# S32G2
# IPCF Hands On

VCNS APPLICATIONS AND SOLUTIONS

**APRIL 2021**

SECURE CONNECTIONS
FOR A SMARTER WORLD


EXTERNAL USE

# Agenda

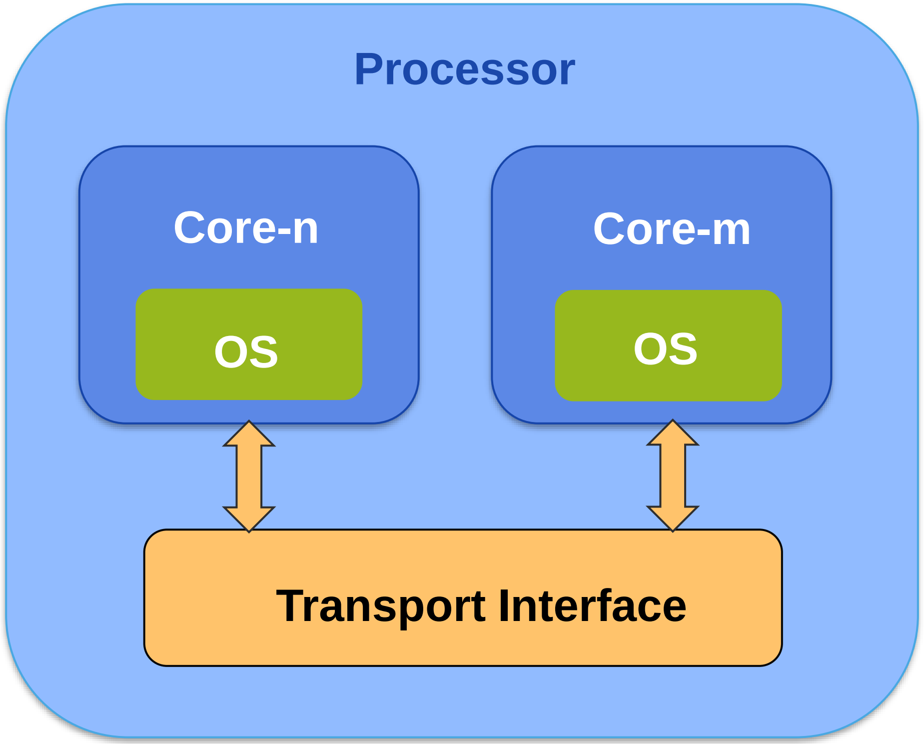
- Introduction
- Underlying HW
- IPCF Architecture
- IPCF Shared Memory Driver
- IPCF Use-Cases
- Hands On

# INTRODUCTION

SECURE CONNECTIONS
FOR A SMARTER WORLD

# INTRODUCTION

- Inter-Platform Communication Framework (IPCF) is a subsystem which enables applications, running on multiple homogenous or heterogenous processing cores, located on the same chip or different chips, running on different operating systems (AUTOSAR® OS, Linux®, FreeRTOS, etc.), to communicate over various transport interfaces (Shared Memory, etc.).
- Designed for closely distributed embedded systems with low-latency and tiny-footprint.
- Exposes a Zero-copy API for maximum performance, minimum overhead and low CPU load.
- IPCF SW release for S32G2 performs communication over Shared Memory.

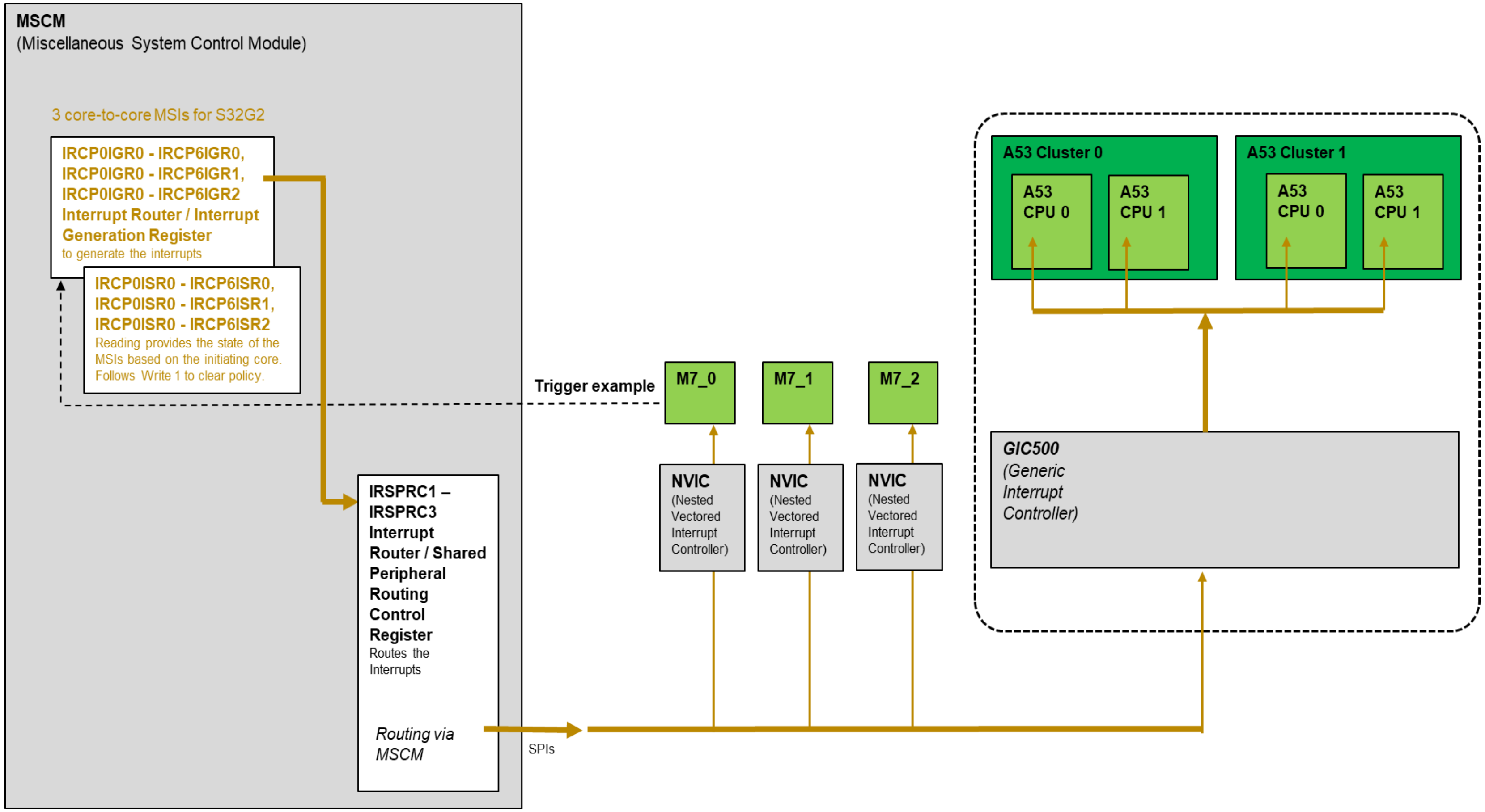


# Underlying HW

SECURE CONNECTIONS FOR A SMARTER WORLD

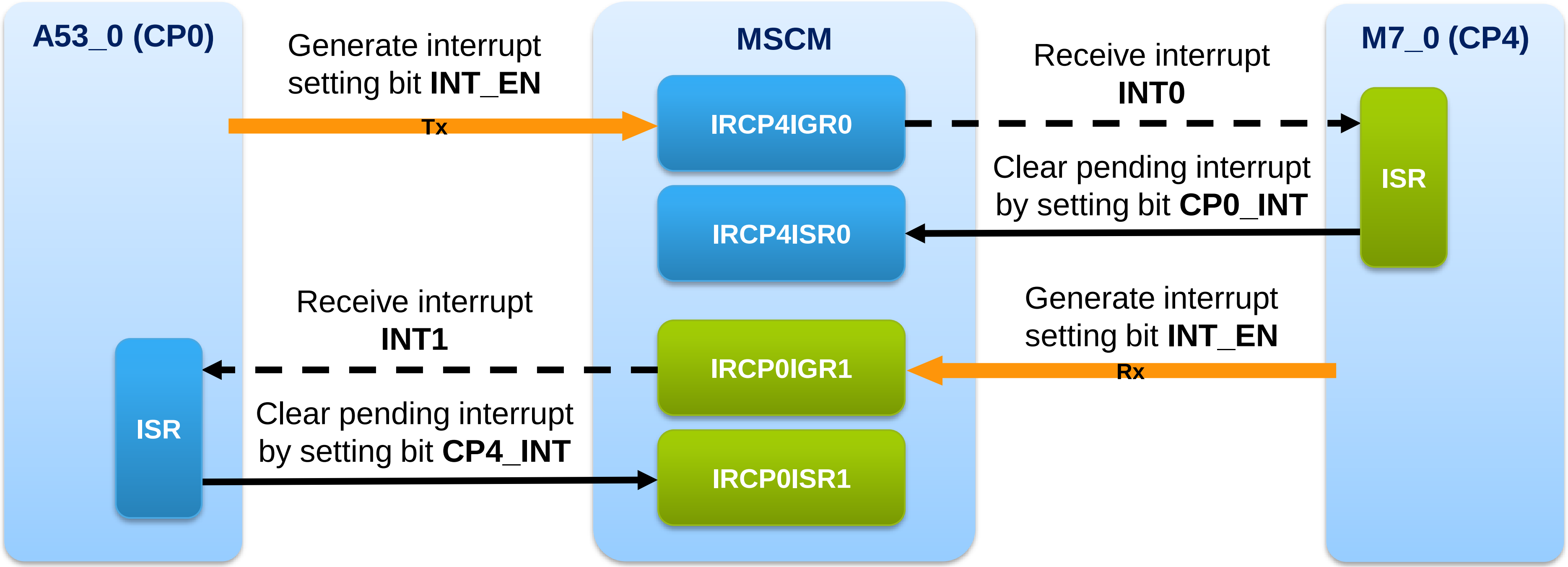
# UNDERLYING HW

- Message-signaled interrupts (MSIs) are interrupts that are indirectly broadcast to a target core by writing configuration bits in MSCM.
- S32G274A has 3 MSIs for core-to-core interrupts and all the application cores can access these.



# MSCM – INTER-CORE INTERRUPTS EXAMPLE

- A53_0 transmit notification is interrupt INT0 and receive notification is INT1



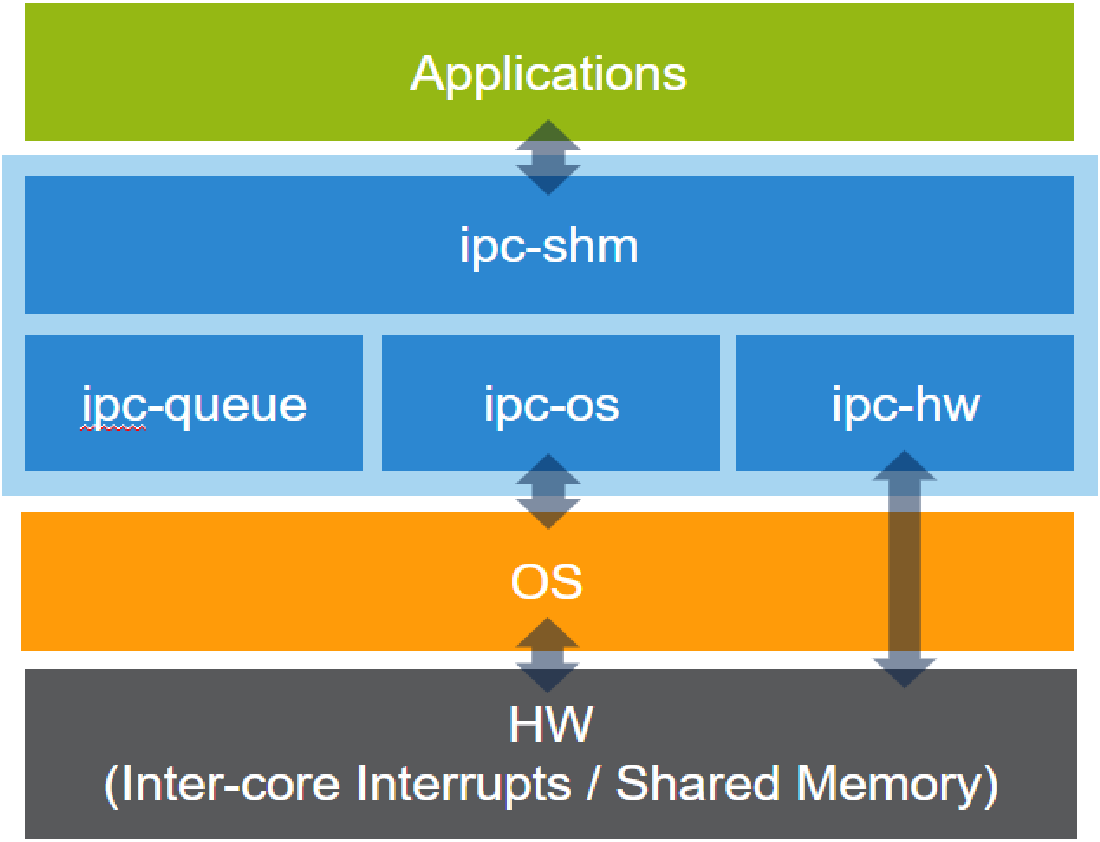
# IPCF Architecture

SECURE CONNECTIONS
FOR A SMARTER WORLD

# IPCF SYSTEM ARCHITECTURE

IPCF driver contains the following layers:

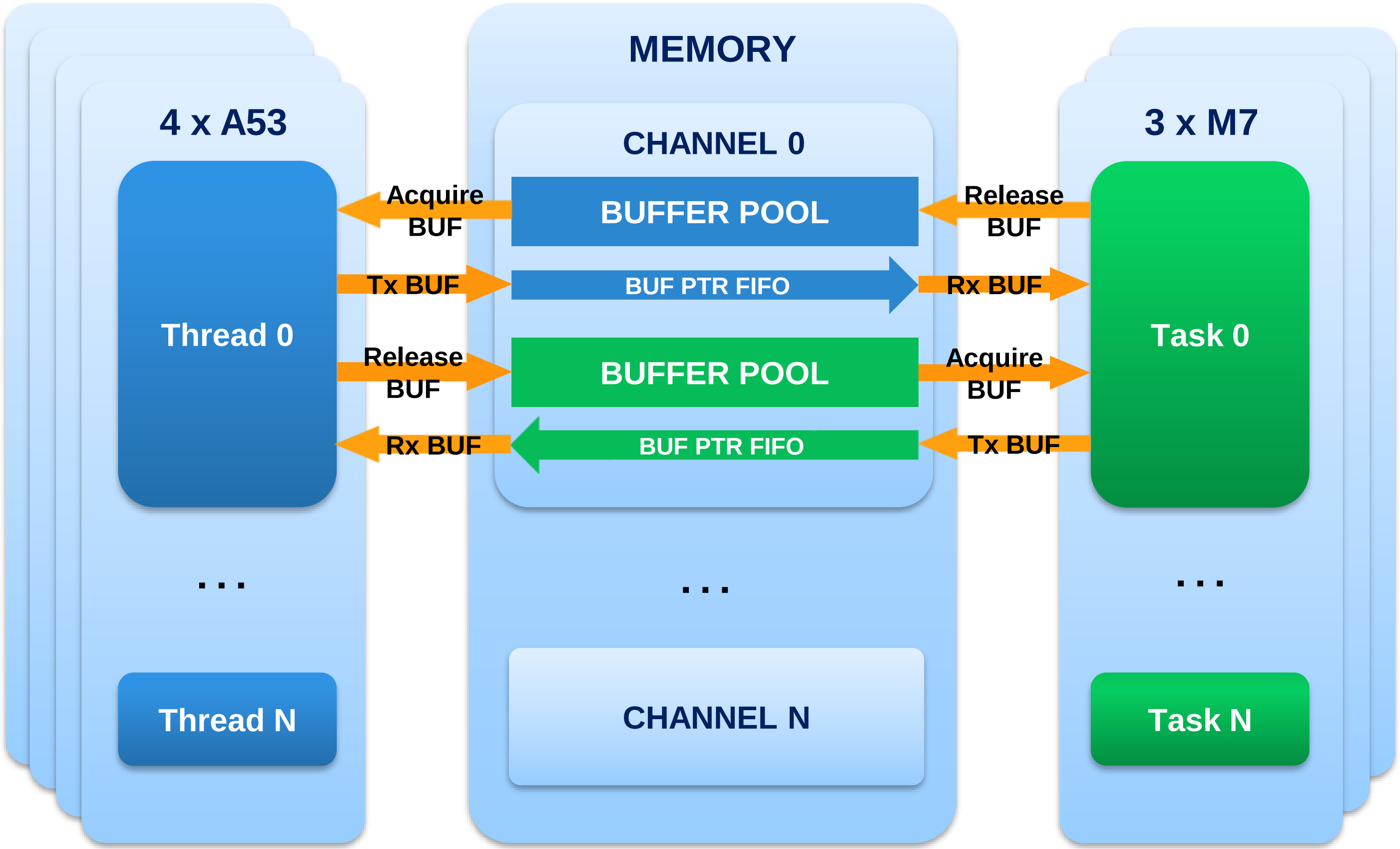
- Shared memory generic implementation that is HW and OS agnostic
- Queue component implementation used in IPCF driver
- HW abstraction component: abstraction over various HW IP modules (MSCM, INTC …)
- OS abstraction component: OS agnostic API for common OS services



# IPCF
# Shared memory driver

SECURE CONNECTIONS
FOR A SMARTER WORLD

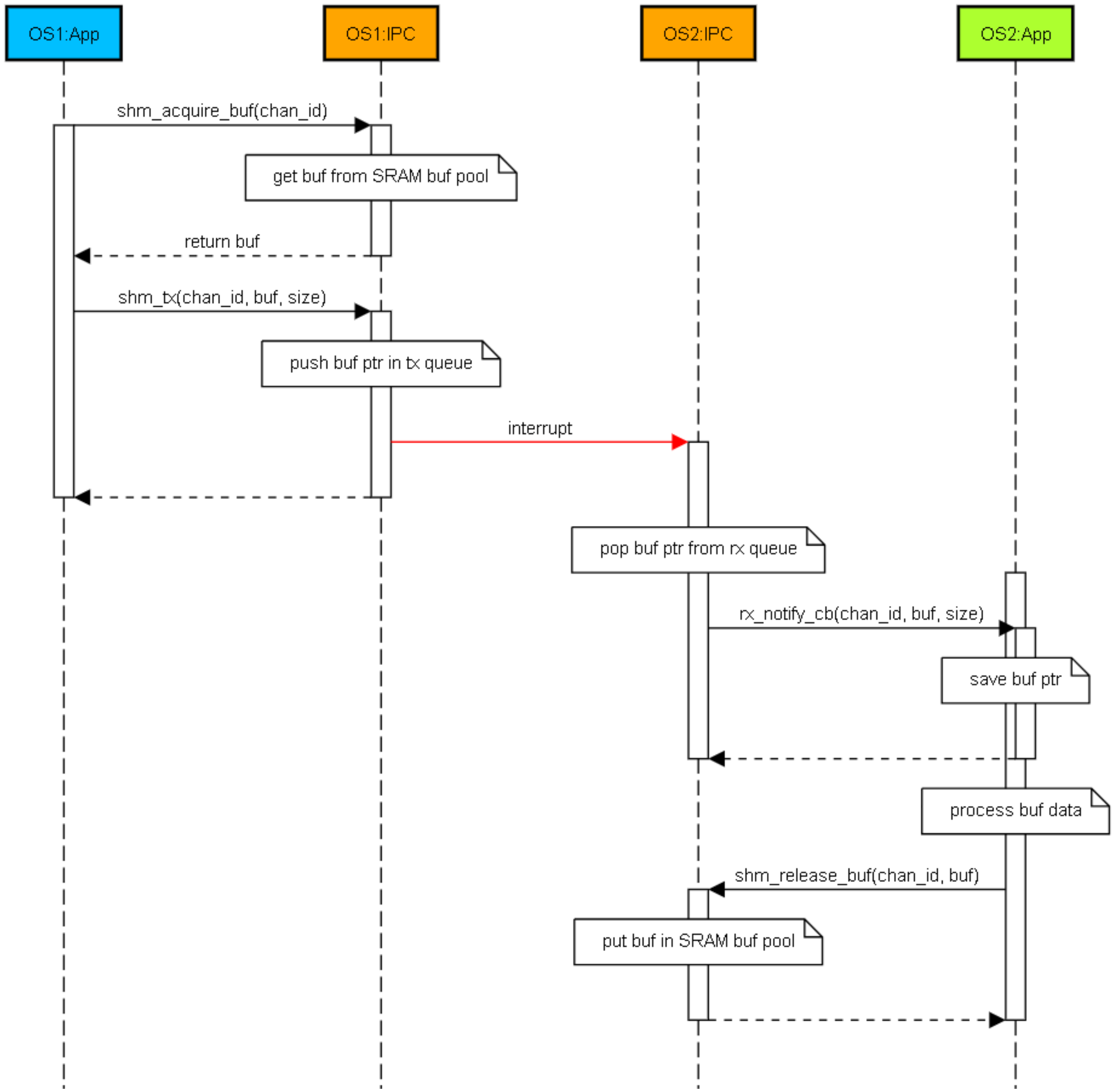
# IPCF SHARED MEMORY DRIVER ARCHITECTURE




- **Zero-Copy architecture**
- **Performance**
- High throughput
- Low CPU load
- Efficient core utilization
- **Freedom from interference**
  - Memory protection
  - Different ASIL partitions

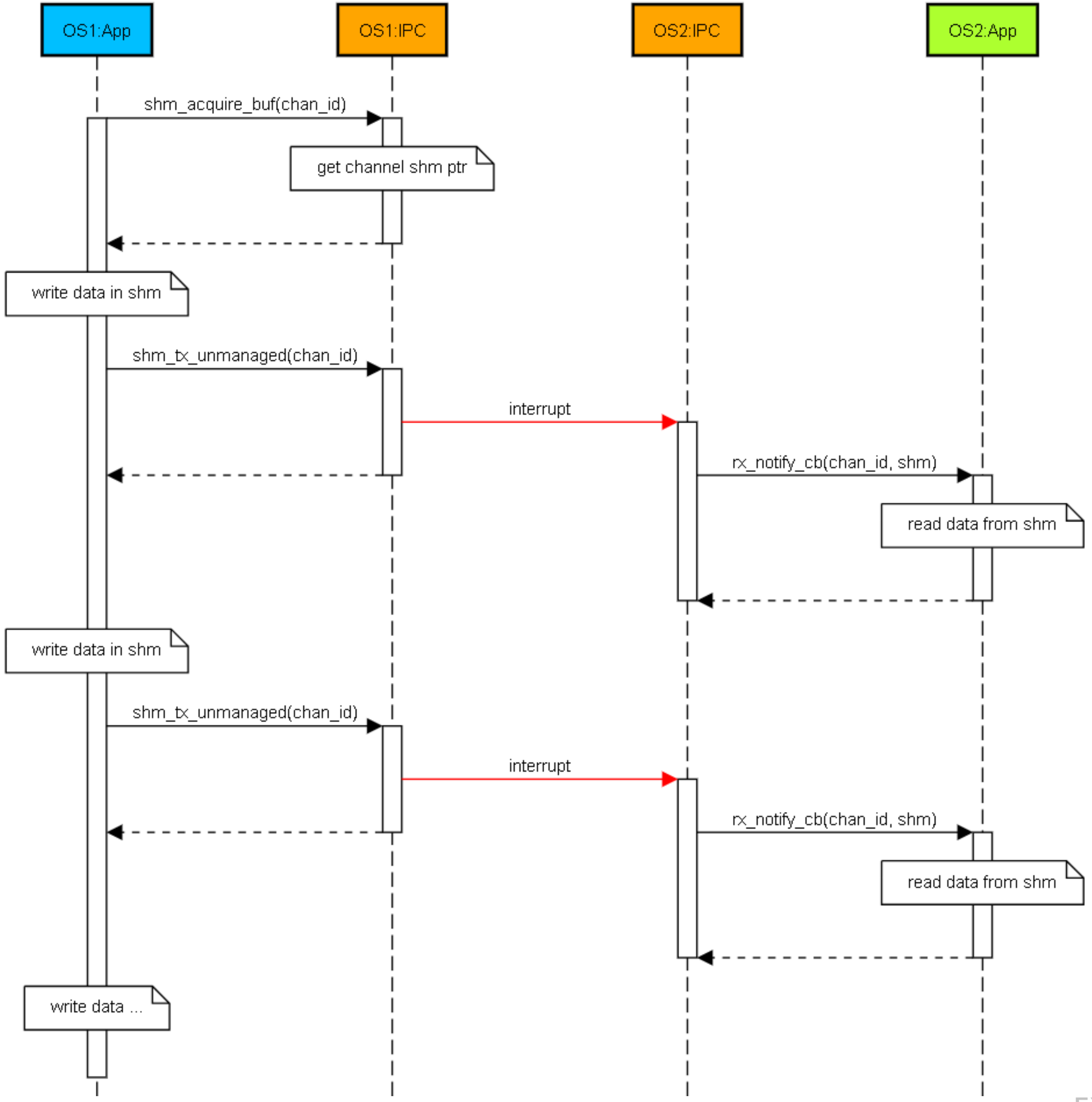
# MANAGED CHANNEL DATA FLOW A53 → M7

The diagram shows data flow from OS1 app to OS2 app, and it is symmetric in the other direction



# UNMANAGED CHANNEL DATA FLOW A53 → M7

- Similar to POSIX ShM
- Each App owns half of the channel memory
- Apps responsible for memory management and sync
- Can be used for streaming use-cases
- It is symmetric in the other direction

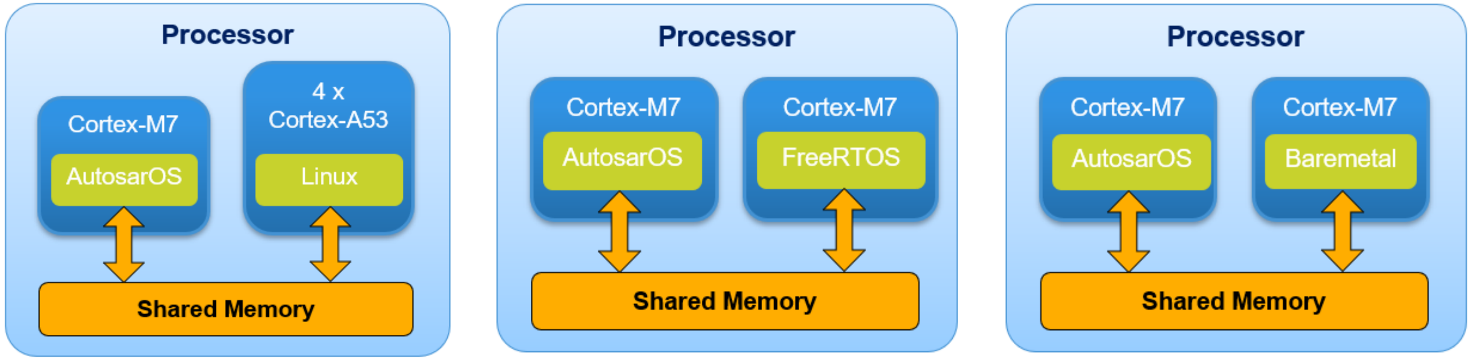


# IPCF Use Cases

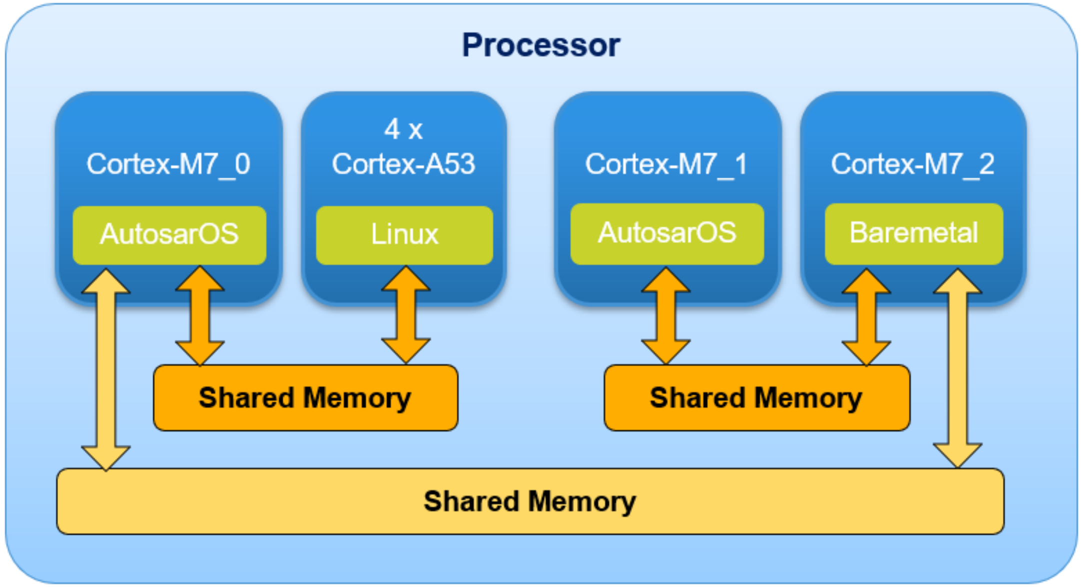
SECURE CONNECTIONS
FOR A SMARTER WORLD

# IPCF USE CASES

- On Multiple homogenous or heterogenous processing cores



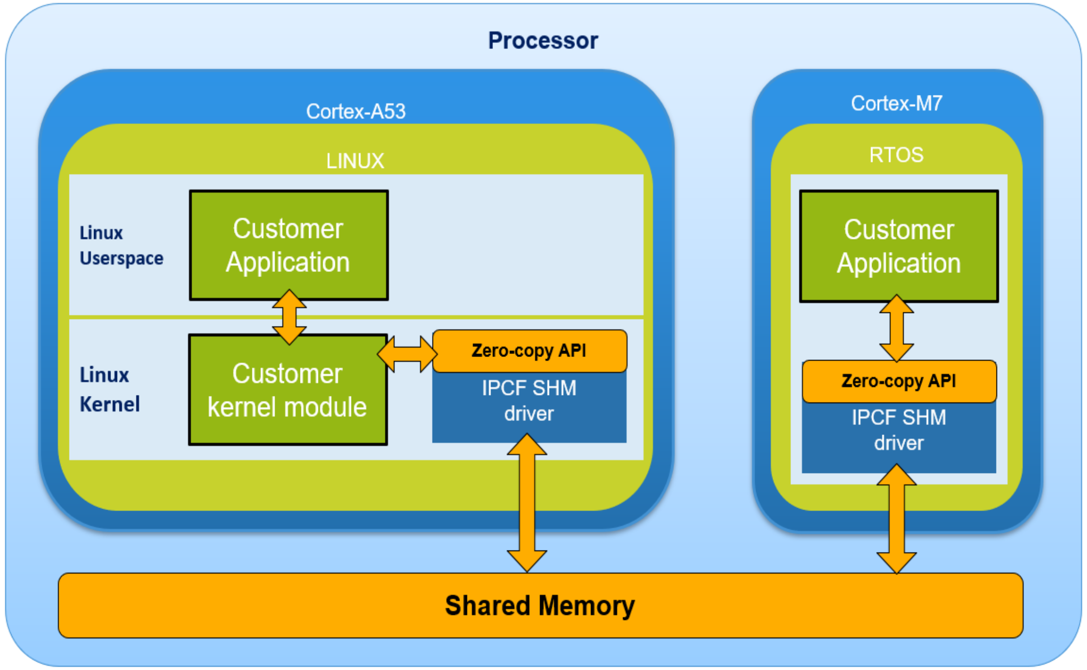

- On Multiple homogenous or heterogenous processing cores with multiple instances.

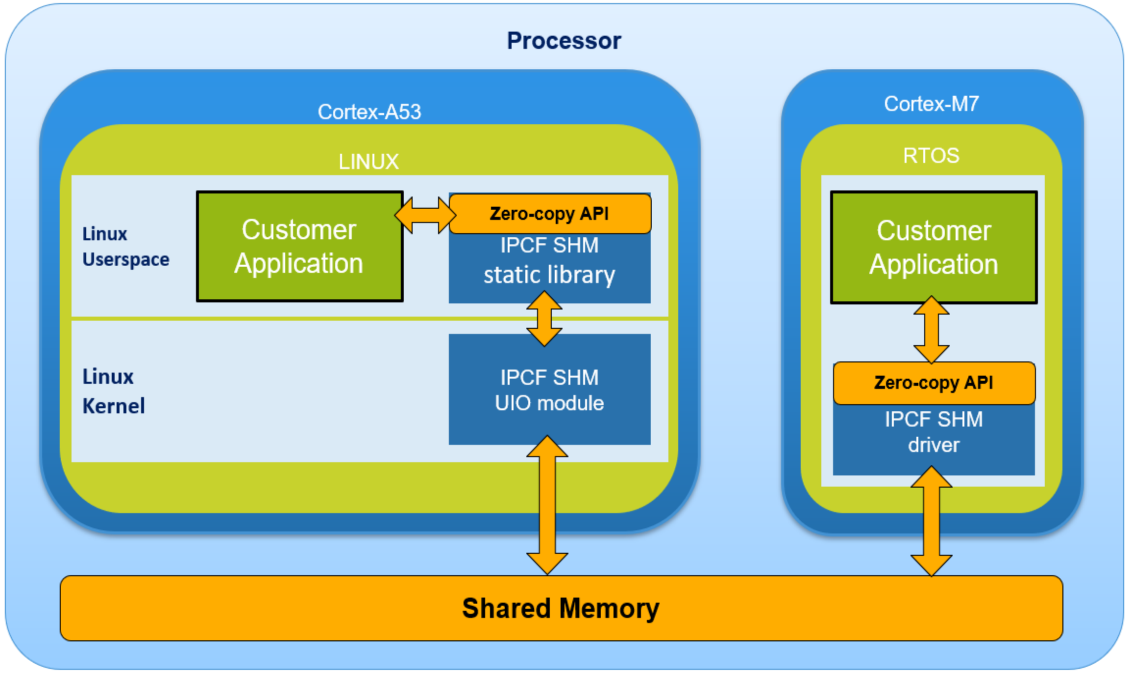



Note: The MSIs are limited to 3. The below use case is applicable with polling feature which is scheduled to be added in the next IPCF SW release.

# IPCF USE CASES

- Use case in Linux



# IPCF Hands-On

SECURE CONNECTIONS
FOR A SMARTER WORLD

# GETTING THE IPCF SW

- Linux®
  - IPCF module is built with Yocto from NXP Auto Linux BSP, but can also be built manually, if needed.

- AUTOSAR®
  - On Flexera, Automotive SW – S32G2 Standard Software → Automotive SW - S32G2 - Inter-Platform Communication Framework
  - Open the latest SW release package, currently latest SW version – D2012
  - Download the installer IPCF_1.1.0_D2012.exe
  - As a prerequisite, the S32XX_AUTOSAR_OS_4_3_106_CODEDROP5_0_8_0 for S32G274A would also be required.

- FreeRTOS**/** Bare-metal
  - On Flexera, Automotive SW – S32G2 Standard Software → Automotive SW - S32G2 - Inter-Platform Communication Framework
  - Open the latest SW release package, currently latest SW version – D2012
  - Download the installer PCF_1.1.0_D2012_updatesite.zip. This is added as module in the S32DS 3.3.
  - For adding FreeRTOS support, install the FreeRTOS version available on Flexera: S32G2 Reference Software → S32G2 - FreeRTOS for Cortex-M7 → S32G2 FreeRTOS 10.3.1 version 0.9.0 → SW32G2_FreeRTOS_10_3_1_UOS_0_9_0_DS_updatesite_D2012.zip

1. Flash the SD card with the Auto Linux BSP image.
2. Copy the stripped binary (IPCF_Example_S32G274.bin) created for CM7 core to the FAT partition on SD-card.
3. Establish a serial connection with the S32G274A board and power it on.
4. Hit any key to stop in the U-Boot console.
5. Disable Data Cache from U-Boot.
   `dcache off`
6. Zero-set SRAM shared memory used by both sample apps.
   `initsram 0x34100000 0x700000`
7. Load the binary in SRAM to the address specified in the linker file.
   `fatload mmc 0:1 0x34300000 /IPCF_Example_S32G274.bin`
8. Start the M7_0 core. (The argument is the address of the Interrupt Vector defined in the Linker file)
   `startm7 0x34501000`
9. Boot Linux.
   `boot`
10. Login with root and run Linux sample application.
    `insmod /lib/modules/`uname -r`/extra/ipc-shm-dev.ko`
    `insmod /lib/modules/`uname -r`/extra/ipc-shm-sample.ko`
    `echo 10 > /sys/kernel/ipc-shm-sample/ping`

## Expected Output

```
root@s32g274aevb:~# echo 10 > /sys/kernel/ipc-shm-sample/ping
root@s32g274aevb:~# [   44.085244] 002: ipc-shm-sample: starting demo...
[   44.085266] 002: ipc-shm-sample: ch 0 >> 20 bytes: SENDING MESSAGES: 10
[   44.085280] 002: ipc-shm-sample: ch 1 >> 16 bytes: #1 Hello world!
[   44.085313] 000: ipc-shm-sample: ch 1 << 16 bytes: #1 Hello world!
[   44.085343] 002: ipc-shm-sample: ch 2 >> 16 bytes: #2 Hello world!
[   44.085372] 000: ipc-shm-sample: ch 2 << 16 bytes: #2 Hello world!
[   44.085394] 002: ipc-shm-sample: ch 1 >> 16 bytes: #3 Hello world!
[   44.085423] 000: ipc-shm-sample: ch 1 << 16 bytes: #3 Hello world!
[   44.085444] 002: ipc-shm-sample: ch 2 >> 16 bytes: #4 Hello world!
[   44.085472] 000: ipc-shm-sample: ch 2 << 16 bytes: #4 Hello world!
[   44.085492] 002: ipc-shm-sample: ch 1 >> 16 bytes: #5 Hello world!
[   44.085520] 000: ipc-shm-sample: ch 1 << 16 bytes: #5 Hello world!
[   44.085541] 002: ipc-shm-sample: ch 2 >> 16 bytes: #6 Hello world!
[   44.085569] 000: ipc-shm-sample: ch 2 << 16 bytes: #6 Hello world!
[   44.085590] 002: ipc-shm-sample: ch 1 >> 16 bytes: #7 Hello world!
[   44.085618] 000: ipc-shm-sample: ch 1 << 16 bytes: #7 Hello world!
[   44.085638] 002: ipc-shm-sample: ch 2 >> 16 bytes: #8 Hello world!
[   44.085666] 000: ipc-shm-sample: ch 2 << 16 bytes: #8 Hello world!
[   44.085687] 002: ipc-shm-sample: ch 1 >> 16 bytes: #9 Hello world!
[   44.085715] 000: ipc-shm-sample: ch 1 << 16 bytes: #9 Hello world!
[   44.085735] 002: ipc-shm-sample: ch 2 >> 16 bytes: #10 Hello world
[   44.085763] 000: ipc-shm-sample: ch 2 << 16 bytes: #10 Hello world
[   44.085779] 000: ipc-shm-sample: ch 0 << 20 bytes: REPLIED MESSAGES: 10
[   44.085779] 002: ipc-shm-sample: exit demo
```

SECURE CONNECTIONS
FOR A SMARTER WORLD