

IDO-EVB3568-V1 Linux5.10开发手册

1、源码获取

1.1 下载源码

1.2 解压源码

2、开发环境搭建

3、SDK编译

3.1 选择屏幕配置

3.2 文件系统选择

3.3 开始编译

3.3.1 一键编译

3.3.2 部分编译

4、FAQ



IDO-EVB3568-V1
Linux5.10开发手册

文档修订历史

版本	PCBA版本	修订内容	修订	审核	日期
V1.0	V1D	创建文档	TWX	IDO	2024/11/27

1、源码获取

1.1 下载源码

链接: <https://pan.baidu.com/s/1Jg9DYMssLLFYjLpD8zo4Gw?pwd=1234>

提取码: 1234

1.2 解压源码

```
▼ Shell |
1  $ cat ido-evb3568-v1d-Linux5.10-sdk-V1.1-241220.tar.gz.* > ido-evb3568-v1d-
   -Linux5.10-sdk-V1.1-241220.tar.gz
2  $ tar -zxvf ido-evb3568-v1d-Linux5.10-sdk-V1.1-241220.tar.gz
```

解压得到ido-evb3568-v1d-Linux5.10-sdk目录即为SDK目录。

2、开发环境搭建

建议使用Ubuntu18.04及以上版本系统上开发, 并安装依赖软件包。

```
▼ Bash |
1  sudo apt-get install repo git ssh make gcc libssl-dev liblz4-tool expect
2  sudo apt-get install g++ patchelf chrpath gawk texinfo chrpath diffstat
3  sudo apt-get install binfmt-support qemu-user-static live-build bison flex
4  sudo apt-get install fakeroot cmake gcc-multilib g++-multilib unzip
5  sudo apt-get install device-tree-compiler python-pip libncurses5-dev python-
   -pyelftools
```

3、SDK编译

3.1 选择屏幕配置

▼ Shell

```
1 $ ./build.sh lunch
2 1. rockchip_rk3568_EVB3568V1_HDMI_EDP-1920x1080_defconfig
3 2. rockchip_rk3568_EVB3568V1_HDMI_LVDS-1024x600_defconfig
4 3. rockchip_rk3568_EVB3568V1_HDMI_MIPI1-1200x1920_Buildroot_defconfig
5 4. rockchip_rk3568_EVB3568V1_HDMI_MIPI1-1200x1920_Debian_defconfig
6 5. rockchip_rk3568_EVB3568V1_HDMI_MIPI1-1200x1920_Ubuntu_defconfig
7 6. rockchip_rk3568_EVB3568V1_HDMI_MIPI2LVDS-1920x1080_defconfig
8 7. rockchip_rk3568_EVB3568V1_HDMI_defconfig
```

每个defconfig配置对应的显示屏不同。

3.2 文件系统选择

通过环境变量RK_ROOTFS_SYSTEM进行文件系统的选择：

buildroot
debian
ubuntu20

例如选择debian文件系统，则将RK_ROOTFS_SYSTEM赋值debian：

▼ Shell

```
1 export RK_ROOTFS_SYSTEM=debian
```

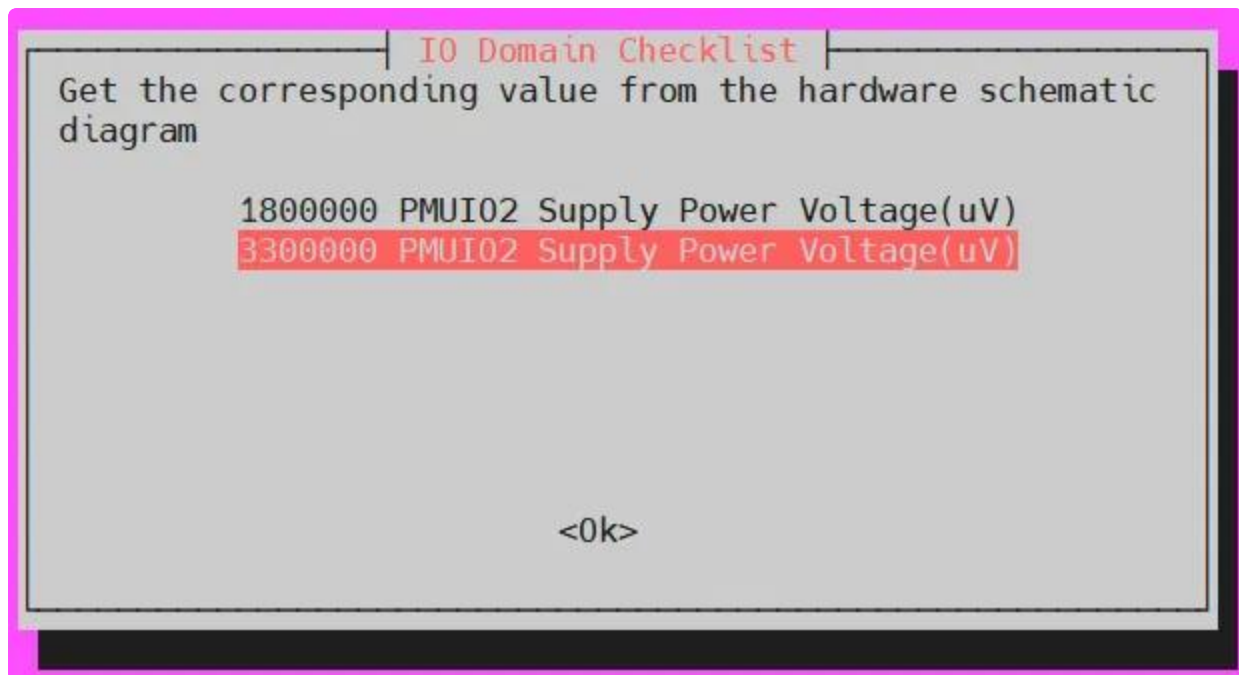
3.3 开始编译

3.3.1 一键编译

▼ Shell

```
1 $ ./build.sh
```

如果前面lunch切换过mk，则编译会弹出IO Domain Checklist弹窗：



注意vccio4 和 vccio6 选择1.8V, 其余为3.3v, 在提示弹窗内, 使用键盘上下键选择正确的值
等待编译完成后, 将生成镜像在rockdev目录下:

```
Shell |
1  $ ls rockdev/
2  boot.img MiniLoaderAll.bin misc.img oem.img parameter.txt recovery.im
   g  rootfs.ext4 rootfs.img uboot.img update.img userdata.img
```

其中update.img为完整固件包。

3.3.2 部分编译

SDK支持单独编译某个模块, 如loader、uboot和kernel等。

以单独编译kernel为例:

```
Shell |
1  $ ./build.sh kernel
```

编译完成后, 会生成rockdev/boot.img。

4、FAQ

编译Buildroot提示: g++:internal compiler error: Killed (program cclplus)

```
8/host/include -I/home/luffy/share/3568/rk356x_linux_sdk_211116/buildroot/output/rockchip_rk3568/host/include -I../gcc/../libdecnumber -I../gcc/../libdecnumber/dpd -I../libdecnumber -I../gcc/..
h.o -MT generic-match.o -MD -MF ./.deps/generic-match.o -g -c /home/luffy/share/3568/rk356x_linux_sdk_211116/buildroot/output/rockchip_rk3568/host/include
2022-01-05T10:38:16 g++: internal compiler error: Killed (program cc1plus)
2022-01-05T10:38:16 Please submit a full bug report,
2022-01-05T10:38:16 with preprocessed source if appropriate.
2022-01-05T10:38:16 See <file:///usr/share/doc/gcc-5/README.Bugs> for instructions.
2022-01-05T10:38:16 Makefile:1116: recipe for target 'gimple-match.o' failed
2022-01-05T10:38:16 make[3]: *** [gimple-match.o] Error 4
2022-01-05T10:38:16 make[3]: *** Waiting for unfinished jobs....
2022-01-05T10:38:23 rm gcc.pod
2022-01-05T10:38:23 Makefile:4300: recipe for target 'all-gcc' failed
2022-01-05T10:38:23 make[2]: *** [all-gcc] Error 2
2022-01-05T10:38:23 package/pkg-generic.mk:228: recipe for target '/home/luffy/share/3568/rk356x_linux_sdk_211116/buildroot/output/rockchip_rk3568/build/host-gcc-initial-9.3.0/.stamp_built' failed
2022-01-05T10:38:23 make[1]: *** [/home/luffy/share/3568/rk356x_linux_sdk_211116/buildroot/output/rockchip_rk3568/build/host-gcc-initial-9.3.0/.stamp_built] Error 2
2022-01-05T10:38:26 /home/luffy/share/3568/rk356x_linux_sdk_211116/buildroot/output/rockchip_rk3568/Makefile:16: recipe for target '_all' failed
2022-01-05T10:38:26 make: *** [_all] Error 2
Command exited with non-zero status 1
you take 26:56.07 to build buildroot
ERROR: Running build_buildroot failed!
ERROR: exit code 1 from line 556:
/usr/bin/time -f "you take %E to build buildroot" $COMMON_DIR/mk-buildroot.sh $BOARD_CONFIG
root@luffy-virtual-machine:~/share/3568/rk356x_linux_sdk_211116# dd if=/dev/
```

提示memory 不够, 可以通过增加swap空间来避免

- 查看Swap空间大小

▼ Shell

1	free -m					
2						
3		total	used	free	shared	buff/cache avail
4	Mem:	7868	1144	5495	352	1228
5	Swap:	8075	0	8075		

- 在原有的Swap基础上增加8G空间 (大小可自己定)

▼ Shell

1	sudo dd if=/dev/zero of=swapfile bs=1G count=8
2	
3	记录了8+0 的读入
4	记录了8+0 的写出
5	8589934592 bytes (8.6 GB, 8.0 GiB) copied, 45.0893 s, 191 MB/s

- 创建一个Swap文件

▼ Shell

1	sudo mkswap /swap/swapfile
2	
3	Setting up swapspace version 1, size = 8 GiB (8589930496 bytes)
4	无标签, UUID=179cba9a-944b-458a-9e1c-c924ad578662

- 尝试激活Swap文件

```

1 swapon /swap/swapfile
2
3 swapon: /swap/swapfile: 不安全的权限 0644, 建议使用 0600。
4 swapon: /swap/swapfile: swapon 失败: 不允许的操作

```

- 修改文件权限

```

1 chmod 0600 /swap/swapfile

```

- 再次查看Swap空间大小

```

1 free -m
2
3      total        used        free      shared  buff/cache   avail
4  Mem:    7868      1178      1174        371       5516
5  Swap:   16267         0      16267

```

- 重启计算机可能会退回8G，如果需要一直保持这个swap空间，可以把它写入/etc/fstab文件中

```

1 sudo vim /etc/fstab
2
3 #增加以下两行:
4
5 (add swap space on /swap/swapfile)
6 /swap/swapfile          /swap          swap          defaults
   0 0

```