Support of the Nezha Allwinner D1 in meta-riscv

Prepared by Cezary Sobczak
Some facts:

- Junior Embedded System Developer at 3mdeb - 2 years of experience.
- Student in the last year of master's at the Gdańsk University of Technology with specialties in Embedded Systems and Autonomous Vehicles.
- My main interests are automotive, IoT, embedded systems, and microcontrollers.
- Project background
- What is a meta-riscv?
- Nezha board introduction
- Machine configuration
- Boot flow
- SD Card storage layout
- Preparing recipe for boot0
- Patches for OpenSBI
- U-Boot recipe adaptation
- Linux recipe adaptation
- Build minimal image
- Demo
- Known issues & needs
Willingness to learn about RISC-V architecture as a part of master's thesis.
At first, the plan was to create a simple OS for the BeagleV board.
Market research for other SBCs or processors based at RISC-V ISA.
SBC Nezha appears on the market with basic support of Linux base systems.
Nezha board didn't have support for Yocto Project.
meta-riscv is a OpenEmbedded / a Yocto layer for RISC-V-based boards and it contains a BSP for it. Here you can find machine configurations, recipes for specific firmware/software of the board, and examples of how to build the basic Yocto image for these machines eg. using kas.
Nezha board is a development board that is designed by an AWOL. This project uses a D1 SoC from Allwinner which is used for the first time by the general public. Probably this board is the first massive produced and available SBC based at RISC-V architecture taking in mind a fact that BeagleV™ pilot program with version beta of the board was canceled in August 2021.
## Specification of Allwinner D1 SoC

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameter description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XuanTie C906</td>
<td>Single core 1.0GHz 64-bit RISC-V processor</td>
</tr>
<tr>
<td>HiFi4 DSP</td>
<td>Cadence® Tensilica® HiFi 4</td>
</tr>
<tr>
<td>G2D 2D</td>
<td>graphics accelerators</td>
</tr>
<tr>
<td>DDR3 RAM</td>
<td>three variants - 512MB, 1GB or 2GB</td>
</tr>
<tr>
<td>SPI NAND</td>
<td>256MB of flash memory</td>
</tr>
</tbody>
</table>
## Peripherals

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameter description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Storage</strong></td>
<td>Onboard 256MB spi-nand, support USB external U disk and SD card to expand storage</td>
</tr>
<tr>
<td><strong>Network</strong></td>
<td>Support Gigabit Ethernet, support 2.4GHz WiFi and Bluetooth, onboard antenna</td>
</tr>
<tr>
<td><strong>Display</strong></td>
<td>Support MIPI-DSI+TP screen interface, support HDMI output, support SPI screen</td>
</tr>
<tr>
<td><strong>Audio</strong></td>
<td>Microphone daughter board interface * 1, 3.5mm headphone jack * 1 (CTIA)</td>
</tr>
<tr>
<td><strong>USB</strong></td>
<td>power type-c, OTG type-c, HOST type-a</td>
</tr>
<tr>
<td><strong>GPIO HEADER</strong></td>
<td>Raspberry Pi like header</td>
</tr>
<tr>
<td><strong>DEBUG</strong></td>
<td>Dedicated header for serial communication (UART)</td>
</tr>
</tbody>
</table>
Machine configuration for boards can be found in `meta-riscv` at path `conf/machine`. Here you can check and change configuration for particular machine. Key features of `nezha-allwinner-d1.conf`:

- install kernel `fitImage` format
- install `boot0`, `U-Boot` and `OpenSBI`
- use `linux-nezha-dev` and `u-boot-nezha` as a virtual preferred provider
- `U-Boot` isn't set as the SBI payload, because D1 SoC using the U-Boot TOC1 image instead
- `KERNEL_DEVICETREE` and `RISCV_SBI_FDT` aren't set because the DTB is loaded from RAM at address `${fdtcontroladdr}`
- set proper `U-Boot` defconfig, entrypoint, device tree load address and binary file name:

```plaintext
UBOOT_MACHINE = "nezha_defconfig"
UBOOT_ENTRYPOINT = "0x40200000"
UBOOT_DTB_LOADADDRESS = "0x4FA00000"
UBOOT_DTB_BINARY ?= "sun20i-d1-nezha.dtb"
```
Boot firmware on D1 consists of three parts, which largely correspond to the components used by 64-bit ARM SoCs:

- **boot0** - it is modified for this board and used as SPL due to features such as enabling the T-HEAD ISA and MMU extensions. Used instead of U-Boot SPL.
- **OpenSBI** - supervisor which is an interface between two less privileged modes boot0 and TPL bootloader.
- **U-Boot** - TPL bootloader which initializes additional hardware and loads kernel from storage or the network.

More information can be found at [linux-sunxi](https://www.linux-sunxi.org/) wiki.
In meta-riscv you can find a formal description of the structure in `nezha.wks` file.
To fit in the Yocto Project build system some adjustments to boot0 Makefile had to be made:

- Allow overriding the variable which contains information about the used tool eg. C compiler and linker,
- Remove `nostdinc` from `config.mk` which helps build on different kinds of toolchains
- Fix build with `binutils v2.28` - it was necessary due to the new ISA specification version 20191213

Link to the recipe:

https://github.com/riscv/meta-riscv/blob/master/recipes-bsp/boot0/boot0.bb
Mainline OpenSBI supports the C906 out of the box, but it needs a few tweaks and a new reset driver for the sunxi watchdog. Here we have two patches prepared by Samuel Holland which are applied during build process:

- Add a separate compatible timer for the D1 CLINT which does not support 64-bit MMIO access
- FDT requires match data to be constant. Match data stores hardware attributes that do not change at runtime, so it does not need to be mutable

Link to recipe:

Stand-alone recipe only for Nezha board was prepared. Main changes compared to mainline U-Boot recipe:

- Use a patched version of 2022.01 version of U-Boot from **Fu Wei** (Fedora) fork:

  SRC_URI = "git://github.com/tekkamanninja/u-boot.git;branch=allwinner_d1"
  SRCREV = "6db9960b2443ef84b88a573cb5817f8e0ef3712e"

- Apply a patch that fixes a problem during uncompressing the Kernel Image:

  Error: inflate() returned -5
  Image too large: increase CONFIG_SYS_BOOTM_LEN

- Fix build with **binutils v2.28** - it was necessary due to the new ISA specification version 20191213
• Provide `toc.cfg` used by `mkimage` tool to create TOC1 image

```ini
[opensbi]
file = fw_dynamic.bin
addr = 0x40000000
[dtb]
file = u-boot.dtb
addr = 0x44000000
[u-boot]
file = u-boot.bin
addr = 0x4a000000
```

• Provide custom U-Boot Environment file `uEnv-nezha.txt`

```ini
bootargs=earlycon=sbi clk_ignore_unused initcall_debug=0 console=ttyS0,115200
       loglevel=8 root=/dev/mmcblk0p2 rootwait init=/sbin/init
bootcmd_load_f=load ${devtype} ${devnum}:${distro_bootpart} ${ramdisk_addr_r} fitImage
bootcmd_run=bootm ${ramdisk_addr_r} -${fdtcontroladdr}
bootcmd=run bootcmd_load_f; run bootcmd_run
```
Add new task `do_make_toc1_image` which is executed after `do_compile` and before `do_deploy`. It has dependency at OpenSBI `do_deploy` task too.

```bash

    do_make_toc1_image() {
        cd ${B}
        cp ${DEPLOY_DIR_IMAGE}/fw_dynamic.bin ${B}
        ${B}/tools/mkimage -T sunxi_toc1 -d ${WORKDIR}/toc.cfg ${B}/u-boot.toc1
    }

```

Link to recipe:

Same as for U-Boot, a stand-alone recipe was prepared. The new recipe provides the following things:

- Current version of kernel: v5.16
- Use a Fu Wei (Fedora) fork with support of the Allwinner D1 chip

```
SRCREV_meta := "ea948a0983d7b7820814e5bce4eda3079201bd95"
SRCREV_machine := "af3f4a1caec12845b809fba959e6334ab3b52a40"
FORK := "tekkamanninja"
BRANCH := "allwinner_nezha_d1_devel"
KMETA = "kernel-meta"
```
Same as for U-Boot, a stand-alone recipe was prepared. The new recipe provides the following things:

- Add **cgroups** and **autofs4** kernel features with the following:

  ```
  KERNEL_FEATURES += "features/cgroups/cgroups.cfg"
  KERNEL_FEATURES += "ktypes/standard/standard.cfg"
  ```

- Fix build with **binutils v2.28** - it was necessary due to the new ISA specification version 20191213

Link to recipe:

Before building you should have the following things on your host PC:

- **Install kas-container**

  ```bash
  $ sudo wget https://raw.githubusercontent.com/siemens/kas/master/kas-container
  -O /usr/bin/kas-container
  $ sudo chmod 755 /usr/bin/kas-container
  ```

- **Clone meta-riscv repository**

  ```bash
  $ git clone https://github.com/riscv/meta-riscv.git
  ```

To build core-image-minimal Poky distro run the following command:

```bash
$ SHELL=/bin/bash kas-container build meta-riscv/nezha.yml
```
- **U-Boot SPL** - is currently available in some form:
  
  [https://github.com/smaeul/u-boot/commit/7f9f2708f1b49f1936731aab4019cdff47b8dc29](https://github.com/smaeul/u-boot/commit/7f9f2708f1b49f1936731aab4019cdff47b8dc29)

- **rng-tools** - for some reason it crashes during the start with SIGSEGV in libc:
  
  ```
  [   10.792295] rngd[139]: unhandled signal 11 code 0x2 at 0x0000003fc72e1378 in libc-2.35.so[3fc727e000+fd00]
  [   10.948096] CPU: 0 PID: 139 Comm: rngd Not tainted 5.16.0-nezha #1
  ```

  This problem doesn't exist when the haveged random number generator is used in the build.

- **WiFi & Bluetooth module** - for now it isn't possible to use a wireless interface. There is a need to port XR829 kernel module for version v5.16 and higher from [Tina-Linux](https://github.com/smaeul/u-boot/commit/7f9f2708f1b49f1936731aab4019cdff47b8dc29) (kernel version: v5.4)