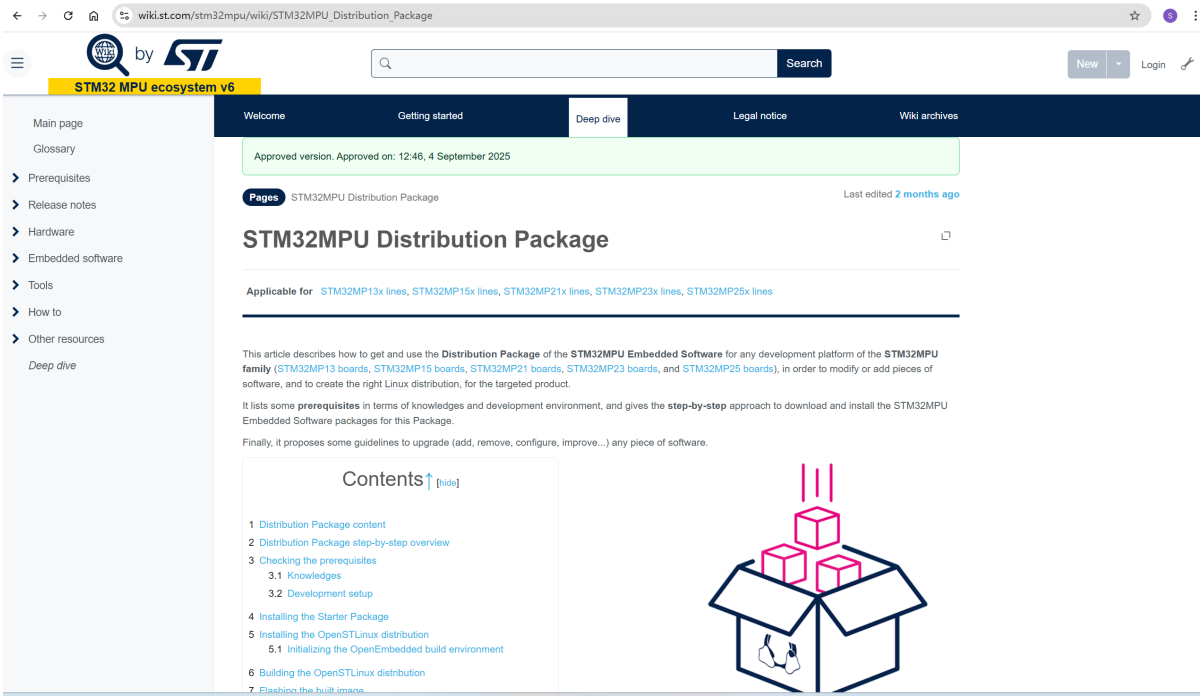
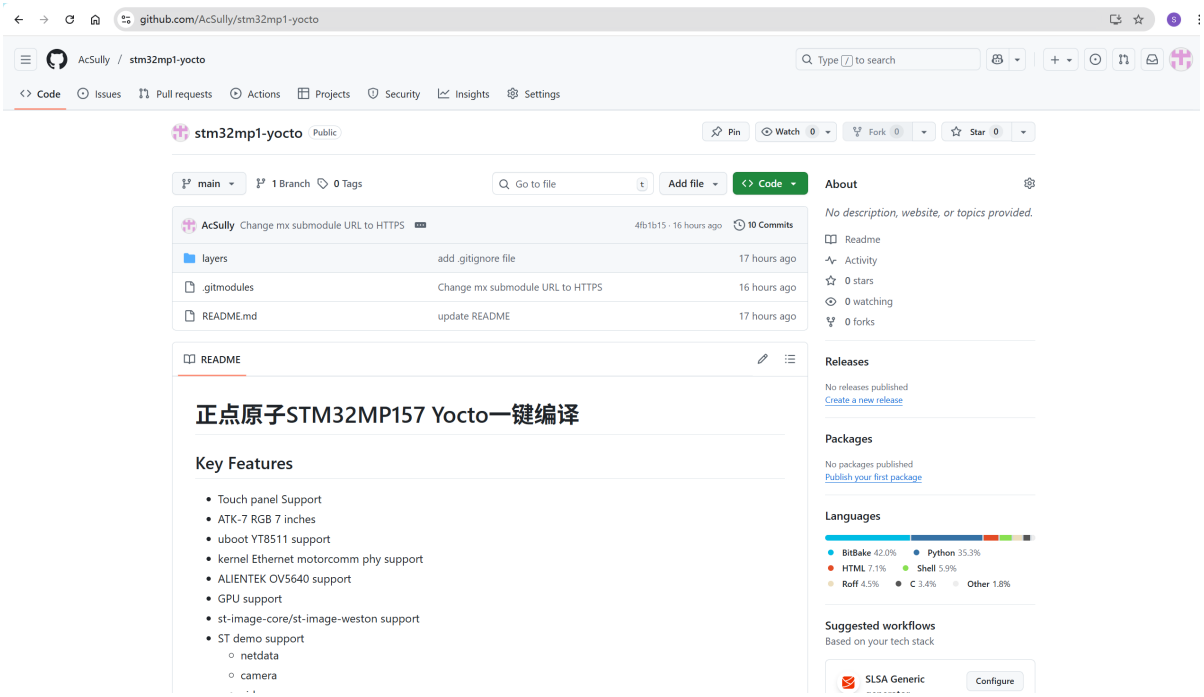
Yocto构建教程

官方wiki文档，强烈推荐！！！里面的内容几乎能解决任何问题！



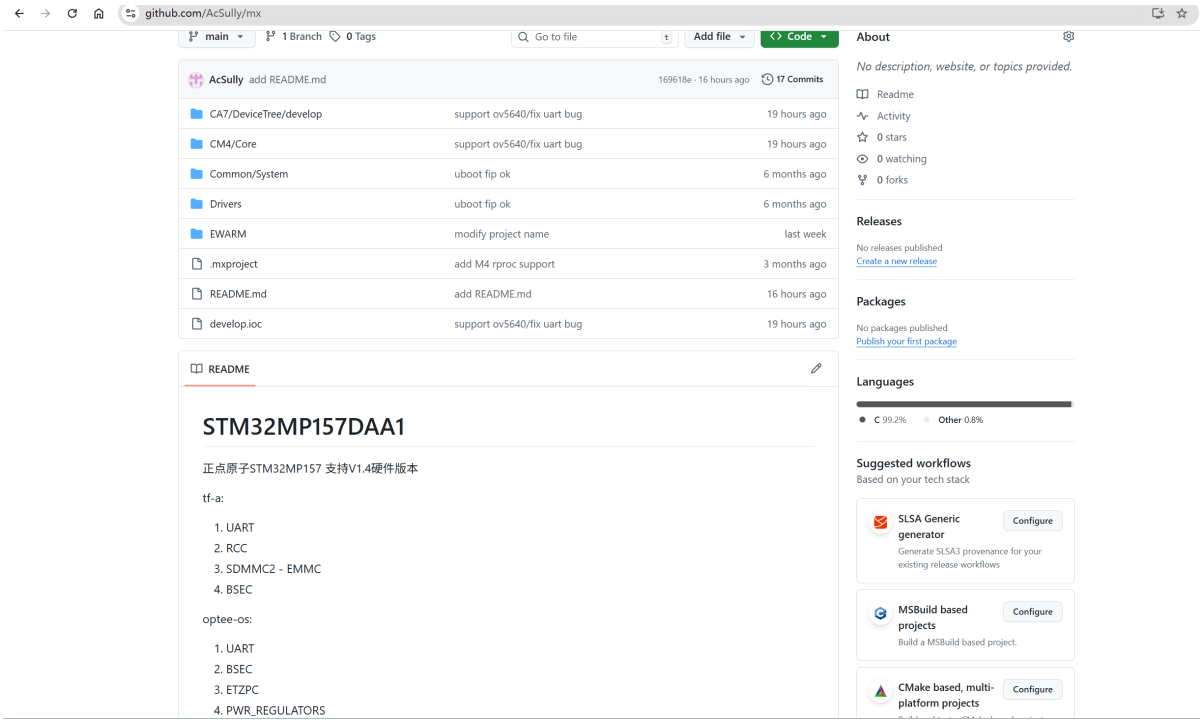
将官方repo仓库更新成github仓库(官方仓库需要访问google)

- 新增meta-st-develop
  - 增加u-boot补丁(meta-st-develop/meta-st-stm32mp-addons/recipes-extended/uboot/u-boot-stm32mp/0001-support-panel-eth-phy.patch)
  - 增加kernel补丁和外部kernel配置(meta-st-develop/recipes-kernel/linux/ patch和external kernel config)
  - 增加mx仓库，完全兼容STM32CUBEMX工具



mx项目目录

- 设备树
  - tf-a
  - optee
  - u-boot
  - kernel
- TODO
  - audio
  - beep
  - adc
  - RS232/RS485
  - FD CAN
  - 6 AIXS SENSOR
  - ALS&PS SENSOR
  - TF Card
  - HDMI
  - USB OTG - 待测试
  - SDIO WIFI&BT
  - TEMP&HUMI SENSOR



下载github代码

```
git clone --recurse-submodules https://github.com/AcSully/stm32mp1-yocto.git
cd stm32mp1-yocto
git submodule update --remote
```

下载Yocto下载包(里面包含需要联网下载的源码)

链接: https://pan.baidu.com/s/15_UaWHm3nBICPwpJyYhACw 提取码: 48h4

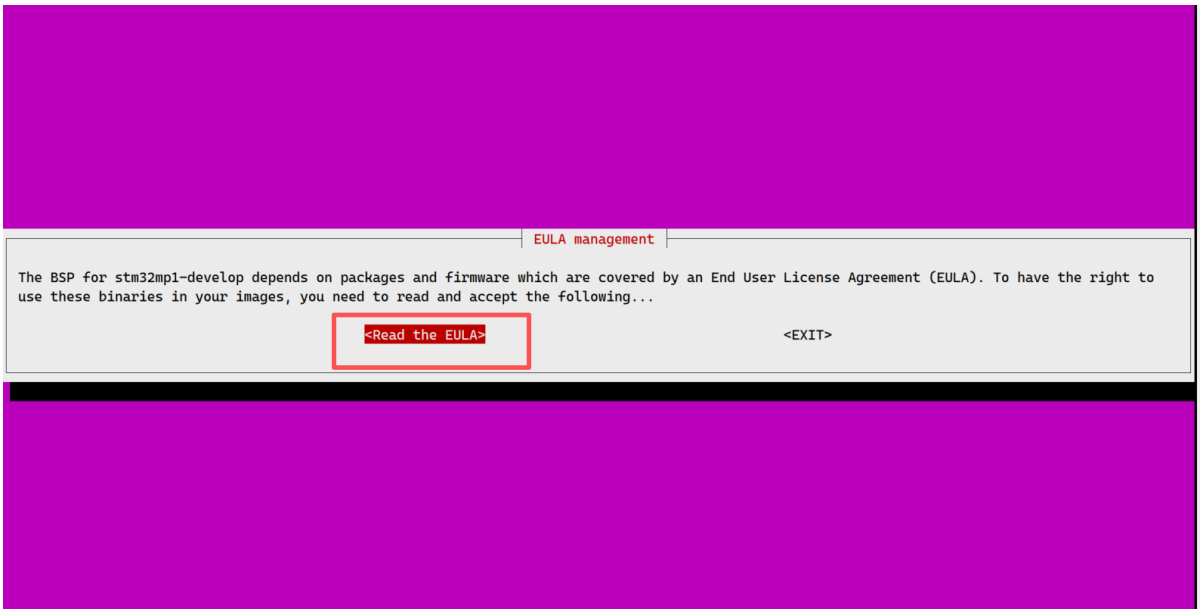
下载完成后解压，记住解压后的目录，我的放在/home/niuke/Downloads/STM32MP1，需要实际解压目录

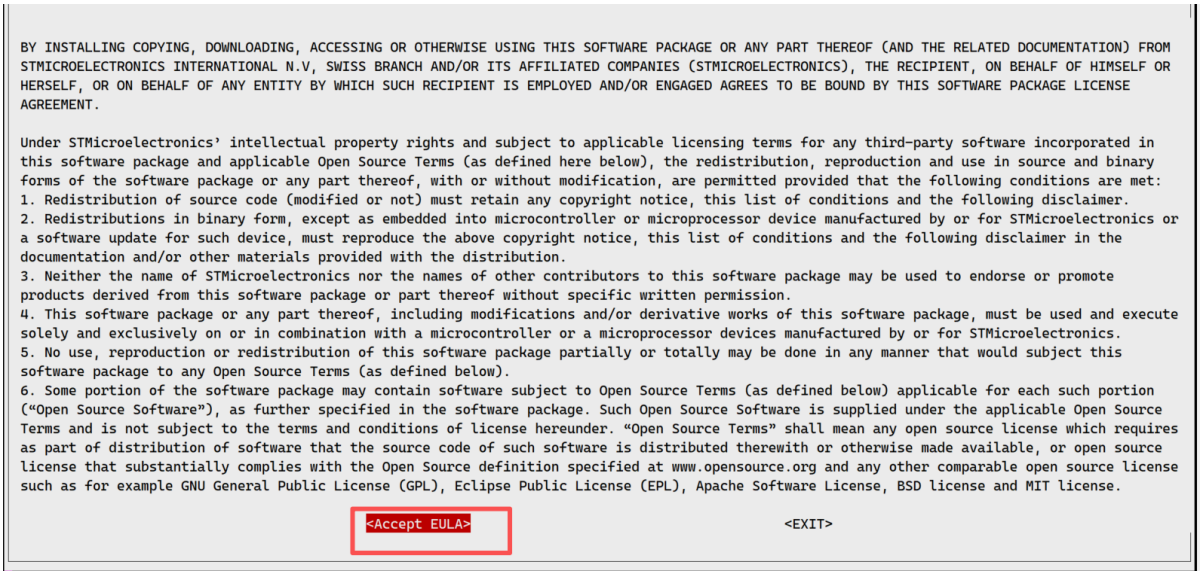
```
7z x STM32MP1_Yocto.zip.001

/home/niuke/Downloads/temp/STM32MP1_Yocto
```

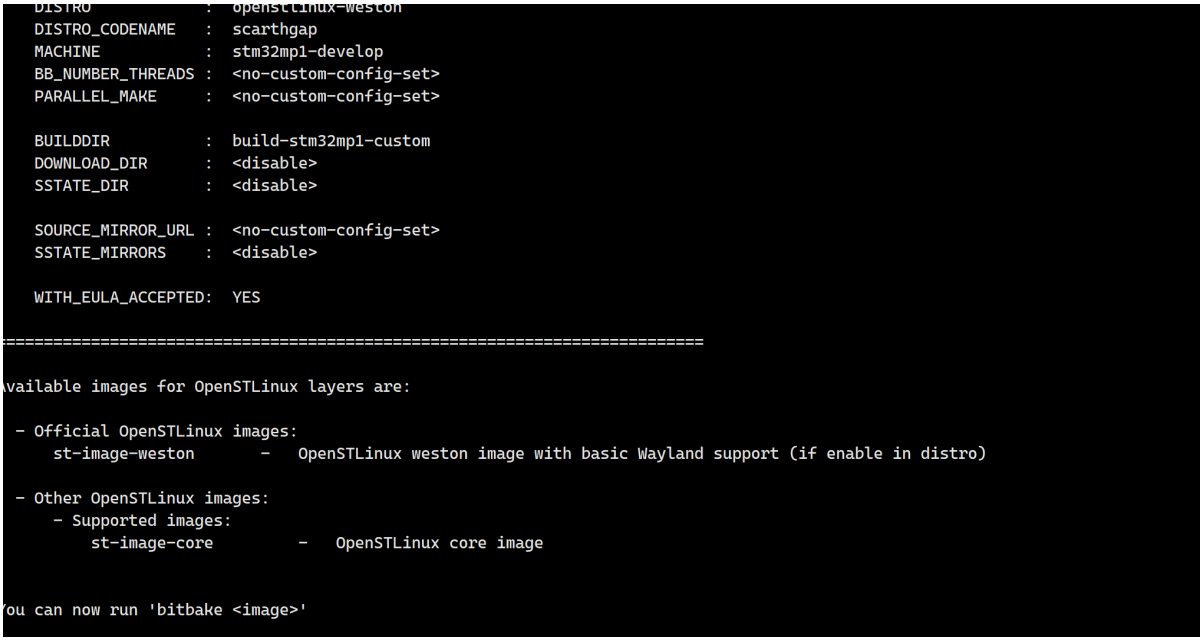
```
Everything is Ok

Folders: 2375
Files: 5622
Size:       9006655167
Compressed: 8865681895
niuke@zzhisome:~/Downloads/git$ ls
STM32MP1_Yocto          STM32MP1_Yocto.zip.002  STM32MP1_Yocto.zip.005  STM32MP1_Yocto.zip.008
STM32MP1_Yocto.tgz      STM32MP1_Yocto.zip.003  STM32MP1_Yocto.zip.006  STM32MP1_Yocto.zip.009
STM32MP1_Yocto.zip.001  STM32MP1_Yocto.zip.004  STM32MP1_Yocto.zip.007
[redacted]~/Downloads/git$ 7z x STM32MP1_Yocto.zip.001
```

声明环境变量

```
DISTRO=openstlinux-weston MACHINE=stm32mp1-develop source layers/meta-st-
develop/scripts/envsetup.sh build-stm32mp1-custom

两个都选择接受
```







vi conf/local.conf
找到DL_DIR，修改路径为刚才解压的路径(STM32MP1_Yocto.zip.001)，保存

```
# connection is slow. These are all stored in DL_DIR. When wiping and rebuilding yo
# can preserve this directory to speed up this part of subsequent builds. This dire
# is safe to share between multiple builds on the same machine too.
#
# The default is a downloads directory under TOPDIR which is the build directory.
#
#DL_DIR ?= "${TOPDIR}/downloads"
DL_DIR ?= "/home/niuke/Downloads/STM32MP1"

#
# Where to place shared-state files
#
# BitBake has the capability to accelerate builds based on previously built output.
# This is done using "shared state" files which can be thought of as cache objects
# and this option determines where those files are placed.
#
# You can wipe out TMPDIR leaving this directory intact and the build would regener
```

编译镜像

```
bitbake st-image-weston
```

```
niuke@zzhisome:~/develop/github/stm32mp1-yocto/build-stm32mp1-custom$ bitbake st-image-weston
NOTE: Started PRServer with DBfile: /home/niuke/develop/github/stm32mp1-yocto/build-stm32mp1-custom/cache/prserv.sqlite3, Address: 127.0.0.1:37411,
PID: 1213260
Loading cache: 100% |                                                                                                  | ETA:  --:--:--
Loaded 0 entries from dependency cache.
Parsing recipes:   1% |##                                                                                               | ETA:  0:02:03
```

```
/usr/lib/pipewire-0.3/libpipewire-module-netjack2-manager.so uses 32-bit api 'setsockopt'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-netjack2-driver.so uses 32-bit api 'setsockopt'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-pipe-tunnel.so uses 32-bit api 'fstat64'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-protocol-pulse.so uses 32-bit api 'fstat64'
/usr/lib/pipewire-0.3/libpipewire-module-protocol-pulse.so uses 32-bit api 'gettimeofday'
/usr/lib/pipewire-0.3/libpipewire-module-protocol-pulse.so uses 32-bit api 'getsockopt'
/usr/lib/pipewire-0.3/libpipewire-module-protocol-pulse.so uses 32-bit api 'stat64'
/usr/lib/pipewire-0.3/libpipewire-module-protocol-pulse.so uses 32-bit api 'setsockopt'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-vban-send.so uses 32-bit api 'setsockopt'
/usr/lib/pipewire-0.3/libpipewire-module-vban-send.so uses 32-bit api 'sendmsg'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-rtp-sink.so uses 32-bit api 'setsockopt'
/usr/lib/pipewire-0.3/libpipewire-module-rtp-sink.so uses 32-bit api 'sendmsg'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-access.so uses 32-bit api 'fstat64'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-client-node.so uses 32-bit api 'clock_gettime'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/pipewire-0.3/libpipewire-module-protocol-simple.so uses 32-bit api 'setsockopt'
Suppress with INSANE_SKIP = "32bit-time"
/usr/lib/libpipewire-0.3.so.0.1203.0 uses 32-bit api 'fcntl64'
/usr/lib/libpipewire-0.3.so.0.1203.0 uses 32-bit api 'clock_gettime'
/usr/lib/libpipewire-0.3.so.0.1203.0 uses 32-bit api 'prctl'
/usr/lib/libpipewire-0.3.so.0.1203.0 uses 32-bit api 'fstat64'
/usr/lib/libpipewire-0.3.so.0.1203.0 uses 32-bit api 'stat64'
/usr/lib/libpipewire-0.3.so.0.1203.0 uses 32-bit api 'pthread_cond_timedwait'
Suppress with INSANE_SKIP = "32bit-time"
/usr/bin/pw-cat uses 32-bit api 'fstat64'
Suppress with INSANE_SKIP = "32bit-time"
/usr/bin/pw-mididump uses 32-bit api 'fstat64'
Suppress with INSANE_SKIP = "32bit-time"
/usr/bin/spa-json-dump uses 32-bit api 'fstat64'
Suppress with INSANE_SKIP = "32bit-time"
/usr/bin/pw-dot uses 32-bit api 'fstat64'
Suppress with INSANE_SKIP = "32bit-time" [32bit-time]
WARNING: netdata-1.44.3-r0 do_package_qa: QA Issue: File /usr/sbin/netdata in package netdata contains reference to TMPDIR [buildpaths]
NOTE: Tasks Summary: Attempted 10846 tasks of which 0 didn't need to be rerun and all succeeded.
NOTE: Writing buildhistory
NOTE: Writing buildhistory took: 45 seconds

Summary: There were 5 WARNING messages.
```

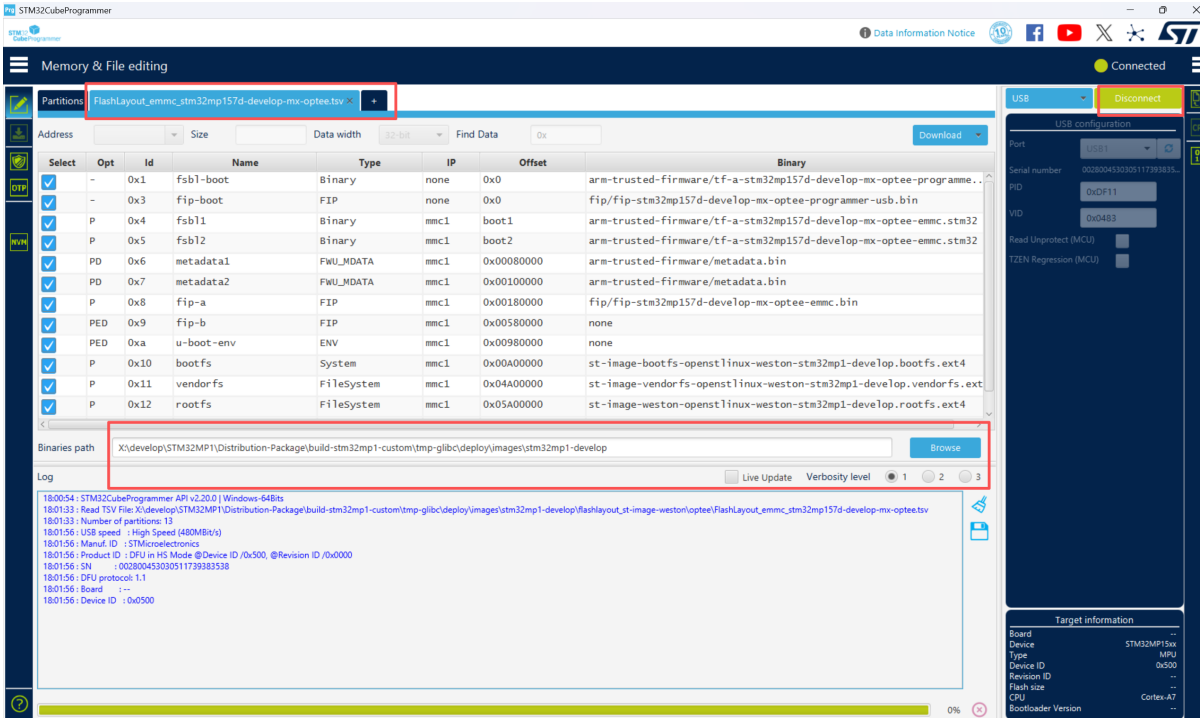
烧录镜像到开发板

镜像位置在build-stm32mp1-custom\tmp-glibc\deploy\images\stm32mp1-develop

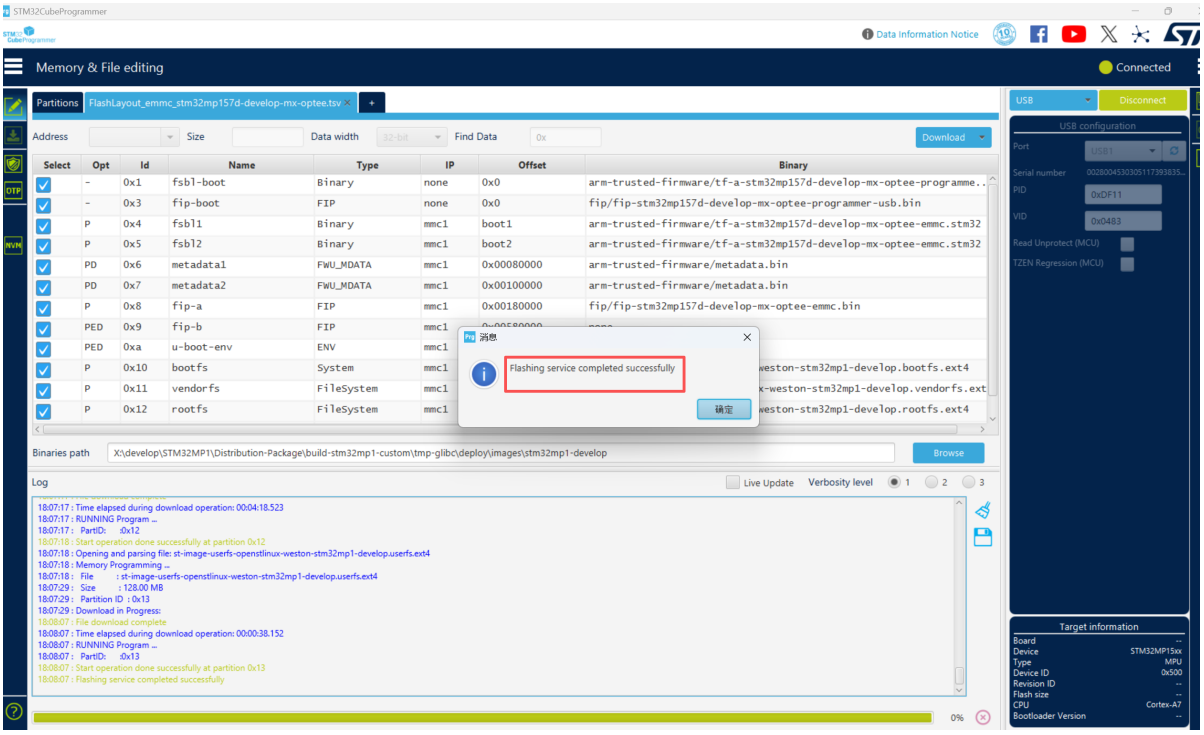
打开烧录软件

打开build-stm32mp1-custom\tmp-glibc\deploy\images\stm32mp1-develop\flashlayout_st-image-weston\optee\FlashLayout_emmc_stm32mp157d-develop-mx-optee.tsv

将拨码开关调整000到烧录模式，烧录器选择USB，connect，Download

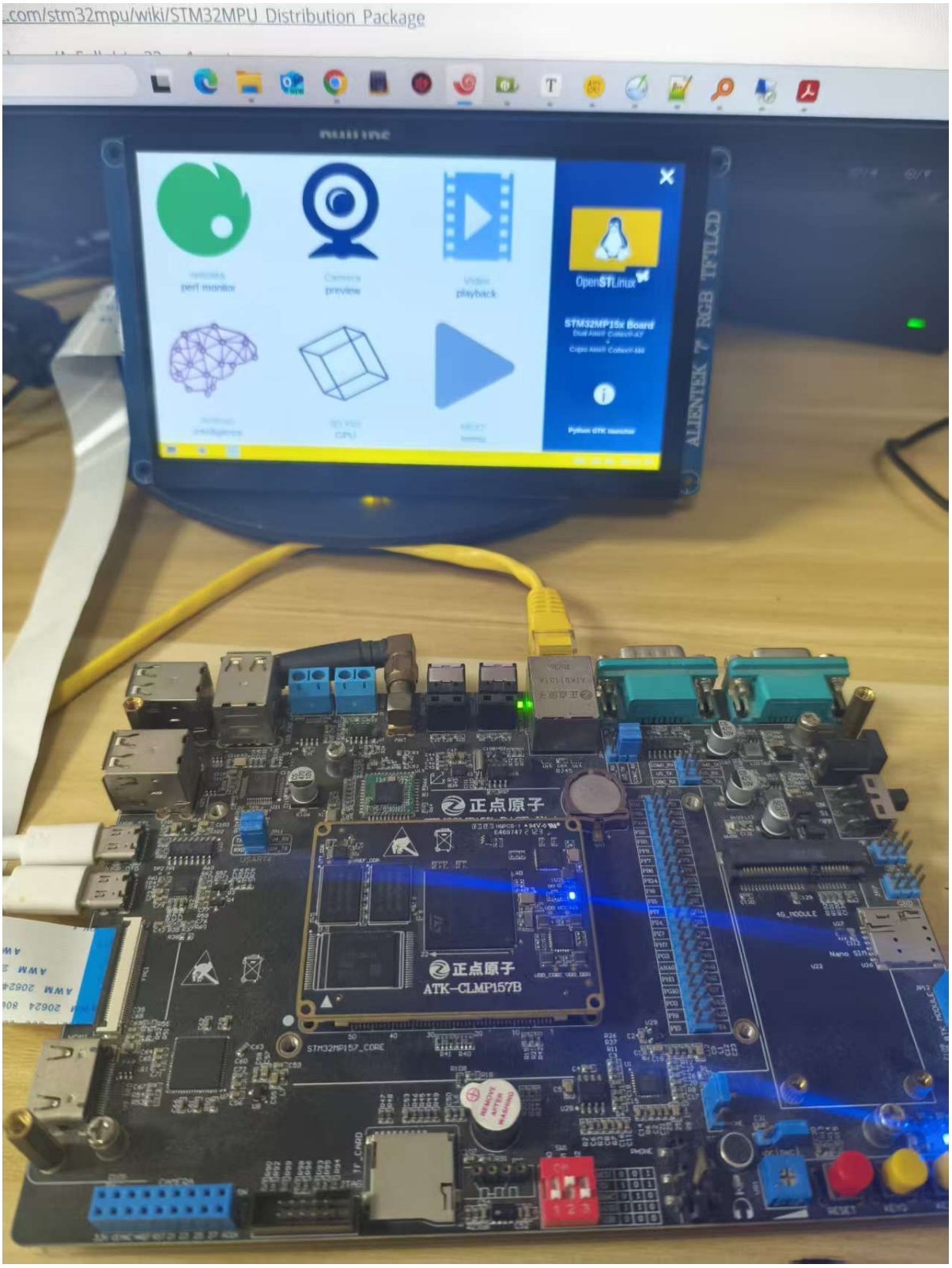


出现"Flashing service completed successfully"，则烧录成功



开发板验证

将启动模式修改成EMMC启动，按RESET键，完美！！！



参考地址

https://wiki.st.com/stm32mpu/wiki/STM32MPU_Distribution_Package

https://github.com/AcSully/stm32mp1-yocto

https://github.com/AcSully/mx