RT-Thread getting started guide for Bouffalo Lab RISC-V AIoT MCU series

Original post: https://zhuanlan.zhihu.com/p/627481115

The `bouffalo_lab` BSP under `rt-thread` is

The BSP for Bouffalo Lab's AIoT chips is located in `rt-thread/bsp/bouffalo_lab` (https://github.com/RT-Thread/rt-thread/tree/master/bsp/bouffalo_lab). It currently has 4 subdirectories including:

Chip model	Core
bl60x (BL602/BL604)	SiFive E24
bl70x (BL702/BL704/BL706)	SiFive E24
bl61x (BL616/BL618)	T-Head E907
bl808	T-Head E902 (lp) + E907 (m0) + C906 (d0)

The latest chips such as BL616/BL618 that support WiFi 6, and the three-core heterogeneous multimedia chip BL808 all use T-Head RISC-V cores.

Chip-related information can be downloaded at https://github.com/bouffalolab/bl_docs, the information is very detailed, with all register-level instructions.

1. rt-thread driver adaptation

The current BSP uses the latest LHAL driver library of `bouffalo_lab`, which is the same as https://github.com/bouffalolab/bouffalo_sdk.

LHAL is a driver library designed by Bouffalo Lab to unify the general peripheral interface. The code is refined and supports all series of Bouffalo Lab chips.

With the support of many small partners, the basic peripheral driver adaptation has been completed, including UART/GPIO/I2C/SPI/PWM/RTC/ADC/WDT/HWTIMER/FLASH, and the RT-Thread driver is also a set of codes to support all the above chips. Other drivers are also being adapted. Interested partners can also participate in submitting PR together, or contact me directly (WeChat: flyingcys).

- UART: supported (default baud rate 2000000)
- GPIO: supported
- I2C: supported
- SPI: supported (supports DMA)
- PWM: supported
- ADC: supported
- RTC: supported
- WDT: supported
- HWTIMER: supported

- FLASH: supported

Friends interested in the Open Source Promotion Plan summer program can also pay attention to Bouffalo Lab BSP, one of the high-level projects of the Summer of Open Source rt-thread community, "RT-thread uses BL602/BL702 chips to support WiFi, BLE, and Thread functions". To complete "BL60X/BL70X's IoT communication capabilities, BL602's WiFi and BLE capabilities, and BL702's BLE and Thread communication capabilities", click this link: https://summer-ospp.ac.cn/org/prodetail/238bc0129?list=org&navpage=org

2. Compile

2.1. Cross-compiler download

Please download the toolchain matching your chip and platform:

BL60x/BL70x

- Bouffalo Lab's developer zone:

  https://dev.bouffalolab.com/download

- or Bouffalo Lab's Gitee:

  https://gitee.com/bouffalolab/toolchain_gcc_sifive_linux

  https://gitee.com/bouffalolab/toolchain_gcc_sifive_windows

BL61x/BL808

- T-Head official website:

  https://occ.t-head.cn/community/download?id=4073475960903634944

- or Bouffalo Lab's GitHub:

  https://github.com/bouffalolab/toolchain_gcc_t-head_linux

  https://github.com/bouffalolab/toolchain_gcc_t-head_windows

2.2. Settings

Under Windows, please use [env tool][1], use the command `tar -xvf Xuantie-900-gcc-elf-newlib-mingw-V2.6.1-20220906.tar.gz` to decompress the cross-compiler. Direct decompression using Windows' decompression tool may cause compilation errors.

Add the local path of the RISC-V toolchain to `EXEC_PATH` in `rtconfig.py` or specify the path through the `RTT_EXEC_PATH` environment variable

Windows:

```
set RTT_EXEC_PATH=C:\Users\xxxx\Downloads\Xuantie-900-gcc-elf-newlib-x86_64-V2.6.1\bin
```

Linux:

```
export RTT_EXEC_PATH=/opt/Xuantie-900-gcc-elf-newlib-x86_64-V2.6.1/bin
```

2.3. Compiling

It is recommended to use [env tool][1] under Windows. Execute:

```
cd bsp/bouffalo_lab/bl61x
menuconfig
pkgs --update
```

If you are on the Linux platform, you can first execute:

```
scons --menuconfig
```

It will automatically download env related scripts to the `~/.env` directory. Then execute:

```
source ~/.env/env.sh
cd bsp/bouffalo_lab/bl61x
pkgs --update
```

After updating the package, execute `scons -j10` or `scons -j10 --verbose` to compile the board support package. Or use the `scons --exec-path="<GCC toolchain path>"` command to compile directly while specifying the toolchain location.

If the compilation is correct, `rtthread.elf` and `rtthread.bin` files will be generated.

After compiling, it will automatically call `libraries/bl_mcu_sdk/tools/bflb_tools/bflb_fw_post_proc` to package `rtthread.bin` for subsequent burning of `bouffalo_flash_cube` tool.

The script will automatically use the `curl` command line to download `bflb_fw_post_proc`. If the automatic download fails, you can manually download the corresponding operating system file from the URL below and save it to `libraries/bl_mcu_sdk/tools/bflb_tools/bflb_fw_post_proc`:

> Linux: https://github.com/bouffalolab/bouffalo_sdk/raw/master/tools/bflb_tools/bflb_fw_post_proc/bflb_fw_post_proc-ubuntu
>
> Windows: https://github.com/bouffalolab/bouffalo_sdk/raw/master/tools/bflb_tools/bflb_fw_post_proc/bflb_fw_post_proc.exe
>
> MacOS: https://github.com/bouffalolab/bouffalo_sdk/raw/master/tools/bflb_tools/bflb_fw_post_proc/bflb_fw_post_proc-macos

3. Download and burn

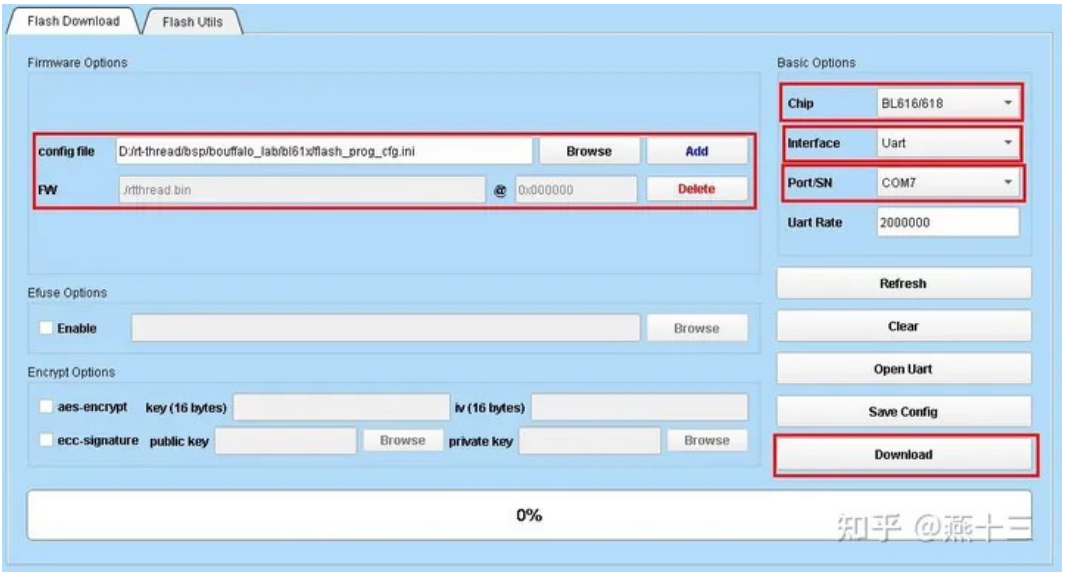
3.1. Burning tool download

The current BSP must be burned using the `bouffalo_flash_cube` tool, and it cannot run normally with other tasks.

> Linux: https://github.com/bouffalolab/bouffalo_sdk/raw/master/tools/bflb_tools/bouffalo_flash_cube/BLFlashCube-ubuntu
>
> Windows: https://github.com/bouffalolab/bouffalo_sdk/raw/master/tools/bflb_tools/bouffalo_flash_cube/BLFlashCube.exe
>
> MacOS: https://github.com/bouffalolab/bouffalo_sdk/raw/master/tools/bflb_tools/bouffalo_flash_cube/BLFlashCube-macos

3.2. GUI download

1. Connect the serial port and select the corresponding serial port number on the tool
2. Open the `flash_prog_cfg.ini` file under the corresponding chip folder
3. Press and hold the boot button on the development board and then power on again to enter the download state
4. Click "Download" to start downloading



3.3. Command line download

The command line download can use the `bouffalo_flash_cube.sh` script in the `bsp/bouffalo_lab` directory, enter `./bouffalo_flash_cube.sh bl616 /dev/ttyUSB1`, the script will automatically use the `curl` command line to download `bouffalo_flash_cube`.

Arguments:

> `bl616`: chip name
>
> `/dev/ttyUSB1`: download serial port number, `/dev/ttyUSBx` or `/dev/ttyACMx` under Linux, `COMx` under Windows

If the automatic download fails, you can manually download the corresponding operating system files and save them to the `libraries/bl_mcu_sdk/tools/bflb_tools/bouffalo_flash_cube` directory.

4. Execution

If the compilation and programming are correct, after resetting the device, you will see the RT-Thread startup logo information on the serial port:

5. List of supported development boards

BL602: BL602-IoT-3S/BL-HWC-G1

BL702: Maix Zero Sense

BL616/BL618: M0S Dock/M0P Dock

BL808: M1s Dock

6. Follow-up promotion plan

Include:

- WiFi driver adaptation
- SD card driver adaptation
- USB driver adaptation
- I2S driver adaptation
- display driver adaptation
- BLE driver adaptation
- Heterogeneous inter-core communication

Wait;

I hope that more friends will participate together, learn RISC-V together, and learn IoT together;

Anyone who is interested in rt-thread, RISC-V, rt-smart, and Bouffalo Lab BSP can add me on WeChat (flyingcys).