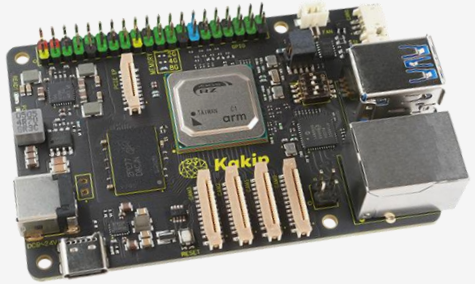
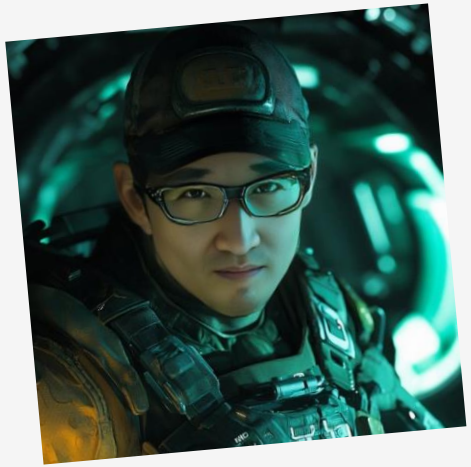
Kakip's Transformer: Dynamic 40-Pin Header Magic!

OSC-Nagoya 2025. Wig Cheng



About Me



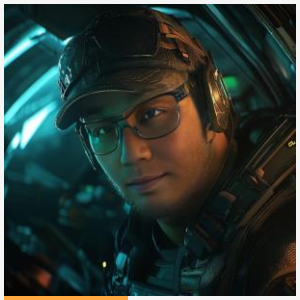
Wig Cheng

One of Kakip Community maintainers

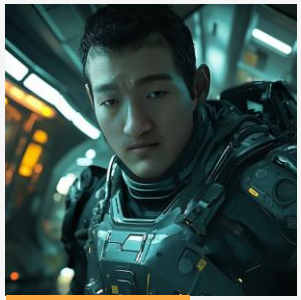
- Android BSP and Enterprise
- Linux OS
  - Ubuntu/Debian
  - Yocto
- Open Source contribution
  - Linux Kernel upstream
  - Google AOSP upstream
  - Robotic arm community
  - E-paper community
- Github: wigcheng

Other kakip contributors

CM



LC



Jason

arm64: dts: renesas: overlays: add device overlay for pwm0
jason9816 committed last month

arm64: dts: renesas: overlays: implement device overlay for pwm0
jason9816 committed last month

arm64: dts: renesas: kakip-es1: add pwm0 definition
jason9816 committed last month

Commits on Feb 8, 2025

drivers: pwm: pwm-gpio.c: backport driver from mainline v6.13, fix compile error
jason9816 committed on Feb 8

Commits on Feb 5, 2025

pwm: Add GPIO PWM driver
vwax authored and jason9816 committed on Feb 5

arm64: dts: renesas: overlays: add device overlay for spi0
lc-wang committed on Feb 5

arm64: dts: renesas: overlays: implement device overlay for spi0
lc-wang committed on Feb 5

Outline

- **GENEIRC 40-PIN Header Definition**
- **Device Tree Overlay**
- **Implementing on Kakip-AI SBC**
- **Yocto Scarthgap traveling**
- **Future work**

Note: I set this session as beginner-level, so there won't be too much driver coding in today's session. The focus will be more on the overall architecture.

01

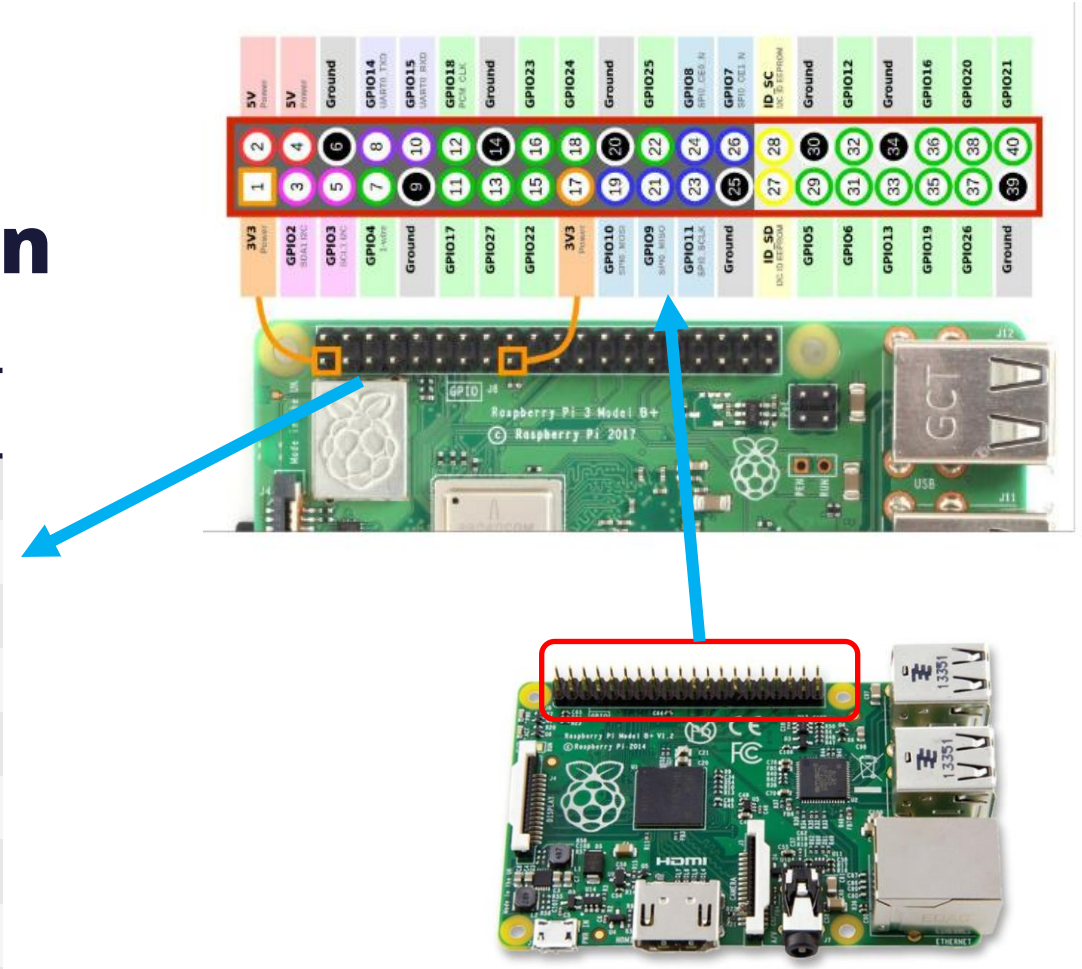
GENEIRC 40-PIN Header Definition

The History of the 40-PIN

First 40-PIN Born

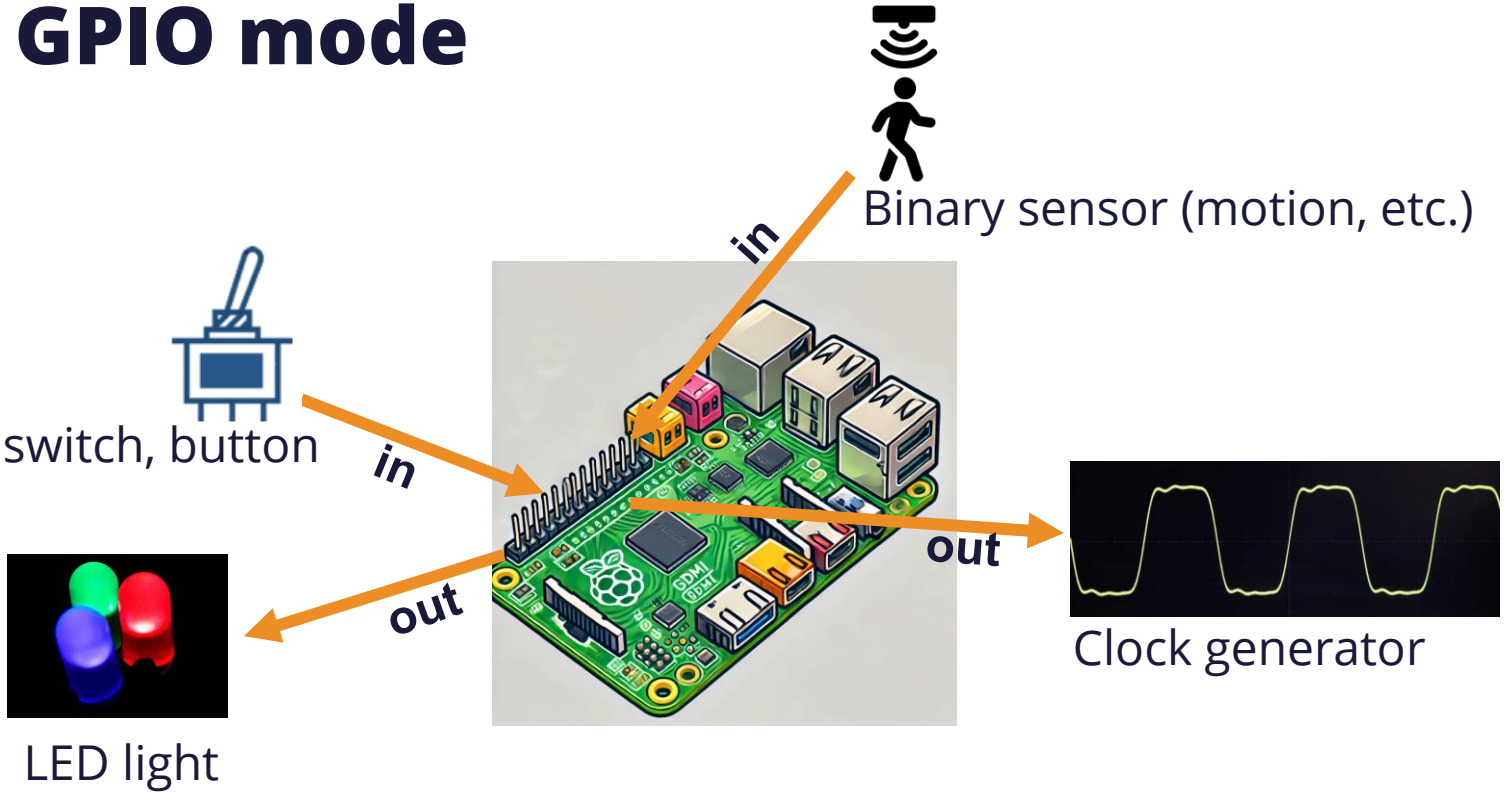
2014/07 for Raspberry PI Model B+

Function	Number
5Volt	2
3.3Volt	2
GND	8
I2C	1 => 2x GPIO
UART	1 => 2x GPIO
SPI	1 => 4x GPIO
PWM	2 => 2x GPIO
PCM	1 => 4x GPIO
GPIO	13



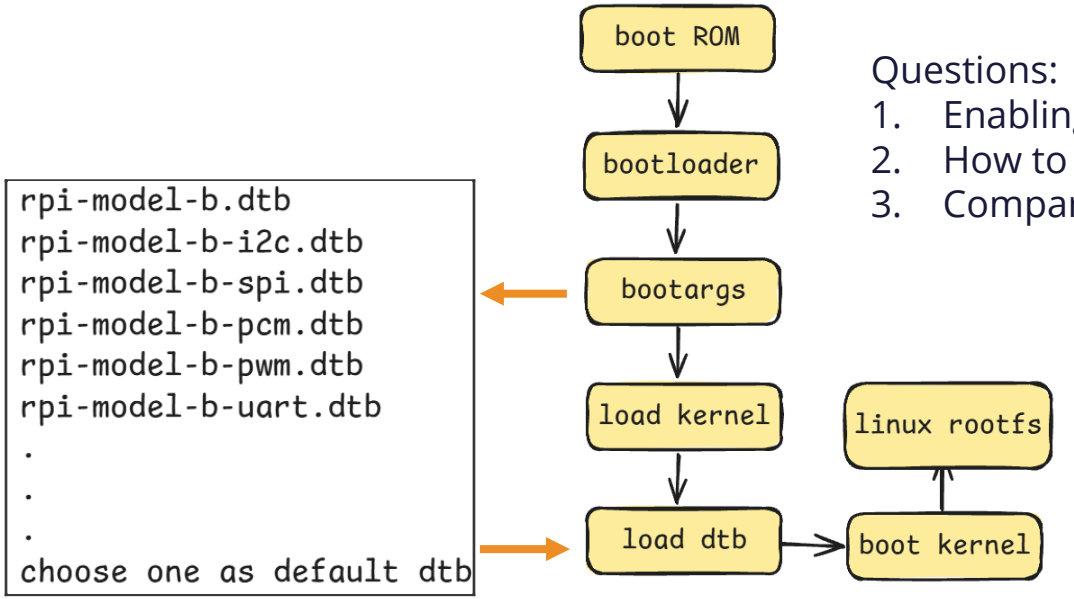
Raspberry Pi 1 Model B+

GPIO mode


Binary sensor (motion, etc.)
in
switch, button
in
out
out
Clock generator
LED light

40-PIN Multi-function on Linux

static change (simplified boot flow)



Questions:

1. Enabling i2c, spi and uart at the same time?
2. How to implement a dynamic change function?
3. Compare the implementing complexity

02

Device Tree Overlay

A Journey Through DTBO’s History

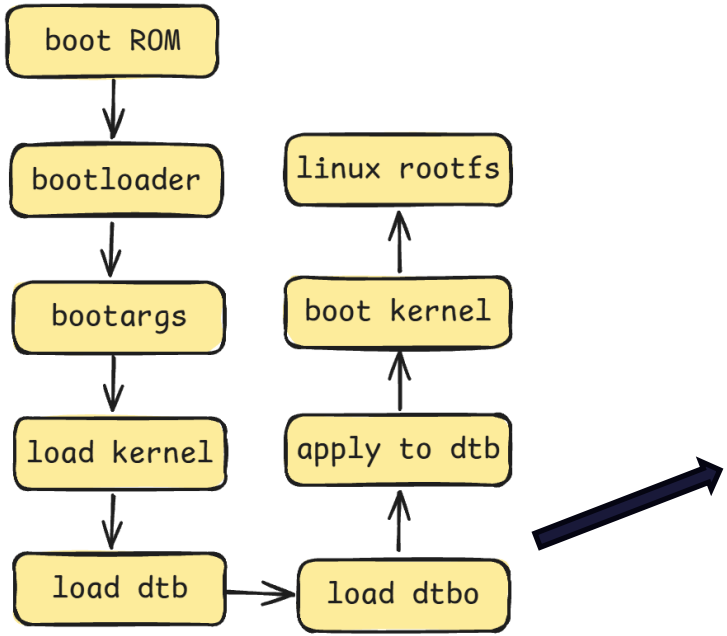
Device-Tree Overlay

First DTBO on Linux Kernel upstream

Date: 2014/07 submitted, 2015/5 merged
Supported platform: BeagleBone Black (ARM32 TI AM3358)
patches: 201c910

Device-Tree Overlay

dynamic change (simplified boot flow)




```
rpi2-i2c.dtbo
rpi2-spi.dtbo
rpi2-pcm.dtbo
rpi2-pwm.dtbo
rpi2-uart.dtbo
.
.
.
multiple dtbo can be loaded!
```

Device-Tree Overlay

Implementing complexity comparison

	With dtbo	Without dtbo
Prepare dts number functions = **n**	**n**	: $a_n = n(n-1) + \sum_{k=1}^{n-2} k$ (for n ≥ 2, and $a_1 = 1$)
Dynamic change	YES	Partially
Development level	Medium?	Easy

03

Implementing on Kakip-AI SBC

The Implementation Section Officially Begins

Kakip-ai

ARM64 Renesas RZ/V2H

- Features
  - ✓ Powerful NPU (Up to 80TOPS)
  - ✓ **Raspberry-PI compatible 40-PIN header**
  - ✓ 4x MIPI-CSI connector
  - ✓ 2x CAN-bus interface
- OS support
  - ✓ CIP Kernel 5.10 (6.1 will be released soon)
  - ✓ Ubuntu 24.04 LTS
- Applications
  - ✓ Edge AI & robotics

Support libgpiod

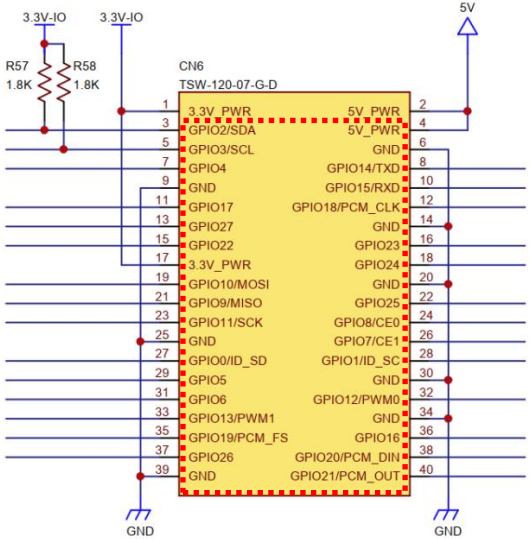
Feature	sysfs GPIO	libgpiod (Character Device)
Year Introduced	~2009	**2017**
Interface Type	Filesystem (/sys/class/gpio/)	Character device (/dev/gpiochipX)
Multi-line Operations	❌ Not supported	✅ Supported
Interrupt / Event	❌ Not user-friendly	✅ poll/select/epoll supported
Performance	⚠️ Relatively slow	✅ Fast and efficient
Race Condition Handling	❌ Prone to issues	✅ Handled via kernel APIs
Kernel Support Since	Before Linux 4.8	Linux 4.8 and newer
Deprecation Status	✅ Deprecated since Linux 5.10	✅ Recommended for new development
CLI Tools	echo and cat the sysfs nodes	gpioinfo, gpiodetect, gpioset, gpioget

Support libgpiod (cont.d)

Major changed: [5a33a30](#) [20a5c759](#) [22baace](#)
Vendor's original architecture base on sysfs way

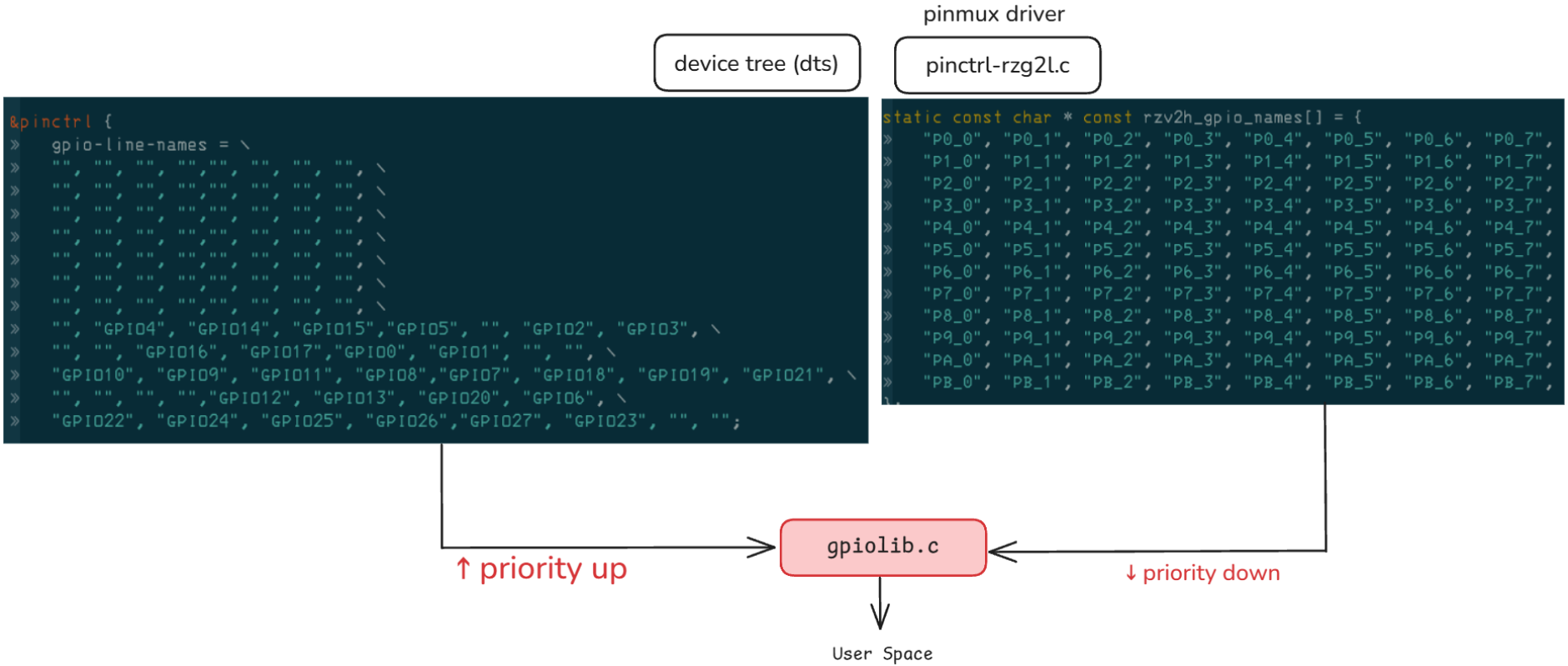
Target:

Linux OS

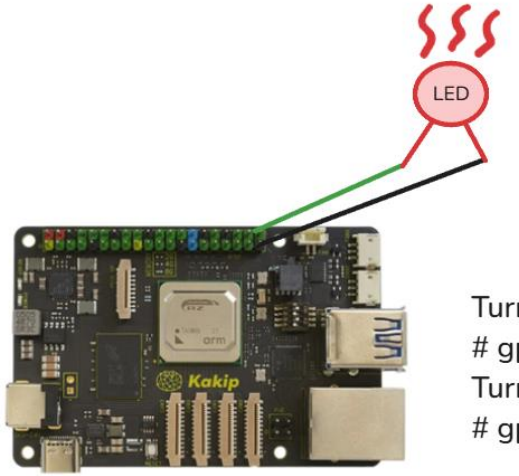


gpioinfo

```
line  66:      "GPIO16"       kernel   input  active-high [used]
line  67:      "GPIO17"       kernel   input  active-high [used]
line  68:      "GPIO0"        kernel   input  active-high [used]
line  69:      "GPIO1"        kernel   input  active-high [used]
line  70:      unnamed        kernel   input  active-high [used]
line  71:      unnamed        kernel   input  active-high [used]
line  72:      "GPIO10"       kernel   input  active-high [used]
line  73:      "GPIO9"        kernel   input  active-high [used]
line  74:      "GPIO11"       kernel   input  active-high [used]
line  75:      "GPIO8"        kernel   input  active-high [used]
line  76:      "GPIO7"        kernel   input  active-high [used]
line  77:      "GPIO18"       unused   input  active-high
line  78:      "GPIO19"       unused   input  active-high
line  79:      "GPIO21"       unused   input  active-high
line  80:      unnamed "SDHI0 VccQ"  output  active-high [used]
line  81:      unnamed "sd0_pwr_en"  output  active-high [used]
line  82:      unnamed "SDHI1 VccQ"  output  active-high [used]
line  83:      unnamed "sd1_pwr_en"  output  active-high [used]
line  84:      "GPIO12"       unused   input  active-high
line  85:      "GPIO13"       kernel   input  active-high [used]
line  86:      "GPIO20"       unused   input  active-high
line  87:      "GPIO6"        unused   input  active-high
line  88:      "GPIO22"       unused   input  active-high
line  89:      "GPIO24"       unused   input  active-high
line  90:      "GPIO25" "cam0_pwr_en" output active-high [used]
line  91:      "GPIO26"       unused   input  active-high
line  92:      "GPIO27"       unused   input  active-high
line  93:      "GPIO23"       unused   input  active-high
line  94:      unnamed        unused   input  active-high
line  95:      unnamed        unused   input  active-high
```


pinmux driver
device tree (dts)
pinctrl-rzg2l.c
&pinctrl {
gpio-line-names = \
"", "", "", "", "", "", "", "", \
"", "", "", "", "", "", "", "", \
"", "", "", "", "", "", "", "", \
"", "", "", "", "", "", "", "", \
"", "", "", "", "", "", "", "", \
"", "", "", "", "", "", "", "", \
"", "", "", "", "", "", "", "", \
"", "GPIO4", "GPIO14", "GPIO15","GPIO5", "", "GPIO2", "GPIO3", \
"", "", "GPIO16", "GPIO17","GPIO0", "GPIO1", "", "", \
"GPIO10", "GPIO9", "GPIO11", "GPIO8","GPIO7", "GPIO18", "GPIO19", "GPIO21", \
"", "", "", "","GPIO12", "GPIO13", "GPIO20", "GPIO6", \
"GPIO22", "GPIO24", "GPIO25", "GPIO26","GPIO27", "GPIO23", "", "";
static const char * const rzv2h_gpio_names[] = {
"P0_0", "P0_1", "P0_2", "P0_3", "P0_4", "P0_5", "P0_6", "P0_7",
"P1_0", "P1_1", "P1_2", "P1_3", "P1_4", "P1_5", "P1_6", "P1_7",
"P2_0", "P2_1", "P2_2", "P2_3", "P2_4", "P2_5", "P2_6", "P2_7",
"P3_0", "P3_1", "P3_2", "P3_3", "P3_4", "P3_5", "P3_6", "P3_7",
"P4_0", "P4_1", "P4_2", "P4_3", "P4_4", "P4_5", "P4_6", "P4_7",
"P5_0", "P5_1", "P5_2", "P5_3", "P5_4", "P5_5", "P5_6", "P5_7",
"P6_0", "P6_1", "P6_2", "P6_3", "P6_4", "P6_5", "P6_6", "P6_7",
"P7_0", "P7_1", "P7_2", "P7_3", "P7_4", "P7_5", "P7_6", "P7_7",
"P8_0", "P8_1", "P8_2", "P8_3", "P8_4", "P8_5", "P8_6", "P8_7",
"P9_0", "P9_1", "P9_2", "P9_3", "P9_4", "P9_5", "P9_6", "P9_7",
"PA_0", "PA_1", "PA_2", "PA_3", "PA_4", "PA_5", "PA_6", "PA_7",
"PB_0", "PB_1", "PB_2", "PB_3", "PB_4", "PB_5", "PB_6", "PB_7",
};
↑ priority up
gpiolib.c
↓ priority down
User Space

libgpiod Example



Turn on
gpioset gpiochip0 79=1
Turn off
gpioset gpiochip0 79=0

- Ubuntu 24.04 (Kakip official way)

  Install libgpiod utilities (v1.7)
  $ sudo apt install libgpiod
  Install libgpiod for development (**C code**)
  $ sudo apt install libgpiod-dev

- Yocto Dunfell (3.1) => **doesn't support**
- Yocto Scarghgap (5.0)

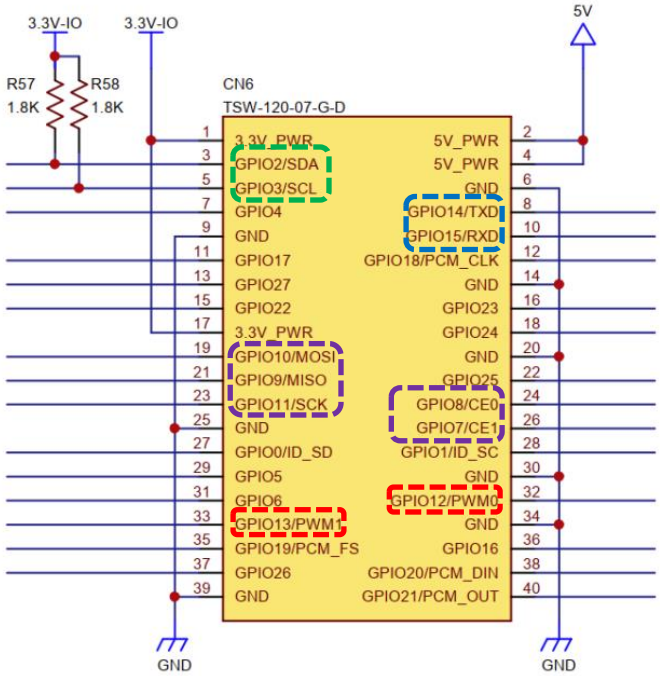
  Add libgpiod package to distro conf of meta-layer
  conf/distro/<distro>.conf

  IMAGE_INSTALL:append =
  libgpiod libgpiod-tools

Implementing multi-function

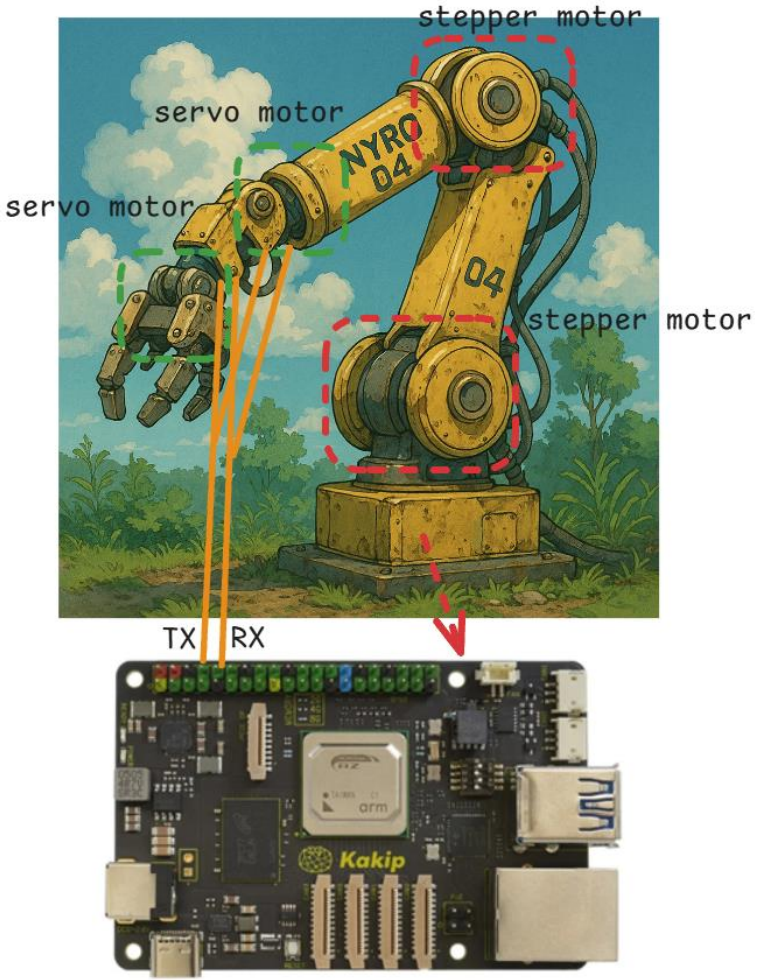
Current configuration

- ✓ 1x I2C
- ✓ 1x UART
- ✓ 2x PWM
- ✓ 1x SPI



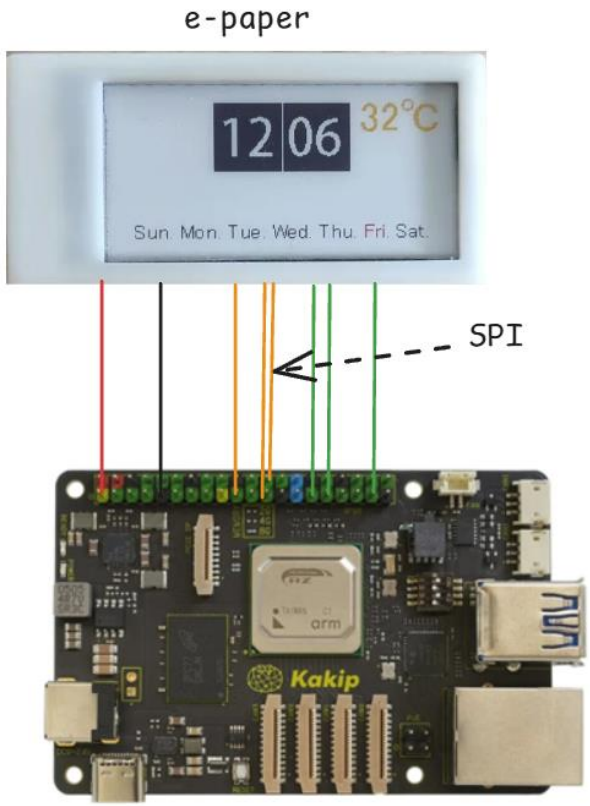
Multi-function – UART

- Add pinmux and SCI node to dts
- loop-back testing
  - TX ⟷ RX
- Baudrate testing
  - 9600bps
  - 115200bps
  - 2Mbps
- Device communication



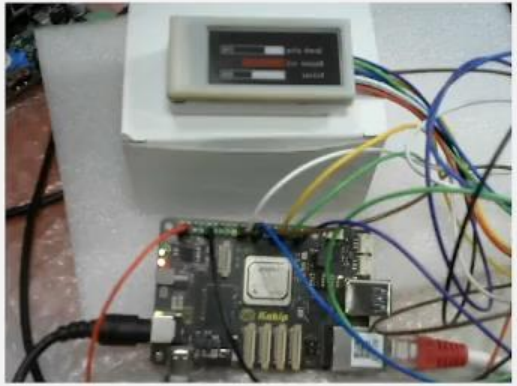
Multi-function – SPI

- Add pinmux and SPI node to dts
- loop-back testing
    - MOSI <-> MISO (spidev_test) (kernel/tools/spi/spidev_test.c)
- Speed testing
    - 5MHz
- Device communication



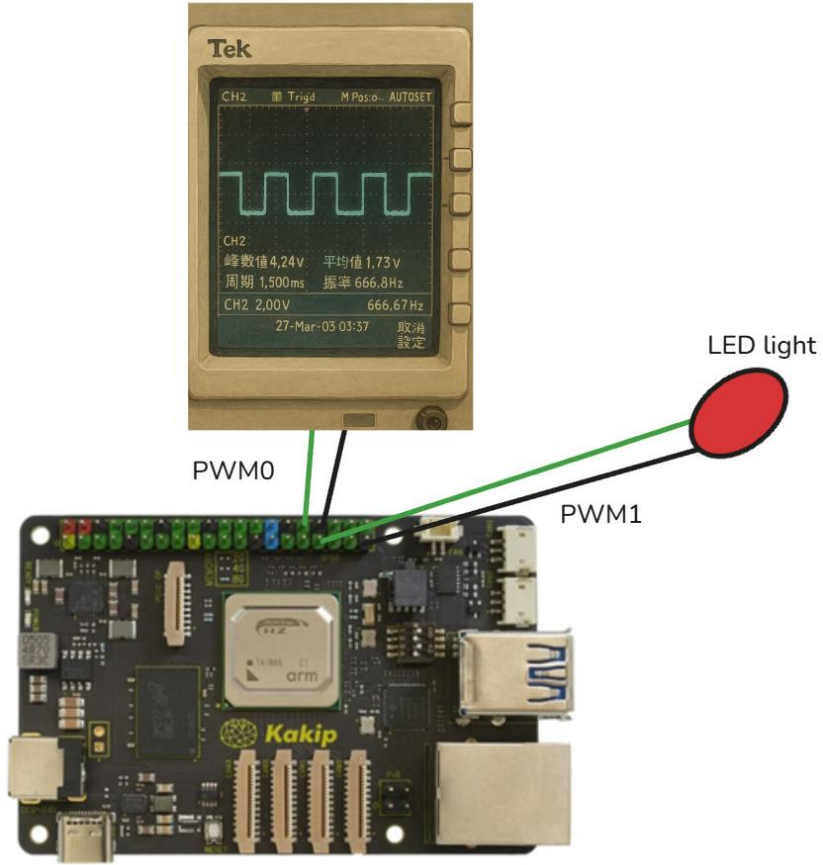
```
root@localhost:/home/ubuntu#
root@localhost:/home/ubuntu# ls
jd79661-epd-image-kakip.c  jd79661_test_flash  jd79661_test_flash_old  ダウンロード  テンプレート  デスクトップ  ドキュメント  ビデオ  ピクチャ  ミュージック  公開
root@localhost:/home/ubuntu#
```

Live Demo!!!
2.13” E-Paper with SPI !



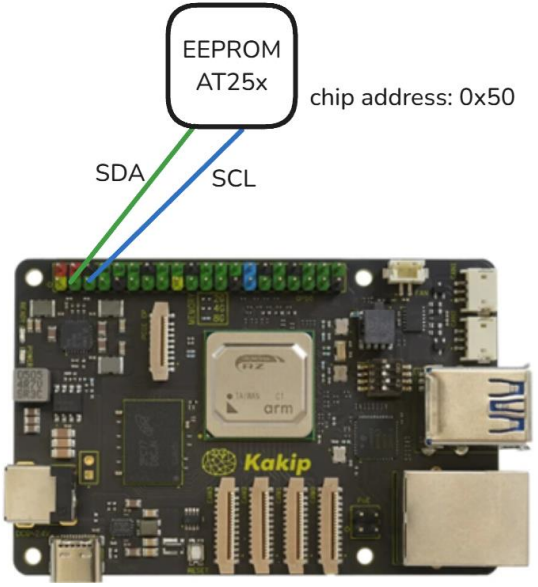
Multi-function – PWM

- Add pinmux and PWM node to dts
- Renesas does not support PWM driver
    - **backport gpio-pwm driver**
- Frequency/duty cycle testing
    - 1500us (667Hz)
    - 0% ~ 100% duty cycle
- Device communication

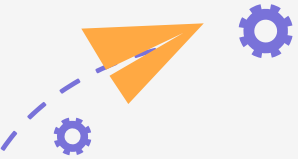


Multi-function – I2C

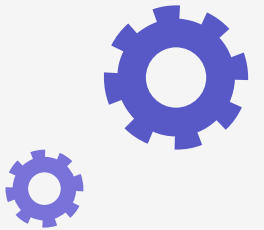
- Add pinmux and I2C node to dts
- **Port simple I2C driver by Renesas**
  5d99348 d799f0f b94da68
- Access testing using i2c-tool
  - i2cdetect
  - i2cdump
  - i2cset
  - i2cget
- Device communication



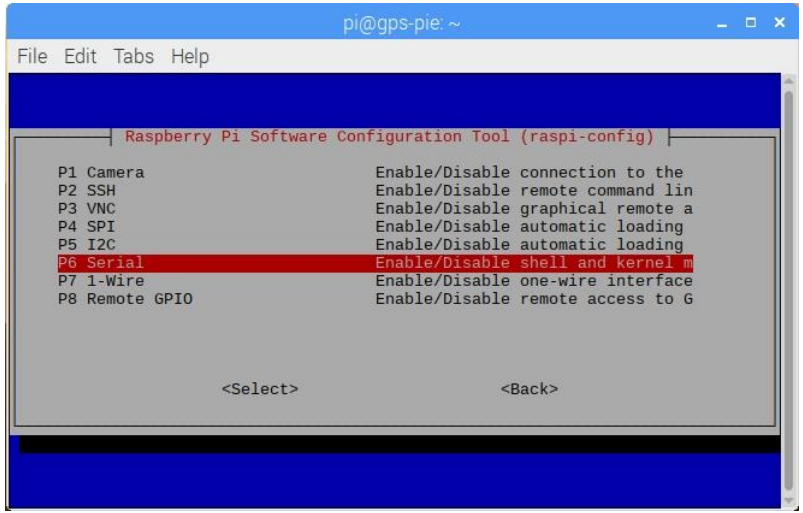

```
root@localhost:/home/ubuntu# i2cdump -y 10 0x50
No size specified (using byte-data access)
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: 01 ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
10: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
20: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
30: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
40: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
50: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
60: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
70: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
80: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
90: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
a0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
b0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
c0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
d0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
e0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
f0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff 05
```



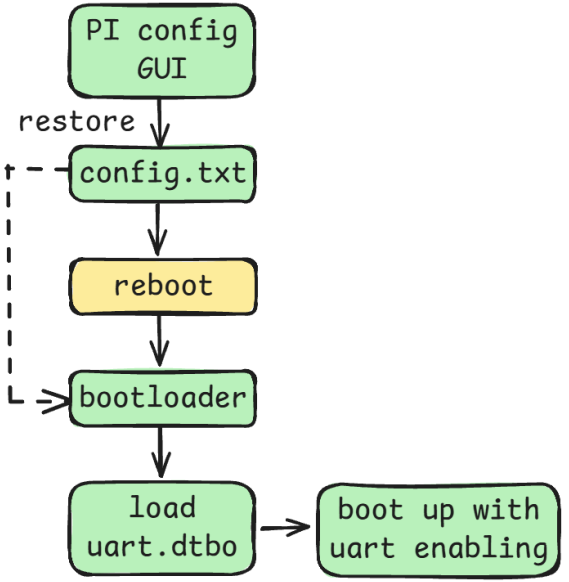
Implementing DTBO



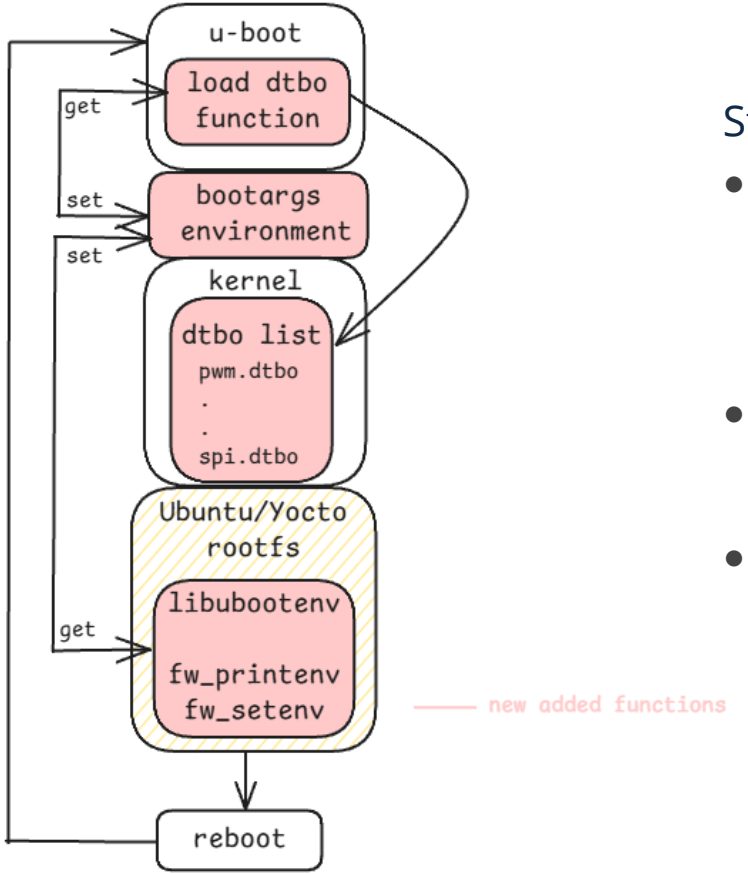
Observing Raspberry PI



Running on First boot, or in the rootfs manually



Design an architecture for Kakip



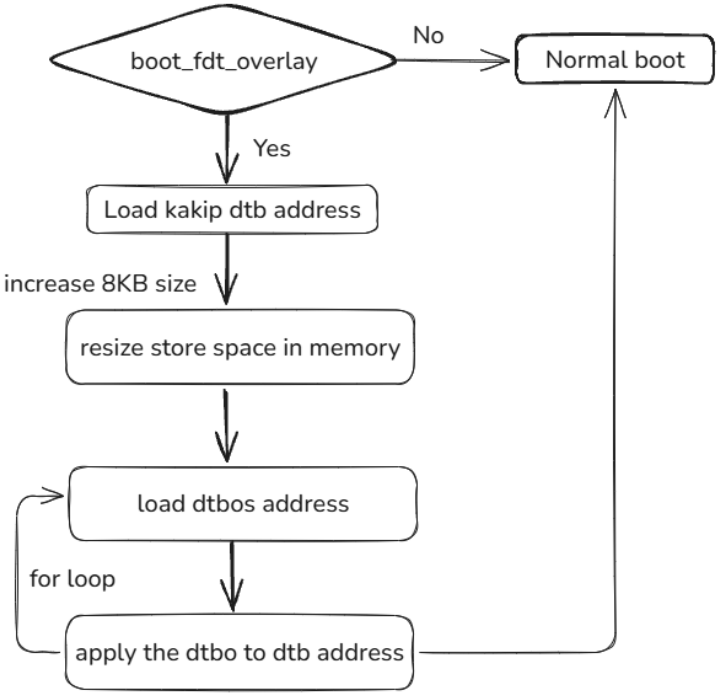
Steps

- U-boot
  - Add dtbo loading function
  - Change the bootargs environment
- Kernel
  - Implement all dtbo files
- Rootfs (Ubuntu/Yocto)
  - Install libubootenv package
  - Setting the config file

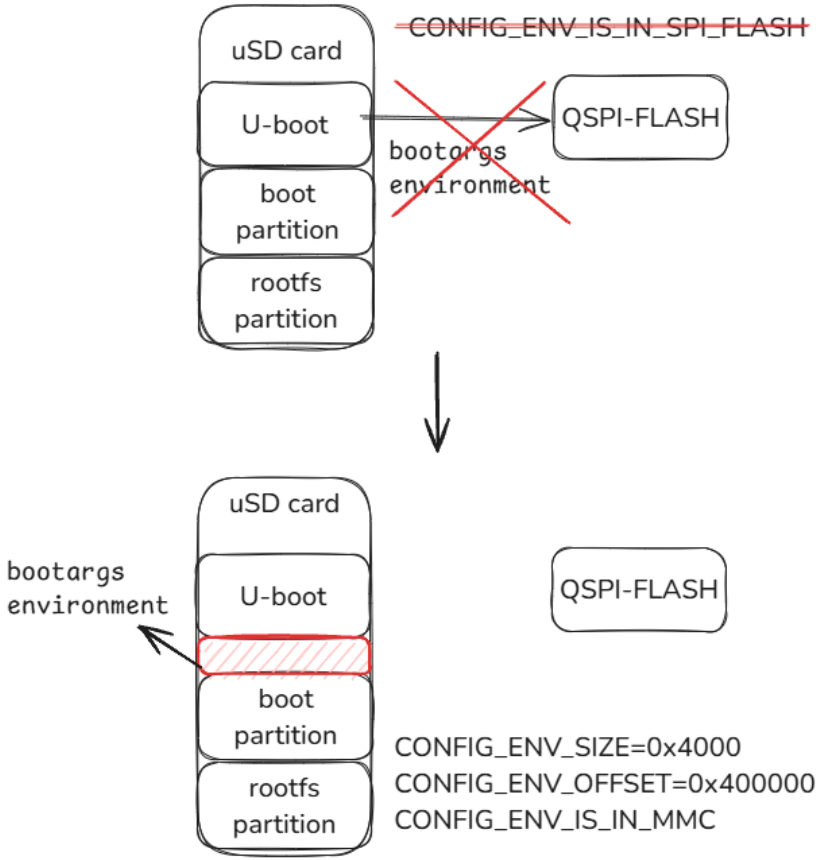
U-boot change

- Add dtbo loading function

1. Modify path: include/configs => kakip header file
2. defconfig: CONFIG_OF_LIBFDT_OVERLAY=y



- Change the bootargs environment



U-boot demo

1. *place dtbo files in /boot directory in rootfs*
2. *boot into u-boot prompt*
3. *enabling device tree overlay function*
   => *setenv boot_fdt_overlay yes*
4. *assigning overlay files what you want to loaded*
   for example: overlay spi0
   => *setenv fdt_overlay_files kakip-es1-spi0*
   for example: overlay pwm0 and pwm1
   => *setenv fdt_overlay_files kakip-es1-pwm0 kakip-es1-pwm1*
   for example: overlay i2c10, pwm0, pwm1
   => *setenv fdt_overlay_files kakip-es1-i2c10 kakip-es1-pwm0 kakip-es1-pwm1*
5. *Issue command to boot into rootfs after assigned*
   => *boot*

Note that it has memory retention, so you don't need to apply the same settings every time you boot.

Kernel change

- Add dtbo loading function
  - commits: 15b88d6 44a18a0 c9caa6b 9d4c10c 3fc9d2e
- Implementing dtbo files

```
kakip-es1-pwm0.dtbo
kakip-es1-pwm1.dtbo
kakip-es1-pwm0.dtbo
kakip-es1-sci5-to-gpio.dtbo
kakip-es1-spi0.dtbo
kakip-es1-imx462.dtbo
kakip-es1-imx219.dtbo
kakip-es1-i2c10.dtbo
```

rootfs change

- Add libubootenv package
- Add environment access config file

Ubuntu
$ sudo apt install libubootenv-tool
$ sudo vim etc/fw_env.config
/dev/mmcblk0 0x400000 0x4000

Yocto
Add a new recipe
DEPENDS = "u-boot-fw-utils"
SRC_URI = file://fw_env.config

fw_env.config content
/dev/mmcblk0 0x400000 0x4000

- U-boot overlay
  - Pros:
    can use in u-boot prompt directly
  - Cons:
    Need a debug console
- libubootenv overlay
  - Pros:
    Easy to use without debug console such as ssh, GUI terminal with keyboard, etc.
  - Cons:
    Need tweak rootfs

libubootenv demo

1. *boot into Ubuntu or Yocto operating system and login.*
2. *Issue command to checking the bootargs status*
   $ sudo fw_printenv
   ... ignore ...
   boot_fdt_overlay=no
   ... ignore ...
   fdt_overlay_files=kakip-es1-spi0
   ... ignore ...
3. *enable overlay function*
   $ sudo fw_setenv boot_fdt_overlay yes
4. *change the overlay files*
   for example, enabling SPI function
   $ sudo fw_setenv fdt_overlay_files kakip-es1-spi0
   for example, enabling spi0, pwm0 and pwm1
   $ sudo fw_setenv fdt_overlay_files 'kakip-es1-spi0 kakip-es1-pwm0 kakip-es1-pwm1'
5. Issue the reboot command, then overlay function will be auto stored
   $ **sudo reboot**

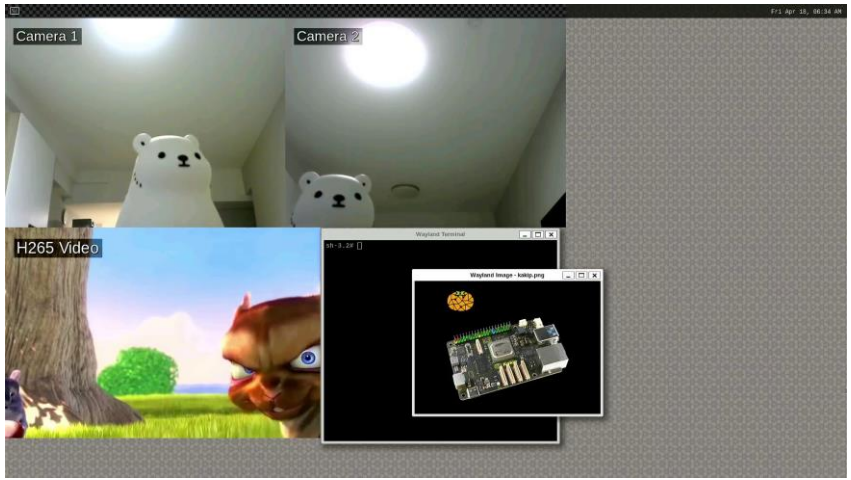
04

Yocto Scarthgap traveling

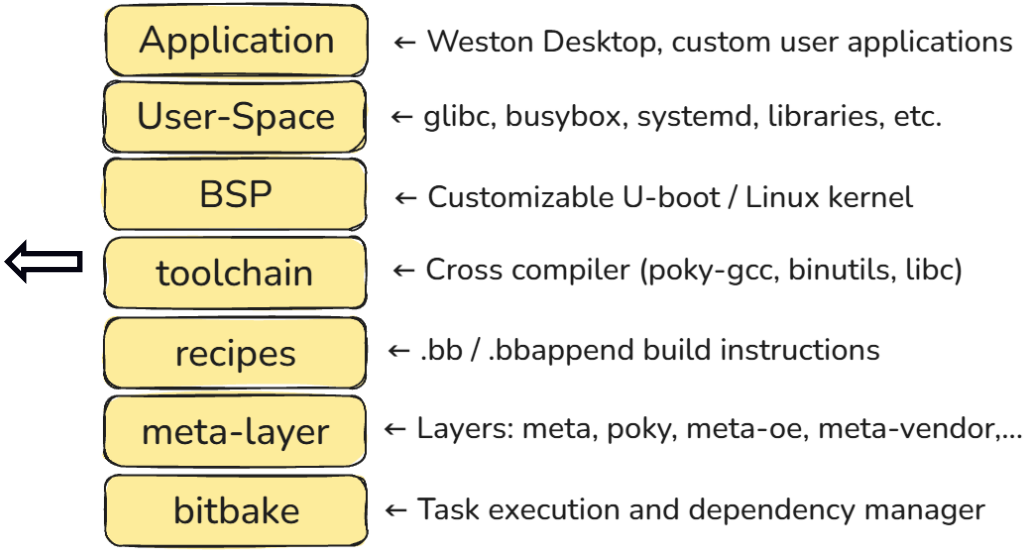
From Origin to Today

Yocto Overview

- A build system for creating custom embedded Linux distributions
- Based on Open-Embedded, using BitBake as the core task
- Focused on reproducibility, modularity (layers), and flexibility



Running Yocto Dunfell on Kakip



Yocto 3.1 (Dunfell) vs 5.0 (Scarthgap)

	Dunfell	Scarthgap
Release Date	**Apr. 2020**	**Apr. 2024**
LTS Support	**Apr. 2024 (EOL)**	**Apr. 2028**
GCC Version	9.3	13.2
Kernel Reference	5.10 and above	6.1 and above
Vendor Support	YES	NO, I implemented a Kakip meta-layer using the community version.
GPU/VPU/NPU support	YES	NO
libgpiod, glmark, libubootenv	NO	YES

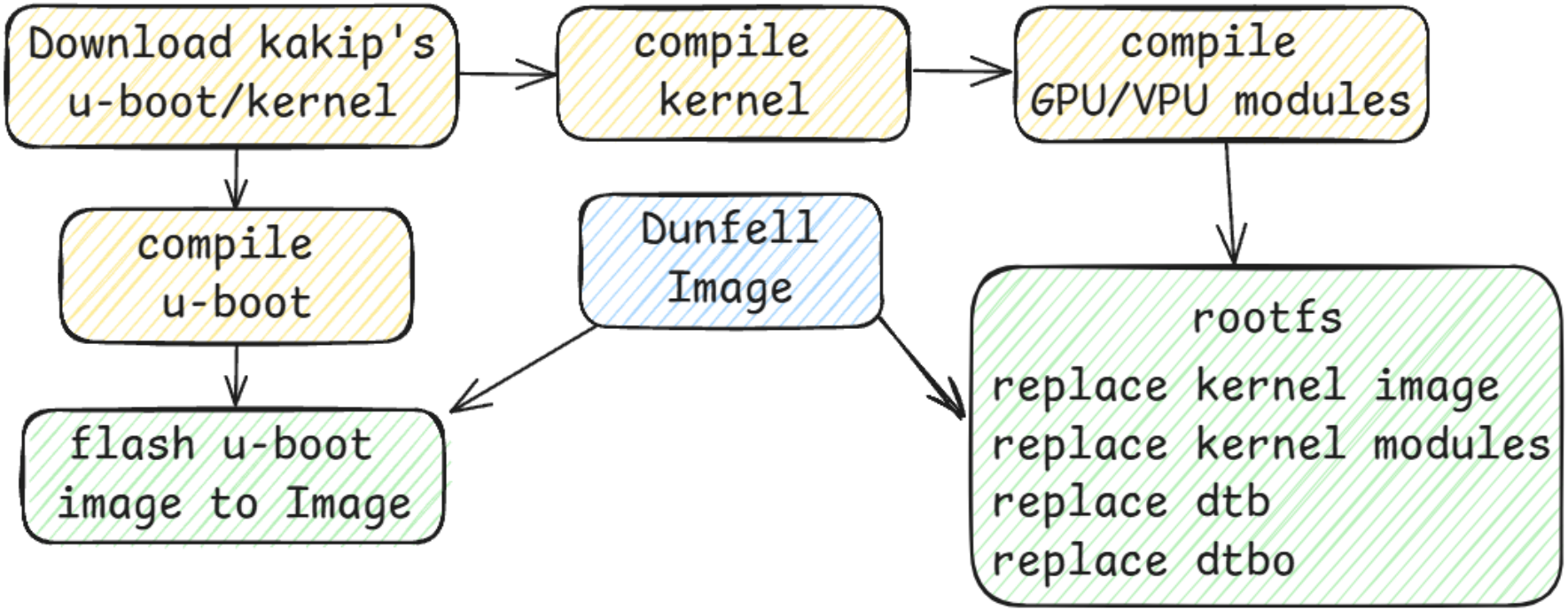
Setup Yocto Dunfell

Download the source code
$ git clone https://bitbucket.org/mqgit/mq-kakip-yocto-bsp-manifest.git -b renesas-linux-dunfell
$ cd mq-kakip-yocto-bsp-manifest

Add meta-layers and source the environment
Read the README.md in the repo
Compile the core image (Weston Desktop version)
$ MACHINE=rzv2h-evk-ver1 bitbake core-image-weston

Output image path:
$ <source folder>/build/tmp/deploy/images/images/rzv2h-evk-ver1/core-image-weston-rzv2h-evk-ver1-20250410102412.rootfs.wic.gz

Make Dunfell image for Kakip



Setup Yocto Scarthgap

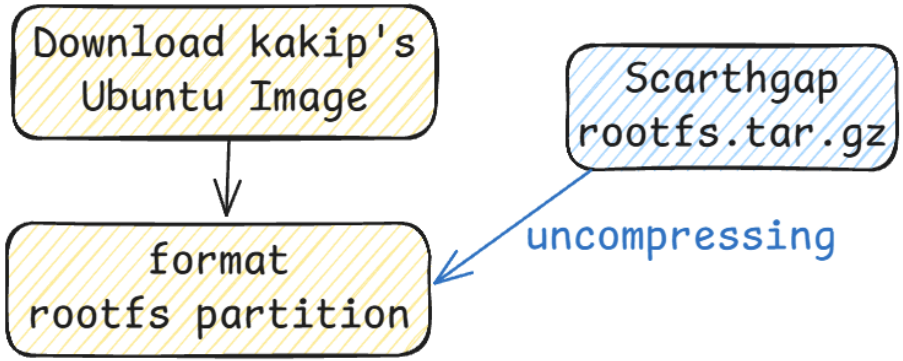
Download the source code
$ repo init -u https://bitbucket.org/mqgit/mq-kakip-yocto-bsp-manifest.git -b renesas-linux-scarthgap-community -m rz-5.10.145-1.0.0.xml
$ repo sync

Compile the image
$ MACHINE=kakip-rzv2h DISTRO=kakip-wayland source kakip-rz-setup-release.sh

Compile the minimal image (Terminal version)
$ bitbake renesas-image-minimal
Compile the minimal image (Weston Desktop version)
$ bitbake renesas-image-demo

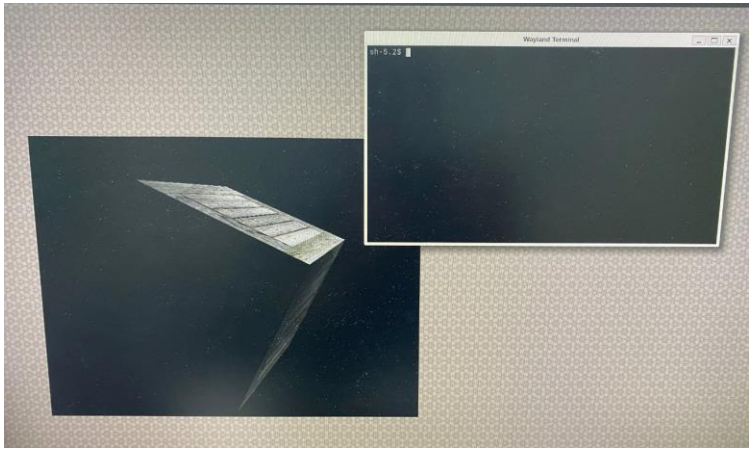
rootfs path:
$ <source folder>/build/tmp/deploy/images/kakip-rzv2h/renesas-image-demo-kakip-rzv2h.tar.gz

Make Scarthgap image for Kakip




```
root@kakip-rzv2h:~# cat /etc/os-release
ID=poky
NAME="Poky (Yocto Project Reference Distro)"
VERSION="5.0.5 (scarthgap)"
VERSION_ID=5.0.5
VERSION_CODENAME="scarthgap"
PRETTY_NAME="Poky (Yocto Project Reference Distro) 5.0.5 (scarthgap)"
CPE_NAME="cpe:/o:openembedded:poky:5.0.5"
```

Terminal version on Kakip



weston version on Kakip

05

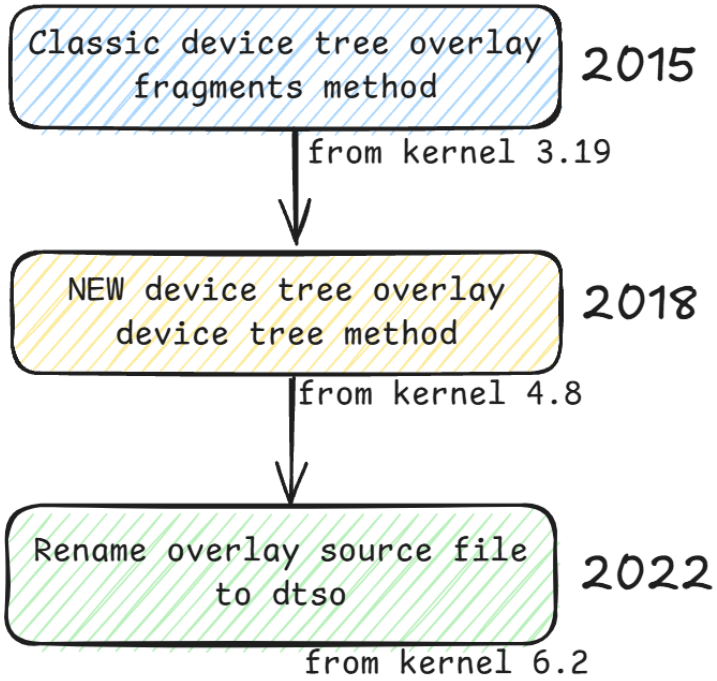
Future Work

DTBO Can Actually Be Improved Further

Overlay improvement

- Current kernel
  - Kernel 5.10
  - dtbo type: Fragments method

- **Future work**
  - **Upgrade to Renesas CIP kernel 6.1**
  - **dtbo type: change to device tree method**
  - **Backport dtso commits for overlay function**



Welcome to try it

U-boot: https://github.com/YDS-Kakip-Team/kakip_u-boot

Kernel: https://github.com/YDS-Kakip-Team/kakip_linux

Thank You!