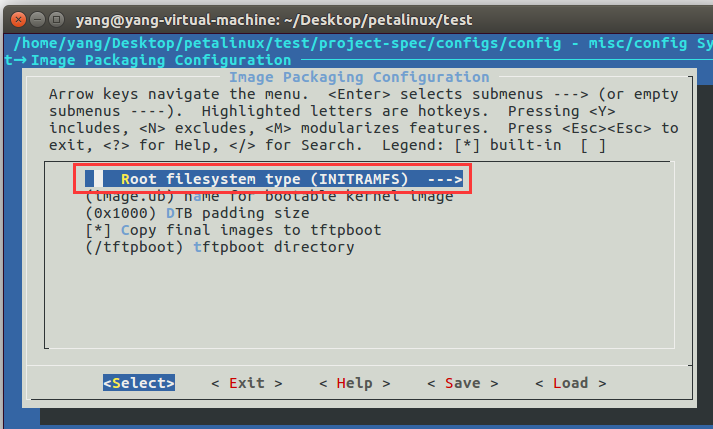
**配置成第二分区启动后，就不能一个分区启动了**。二选一操作。如果RAMdisk启动，就选择INITRD

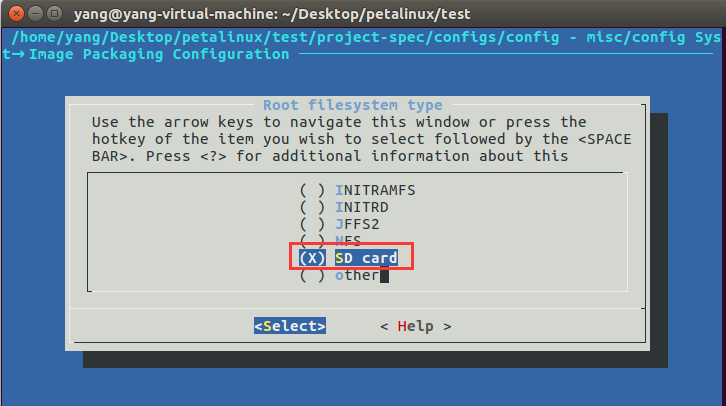


1. PetaLinux Auto Login



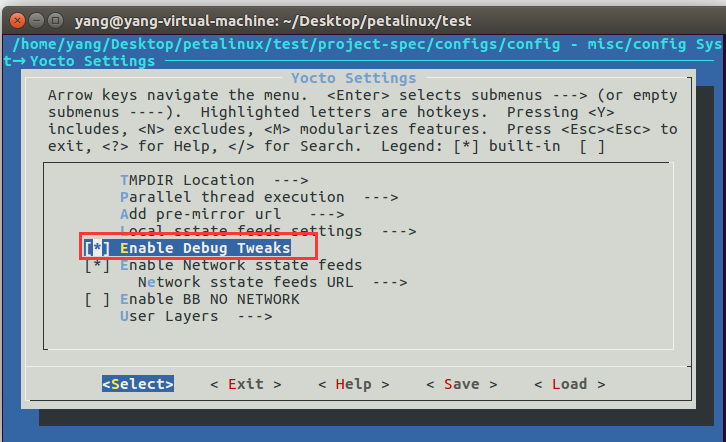


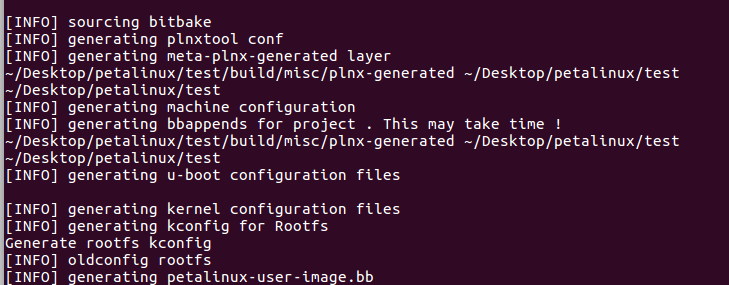




此步不设置 可以仅用第一分区启动。

接下来设置自动登录 在红框处输入y即可

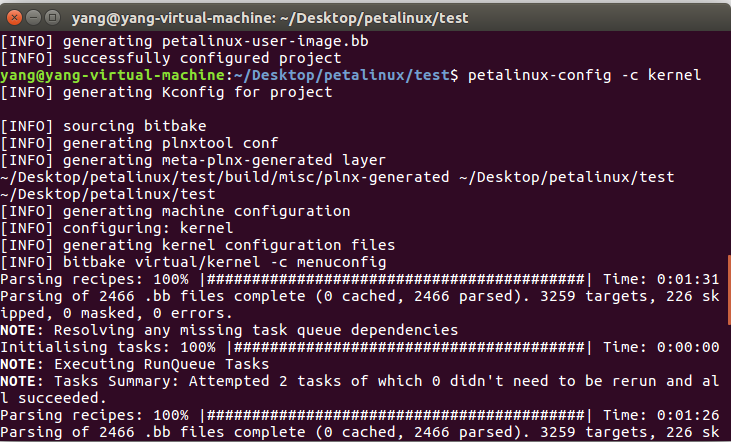


接下来保存退出，出现以下界面可能会花费一段时间。另外如果还想再次修改配置，可以通过petalinux-config指令再次进入。以下7、8、9操作步骤就是再进入后操作的。

## (可选)再配置linux内核

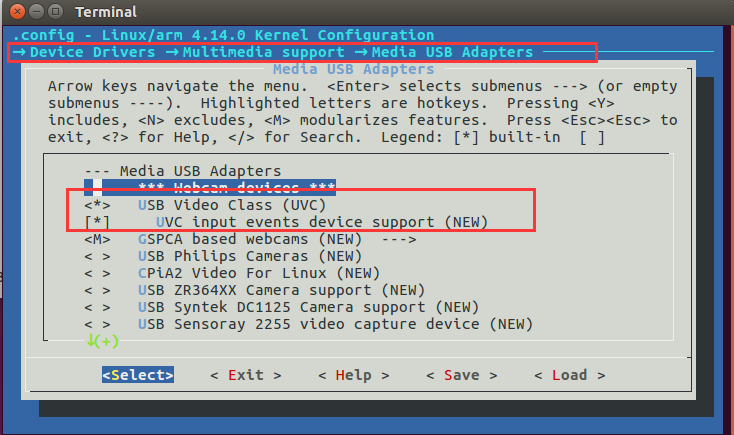
在获取硬件信息完成后，输入以下指令后还可以继续修改内核配置，进入内核配置界面。

petalinux-config -c kernel

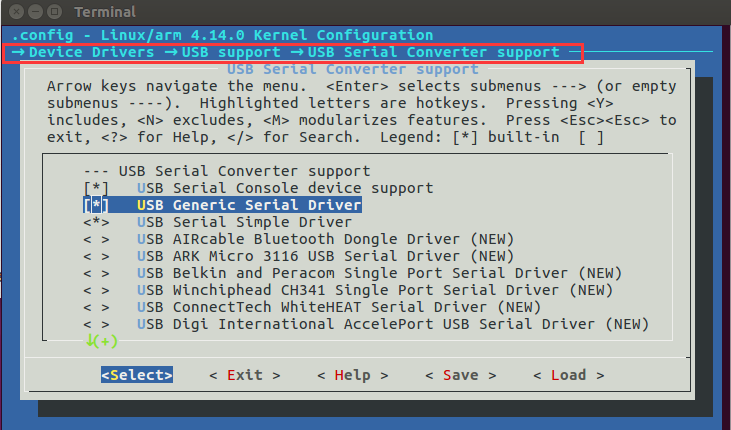


在内核配置中，可以操作很多功能：添加USB支持V4L2，USB转串口，网口驱动等等，还可以添加自己的应用，或者一些SPI，i2c的驱动等等。这里根据经验把用到的都配置起来。后续还能补充：

1. USB支持V4L2：



1. 添加USB转串口驱动：这个路径下可以全部勾选，这样可以支持更多的转串口芯片。通常CP210X的用的多。



1. Sdsoc的配置，在这里还有一些特别要求，详情见Sdsoc的内核配置文档UG1146，或者个人整理的中文简述。他用的是该指令：

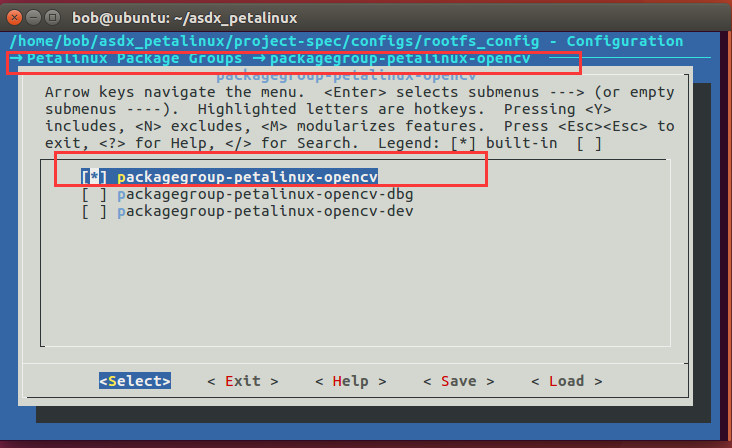
petalinux-config -p < petalinux\_project > -c kernel 一样的,只要在工程目录下

## (可选)在配置内核中配置文件系统

输入以下指令，运行出现以下界面，如图选择，进入，然后在自己的apps面前输入y保存退出。

petalinux-config -c rootfs

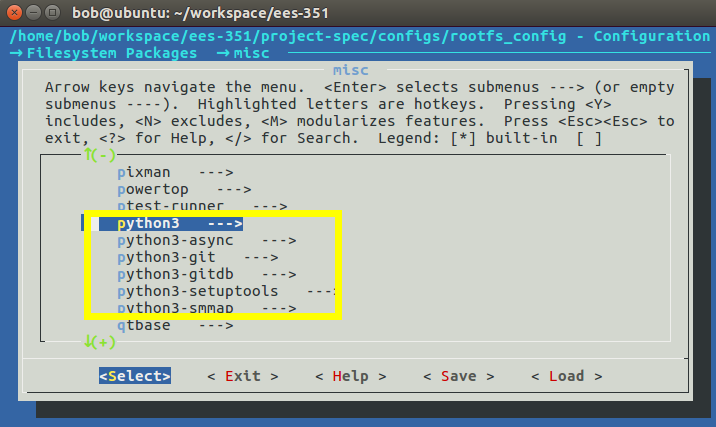
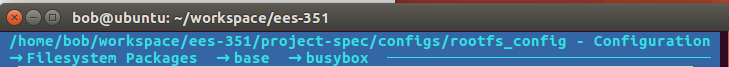
1. Sdsoc的配置，在这里还有一些特别要求，详情见Sdsoc的ROOTFS配置文档UG1146，或者个人整理的中文简述。
2. 添加一些第三方编译库，xilinx提前已经提供了，这里可以加载。比如opencv的库。Python3的支持等等，如果版本不是自己需要的，也可以后续自己添加。



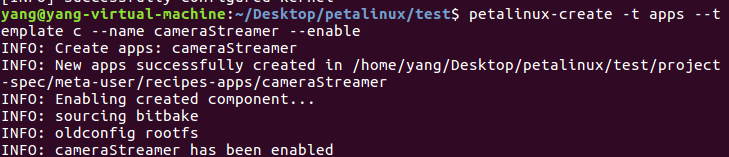
1. Icluding Prebuilt Libraries（添加预编译的库）：

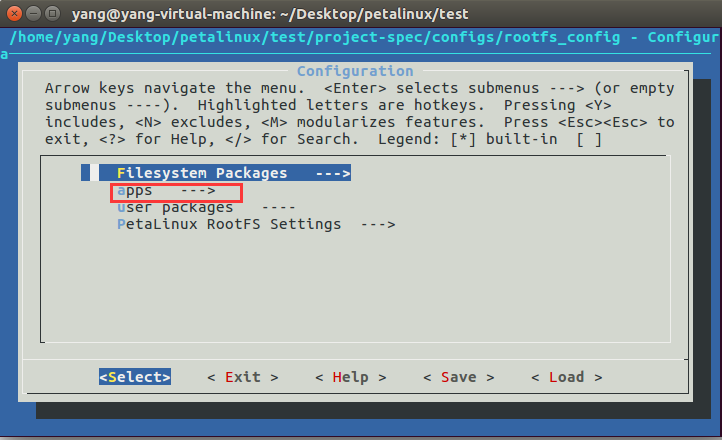
在rootfs中 比如添加Petalinux自带的opencv.so 或者自己编译的opencv.so.

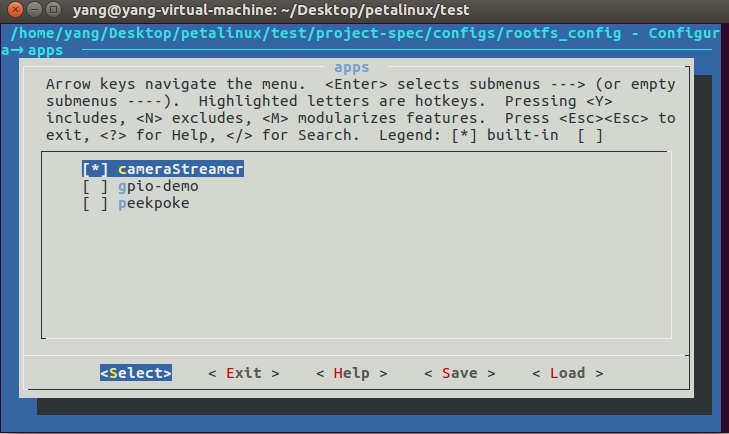
本人发现petalinux默认没有安装apt-get工具，gcc,等等，很多Linux常用的开发工具，安装岸上所说，最好安装一下一个工具包build-essential。这个要找一下。我在其他嵌入式开发的书中也看到过，需要安装这个包。所以最好配置进peta中。好像在misc什么配置中。下面列几个常用的工具和库路径，没有完全验证过。



1. (可选)在配置内核中添加自己的应用（根据需求添加 也可以后期自己添加 此步可以跳过）

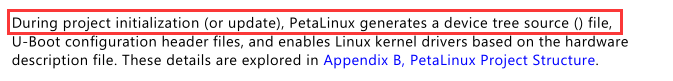
输入以下指令，添加自己的应用。系统反应完成后，将写好的源码放进……./工程目录/ /project-spec/meta-user/recipes-apps/应用名字/files/应用名.c，代替自动生成的代码。

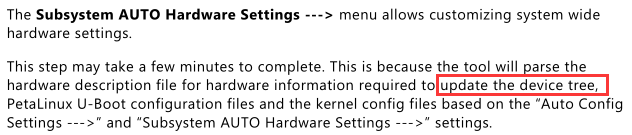


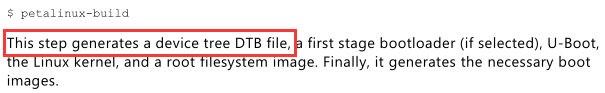


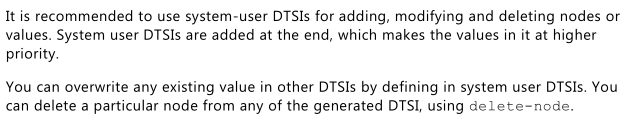
## 设备树修改操作

建议必须在配置内核、rootfs后，才能修改设备树，最后才能build。但是设备树的修改有点“讲究”。根据UG1144上面有如下所说：









上面四句话的意思就是：

1. 在创建系统和更新（有条件的）系统配置的时候会自动生成设备树源文件，这些文件指的是如下目录里的设备树文件：

/petalinux工程目录/components/plnx\_workspace/device-tree/device-tree

1. 只要在petalinux-build操作之前，进行了Petalinux-config操作，或者Petalinux-config –c kernel操作就会在build时候重新更新上述目录的文件。
2. 根据上述原因，和最后一个截图的意思。用户只能通过

“System-user.dtsi”文件来修改添加自己想要的设备节点。同时也能删除系统生成的节点。该设备树源文件在如下目录：

/petalinux工程目录/project-spec/meta-user/recipes-bsp/device-tree/files

那么有两种方法增删改查设备树：

方法一：官方的，通过System-user.dtsi的修改来操作。

方法二：个人尝试，通过petalinux-config配置+二次修改系统生成的设备树+禁止系统刷新掉已经修改的设备树的操作来修改。

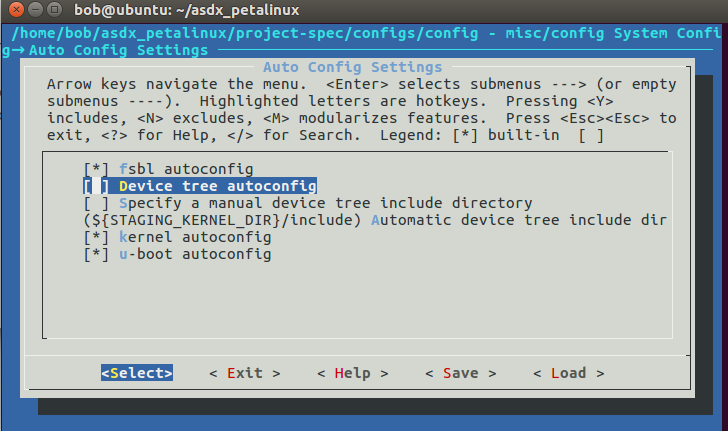
下面以修改USB0口为HOST模式来尝试上述两种方法：

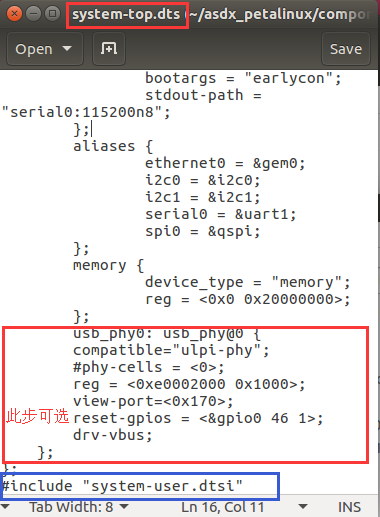
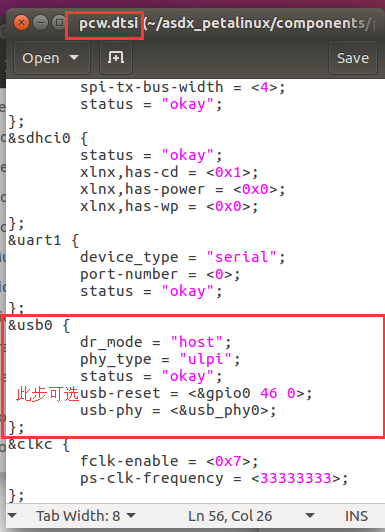
方法一：

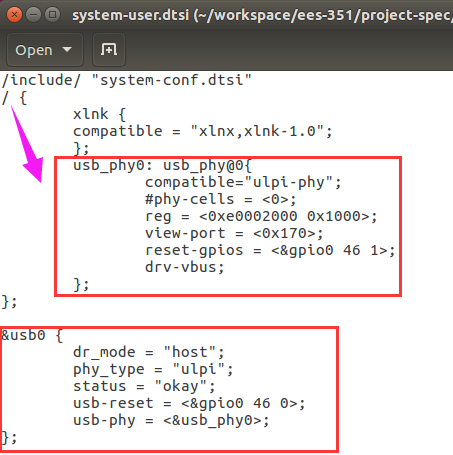
1. 在在配置完内核，rootfs后，去修改System-user.dtsi添加如下内容后，再build。

方法二：有点麻烦，但是很接近原版linux的操作。

1. 当按照前面方法执行到这里的时候，基本配置已经做完。所以，直接先petalinux-build操作。这样借助先前的操作先让系统生成需要的设备树文件。（以后要修改的话，这一步就不要了，已经有了）
2. 再次通过petalinux-config指令进入peta的配置界面，并进入到Auto Config Settings，并去掉Device tree autoconfig前面的\*号。表示不自动生成设备树源文件。
3. 然后去修改/components/plnx\_workspace/device-tree/device-tree下的源文件，如同生成普通linux设备树一样。就不用在通过别的设备树源文件来修改了。
4. Build操作即可。
5. 其实通过第三张截图中的蓝色框能看出来，第二种方法和第一种方法其实本质是一样的。只是把修改的内容放到了不同的地方，但是都会被完全编译进去的。推荐第一种方法。







usb\_phy0: usb\_phy@0{

compatible="ulpi-phy";

#phy-cells = <0>;

reg = <0xe0002000 0x1000>;

view-port = <0x170>;

reset-gpios = <&gpio0 46 1>;

drv-vbus;

};

&usb0 {

dr\_mode = "host";

phy\_type = "ulpi";

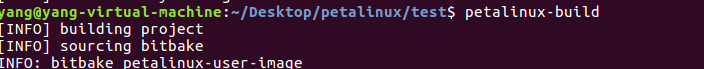
status = "okay";

usb-reset = <&gpio0 46 0>;

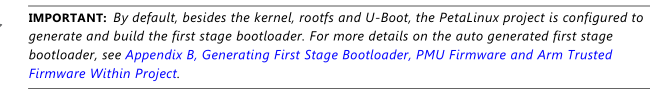
usb-phy = <&usb\_phy0>;

};

## 用 petalinux-build 命令生成系统镜像



注：此步会消耗比较长的时间，请耐心等待。（几十分钟到几小时，根据CPU性能来，这里建议全核运行）这里就是编译linux内核、uboot、文件系统等等操作了。生成的images和设备树在工程目录的images/linux子目录下。除了生成rootfs，kernel,uboot外，还生成first stage bootloader。

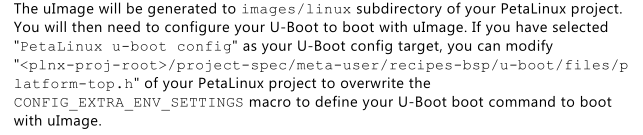


想要生成uImage：

1. 用petalinux-build命令，生成的是FIT image（貌似是uImage.ub）以及RAM disk image还有rootfs.cpio.gz.u-boot。如果想生成uImage，可以用petalinux-package –image代替。生成的uImage被生成在images/linux子目录下。

$ petalinux-package --image -c kernel --format uImage

1. 你还需要配置u-boot去支持启动uImage。

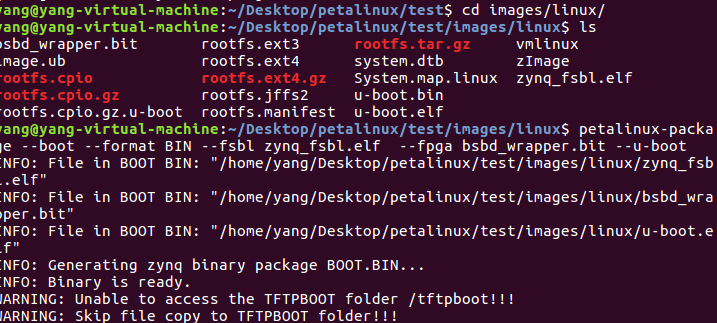


## 生成BOOT.BIN文件

输入以下命令来生成BOOT.BIN文件，这里主要是通过zynq的fsbl文件启动uboot，等uboot加载后，由于已经配置过文件系统和内核，所以uboot就会再加载他们。从这开始后就和其他linux启动一样了。

先用命令 cd images/linux/ 跳转至linux文件夹

再用命令 petalinux-package --boot --format BIN --fsbl zynq\_fsbl.elf --fpga bsbd\_wrapper.bit --u-boot 生成BOOT.BIN文件 其中bsbd\_wrapper.bit 名称为自己定义。2018.3中全部统一生成为system.bit。



**注： 一定要先转到images/linux目录下。**

## 系统启动：

1. 如果只从第一分区启动，不加载第二分区文件系统：
   1. 首先执行6、7、8、9步的操作
   2. 将BOOT.bin和imuge.ub文件拷进FAT32分区，即可启动。但是会掉电丢失。
   3. 毕竟是ramdisk压缩文件系统，没有存放在第二分区。
2. 从第二分区启动：
   1. 执行6、7、8、9步的操作。
   2. 将BOOT.bin和imuge.ub文件拷进FAT32分区
   3. 若想掉电不丢失，还需将rootfs.tar.gz 拷进EXT4分区并解压。（注意获取权限）
   4. 将SD 卡插入开发板，并且设置为SD卡启动，即可启动。
   5. 如果有APP可执行文件要运行。可以放进第一分区，然后挂载，复制到第二分区的某个文件夹再运行。直接拷进第二分区无法运行（权限问题）。

以上是Petalinux搭建最小系统的基本操作。但是还有很多步骤需要完善，下面会陆续完善，诸如内核驱动编译等等。

Icluding Prebuilt App（添加预编译的应用）：

Icluding Prebuilt Modules（添加预编译的模块）：

Building User App（预编译的应用）：

Test user app

Building User Modules（预编译的模块）：

Application Auto Run at Startup(开机自启动)：

Devicetree Configuration:

U-Boot Configuration:

Configuring Out-of-tree Build（petalinux从git上更新uboot,kernal）