OPEN-AMP 2016.10 FEATURE PROPOSALS

PETR LUKAS, MICHAL PRINC, MAREK NOVAK
MCU SW TEAM
APR 28, 2016

SECURE CONNECTIONS
FOR A SMARTER WORLD

Changes proposed for 2016.10 OpenAMP release

- Zero copy support
  - API extension to enable zero copy operation
  - Applications can access directly virtio buffers
  - Essential for low footprint devices
  - Implementation ready and submitted as a patch

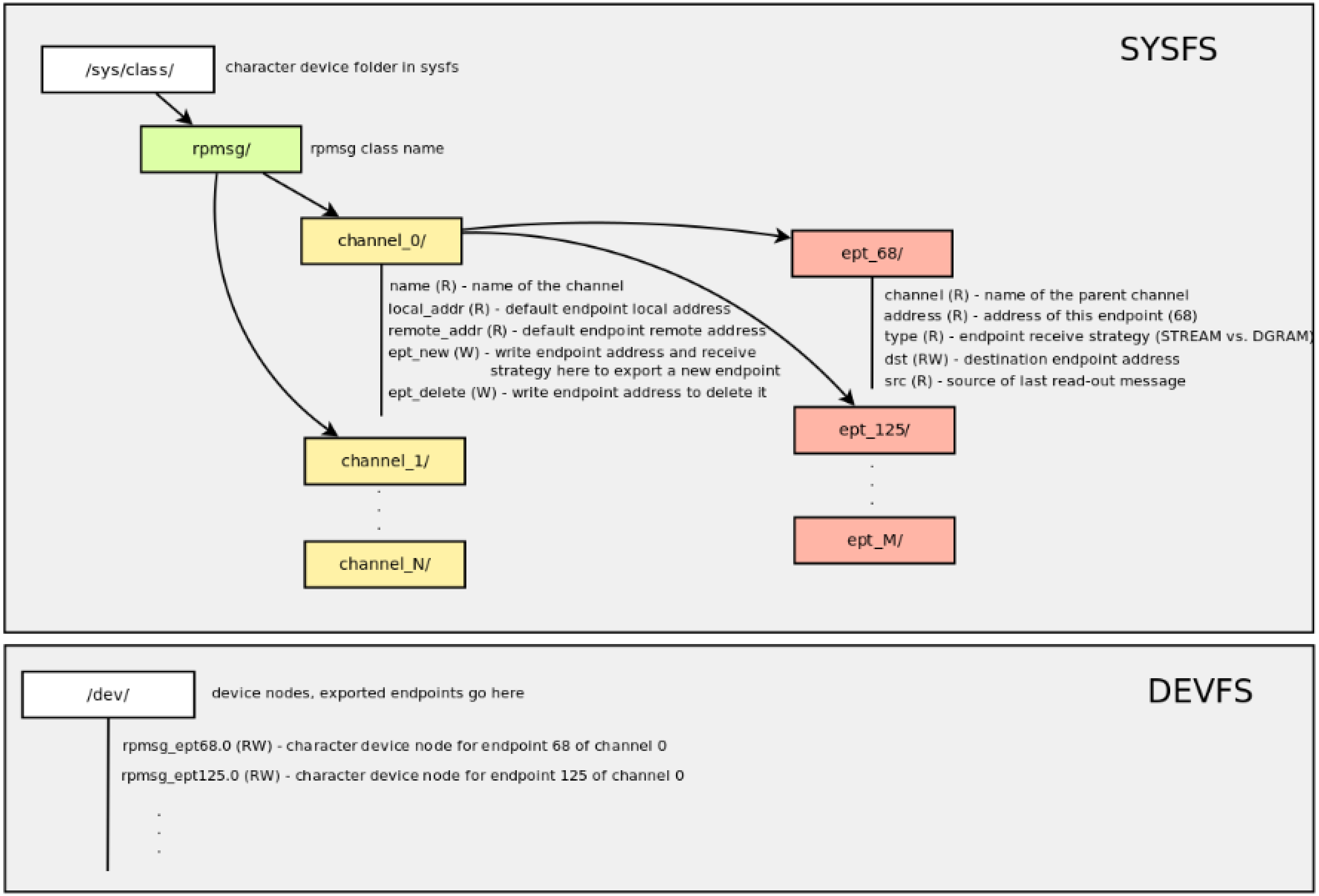
- RPMSG RTOS layer
  - Provide API similar to standard RTOS Messaging
  - Effective in RTOS, copy operation are not done in the ISR context
  - We would like to be maintainers of this component
  - Implementation ready and submitted as a patch

- Standardize RPMSG endpoint operations from Linux user space
  - Sysfs driver – experimental implementation done, available for review
  - User is able to create/destroy endpoint in runtime
  - udev rules to create nodes in /dev are available

- Libmetal
  - Concern with code size and complexity for RTOSes and BM (we use only RPMSG i
  - We would like to use RPMSG porting layer directly without LIBMETAL on RTOS/BM

Backup slides

Sysfs rpmsg_ept driver


SYSFS
/sys/class/
character device folder in sysfs
rpmsg/
rpmsg class name
channel_0/
name (R) - name of the channel
local_addr (R) - default endpoint local address
remote_addr (R) - default endpoint remote address
ept_new (W) - write endpoint address and receive strategy here to export a new endpoint
ept_delete (W) - write endpoint address to delete it
channel_1/
channel_N/
ept_68/
channel (R) - name of the parent channel
address (R) - address of this endpoint (68)
type (R) - endpoint receive strategy (STREAM vs. DGRAM)
dst (RW) - destination endpoint address
src (R) - source of last read-out message
ept_125/
ept_M/
DEVFS
/dev/
device nodes, exported endpoints go here
rpmsg_ept68.0 (RW) - character device node for endpoint 68 of channel 0
rpmsg_ept125.0 (RW) - character device node for endpoint 125 of channel 0

RPMSG RTOS extension - Motivation

- Current RPMsg API relies on ISR
  - all the processing of received data must be done in the interrupt context
  - message must be copied in a temporary application buffer for later processing

- This is not common approach in RTOS environment

- RTOS typically support blocking sequential API

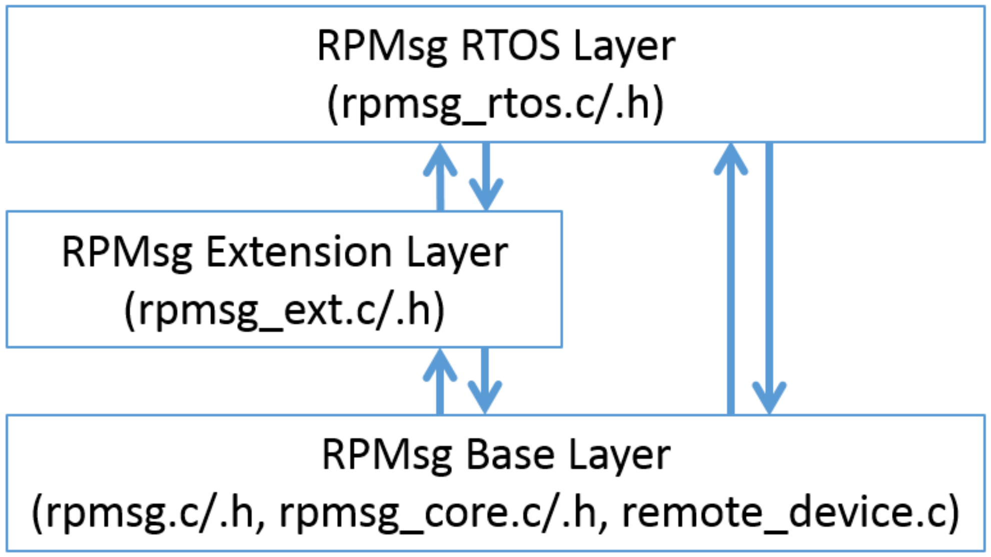
- RTOS-aware extension of RPMsg created

RTOS-aware extension features

- No data processing in the interrupt context
- Blocking receive API
- Zero-copy send and receive API
- Receive with timeout provided by RTOS
- Compatibility with Linux OS upstream kept
- Separation Env and Platform
- FreeRTOS environmental layer
- Baremetal vs. Linux and FreeRTOS vs. Linux communication examples
- Own test code

RTOS-aware extension implementation

- Two layers
  - **RPMsg Extension (ZERO COPY)** layer allows users to allocate and release virtio tx buffers, as well as it implements the zero-copy send functionality, intended for baremetal apps.
  - **RPMsg RTOS layer** addresses RTOS-based application needs (handling received data outside the interrupt context, blocking receive API implementation, zero-copy mechanisms)



RPMsg ZERO COPY API

- rpmsg_hold_rx_buffer
- rpmsg_release_rx_buffer
- rpmsg_alloc_tx_buffer
- rpmsg_sendto_nocopy
- rpmsg_send_nocopy

RPMsg RTOS API

- rpmsg_rtos_init
- rpmsg_rtos_deinit
- rpmsg_rtos_create_ept -> create msg_queue
- rpmsg_rtos_destroy_ept
- rpmsg_recv()
- rpmsg_rtos_recv_nocopy
- rpmsg_rtos_recv_nocopy_free
- rpmsg_rtos_alloc_tx_buffer
- rpmsg_rtos_send
- rpmsg_rtos_send_nocopy

RPMsg porting sub-layers

The RPMsg porting layers have been also modified and consolidated in order to

- Strictly separate platform-related (multicore device) and environment-related (Bare Metal, RTOS) layers.
- Update the environment layer API by functions requested by the RTOS layer. The following *env* functions have been introduced:
  - *int env_create_queue(void queue, int length, int element_size)*
  - *void env_delete_queue(void queue)*
  - *int env_put_queue(void queue, void msg, int timeout_ms)*
  - *int env_get_queue(void queue, void msg, int timeout_ms)*

SECURE CONNECTIONS
FOR A SMARTER WORLD