AidoSDR固件制作

AidoSDR的固件制作方法已经在github上开源，可以在此链接找到 https://github.com/sdrdeepaido/aidosdr-fw 建议您关注github，github上的更新会优先与文档。

构建说明

固件采用 Xilinx Vivado 2023.2(v0.39). 您需要在 Linux PC 上安装正确的 Vivado 版本，然后按照以下说明生成固件：

安装构建要求

```
sudo apt-get install git build-essential fakeroot libncurses5-dev libssl-dev ccache
sudo apt-get install dfu-util u-boot-tools device-tree-compiler mtools
sudo apt-get install bc python cpio zip unzip rsync file wget
sudo apt-get install libtinfo5 device-tree-compiler bison flex u-boot-tools
sudo apt-get purge gcc-arm-linux-gnueabihf
sudo apt-get remove libfdt-de
```

获取源代码并设置 bash

1. 从 git 获取源代码 v0.39
   `git clone -b v0.39 --recursive https://github.com/sdrdeepaido/aidosdr-fw.git`
2. 工具链

由于 Vivado/Vitis 附带的 AMD/Xilinx GCC 工具链与 Buildroot 不兼容，本项目已切换到 Buildroot 外部工具链： Linaro GCC 7.3-2018.05 7.3.1

```
https://releases.linaro.org/components/toolchain/binaries/7.3-2018.05/arm-linux-gnueabihf/
```

3.设置环境变量和工具链

- v0.39

```
export CROSS_COMPILE=arm-linux-gnueabihf-
export PATH=$PATH:/Toolchain-PATH/gcc-linaro-7.3.1-2018.05-i686_arm-linux-gnueabihf/bin
export VIVADO_SETTINGS=/opt/Xilinx/Vivado/2023.2/settings64.sh
```

导出目标

对于 AidoSDR

```
export TARGET=AidoSDR
```

对于 aidosdr_A1

```
export TARGET=aidosdr_A1
```

进入hdl目录

对于 AidoSDR，切换到2023_R2分支

```
cd aidosdr-fw/plutosdr-fw/hdl
git checkout 2023_R2
cd ../../../
```

对于aidosdr_A1

```
保持默认即可
```

Patch

完成以上步骤后，开始Patch。

```
cd aidosdr-fw
```

对于 AidoSDR

```
sh patch.sh AidoSDR
```

对于 aidosdr_A1

`sh patch.sh aidosdr_A1`

如果补丁成功应用，您将看到以下信息。

```
AIdo_sdr/aidosdr-fw$ sh patch.sh AidoSDR
Patch check...
 ...
 ...

Patch...
...
...
patch finish
```

Build

然后你就可以制作固件了。

```
cd plutosdr-fw
sudo -E make
```

固件编译完成后，您将在编译文件夹中看到以下文件。这些文件用于闪存更新。

```
AIdo_sdr/aidosdr-fw/plutosdr-fw$ ls -AGhl build
总计 572M
-rw-r--r-- 1 root  15M  6月 11 17:46 AidoSDR.dfu
-rw-r--r-- 1 root  15M  6月 11 17:47 AidoSDR.frm
-rw-r--r-- 1 root   33  6月 11 17:47 AidoSDR.frm.md5
-rw-r--r-- 1 root  15M  6月 11 17:46 AidoSDR.itb
-rw-r--r-- 1 root   69  6月 11 17:47 boot.bif
-rw-r--r-- 1 root 510K  6月 11 17:47 boot.bin
-rw-r--r-- 1 root 510K  6月 11 17:47 boot.dfu
-rw-r--r-- 1 root 639K  6月 11 17:47 boot.frm
-rw-r--r-- 1 root 480M  6月 11 17:47 legal-info-v0.39-dirty.tar.gz
-rw-r--r-- 1 root 641K  6月 11 17:36 LICENSE.html
-rw-r--r-- 1 root  26M  6月 11 17:47 plutosdr-fw-v0.39-dirty.zip
-rw-r--r-- 1 root 720K  6月 11 17:47 plutosdr-jtag-bootstrap-v0.39-dirty.zip
-rw-r--r-- 1 root 524K  6月 11 17:45 ps7_init.c
-rw-r--r-- 1 root 525K  6月 11 17:45 ps7_init_gpl.c
-rw-r--r-- 1 root 4.2K  6月 11 17:45 ps7_init_gpl.h
-rw-r--r-- 1 root 3.6K  6月 11 17:45 ps7_init.h
-rw-r--r-- 1 root 2.8M  6月 11 17:45 ps7_init.html
-rw-r--r-- 1 root  35K  6月 11 17:45 ps7_init.tcl
-rw-r--r-- 1 root 7.9M  6月 11 17:37 rootfs.cpio.gz
drwxr-xr-x 6 root 4.0K  6月 11 17:46 sdk
-rw-r--r-- 1 root 2.4M  6月 11 17:45 system_top.bit
-rw-r--r-- 1 root 858K  6月 11 17:45 system_top.xsa
-rwxr-xr-x 1 root 473K  6月 11 17:47 u-boot.elf
-rw-r----- 1 root 128K  6月 11 17:47 uboot-env.bin
-rw-r----- 1 root 129K  6月 11 17:47 uboot-env.dfu
-rw-r--r-- 1 root 7.7K  6月 11 17:47 uboot-env.txt
-rwxr-xr-x 1 root 4.4M  6月 11 17:35 zImage
-rw-r--r-- 1 root  23K  6月 11 17:37 zynq-AidoSDR.dtb
```

制作 SD 卡启动映像

固件构建完成后，您可以为设备构建 SD 卡启动映像。只需输入以下命令

```
make sdimg
```

您将在 build_sdimg 文件夹中看到 SD 启动映像。您只需将该文件夹中的所有文件复制到 SD 卡中，将 SD 卡插入 AidoSDR，并将跳线设置为 SD 卡启动模式即可。

通过DFU更新Flash

AidoSDR 和 aidosdr_A1 支持 DFU 模式，您可以通过 DFU 模式更新 Flash。将跳线设置为 Flash Boot 模式。设备通电后，按下 DFU 按钮，您将看到设备中的两个 LED 指示灯亮起绿色，此时即可更新 Flash。您需要先进入 build 文件夹，然后将 Micro USB 插入 OTG 端口。然后，运行以下命令。

```
sudo dfu-util -a firmware.dfu -D ./AidoSDR.dfu
sudo dfu-util -a boot.dfu -D ./boot.dfu
sudo dfu-util -a uboot-env.dfu -D ./uboot-env.dfu
```

现在您可以重新启动设备。

支持 2r2t 模式

AidoSDR支持2T2R模式，如果要使用 2r2t 模式，需要进入系统并运行以下命令将模式配置写入 nor flash。**但是 SD 卡启动模式和 QSPI 启动模式略有不同**

QSPI 启动模式

```
fw_setenv attr_name compatible
fw_setenv attr_val ad9361
fw_setenv compatible ad9361
fw_setenv mode 2r2t
reboot
```

重启后，使用命令检测 flash 中的变量是否已写入。如果写入成功，则可以使用 2r2t 模式。

当然，还有另一种方法可以配置 2r2t 模式，并在 uboot 下使用命令写入 flash，例如

```
setenv attr_name compatible
setenv attr_val ad9361
setenv compatible ad9361
setenv mode 2r2t
saveenv
reset
```

SD启动模式

您需要修改uEnv.txt文件中的某些参数

1.您需要修改 **adi_loadvals** 的值，如下所示：

修复之前：

```
adi_loadvals=fdt addr ${fit_load_address}......
```

修复之后： `txt adi_loadvals=fdt addr ${devicetree_load_address}......`

2.您需要修改**mode**的值，如下所示：

修复之前：

```
maxcpus=1
mode=1r1t
```

修复之后：

```
maxcpus=1
mode=2r2t
```

3.您需要修改**sdboot（添加运行 adi_loadvals 和 #{fit_config}）**的值，如下所示：

修复之前：

```
sdboot=if mmcinfo; then run uenvboot; echo Copying Linux from SD to RAM... && load mmc 0 $
{fit_load_address} ${kernel_image} && load mmc 0 ${devicetree_load_address} ${devicetree_image}
&& load mmc 0 ${ramdisk_load_address} ${ramdisk_image} bootm ${fit_load_address} $
{ramdisk_load_address} ${devicetree_load_address}; fi
```

修复之后：

```
sdboot=if mmcinfo; then run uenvboot; echo Copying Linux from SD to RAM... && load mmc 0 $
{fit_load_address} ${kernel_image} && load mmc 0 ${devicetree_load_address} ${devicetree_image}
&& load mmc 0 ${ramdisk_load_address} ${ramdisk_image} && run adi_loadvals;bootm $
{fit_load_address} ${ramdisk_load_address} ${devicetree_load_address}#{fit_config}; fi
```

4.您需要在最后一行**添加以下参数（attr_name attr_val 兼容）**：

修复之前：

```
usbboot=if usb start; then run uenvboot; echo Copying Linux from USB to RAM... && load usb 0 $
{fit_load_address} ${kernel_image} && load usb 0 ${devicetree_load_address} ${devicetree_image}
&& load usb 0 ${ramdisk_load_address} ${ramdisk_image} && bootm ${fit_load_address} $
{ramdisk_load_address} ${devicetree_load_address}; fi
```

修复之后：

```
usbboot=if usb start; then run uenvboot; echo Copying Linux from USB to RAM... && load usb 0 $
{fit_load_address} ${kernel_image} && load usb 0 ${devicetree_load_address} ${devicetree_image}
&& load usb 0 ${ramdisk_load_address} ${ramdisk_image} && bootm ${fit_load_address} $
{ramdisk_load_address} ${devicetree_load_address}; fi
attr_name=compatible
attr_val=ad9361
compatible=ad9361
```

然后你就可以享受2r2t模式了。