WInnF Transceiver Facility

Eric Nicollet (Thales)
Co-chair of the Transceiver Next project

WInnComm Europe 2019
15-16 May 2019 – Berlin

WInnForum Standards™

WIRELESS INNOVATION FORUM®

Agenda

Introduction

The PIM (Platform Independent Model) Specification

The PSM (Platform Specific Model) Specifications

Conclusions

Introduction

WInnF Transceiver Facility Project

Motivation

- No recognized standard for Transceiver API
- A diversity of background APIs
  - JTNC MHAL RF Chain Coordinator (USA)
  - WInnF Transceiver Facility v1.0 (WInnF, from EU project)
  - ESSOR Architecture Transceiver API (ESSOR Countries)
- Need for international harmonization
- Need for open development approach

Key milestones

- Kicked-off: 2015 – PIM Spec: Jul 2017
- 3 PSMs Specs (as currently planned): Jul 2019

Work product

- Transceiver Facility v2.0, a WInnF SDR Standard

WInnF Transceiver Facility v2.0

XCVR v2.0 is a WInnF Facility (SDR Standard)

- One PIM Spec: fully implementation agnostic API and Properties
- Several PSM Specs: PIM Spec mapping to programming paradigms
  - Native C++, FPGA and SCA

An unprecedented international harmonization

- 12 contributors from 6 countries (CAN, FRA, GER, ITA, JAP, USA)
  - DGA, ENSTA, FKIE, Harris Corporation, Hitachi Kokusai Electric, JTNC, Leonardo, NordiaSoft, Rockwell-Collins, Rohde & Schwarz, Thales, Viavi Solutions (Cobham)
- Reflecting lessons learnt from programs
  - ESSOR (ESP, FIN, FRA, ITA, POL, SWE)
  - SVFuA (GER)
- Reflecting the SDR expertise of worldwide manufacturers

Background APIs

JTNC MHAL RF Chain Coordinator

- Initially developed by the JTRS program
- First published API for Transceiver functionality
- Release: 2007 (latest update: v3.0, oct. 2013)

WInnF Transceiver Facility v1.0

- Openly developed as a WInnF standard
- Leveraging the European project End-to-End Efficiency (E3) and FRA MoD R&T project PEA AL
- Release: Jan 2009

ESSOR Architecture Transceiver API

- Developed by the ESSOR program
- Considered WInnF Facility v1.0 as one reference starting point
- Finalized: 2010 – Released to WInnF: 2016

The PIM Specification

WInnForum Standards™

WIRELESS INNOVATION FORUM®

XCVR Facility PIM – Overview

As a WInnF Facility, the Transceiver Facility (XCVR) standardizes

- A service-oriented *Application Programming Interface*
- Portability-oriented *Properties*

It therefore supports

- Portability of *applications*
- Openness of *radio platforms* (a.k.a. *hospitality*)

***Transceiver* is the processing stage between**

- The antenna
- The radio physical layer baseband processing

***Tranceiver* I/Os are**

- The baseband signal
- The radio signal

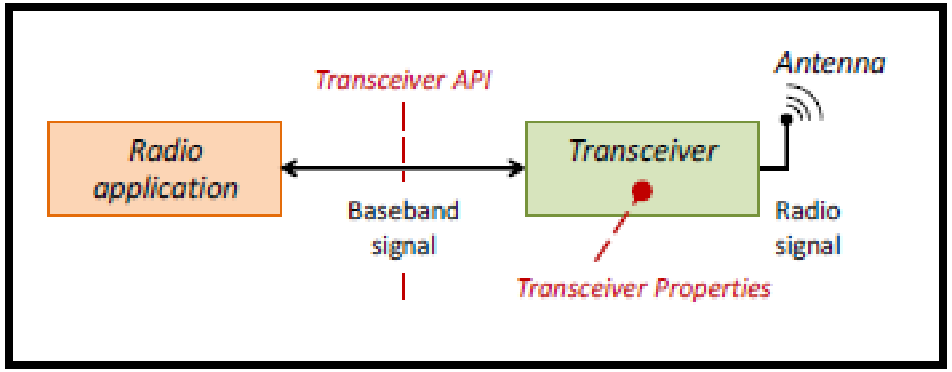


Figure 1 Overview of Transceiver Facility

XCVR Facility PIM – Applicability

For any type of radio

- Simplex, half or full fuplex
- Single or multichannel

For any type of real-time control needs

- Timed
- Strobe-based waveforms

Extremely scalable

- From low cost to high performance *transceivers*
- From basic to advanced *radio applications*

Extensive debug and integration support

- Standard exceptions and errors

Tx processing phases

A *Tx channel* operates a succession of Tx processing phases

A *Tx processing phase* continuously up-converts a *baseband signal $\underline{s}_{BB}[n]$* into a *radio signal $s_{RF}(t)$*

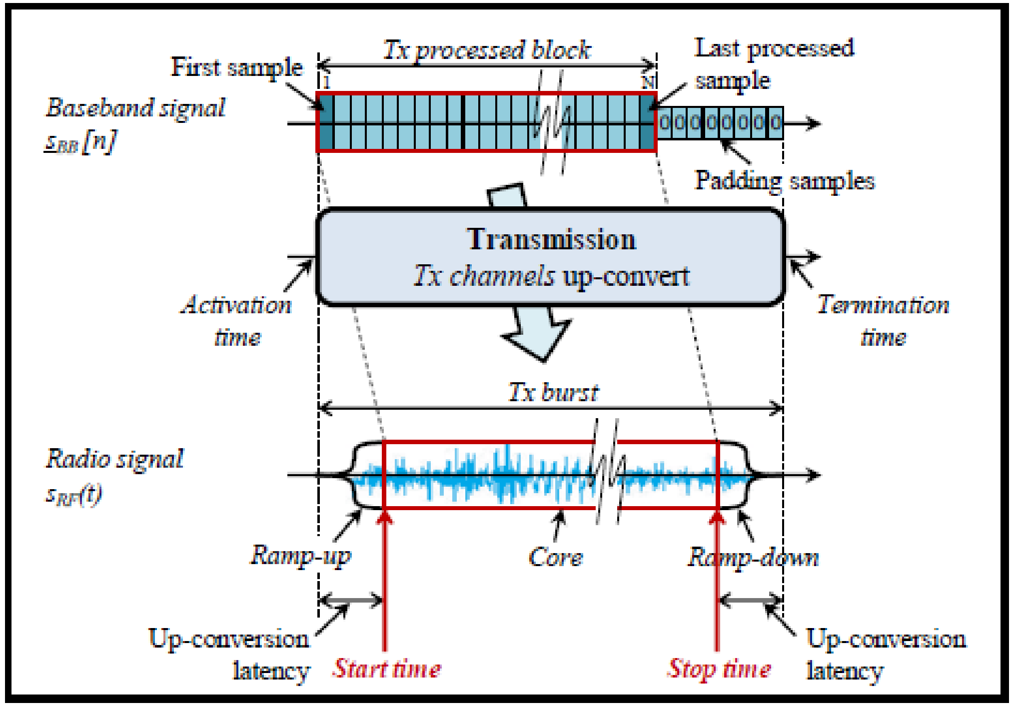


Figure 2 Principle of transmission processing phase

Ideal model

$$\dot{\underline{s}}_{RF}(f+f_c) = \alpha.\mathrm{rect}(f/B).\dot{\underline{s}}_{BB}(f),\ f \in [-F_s^{BB}/2 ; +F_s^{BB}/2]$$

Real model

$$\dot{\underline{s}}_{RF}(f+f_c) = \underline{H}_{Tx}(f).\dot{\underline{s}}_{BB}(f),\ f \in [-F_s^{BB}/2 ; +F_s^{BB}/2]$$

Rx processing phases

A *Rx channel* operates a succession of *Rx processing phases*

An *Rx processing phase* continuously down-converts a radio signal $s_{RF}(t)$ into a baseband signal $\underline{s}_{BB}[n]$

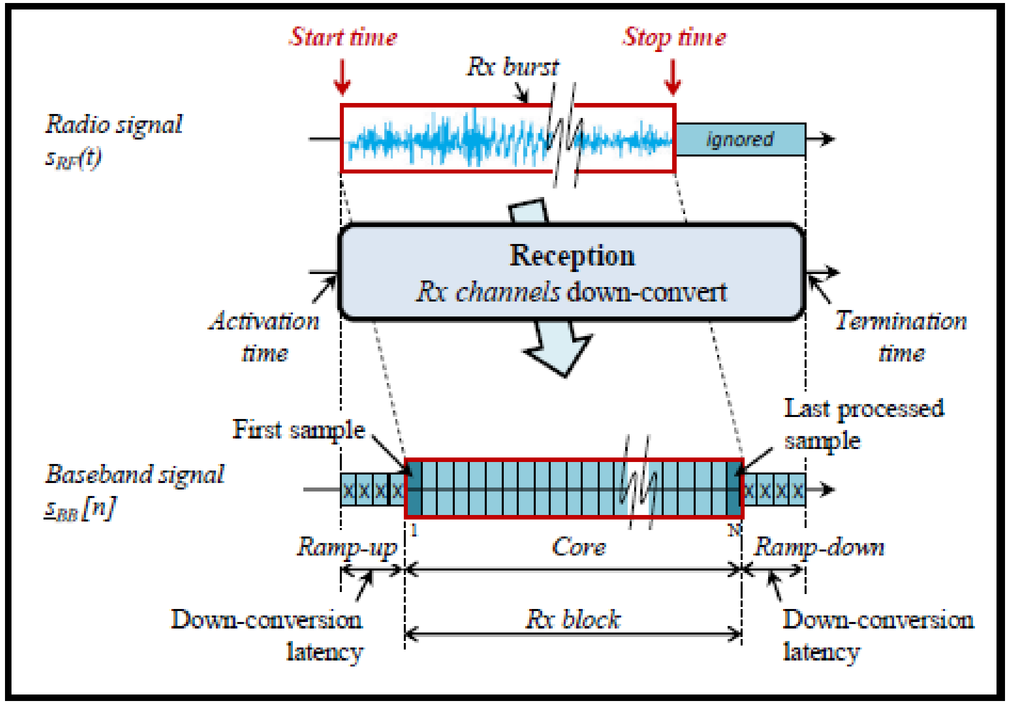


Figure 5 Principle of reception processing phase

Ideal model

$$\underline{\dot{s}}_{BB}(f) = \alpha.\mathrm{rect}(f/B).\underline{\dot{s}}_{RF}(f-f_c),\ f \in [-F_s^{BB}/2\,;+F_s^{BB}/2]$$

Real model

$$\underline{\dot{s}}_{BB}(f) = \underline{H}_{Rx}(f).\underline{\dot{s}}_{RF}(f-f_c),\ f \in [-F_s^{BB}/2\,;+F_s^{BB}/2]$$

WInnForum Standards™

Wireless Innovation Forum

PIM services groups

Essential SGs

• `BurstControl`	Creation and termination of bursts
• `Tuning`	Control of the tuning parameters
• `BasebandSignal`	Packet-based exchange of samples blocks

Additional SGs

• `Management`	General control
• `Transceiver Time`	Access to transceiver time
• `Strobing`	Strobes triggering for creation of bursts
• `GainControl`	Automated gain control
• `Notifications`	Notification of events and errors

Essential SGs – BurstControl (1)

4 possible provide services for bursts start time control

- `DirectCreation` unspecified start time
- `RelativeCreation` from start time of the previous burst
- `AbsoluteCreation` using transceiver time (e.g. terminal time)
- `StrobedCreation` from the next occurrence of a strobe signal

Choice depends on the nature of each radio application

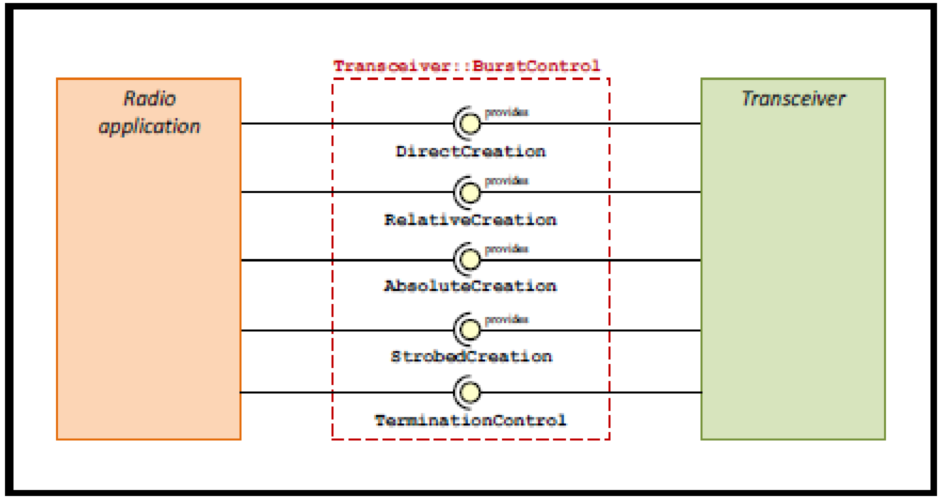


Figure 16 Services of BurstControl services group

Essential SGs – BurstControl (2)

Respect of the specified *start time* is fully under *transceiver* responsibility

Offloads the *radio application* from all platform-specific real-time control aspects for tuning and initiation of the Tx or Rx processing phases

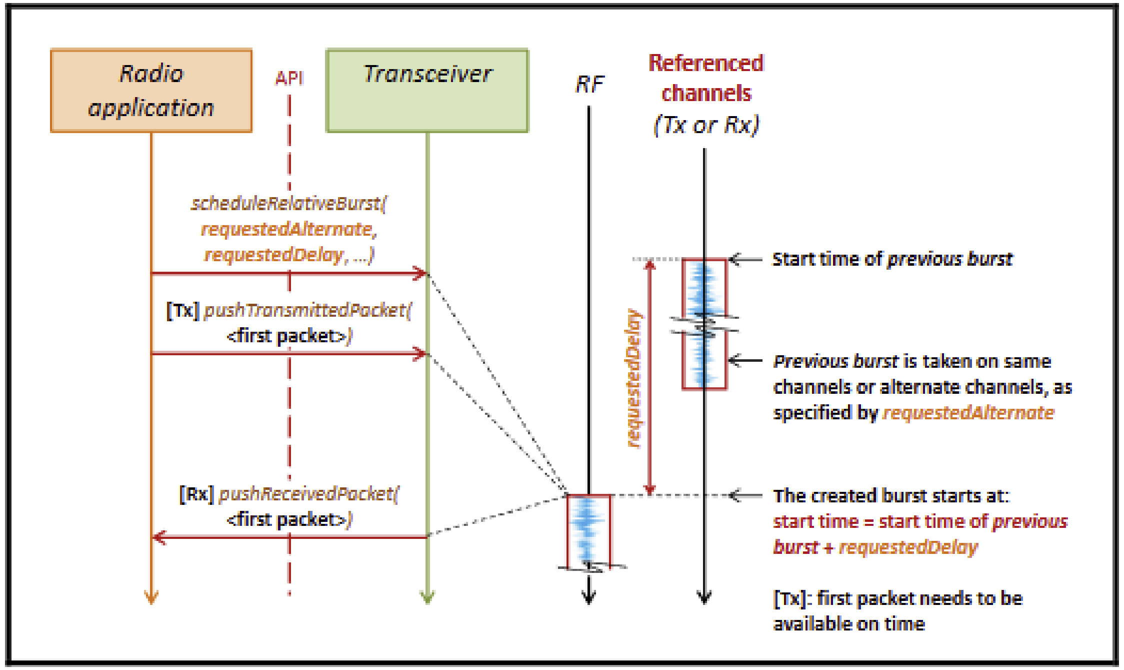


Figure 42 Principle of *scheduleRelativeBurst()*

Essential SGs – Tuning (1)

2 provide services

- `InitialTuning` tuning applicable at beginning of the burst
- `Retuning` changing tuning within an on-going burst

Essential concepts of tuning

- *TuningPreset* integer Id of the applicable *tuning preset*
- *CarrierFrequency* value of carrier frequency f_c
- *Gain* value of conversion gain G

Protocols for tuning association: *sequential* and *burst referencing*

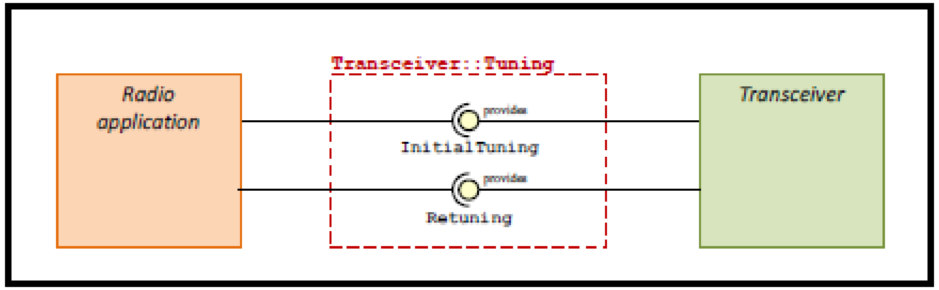
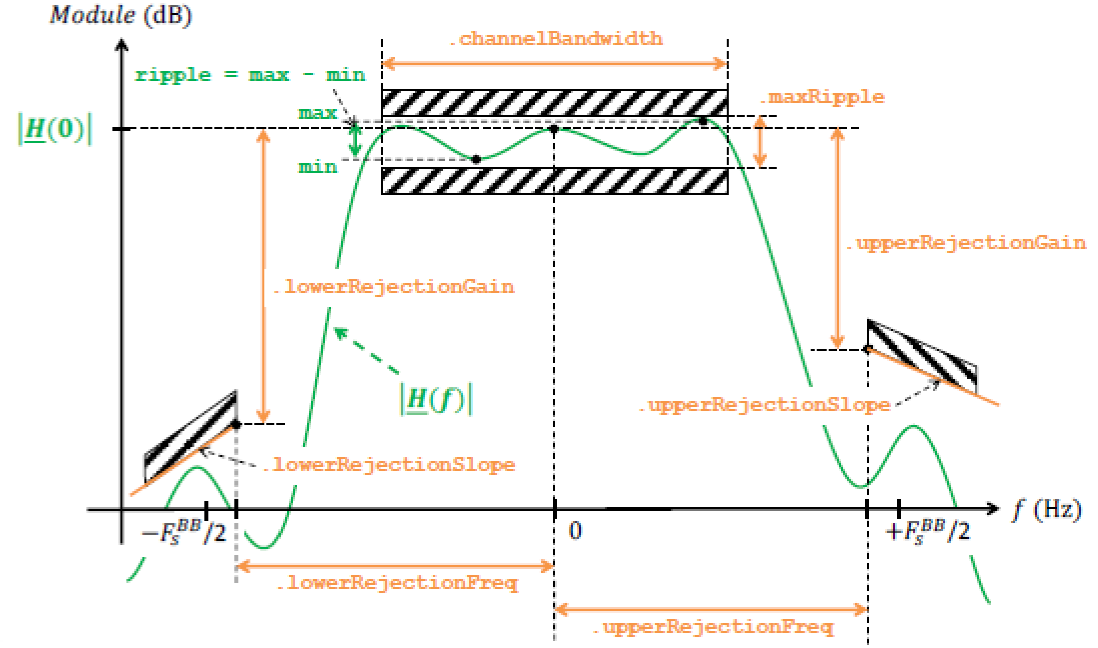


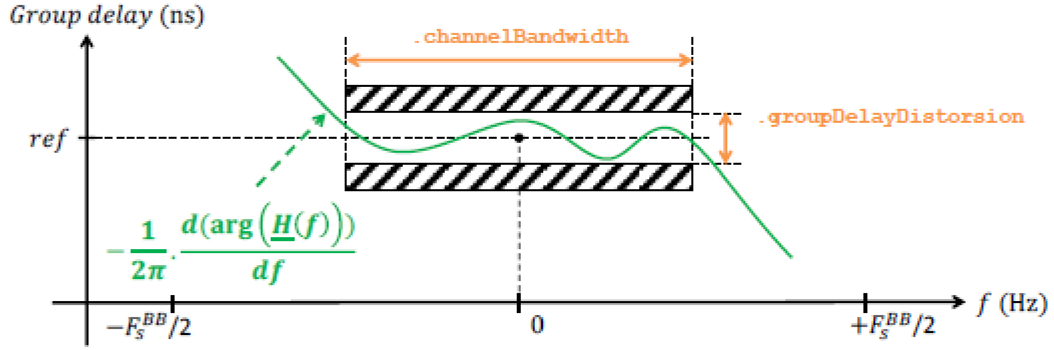
Figure 26 Services of Tuning services group

Essential SGs – Tuning (2)

Implementation of the requested *TuningPreset* is fully under *transceiver* responsibility

Offloads the *radio application* from all platform-specific channelization programming aspects





WInnForum Standards

WIRELESS INNOVATION FORUM

Essential SGs – BasebandSignal (1)

2 essential services

- `SamplesTransmission` packets to *Tx channel* (provide)
- `SamplesReception` packets from *Rx channel* (use)

For each processing phase, a succession of packets are pushed to exchange the samples block

A boolean flag enables to indicate last packet of a block

Packets can optionnally be piggy-packed with metadata

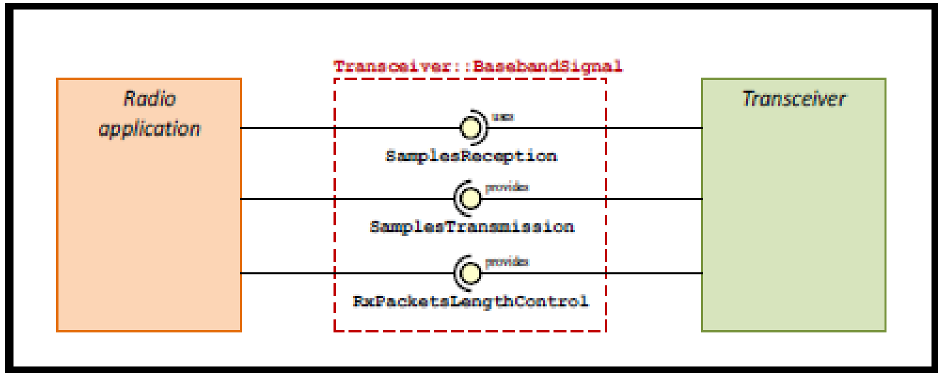


Figure 22 Services of BasebandSignal services group

Essential SGs – BasebandSignal (2)

Synchronization of baseband samples with the specified start time is fully under *transceiver* responsibility

Offloads the *radio application* from all implementation-specific sample chain activation and Tx latency handling aspects

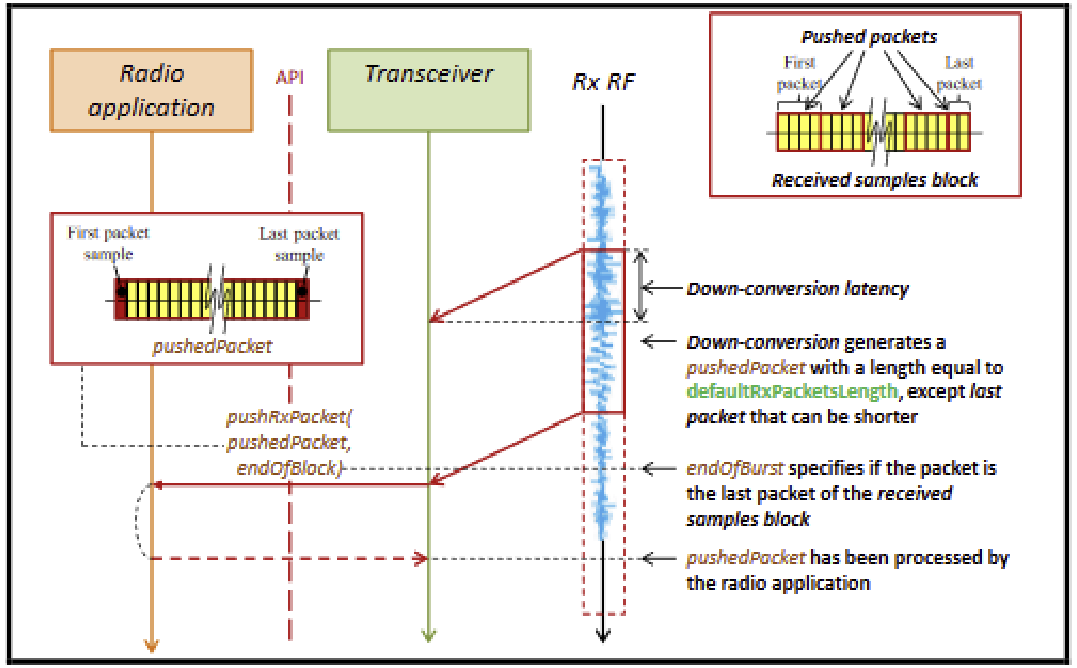


Figure 45 Principle of *pushRxPacket()*

WInnForum Standards™

WIRELESS INNOVATION FORUM

Tx and Rx channels handling

Services instances attachment

- Control services
  - Distinct instances are attached to *Tx channels* and *Rx channels*
  - Each *Tx channels* service instance controls all the *Tx channels*
  - Each *Rx channels* service instance controls all the *Rx channels*
- Data services
  - 1 instance of `SamplesTransmission` service per *Tx channel*
  - 1 instance of `SamplesReception` service per *Rx channel*

Suitable to synchronized usage of multiple channels (e.g. MIMO, Direction Finding)

The possibility to instantiate several *transceivers* enables to address the needs of applications operating independent *transceivers*

PIM provide services

Provide services = **calls from the *radio application* to the *transceiver***

Services groups / Modules	Services / Interfaces	Primitives
Management	::Management::Reset	*reset()*
	::Management::RadioSilence	*startRadioSilence()* *stopRadioSilence()*
BurstControl	::BurstControl::DirectCreation	*startBurst()*
	::BurstControl::RelativeCreation	*scheduleRelativeBurst()*
	::BurstControl::AbsoluteCreation	*scheduleAbsoluteBurst()*
	::BurstControl::StrobedCreation	*scheduleStrobedBurst()*
	::BurstControl::Termination	*setBlockLength()* *stopBurst()*
BasebandSignal	::BasebandSignal::SamplesTransmission	*pushTxPacket()*
	::BasebandSignal::RxPacketsLengthControl	*setRxPacketsLength()*
Tuning	::Tuning::InitialTuning	*setTuning()*
	::Tuning::Retuning	*retune()*
GainControl	::GainControl::GainLocking	*lockGain()* *unlockGain()*
TransceiverTime	::TransceiverTime::TimeAccess	*getCurrentTime()* *getLastStartTime()*
Strobing	::Strobing::ApplicationStrobe	*triggerStrobe()*

PIM use services

Use services = **calls from the *transceiver* to the *radio application***

Services groups	Service / Interface	Primitives
BasebandSignal	::BasebandSignal::SamplesReception	*pushRxPacket()*
Notifications	::Notifications::Events	*notifyEvent()*
	::Notifications::Errors	*notifyError()*
GainControl	::GainControl::GainChanges	*indicateGain()*

The PSM Specifications

PSM Specifications Overview

PSM Specs under finalization

- Native C++
- FPGA
- SCA

PSM Specs under consideration

- Native C
- Other IDL-based compent frameworks, e.g. Redhawk

Reported usages

- Native C++ and FPGA PSMs: consistent content used in ESSOR Program and derivate products
- SCA PSM: consistent content used in current developments of a radio manufacturer and one Viavi Solutions product

User-specific PSMs are possible, although discouraged from portability standpoint

Native C++ PSM Spec

Approach

- For C++03 (DSP compilers) and C++11 (state-of-the art compilers)
- Compliant façades are accessed in C++, locally or via a proxy
- Mapping from PIM IDL signatures based on OMG mapping rules

Specifies

- The Facade class, for façade instances access
- Standard C++ header files (5)
- Optional standard active service access solutions

Development status

- Initial proposal by Thales, 2016
- Finalization under Work Group review
- Ready for WG ballot since Summer 2017

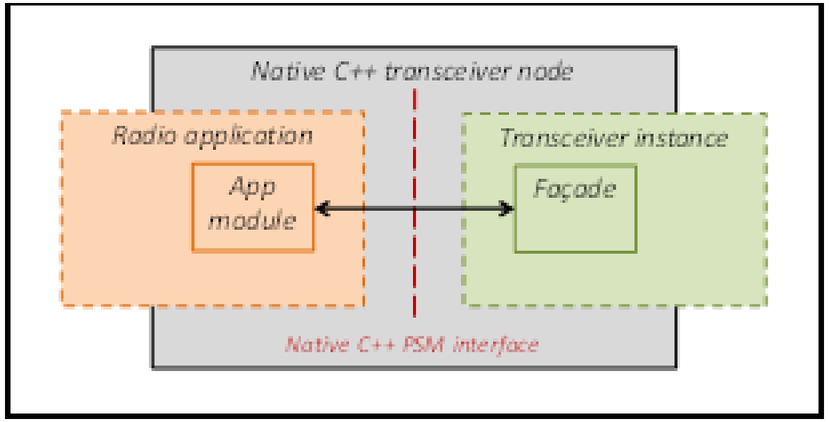


Figure 1 Positioning of Native C++ PSM interfaces

FPGA PSM Spec (1)

Approach

- For all FPGA designs, independently of programming approach
- Compliant façades are accessed locally or via a proxy
- Mapping from PIM IDL signatures based on
  - One-to-one message-oriented mapping for control primitives
  - Specific stream-oriented mapping for data primitives

Specifies

- Standard RTL signals and chronograms
- Appendix: standard VHDL packages (4)

Development status

- Initial proposal by Thales, 2016
- Finalization under Work Group review
- Ready for WG ballot since Summer 2017

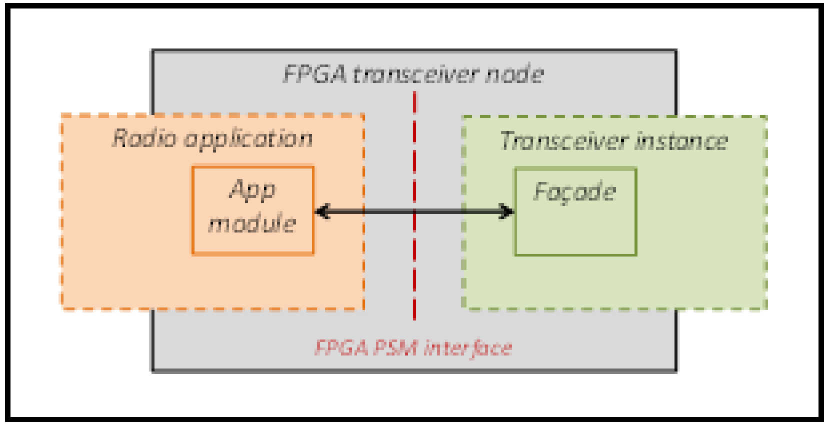


Figure 1 Positioning of FPGA PSM interfaces

FPGA PSM Spec (2)

Data services of the PIM have a specific mapping

- `SamplesTransmission` and `SamplesReception`
- Packet-oriented primitives
  - *pushTxPacket(in BasebandPacket txPacket, in boolean endOfBlock)*
  - *pushRxPacket(in BasebandPacket rxPacket, in boolean endOfBlock)*
- A block of samples is transmitted as a succession of packets

A message-oriented mapping to FPGA would

- Specify a boundary signal for the endOfBlock flag
- Secify an additional boundary signal for the pushed packets
- Only the block boundary signal is necessary

The mapping for data services is therefore stream-oriented

- Only specifies one boundary signal, for the block boundary
- Consistent with FPGAs stream-oriented computing paradigm

SCA PSM Spec (1)

Approach

- Transceiver implemented as an SCA Device
- SCA Ports enabling access to the instantiated services
- **Sets the basis of future PSMs, e.g. the Time Service SCA PSM**

Specifies

- **SCA ports** (services groups-wise or service-wise)
- **SCA 2.2.2** and **SCA 4.1** management
- **SCA properties** derived from PIM
- **Standard IDL files (22)**

Development status

- Initial proposal by NordiaSoft, Nov 2018
- Refinement by NordiaSoft and Thales
- A couple of points remain before finalization

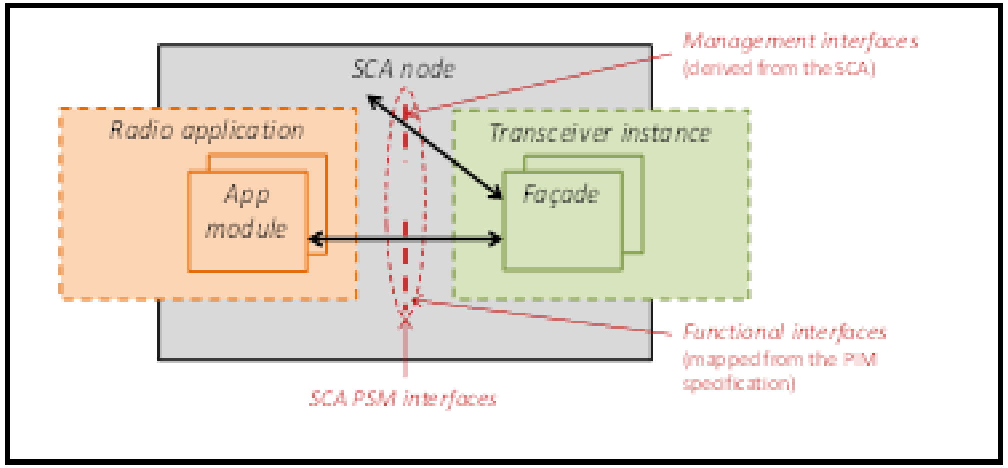


Figure 1 Positioning of SCA PSM interfaces

Conclusions

Conclusions

WInnF Transceiver Facility v2.0 fills a major gap

- Fully implementation agnostic API and portability Properties
- Highly scalable, from GPP to DSPs and FPGAs, from low grade to high performance radio applications and platforms

Unprecedented successful international harmonization

- 12 contributing organizations from 6 countries, leveraging 3 background APIs

Proven technical approach

- Principles validated by successful military programs
- Implemented in several on-going developments

Openly available and maintained by WInnF

- PIM spec published: Jul 2017
- PSM specs published: Sept 2019

A credible candidate for DISR and EDSTAR referencing

End of the presentation
Thank you for your attention

Contacts

- claude.belisle@nordiasoft.com (WG co-chair)
- eric.nicollet@thalesgroup.com (WG co-chair)

- lee.pucker@wirelessinnovation.org (WInnF CEO)

WInnForum Standards™

WIRELESS INNOVATION FORUM®