RUNNING ZEPHYR RTOS ON CADENCE® TENSILICA® HIFI 4 DSP

Iuliana Prodan

JUNE 2023

SECURE CONNECTIONS FOR A SMARTER WORLD

PUBLIC

CONTENT

Hardware overview of i.MX 8M Plus

HiFi4 DSP support

- ➢ Current state
- ➢ What's next?

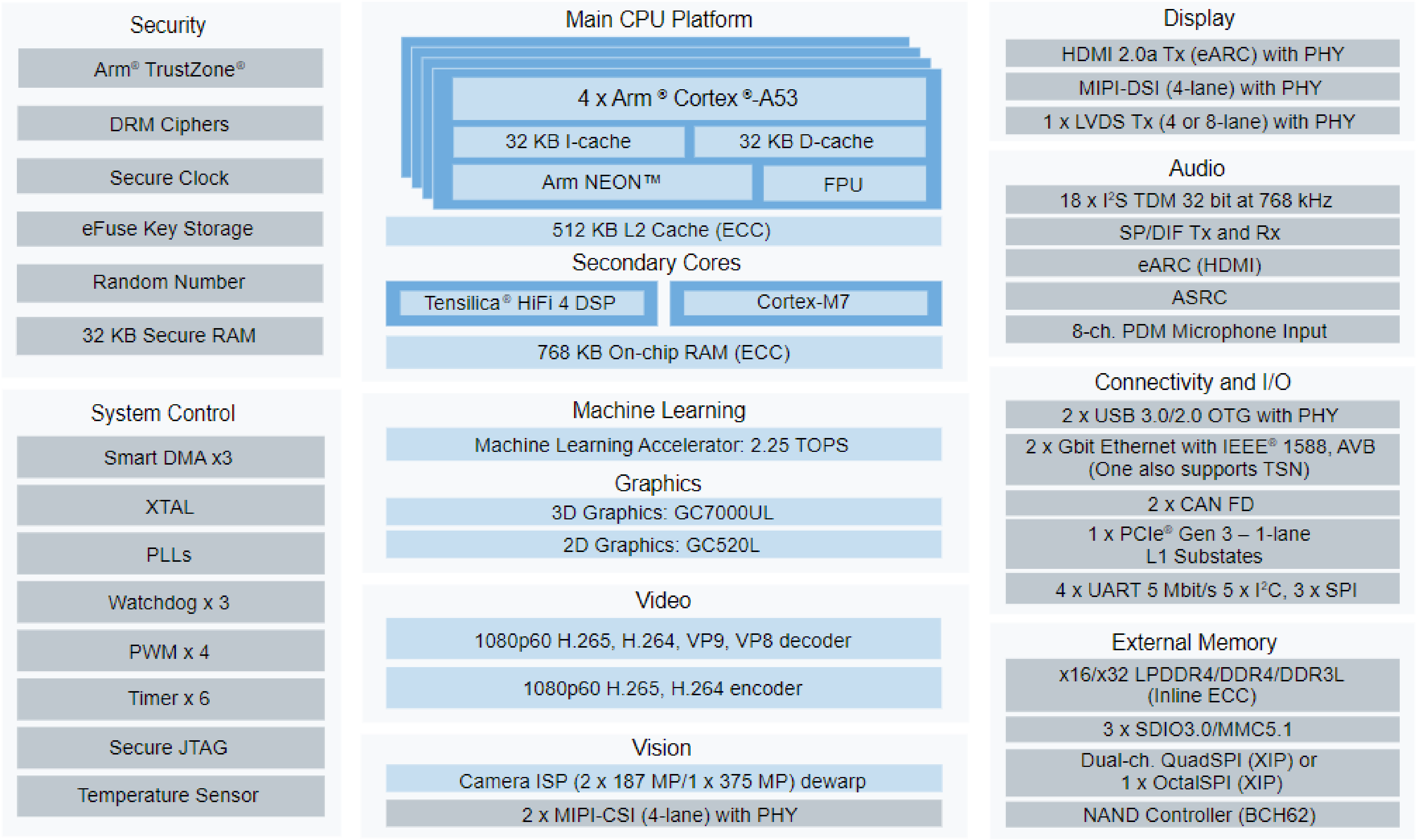
Generic Linux and Zephyr communication setup

- ➢ Remoteproc
- ➢ Rpmsg
- ➢ Mailbox
- ➢ OpenAMP

Challenges

Future work

HARDWARE OVERVIEW – I.MX 8M PLUS



Security

- Arm® TrustZone®
- DRM Ciphers
- Secure Clock
- eFuse Key Storage
- Random Number
- 32 KB Secure RAM

System Control

- Smart DMA x3
- XTAL
- PLLs
- Watchdog x 3
- PWM x 4
- Timer x 6
- Secure JTAG
- Temperature Sensor

Main CPU Platform

- 4 x Arm ® Cortex ®-A53
- 32 KB I-cache
- 32 KB D-cache
- Arm NEON™
- FPU
- 512 KB L2 Cache (ECC)

Secondary Cores

- Tensilica® HiFi 4 DSP
- Cortex-M7
- 768 KB On-chip RAM (ECC)

Machine Learning

- Machine Learning Accelerator: 2.25 TOPS

Graphics

- 3D Graphics: GC7000UL
- 2D Graphics: GC520L

Video

- 1080p60 H.265, H.264, VP9, VP8 decoder
- 1080p60 H.265, H.264 encoder

Vision

- Camera ISP (2 x 187 MP/1 x 375 MP) dewarp
- 2 x MIPI-CSI (4-lane) with PHY

Display

- HDMI 2.0a Tx (eARC) with PHY
- MIPI-DSI (4-lane) with PHY
- 1 x LVDS Tx (4 or 8-lane) with PHY

Audio

- 18 x I²S TDM 32 bit at 768 kHz
- SP/DIF Tx and Rx
- eARC (HDMI)
- ASRC
- 8-ch. PDM Microphone Input

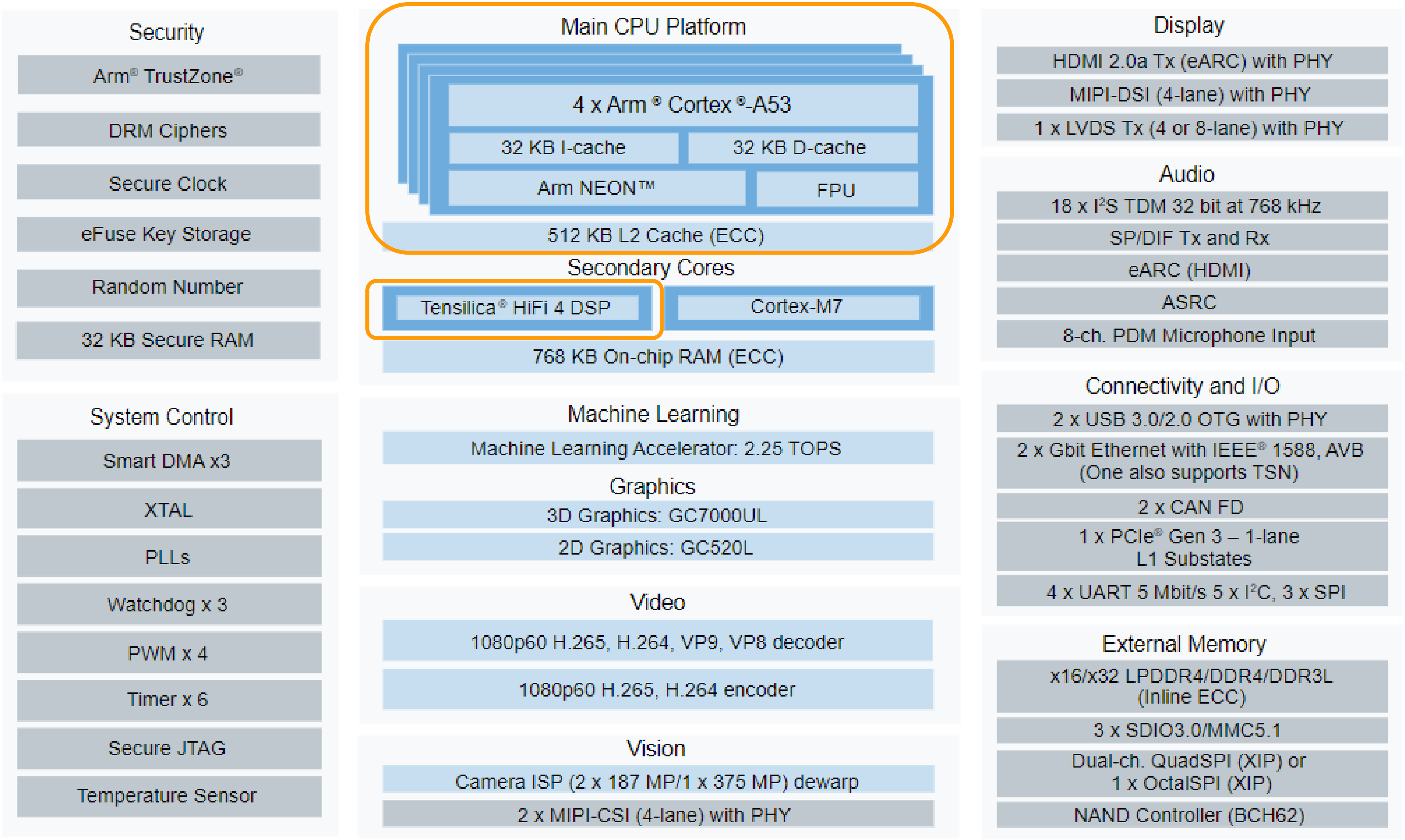
Connectivity and I/O

- 2 x USB 3.0/2.0 OTG with PHY
- 2 x Gbit Ethernet with IEEE® 1588, AVB (One also supports TSN)
- 2 x CAN FD
- 1 x PCIe® Gen 3 – 1-lane L1 Substates
- 4 x UART 5 Mbit/s 5 x I²C, 3 x SPI

External Memory

- x16/x32 LPDDR4/DDR4/DDR3L (Inline ECC)
- 3 x SDIO3.0/MMC5.1
- Dual-ch. QuadSPI (XIP) or 1 x OctalSPI (XIP)
- NAND Controller (BCH62)

HARDWARE OVERVIEW – I.MX 8M PLUS



Security

- Arm® TrustZone®
- DRM Ciphers
- Secure Clock
- eFuse Key Storage
- Random Number
- 32 KB Secure RAM

System Control

- Smart DMA x3
- XTAL
- PLLs
- Watchdog x 3
- PWM x 4
- Timer x 6
- Secure JTAG
- Temperature Sensor

Main CPU Platform

- 4 x Arm ® Cortex ®-A53
- 32 KB I-cache
- 32 KB D-cache
- Arm NEON™
- FPU
- 512 KB L2 Cache (ECC)

Secondary Cores

- Tensilica® HiFi 4 DSP
- Cortex-M7
- 768 KB On-chip RAM (ECC)

Machine Learning

- Machine Learning Accelerator: 2.25 TOPS

Graphics

- 3D Graphics: GC7000UL
- 2D Graphics: GC520L

Video

- 1080p60 H.265, H.264, VP9, VP8 decoder
- 1080p60 H.265, H.264 encoder

Vision

- Camera ISP (2 x 187 MP/1 x 375 MP) dewarp
- 2 x MIPI-CSI (4-lane) with PHY

Display

- HDMI 2.0a Tx (eARC) with PHY
- MIPI-DSI (4-lane) with PHY
- 1 x LVDS Tx (4 or 8-lane) with PHY

Audio

- 18 x I²S TDM 32 bit at 768 kHz
- SP/DIF Tx and Rx
- eARC (HDMI)
- ASRC
- 8-ch. PDM Microphone Input

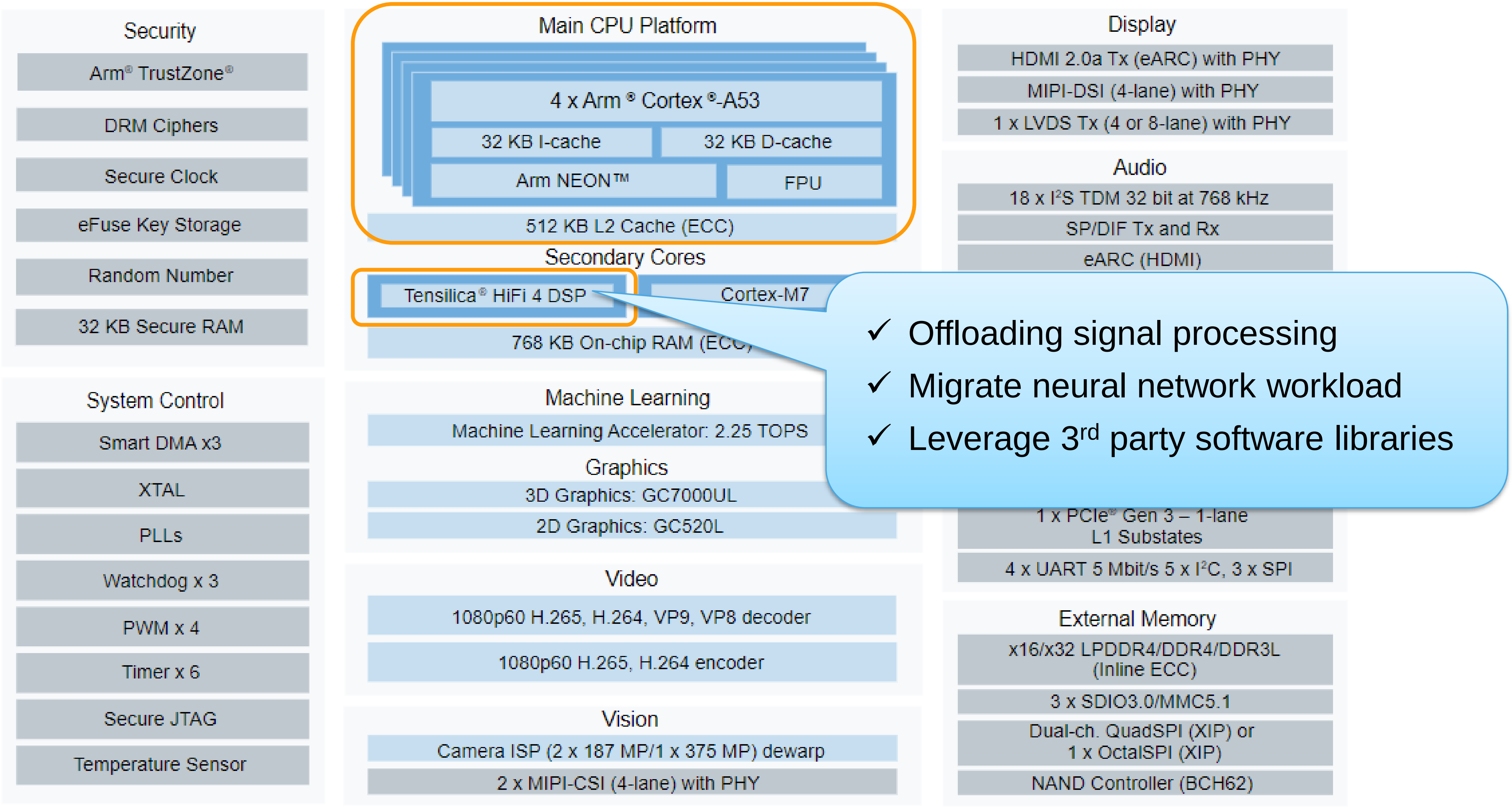
Connectivity and I/O

- 2 x USB 3.0/2.0 OTG with PHY
- 2 x Gbit Ethernet with IEEE® 1588, AVB (One also supports TSN)
- 2 x CAN FD
- 1 x PCIe® Gen 3 – 1-lane L1 Substates
- 4 x UART 5 Mbit/s 5 x I²C, 3 x SPI

External Memory

- x16/x32 LPDDR4/DDR4/DDR3L (Inline ECC)
- 3 x SDIO3.0/MMC5.1
- Dual-ch. QuadSPI (XIP) or 1 x OctalSPI (XIP)
- NAND Controller (BCH62)

HARDWARE OVERVIEW – I.MX 8M PLUS



Security
- Arm® TrustZone®
- DRM Ciphers
- Secure Clock
- eFuse Key Storage
- Random Number
- 32 KB Secure RAM

System Control
- Smart DMA x3
- XTAL
- PLLs
- Watchdog x 3
- PWM x 4
- Timer x 6
- Secure JTAG
- Temperature Sensor

Main CPU Platform
- 4 x Arm ® Cortex ®-A53
- 32 KB I-cache
- 32 KB D-cache
- Arm NEON™
- FPU
- 512 KB L2 Cache (ECC)

Secondary Cores
- Tensilica® HiFi 4 DSP
- Cortex-M7
- 768 KB On-chip RAM (ECC)

Machine Learning
- Machine Learning Accelerator: 2.25 TOPS

Graphics
- 3D Graphics: GC7000UL
- 2D Graphics: GC520L

Video
- 1080p60 H.265, H.264, VP9, VP8 decoder
- 1080p60 H.265, H.264 encoder

Vision
- Camera ISP (2 x 187 MP/1 x 375 MP) dewarp
- 2 x MIPI-CSI (4-lane) with PHY

Display
- HDMI 2.0a Tx (eARC) with PHY
- MIPI-DSI (4-lane) with PHY
- 1 x LVDS Tx (4 or 8-lane) with PHY

Audio
- 18 x I²S TDM 32 bit at 768 kHz
- SP/DIF Tx and Rx
- eARC (HDMI)

- 1 x PCIe® Gen 3 – 1-lane L1 Substates
- 4 x UART 5 Mbit/s 5 x I²C, 3 x SPI

External Memory
- x16/x32 LPDDR4/DDR4/DDR3L (Inline ECC)
- 3 x SDIO3.0/MMC5.1
- Dual-ch. QuadSPI (XIP) or 1 x OctalSPI (XIP)
- NAND Controller (BCH62)

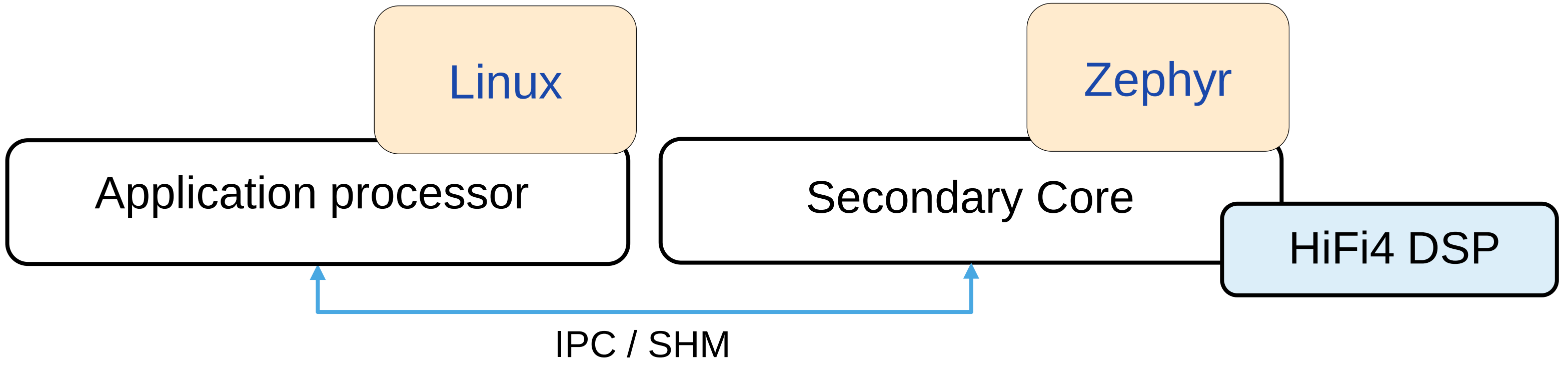
✓ Offloading signal processing

✓ Migrate neural network workload

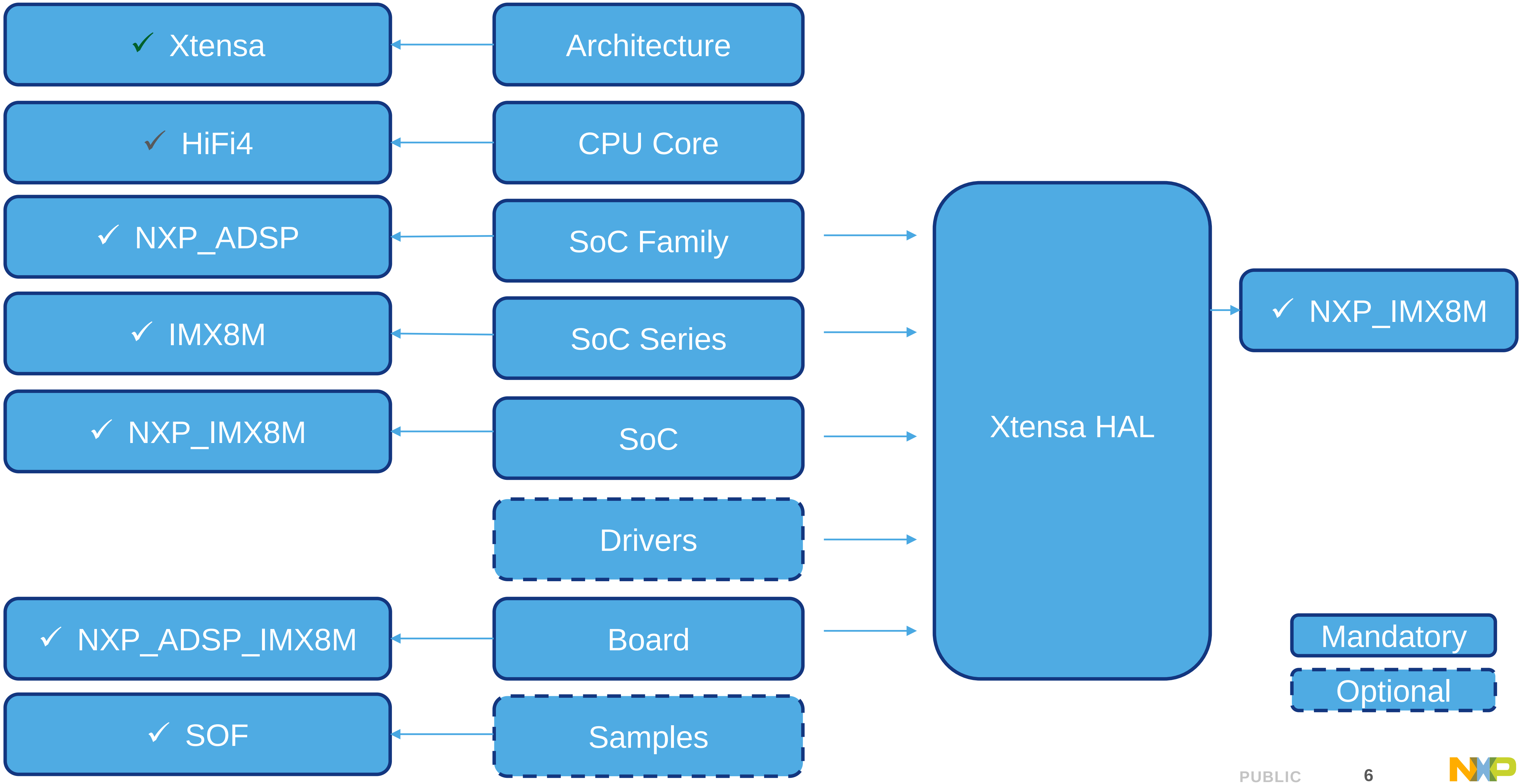
✓ Leverage 3rd party software libraries

APPLICATION PROCESSOR – HIFI4 DSP INTERACTION

- Application processor
  - ✓ loads DSP firmware
  - ✓ starts the DSP
- IPC – optional during various stages



HIFI4 DSP SUPPORT - CURRENT STATE



✓ Xtensa ← Architecture

✓ HiFi4 ← CPU Core

✓ NXP_ADSP ← SoC Family

✓ IMX8M ← SoC Series

✓ NXP_IMX8M ← SoC

Drivers

✓ NXP_ADSP_IMX8M ← Board

✓ SOF ← Samples

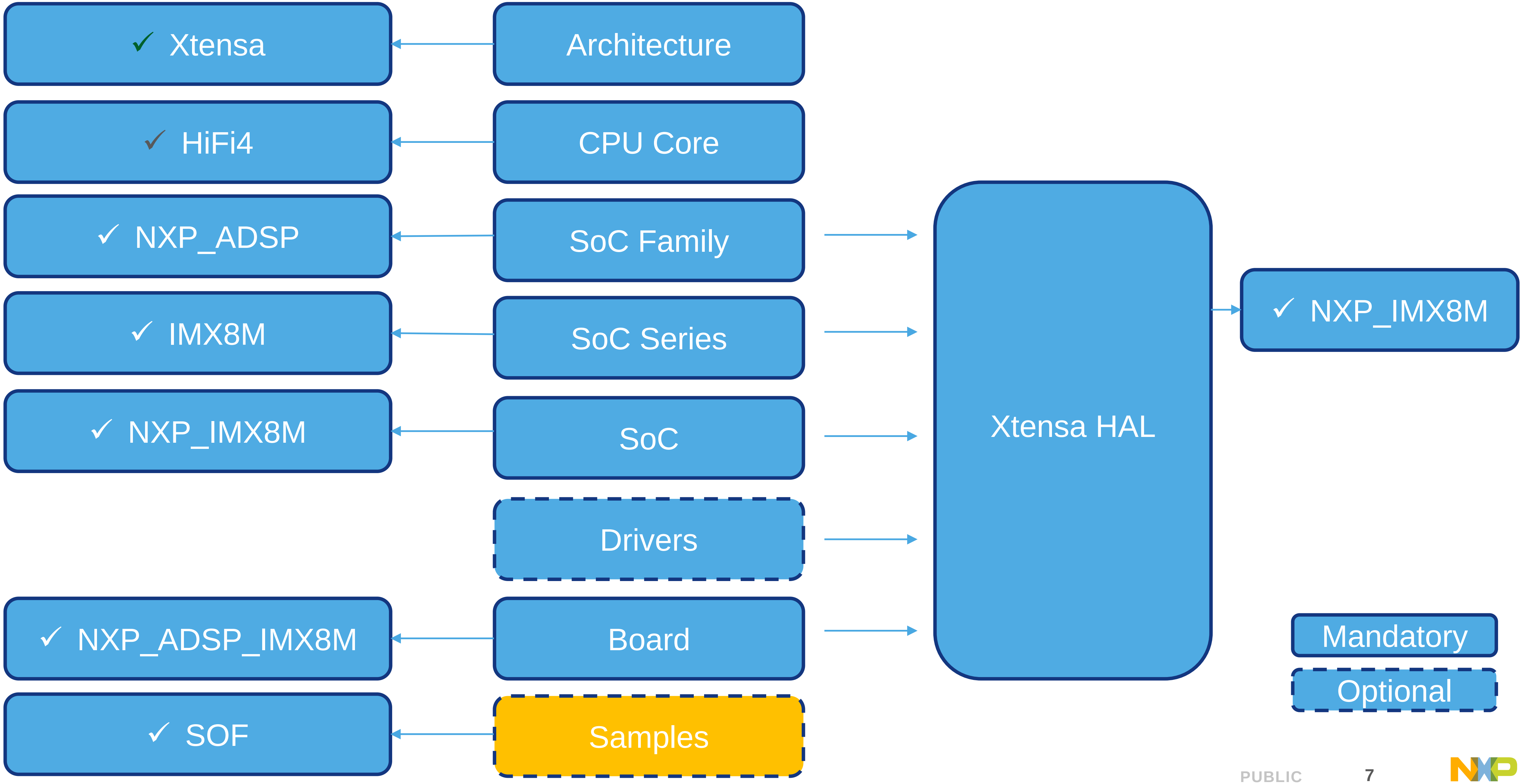
Xtensa HAL

✓ NXP_IMX8M

Mandatory

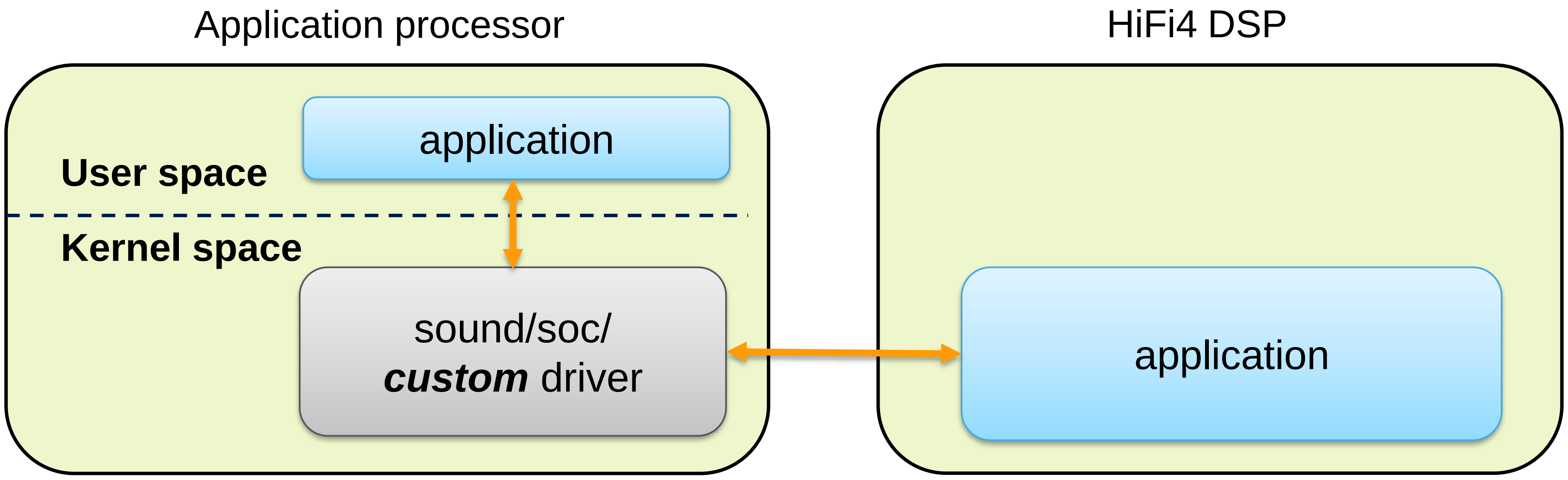
Optional

HIFI4 DSP SUPPORT - CURRENT STATE



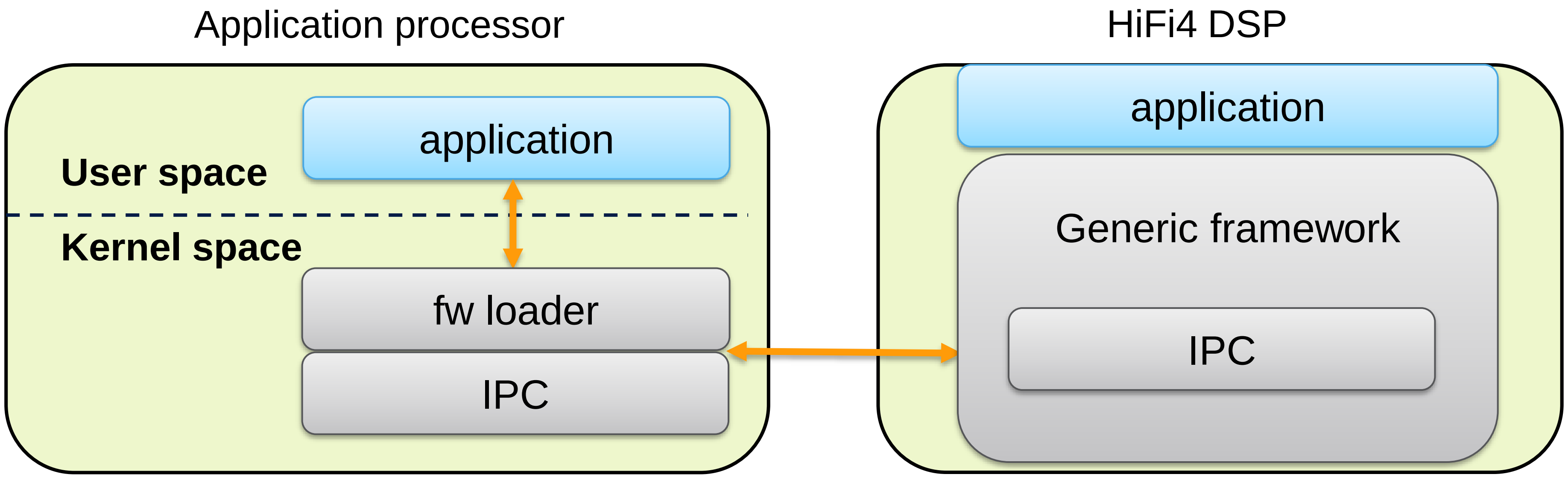
HIFI4 DSP SUPPORT - CURRENT STATE

- Sound Open Firmware (SOF) with Zephyr
- Custom fw loader & IPC – SOF specific APIs



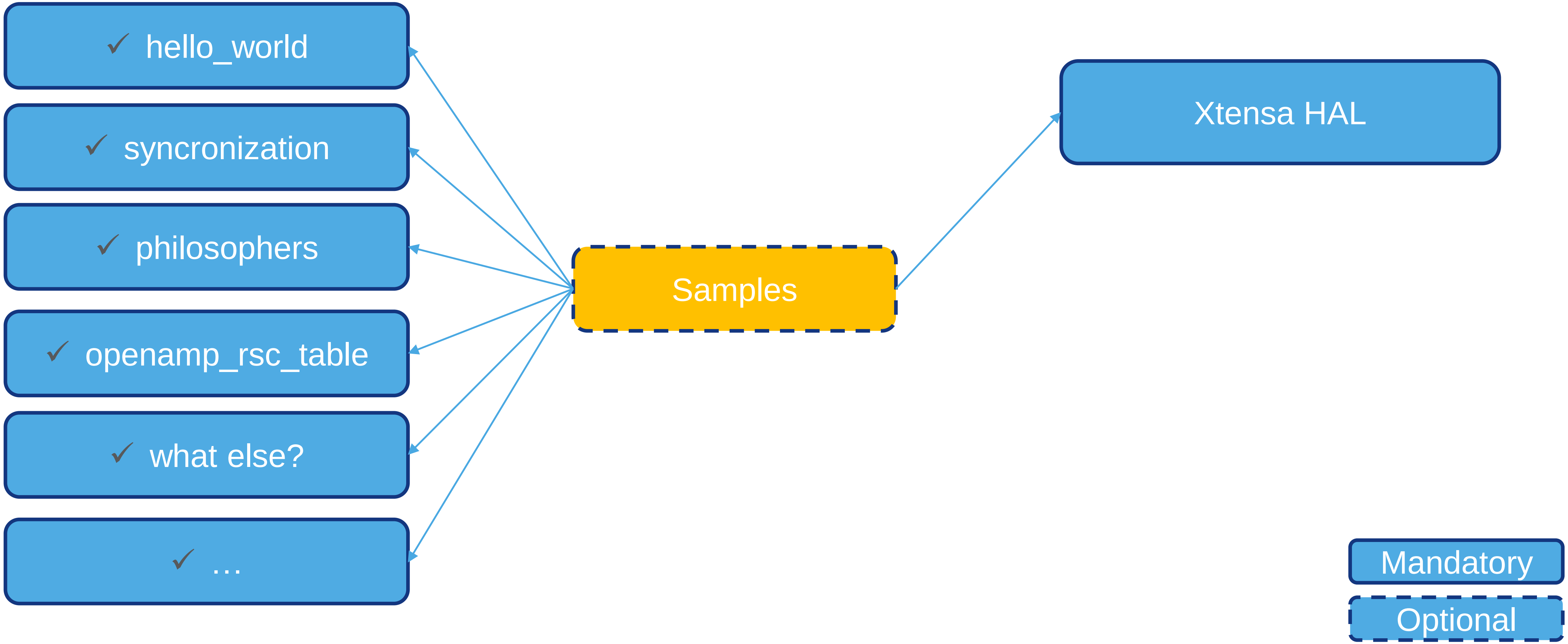
HIFI4 DSP SUPPORT – WHAT'S NEXT

- Generic fw loader & IPC
- Harness DSP power processing

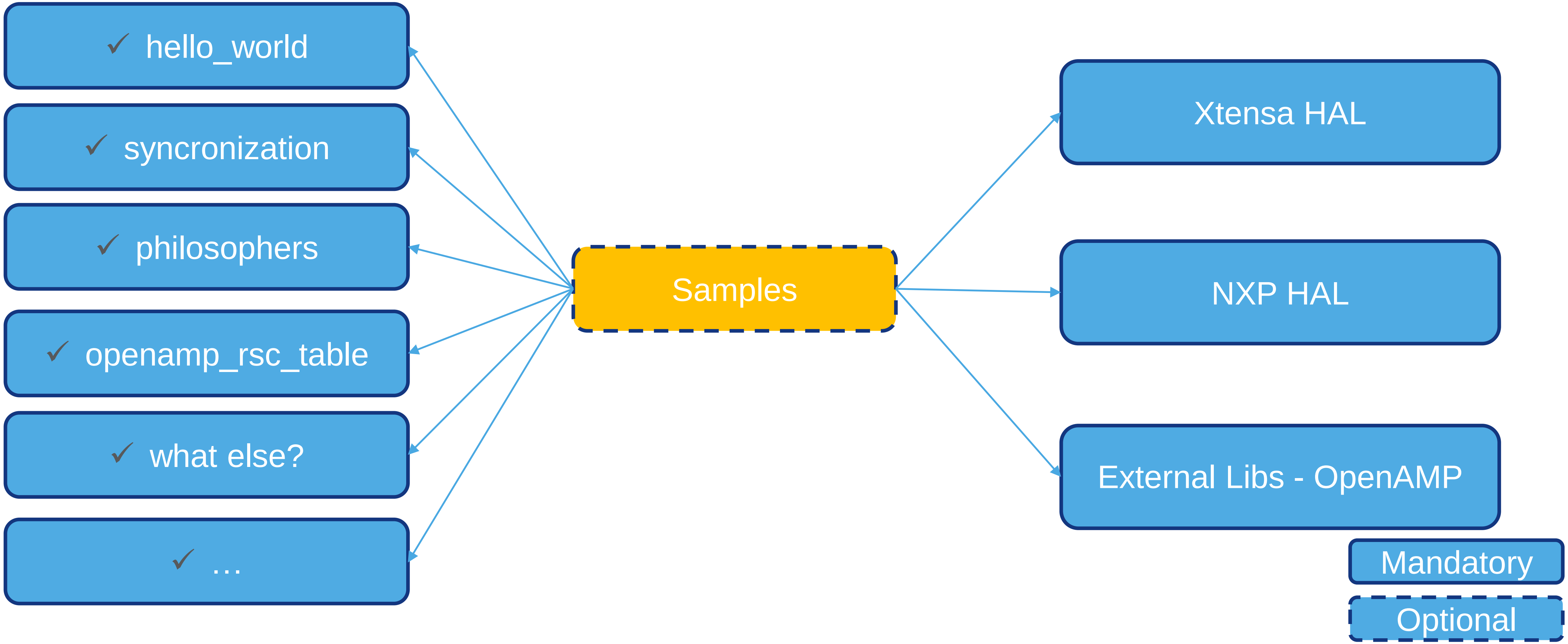


HIFI4 DSP SUPPORT – WHAT'S NEXT

- Generic fw loader & IPC

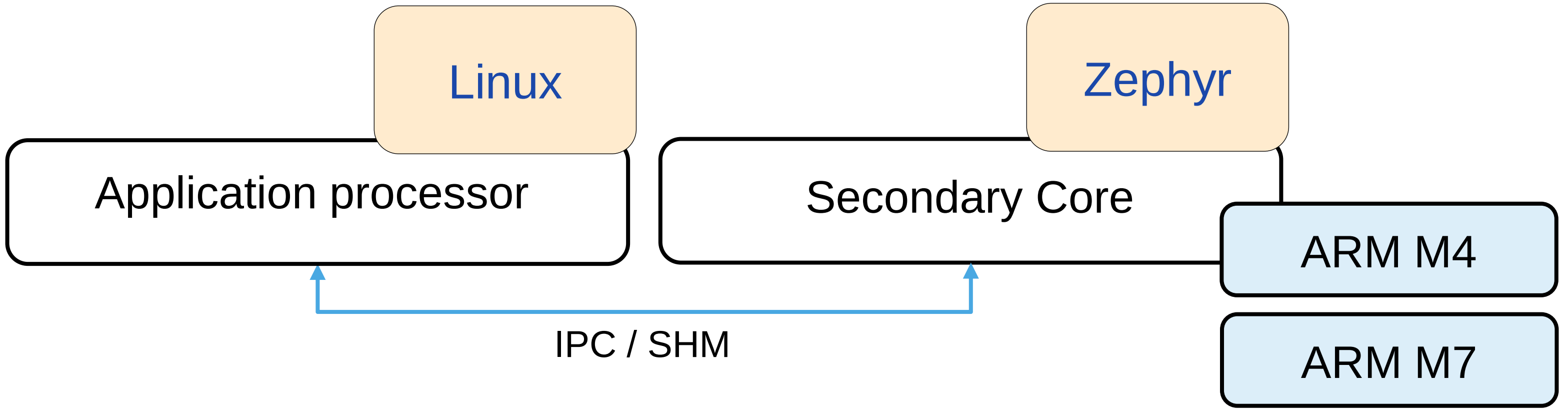


HIFI4 DSP SUPPORT – WHAT'S NEXT



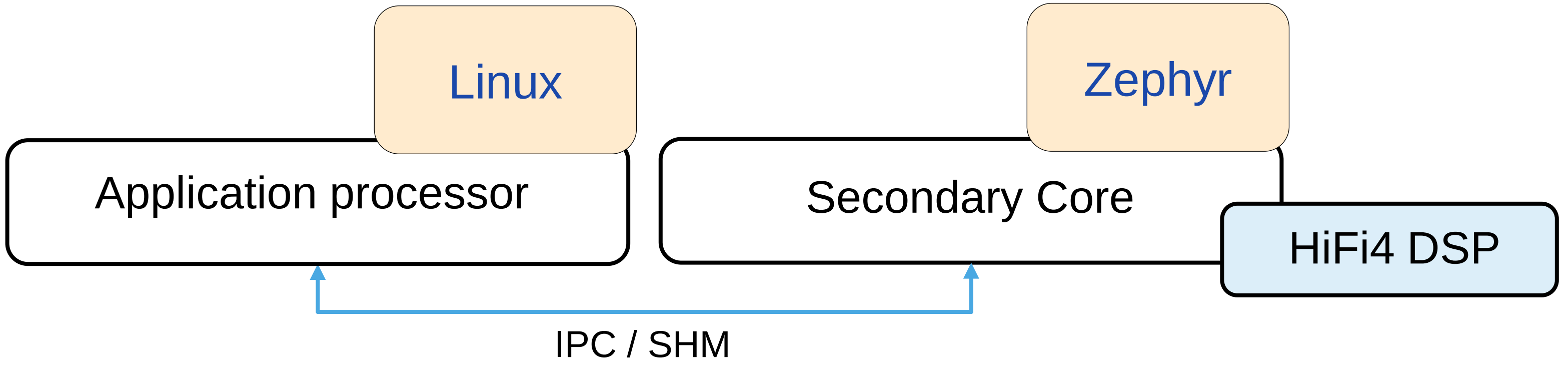
GENERIC LINUX AND ZEPHYR COMMUNICATION SETUP

- An Introduction to Asymmetric Multiprocessing: When this Architecture can be a Game Changer and How to Survive It - Nicola La Gloria & Laura Nao, Kynetics - ELC 2018 - slides
- Linux and Zephyr “talking” to each other in the same SoC - Diego Sueiro, Sepura/Embarcados, ELCE 2018 - slides
- Asymmetric Multiprocessing and Embedded Linux - Marek Novak & Dušan Červenka, NXP Semiconductor - ELCE 2017 – slides
- Asymmetric Multi Processing with Linux & Zephyr - Arnaud Ferraris, Collabora - link



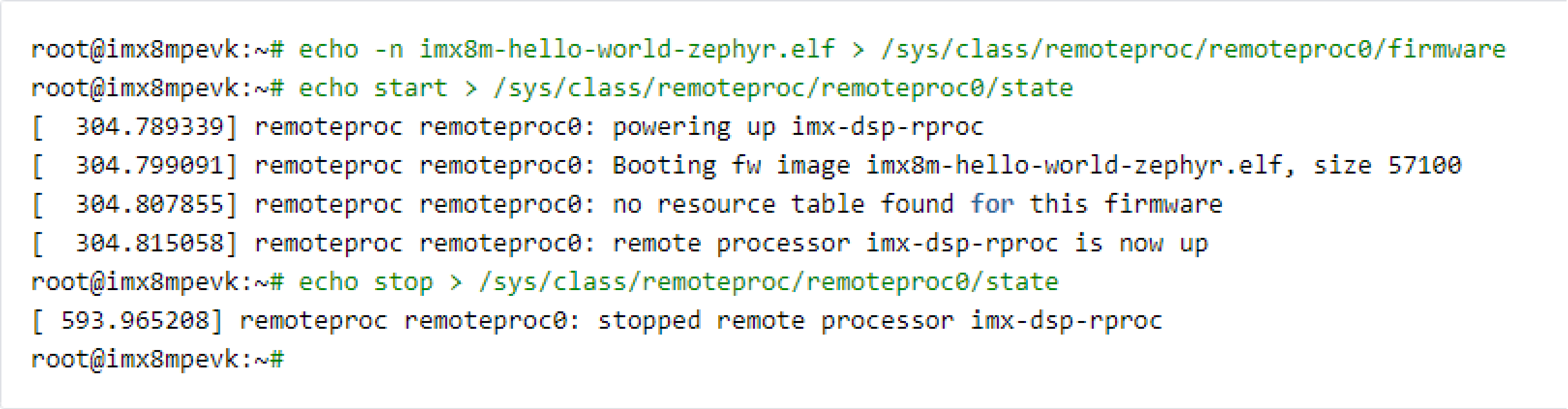
GENERIC LINUX AND ZEPHYR COMMUNICATION SETUP

- Generic vs specific
- How is the application loaded and started on DSP?
  - remoteproc
- How does the cores communicate with one-another?
  - remote processor messaging (rpmsg)
  - mailbox
  - OpenAMP



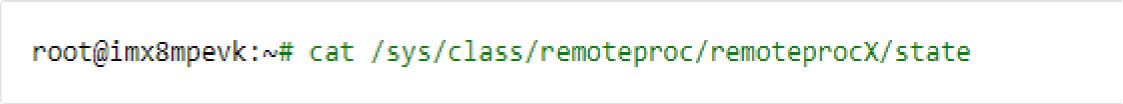
REMOTEPROC

- Framework that allows the different platforms/architectures to control (power on, load firmware, power off) remote processors

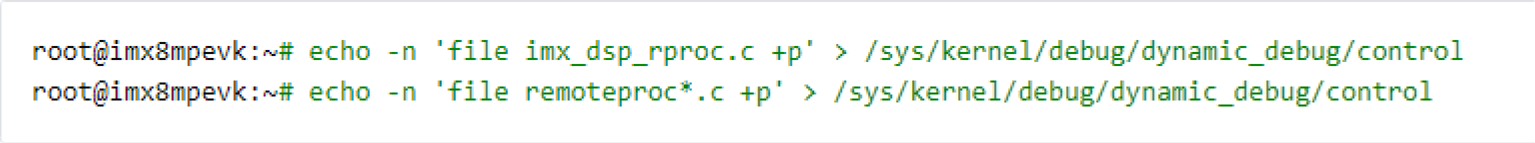


```
root@imx8mpevk:~# echo -n imx8m-hello-world-zephyr.elf > /sys/class/remoteproc/remoteproc0/firmware
root@imx8mpevk:~# echo start > /sys/class/remoteproc/remoteproc0/state
[  304.789339] remoteproc remoteproc0: powering up imx-dsp-rproc
[  304.799091] remoteproc remoteproc0: Booting fw image imx8m-hello-world-zephyr.elf, size 57100
[  304.807855] remoteproc remoteproc0: no resource table found for this firmware
[  304.815058] remoteproc remoteproc0: remote processor imx-dsp-rproc is now up
root@imx8mpevk:~# echo stop > /sys/class/remoteproc/remoteproc0/state
[ 593.965208] remoteproc remoteproc0: stopped remote processor imx-dsp-rproc
root@imx8mpevk:~#
```

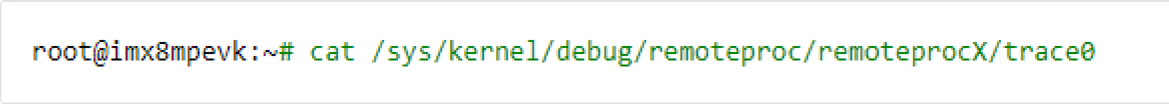
- Offers services to monitor and debug the remote processor



```
root@imx8mpevk:~# cat /sys/class/remoteproc/remoteprocX/state
```

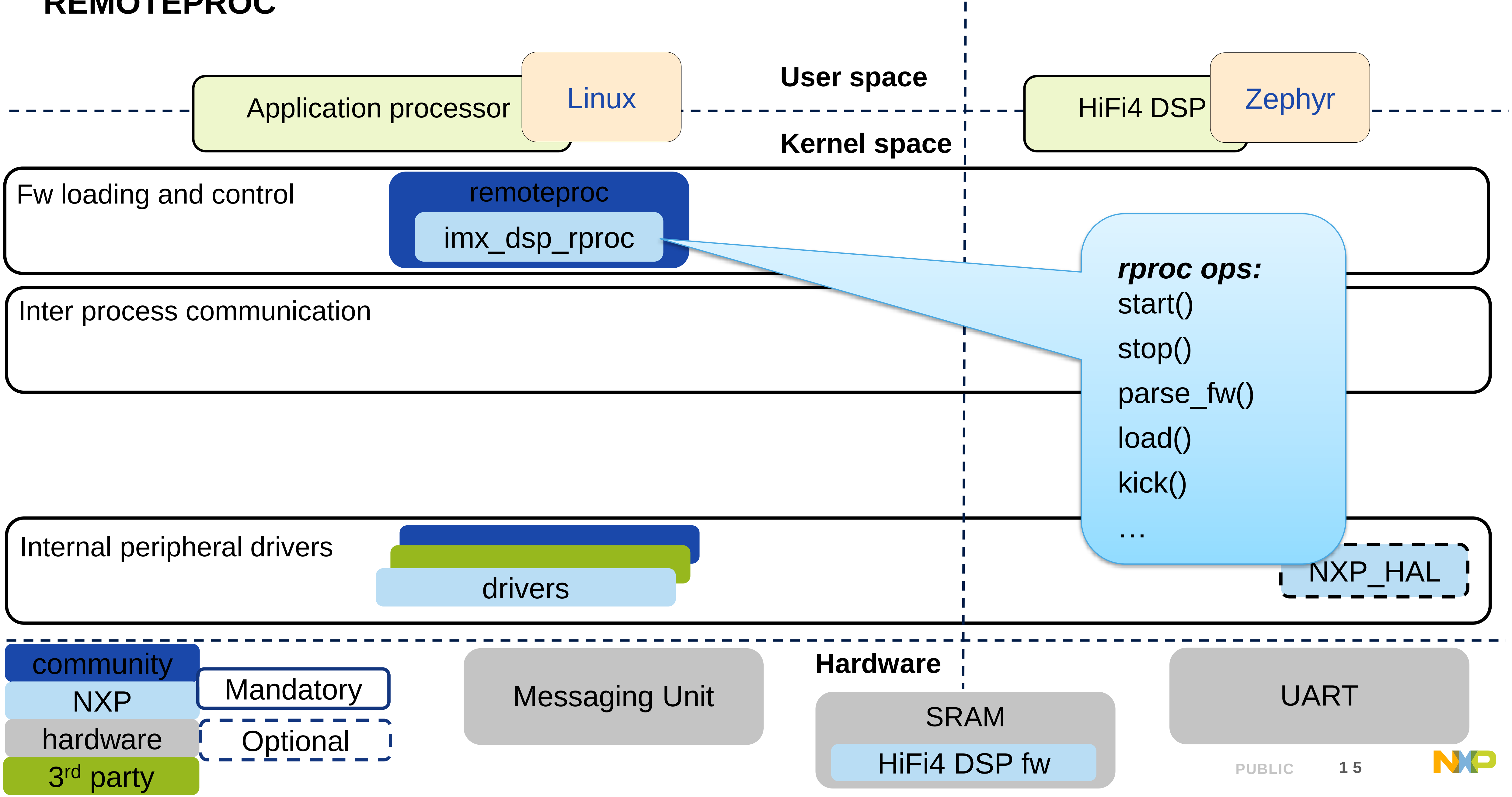


```
root@imx8mpevk:~# echo -n 'file imx_dsp_rproc.c +p' > /sys/kernel/debug/dynamic_debug/control
root@imx8mpevk:~# echo -n 'file remoteproc*.c +p' > /sys/kernel/debug/dynamic_debug/control
```



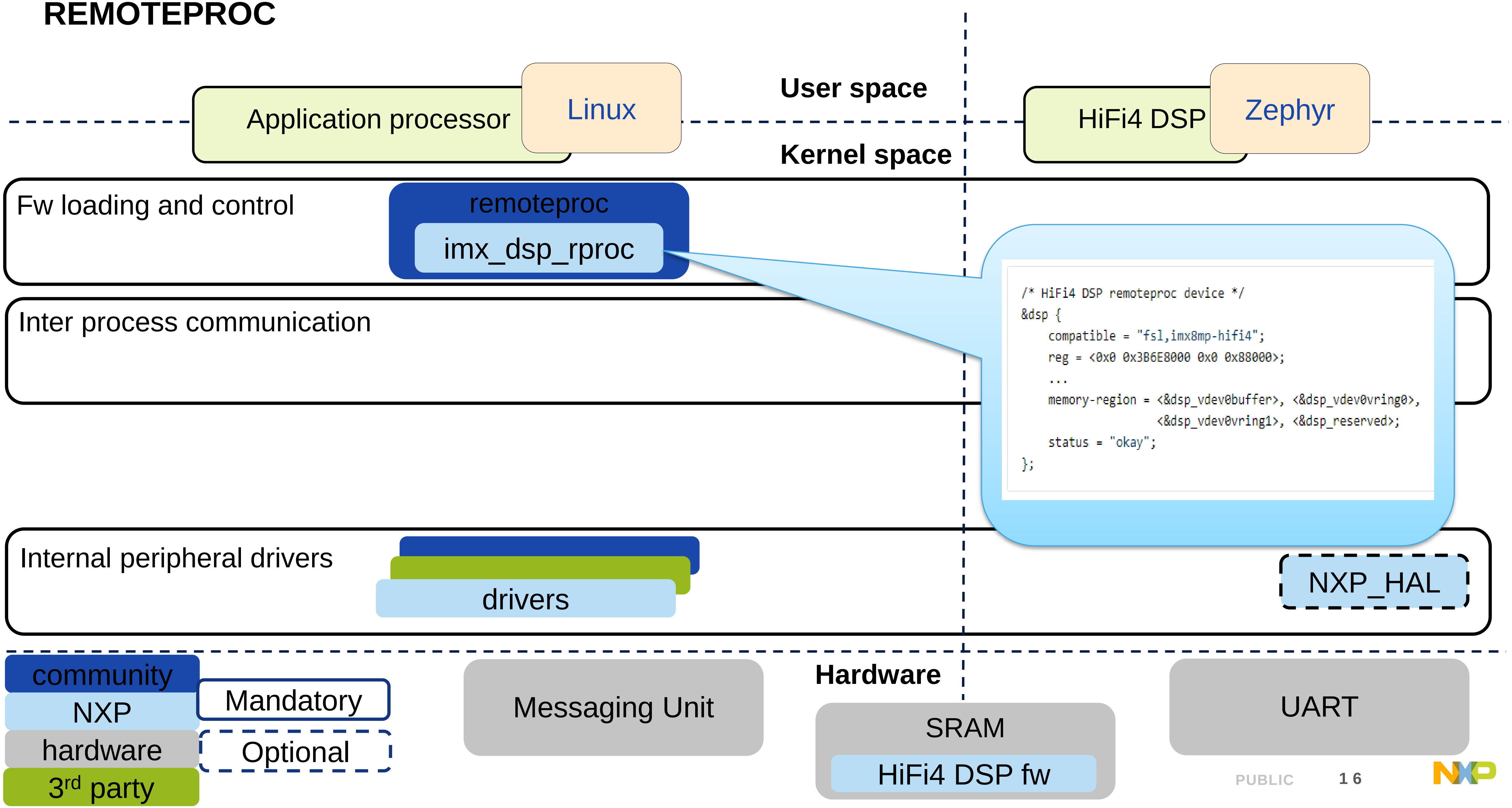
```
root@imx8mpevk:~# cat /sys/kernel/debug/remoteproc/remoteprocX/trace0
```

REMOTEPROC



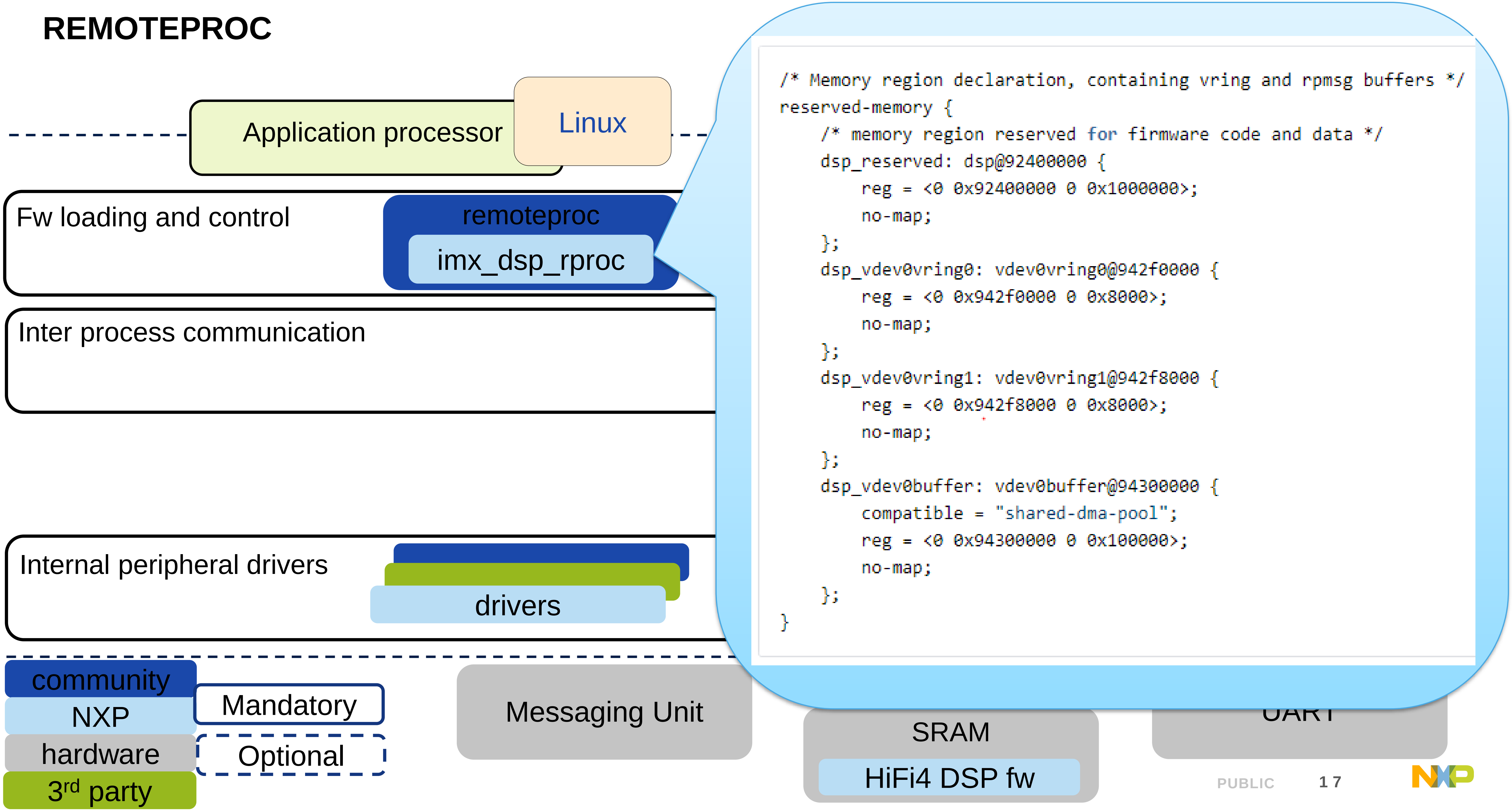
User space
Kernel space
Application processor
Linux
HiFi4 DSP
Zephyr
Fw loading and control
remoteproc
imx_dsp_rproc
Inter process communication
Internal peripheral drivers
drivers
NXP_HAL
rproc ops:
start()
stop()
parse_fw()
load()
kick()
…
Hardware
Messaging Unit
SRAM
HiFi4 DSP fw
UART
community
NXP
hardware
3rd party
Mandatory
Optional

REMOTEPROC



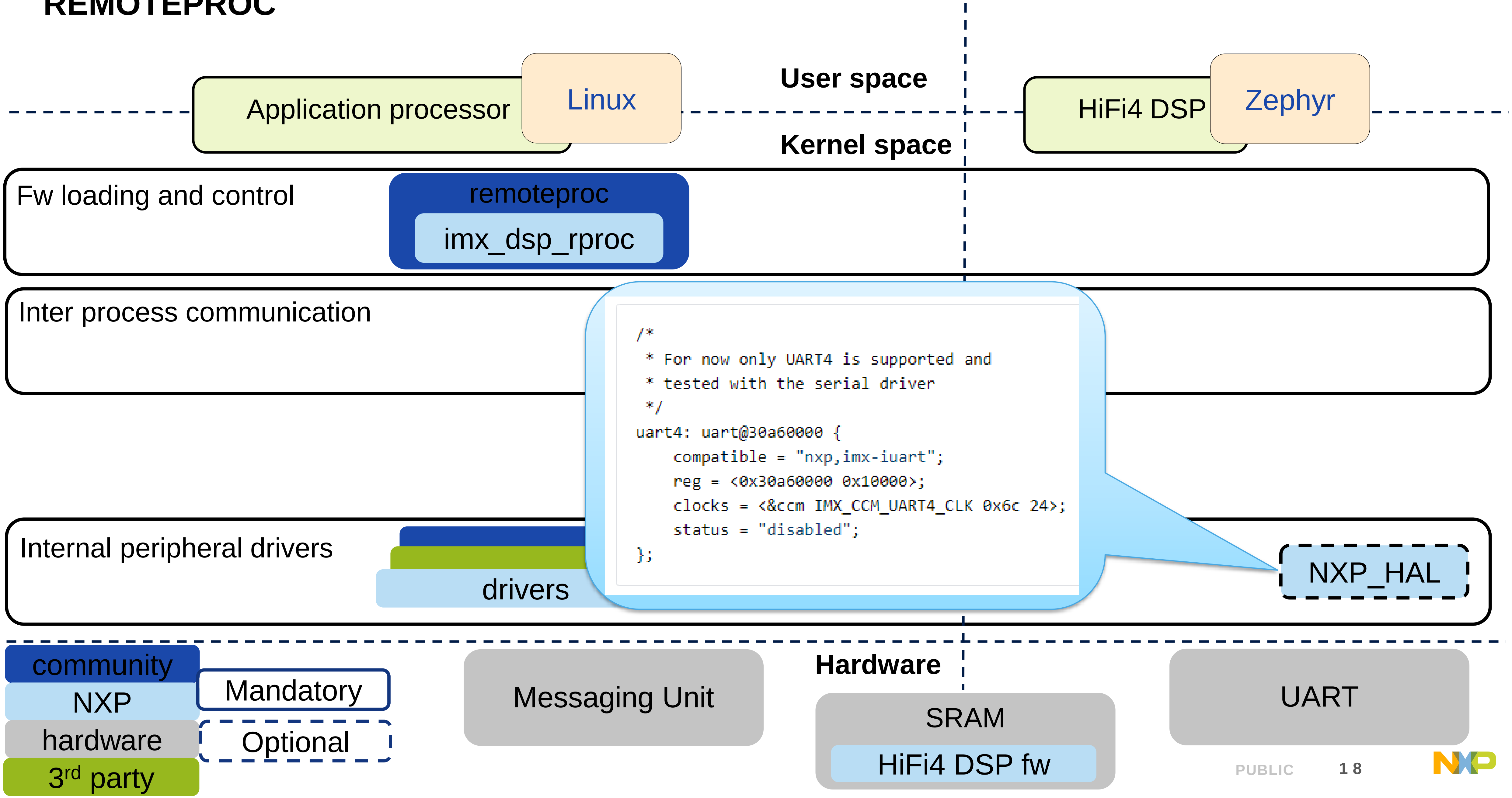
```
/* HiFi4 DSP remoteproc device */
&dsp {
    compatible = "fsl,imx8mp-hifi4";
    reg = <0x0 0x3B6E8000 0x0 0x88000>;
    ...
    memory-region = <&dsp_vdev0buffer>, <&dsp_vdev0vring0>,
                    <&dsp_vdev0vring1>, <&dsp_reserved>;
    status = "okay";
};
```

REMOTEPROC



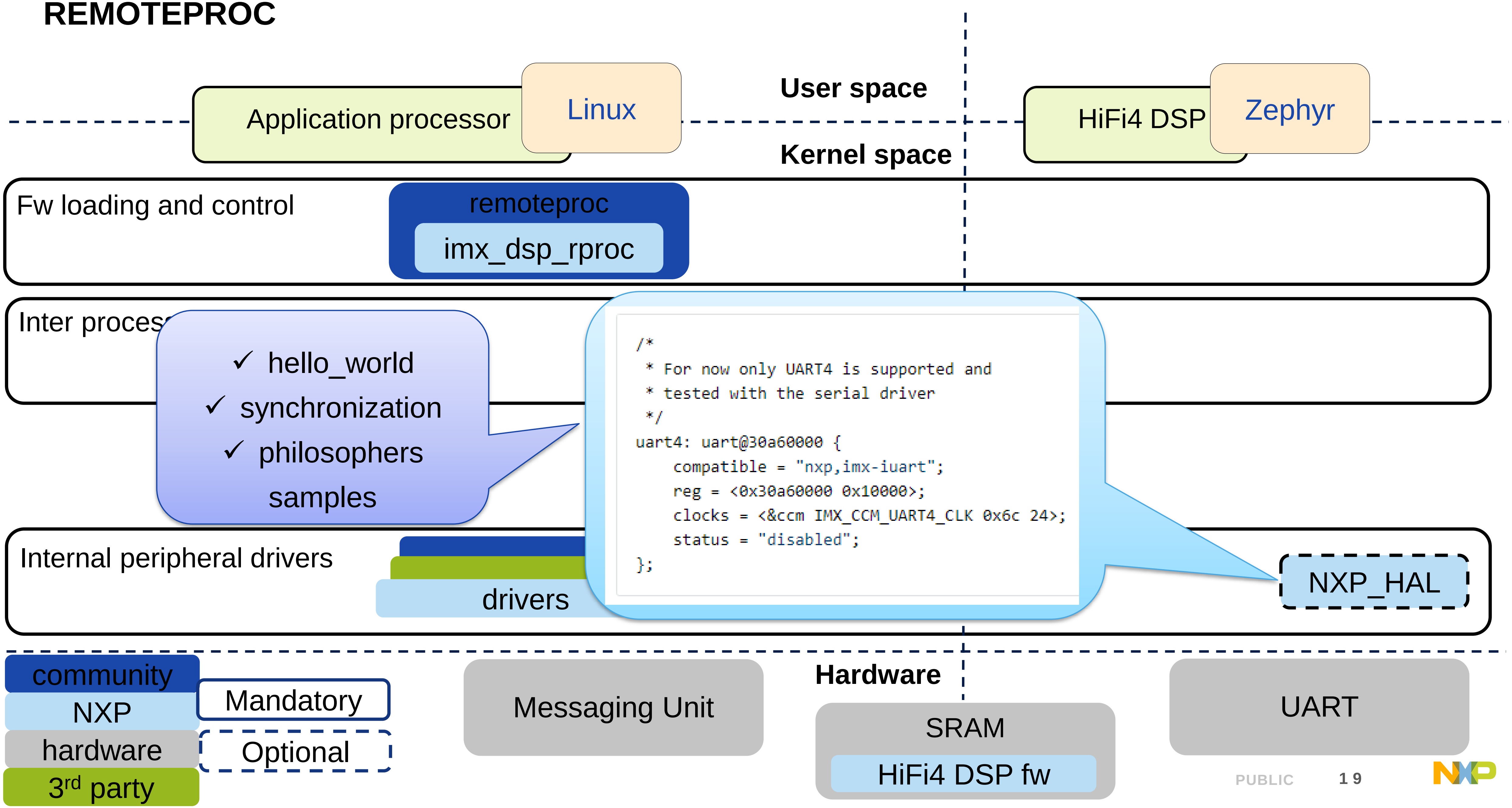
```
/* Memory region declaration, containing vring and rpmsg buffers */
reserved-memory {
    /* memory region reserved for firmware code and data */
    dsp_reserved: dsp@92400000 {
        reg = <0 0x92400000 0 0x1000000>;
        no-map;
    };
    dsp_vdev0vring0: vdev0vring0@942f0000 {
        reg = <0 0x942f0000 0 0x8000>;
        no-map;
    };
    dsp_vdev0vring1: vdev0vring1@942f8000 {
        reg = <0 0x942f8000 0 0x8000>;
        no-map;
    };
    dsp_vdev0buffer: vdev0buffer@94300000 {
        compatible = "shared-dma-pool";
        reg = <0 0x94300000 0 0x100000>;
        no-map;
    };
}
```

REMOTEPROC



```
/*
 * For now only UART4 is supported and
 * tested with the serial driver
 */
uart4: uart@30a60000 {
    compatible = "nxp,imx-iuart";
    reg = <0x30a60000 0x10000>;
    clocks = <&ccm IMX_CCM_UART4_CLK 0x6c 24>;
    status = "disabled";
};
```

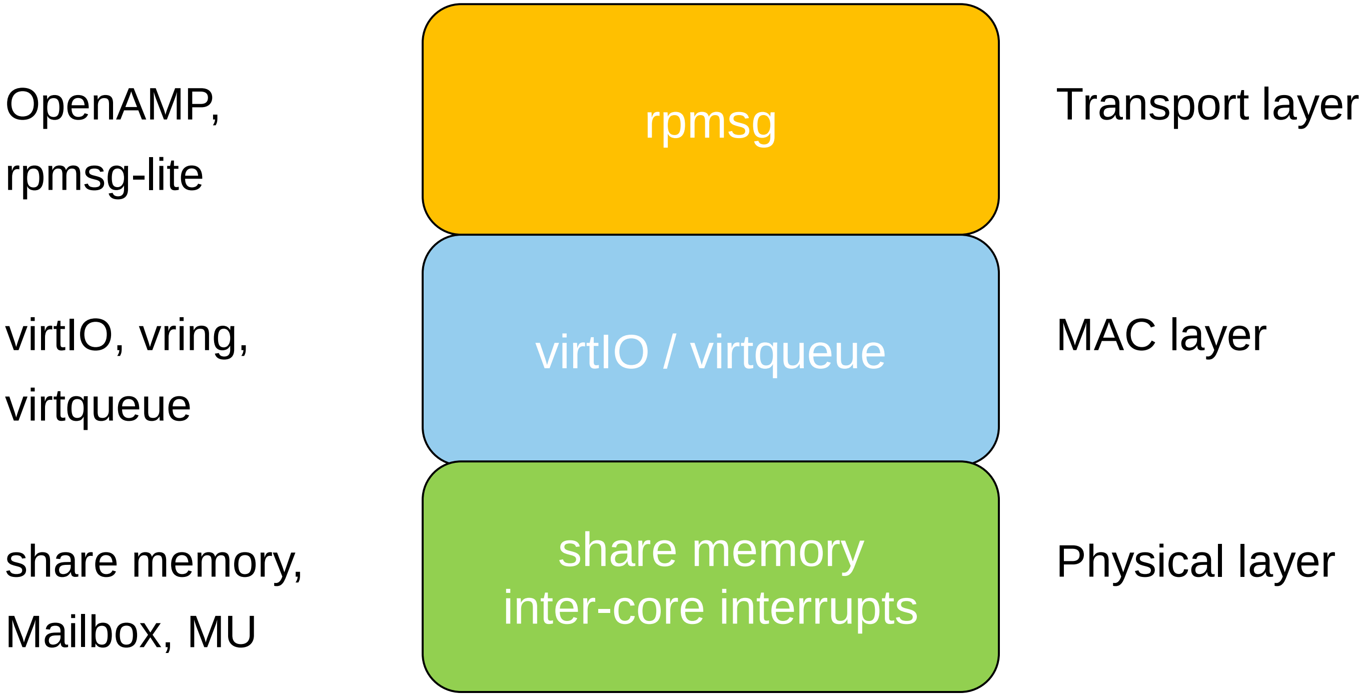
REMOTEPROC



```
/*
 * For now only UART4 is supported and
 * tested with the serial driver
 */
uart4: uart@30a60000 {
    compatible = "nxp,imx-iuart";
    reg = <0x30a60000 0x10000>;
    clocks = <&ccm IMX_CCM_UART4_CLK 0x6c 24>;
    status = "disabled";
};
```

RPMSG

- A virtio-based messaging bus that allows kernel drivers to communicate with remote processors
- Cores cooperate using a shared memory-based communication



OpenAMP,
rpmsg-lite

rpmsg

Transport layer

virtIO, vring,
virtqueue

virtIO / virtqueue

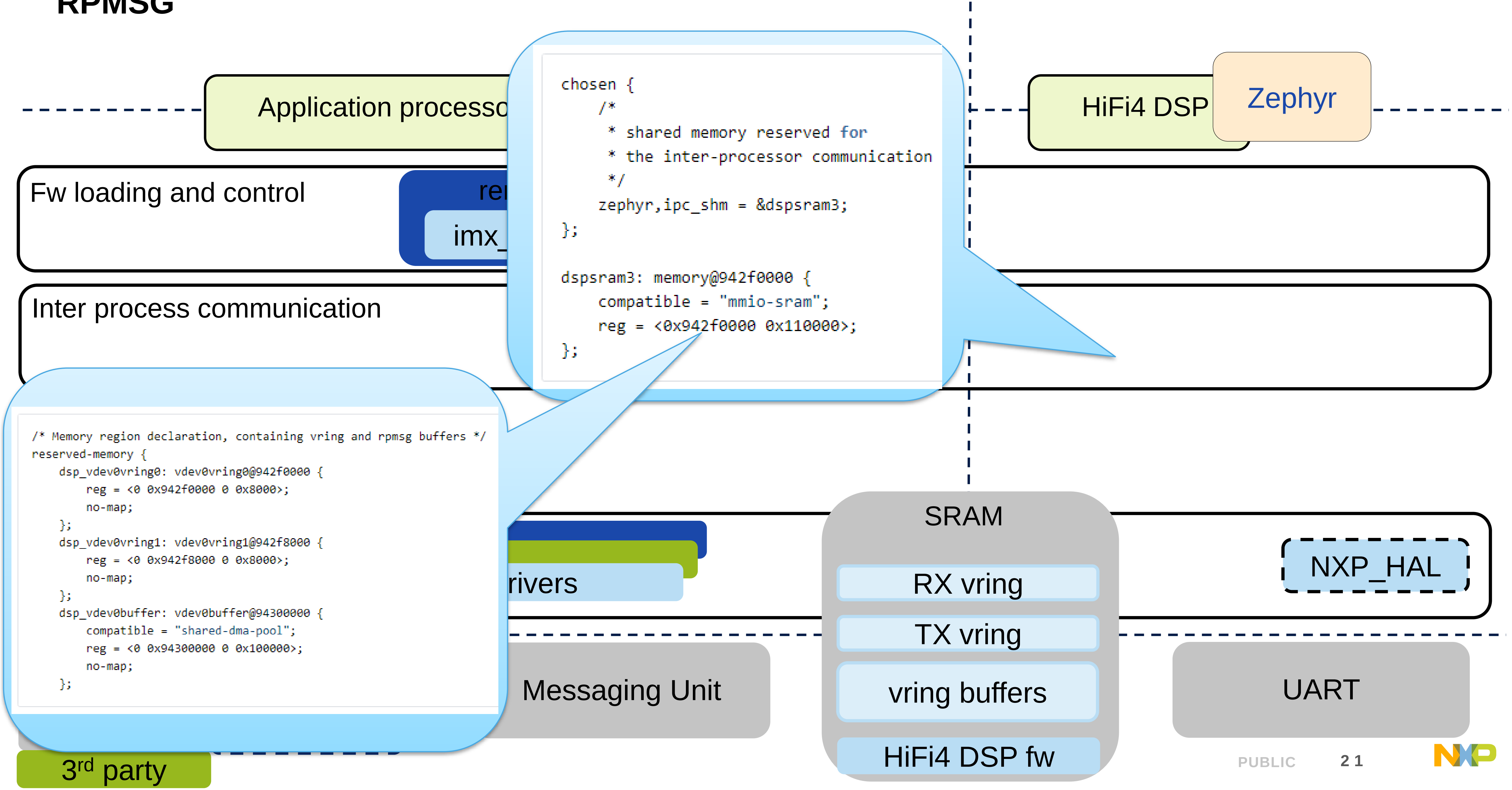
MAC layer

share memory,
Mailbox, MU

share memory
inter-core interrupts

Physical layer

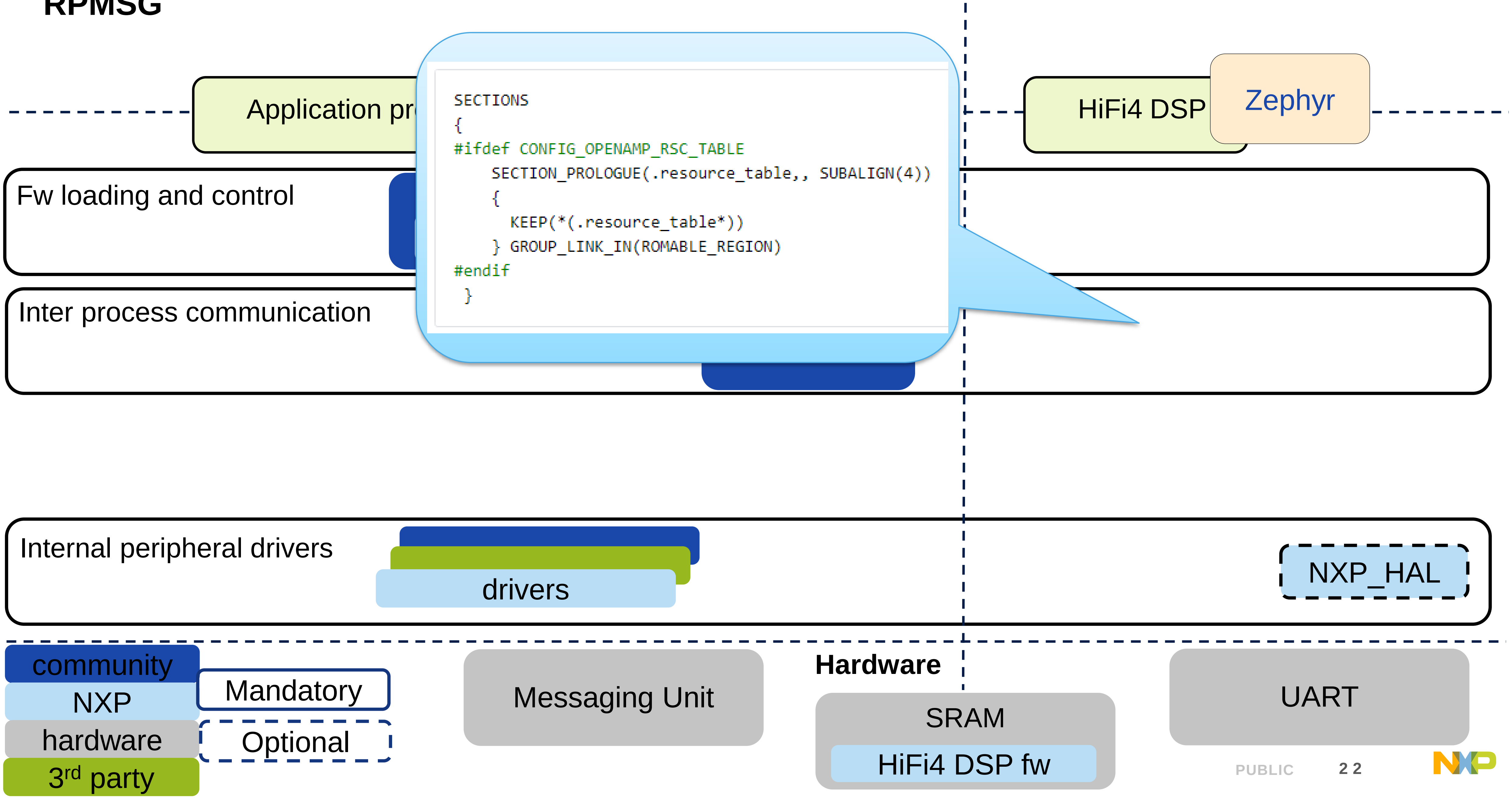
RPMSG



```
chosen {
    /*
     * shared memory reserved for
     * the inter-processor communication
     */
    zephyr,ipc_shm = &dspsram3;
};

dspsram3: memory@942f0000 {
    compatible = "mmio-sram";
    reg = <0x942f0000 0x110000>;
};
```

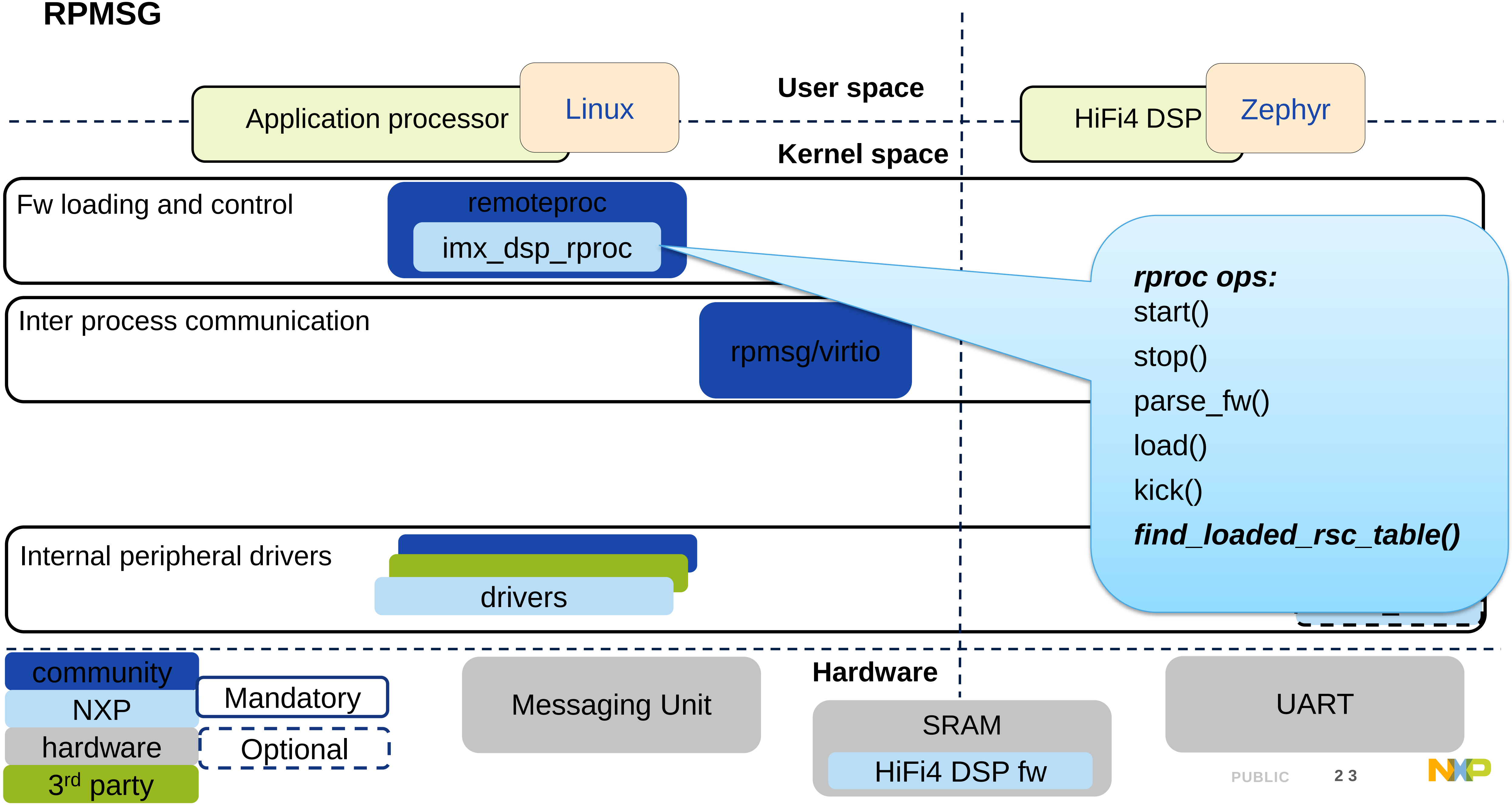
```
/* Memory region declaration, containing vring and rpmsg buffers */
reserved-memory {
    dsp_vdev0vring0: vdev0vring0@942f0000 {
        reg = <0 0x942f0000 0 0x8000>;
        no-map;
    };
    dsp_vdev0vring1: vdev0vring1@942f8000 {
        reg = <0 0x942f8000 0 0x8000>;
        no-map;
    };
    dsp_vdev0buffer: vdev0buffer@94300000 {
        compatible = "shared-dma-pool";
        reg = <0 0x94300000 0 0x100000>;
        no-map;
    };
```

RPMSG



```
SECTIONS
{
#ifdef CONFIG_OPENAMP_RSC_TABLE
    SECTION_PROLOGUE(.resource_table,, SUBALIGN(4))
    {
      KEEP(*(.resource_table*))
    } GROUP_LINK_IN(ROMABLE_REGION)
#endif
 }
```

RPMSG



User space

Kernel space

Application processor

Linux

HiFi4 DSP

Zephyr

Fw loading and control

remoteproc

imx_dsp_rproc

Inter process communication

rpmsg/virtio

Internal peripheral drivers

drivers

Hardware

Messaging Unit

SRAM

HiFi4 DSP fw

UART

community

NXP

hardware

3rd party

Mandatory

Optional

rproc ops:

start()

stop()

parse_fw()

load()

kick()

find_loaded_rsc_table()

MAILBOX

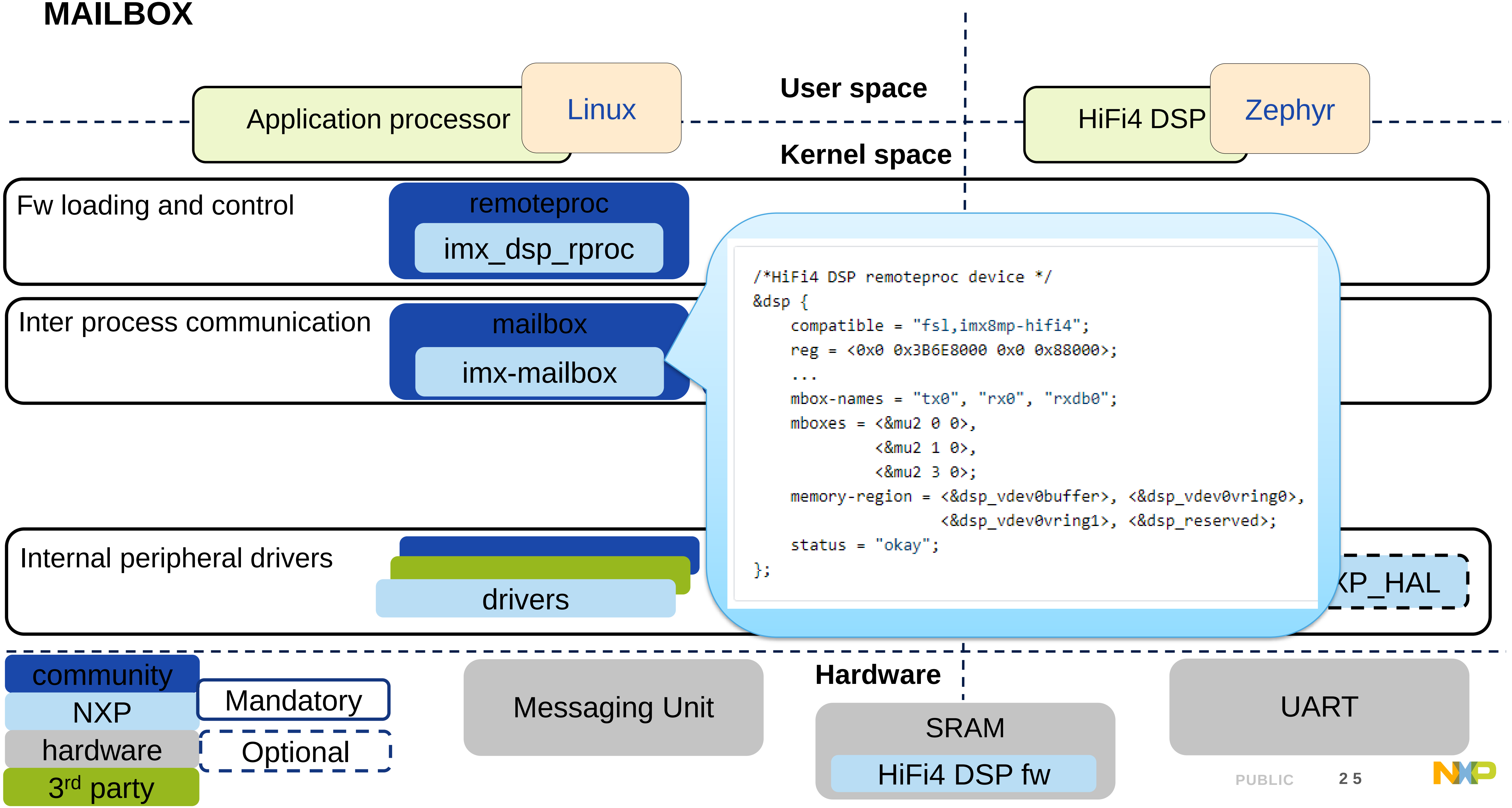
- Framework used in inter-processor communication to exchange messages or signals between the host and the remote processor
- Based on:
  - *mailbox controller*

  that is platform dependent;
  - *mailbox client*

  that oversees the message

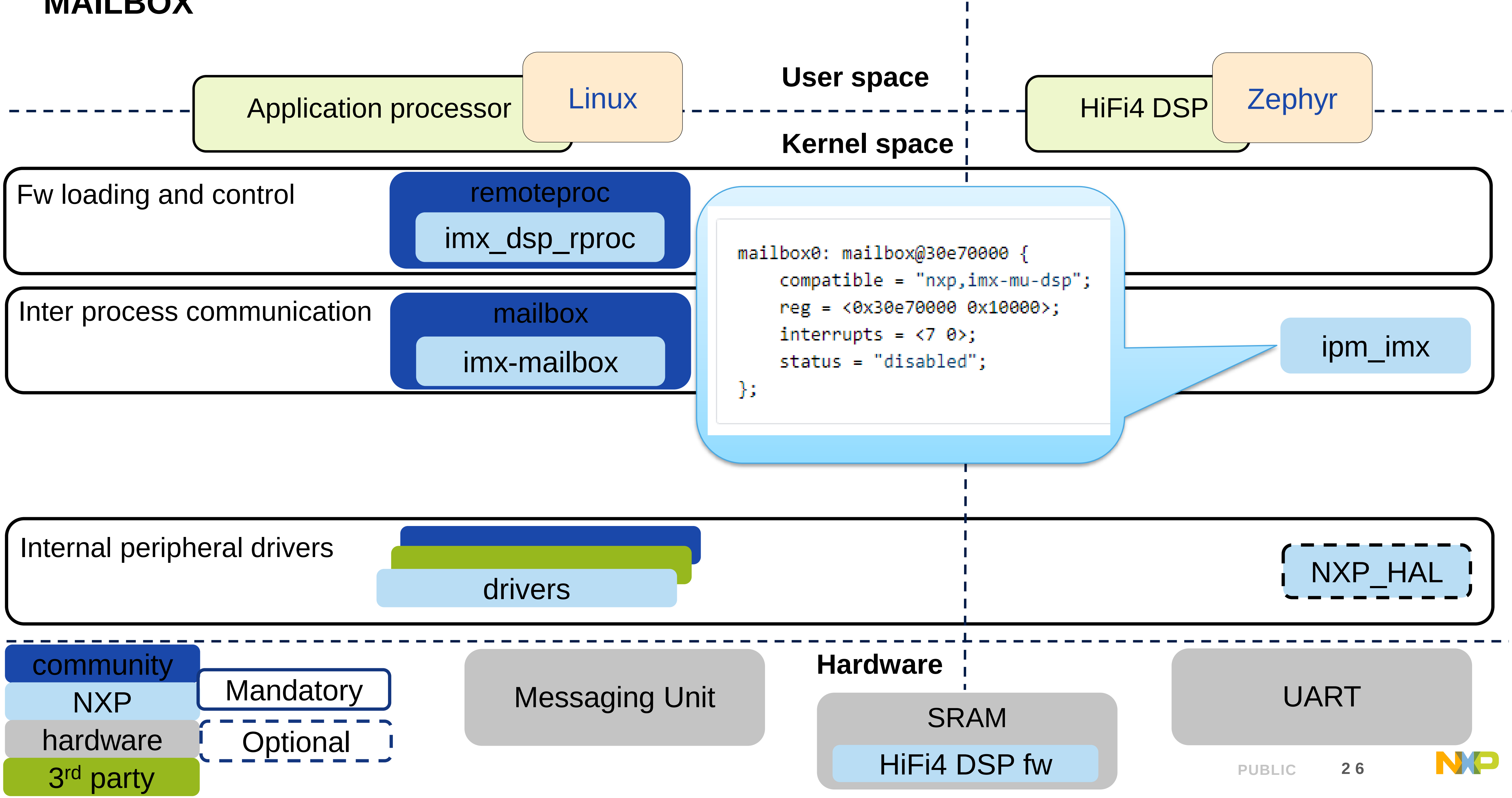
  send/receive.

MAILBOX



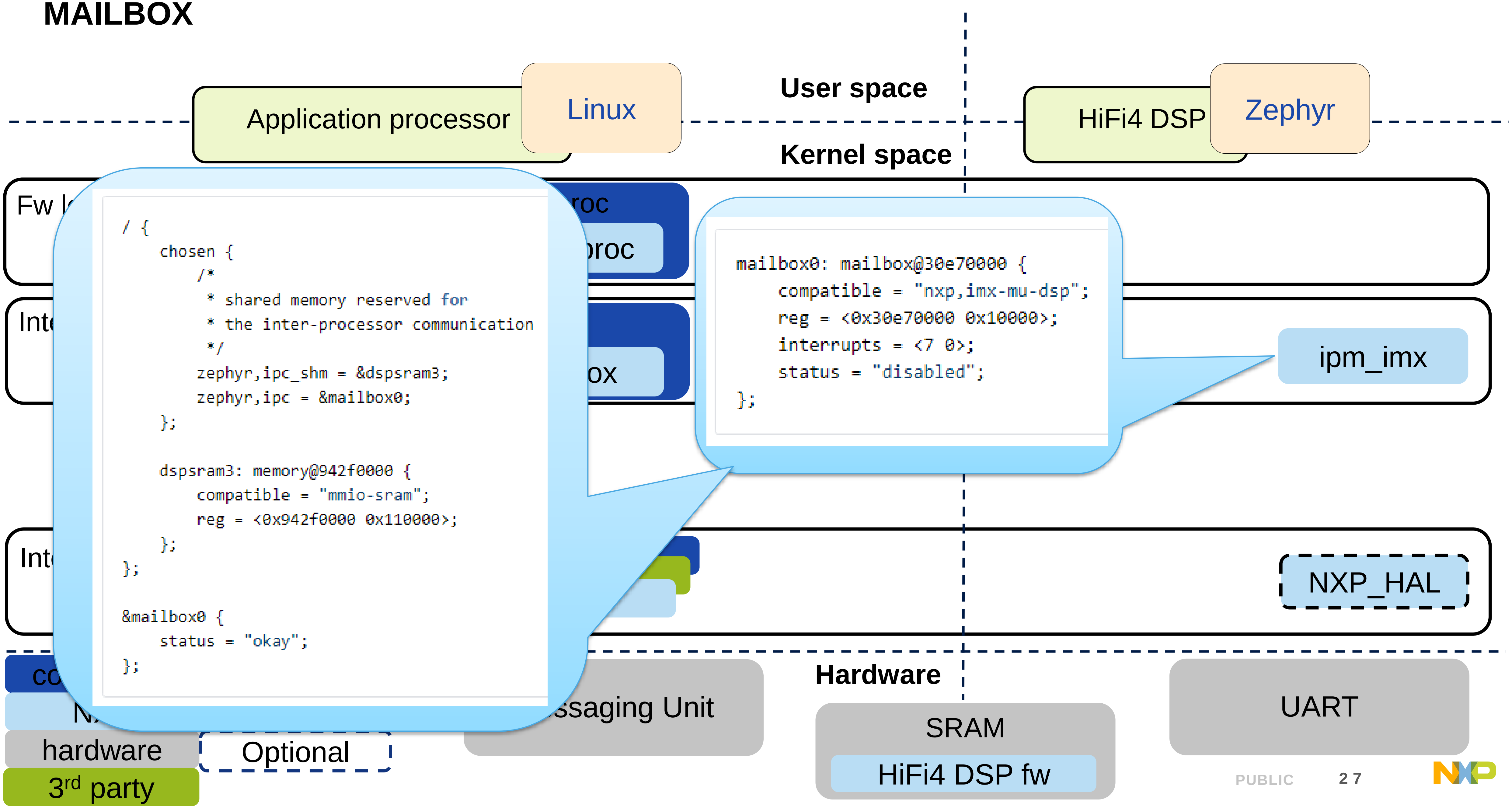
```
/*HiFi4 DSP remoteproc device */
&dsp {
    compatible = "fsl,imx8mp-hifi4";
    reg = <0x0 0x3B6E8000 0x0 0x88000>;
    ...
    mbox-names = "tx0", "rx0", "rxdb0";
    mboxes = <&mu2 0 0>,
             <&mu2 1 0>,
             <&mu2 3 0>;
    memory-region = <&dsp_vdev0buffer>, <&dsp_vdev0vring0>,
                    <&dsp_vdev0vring1>, <&dsp_reserved>;
    status = "okay";
};
```

MAILBOX



```
mailbox0: mailbox@30e70000 {
    compatible = "nxp,imx-mu-dsp";
    reg = <0x30e70000 0x10000>;
    interrupts = <7 0>;
    status = "disabled";
};
```

MAILBOX

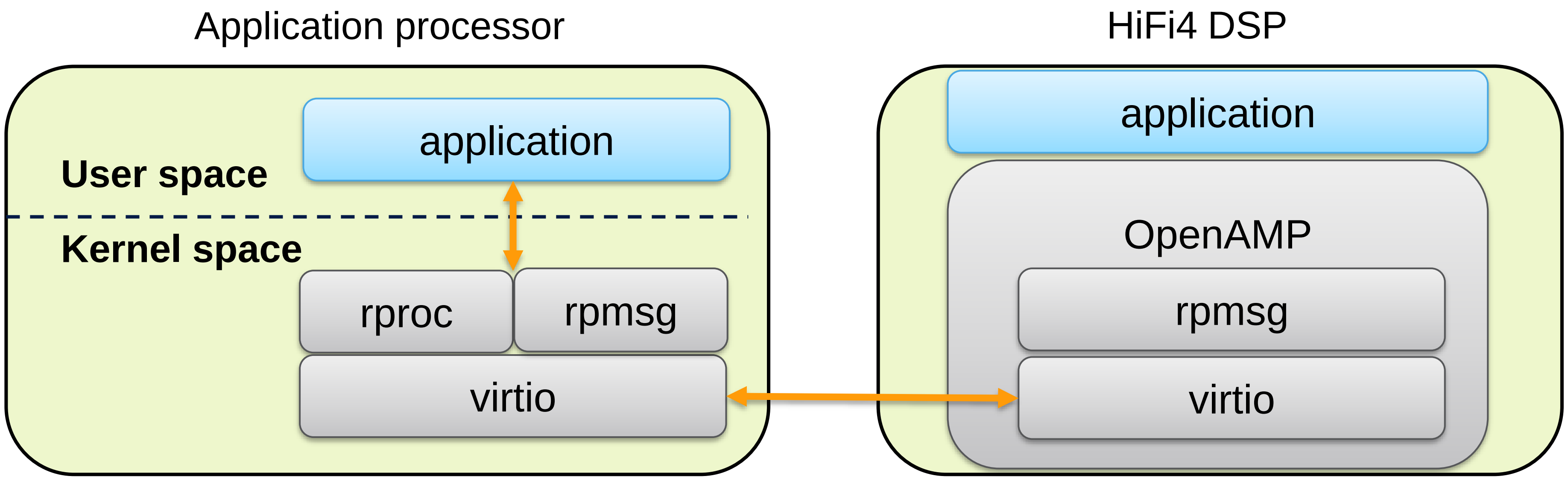


```
/ {
    chosen {
        /*
         * shared memory reserved for
         * the inter-processor communication
         */
        zephyr,ipc_shm = &dspsram3;
        zephyr,ipc = &mailbox0;
    };

    dspsram3: memory@942f0000 {
        compatible = "mmio-sram";
        reg = <0x942f0000 0x110000>;
    };
};

&mailbox0 {
    status = "okay";
};
```

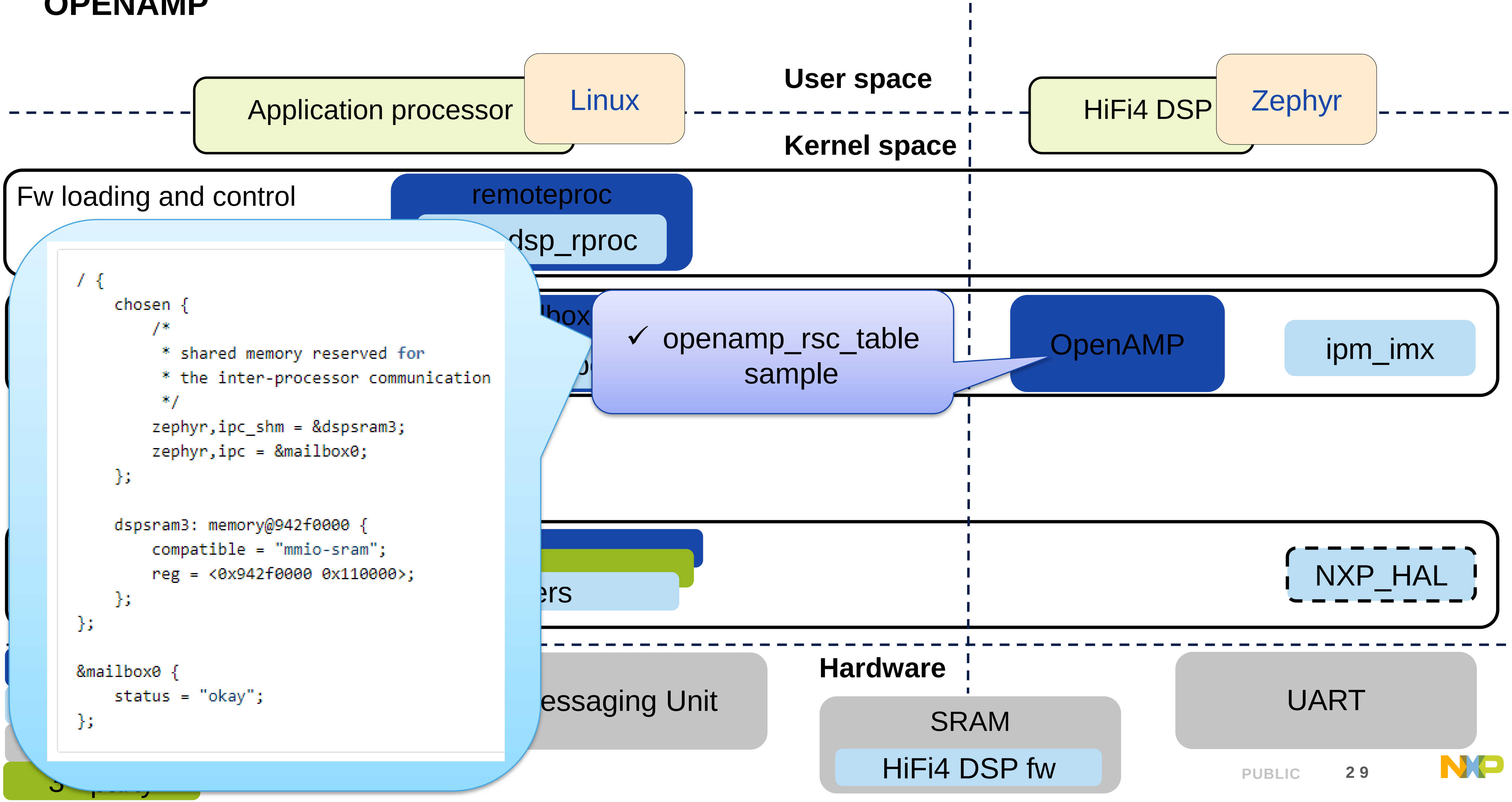
```
mailbox0: mailbox@30e70000 {
    compatible = "nxp,imx-mu-dsp";
    reg = <0x30e70000 0x10000>;
    interrupts = <7 0>;
    status = "disabled";
};
```

OPENAMP

- Open Asymmetric Multi-processing (OpenAMP) framework provides:
  - life cycle management (remoteproc)
  - inter processor communication capabilities (rpmsg)
  - compatibility with upstream Linux remoteproc, rpmsg and VirtIO components
  - uses Libmetal as an abstraction layer to access shared memory



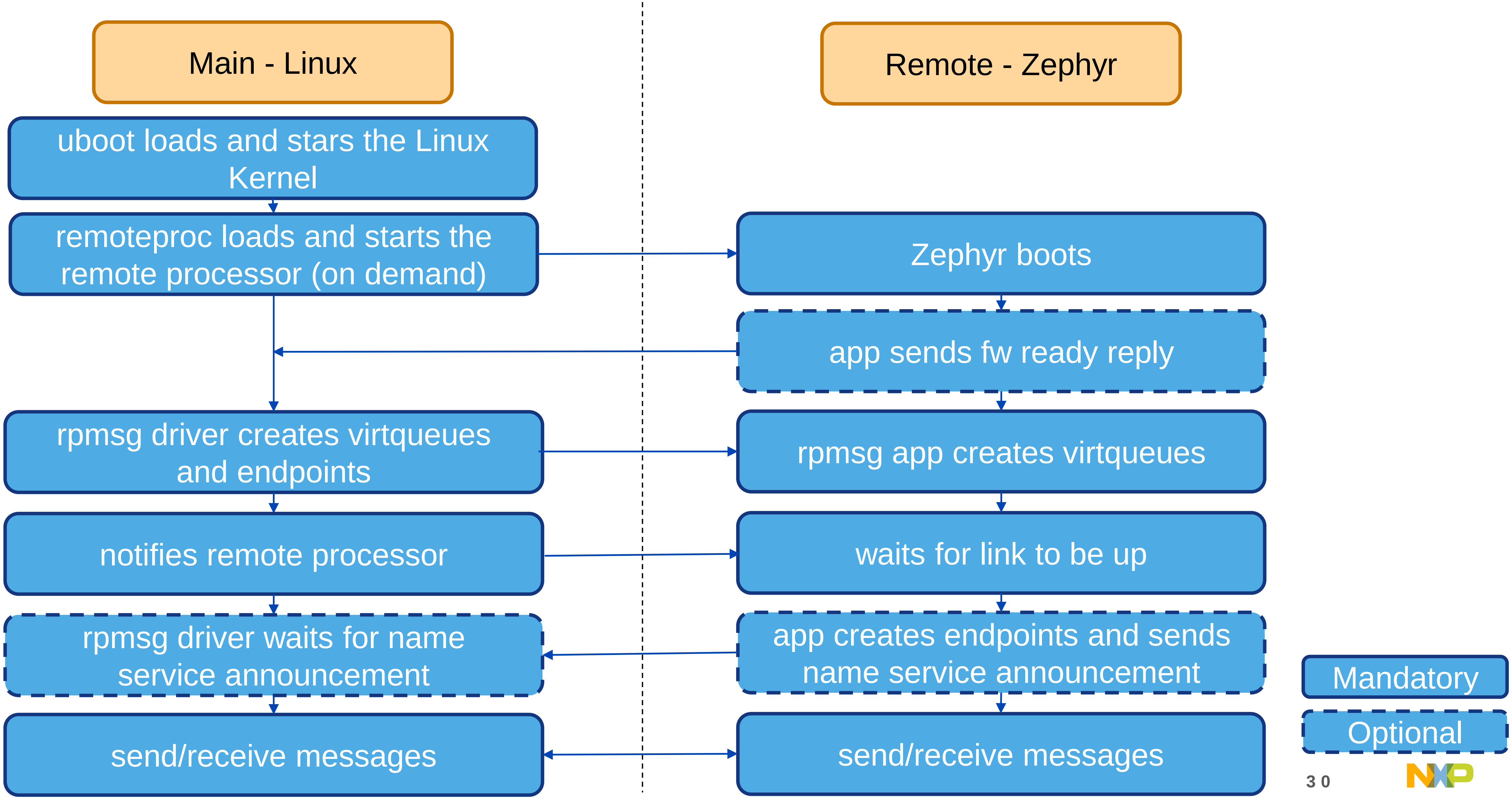
OPENAMP



✓ openamp_rsc_table sample

```
/ {
    chosen {
        /*
         * shared memory reserved for
         * the inter-processor communication
         */
        zephyr,ipc_shm = &dspsram3;
        zephyr,ipc = &mailbox0;
    };

    dspsram3: memory@942f0000 {
        compatible = "mmio-sram";
        reg = <0x942f0000 0x110000>;
    };
};

&mailbox0 {
    status = "okay";
};
```

GENERIC LINUX AND ZEPHYR COMMUNICATION SETUP



Main - Linux

uboot loads and stars the Linux Kernel

remoteproc loads and starts the remote processor (on demand)

rpmsg driver creates virtqueues and endpoints

notifies remote processor

rpmsg driver waits for name service announcement

send/receive messages

Remote - Zephyr

Zephyr boots

app sends fw ready reply

rpmsg app creates virtqueues

waits for link to be up

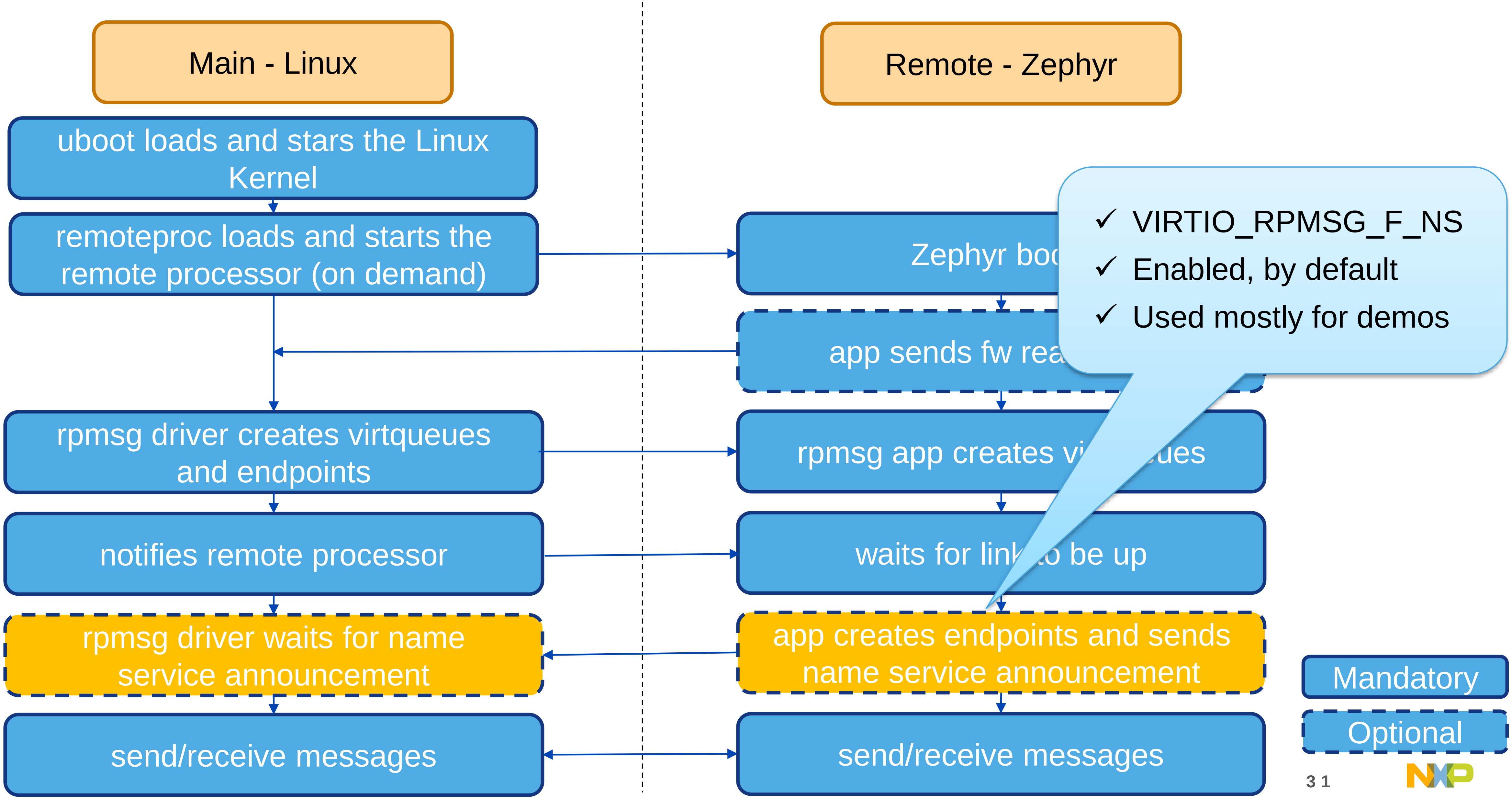
app creates endpoints and sends name service announcement

send/receive messages

Mandatory

Optional

GENERIC LINUX AND ZEPHYR COMMUNICATION SETUP



Main - Linux

uboot loads and stars the Linux Kernel

remoteproc loads and starts the remote processor (on demand)

rpmsg driver creates virtqueues and endpoints

notifies remote processor

rpmsg driver waits for name service announcement

send/receive messages

Remote - Zephyr

Zephyr boo

app sends fw rea

rpmsg app creates vi ues

waits for lin o be up

app creates endpoints and sends name service announcement

send/receive messages

- ✓ VIRTIO_RPMSG_F_NS
- ✓ Enabled, by default
- ✓ Used mostly for demos

Mandatory

Optional

CHALLENGES

- Documentation for remoteproc / rpmsg could be improved
- Linux remoteproc
  - for i.MX, applications are written to IRAM which has a limitation of 4bytes writes
  - for IPC, `find_loaded_rsc_table()` is mandatory
- Zephyr IPC
  - shared memory for core communication must be large enough for both vrings and buffers
  - Messaging Unit driver must be correctly initialized, and all interrupts enabled
- OpenAMP
  - when reading the status from resource table structure, the DCache must be invalidated, otherwise the status is never updated

FUTURE WORK

- Upstream
- Add or enable new samples in Zephyr using DSP API
- Benchmark DSP results
- Use generic loader for other applications (e.g. SOF)

THANK YOU

- Questions?

SECURE CONNECTIONS
FOR A SMARTER WORLD